Methodology for Design and Analysis of Machine Learning Competitions

Adrien Pavão

université PARIS-SACLAY

Methodology for Design and Analysis of Machine Learning Competitions

Méthodologie pour la Conception et l'Analyse de Compétitions en Apprentissage Automatique

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°580 : sciences et technologies de l'information et de la communication (STIC)
Spécialité de doctorat : Informatique
Graduate School : Informatique et Sciences du Numérique
Référent : Faculté des Sciences d'Orsay

Thèse préparée dans l'unité de recherche **Laboratoire Interdisciplinaire des Sciences du Numérique** (LISN), sous la direction d'**Isabelle GUYON**, Professeur

Thèse soutenue à Paris-Saclay, le 5 décembre 2023, par

Adrien PAVÃO

Composition du jury

Membres du jury avec voix délibérative

Bertrand THIRION Chercheur, INRIA, Université Paris-Saclay	Président
Ioannis TSAMARDINOS Professeur, University of Crete	Rapporteur & Examinateur
Mihaela VAN DER SCHAAR Professeur, University of Cambridge	Rapporteur & Examinatrice
Kristin BENNETT Professeur, Rensselaer Polytechnic Institute	Examinatrice
Magali RICHARD Chargée de recherche, Université Grenoble Alpes	Examinatrice

THESE DE DOCTORAT
NNT : 2023UPASG088

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Méthodologie pour la Conception et l'Analyse de Compétitions en Apprentissage Automatique

Mots clés : plan d'experience, apprentissage statistique, concours scientifiques, évaluation

Résumé : Nous développons et étudions une méthodologie systématique et unifiée pour organiser et utiliser les compétitions scientifiques dans la recherche, en particulier dans le domaine de l'apprentissage automatique (intelligence artificielle basée sur les données). De nos jours, les compétitions deviennent de plus en plus populaires en tant qu'outil pédagogique et comme moyen de repousser les limites de l'état de l'art en engageant des scientifiques de tous âges, à l'intérieur ou à l'extérieur du milieu universitaire. On peut y voir une forme de science citoyenne. Cette forme de contribution communautaire à la science pourrait contribuer à la recherche reproductible et démocratiser l'intelligence artificielle. Toutefois, si la distinction entre organisateurs et participants peut atténuer certains biais, il existe un risque que des biais dans la sélection des données, les métriques d'évaluation, et d'autres éléments de conception expérimentale compromettent l'intégrité des résultats et amplifient l'influence du hasard. Dans les cas extrêmes, les résultats pourraient être inutiles, voire préjudiciables à la communauté scientifique et, en conséquence, à la société dans son ensemble. Notre objectif est d'inscrire l'organisation de compétitions scientifiques dans un cadre rigoureux et d'offrir à la communauté des recommandations éclairées. Conjointement avec l'effort de développement des outils d'organisation de compétitions que nous développons dans le cadre du projet CodaLab, nous visons à fournir une contribution utile à la communauté. Cette thèse comprend des contributions théoriques s'appuyant sur la conception expérimentale, les statistiques et la théorie des jeux, ainsi que des résultats empiriques pratiques résultant de l'analyse des données de compétitions passées.

Title : Methodology for Design and Analysis of Machine Learning Competitions

Keywords : experimental design, machine learning, challenges, benchmarks

Abstract : We develop and study a systematic and unified methodology to organize and use scientific challenges in research, particularly in the domain of machine learning (data-driven artificial intelligence). As of today, challenges are becoming more and more popular as a pedagogic tool and as a means of pushing the state-of-the-art by engaging scientists of all ages, within or outside academia. This can be thought of as a form of citizen science. There is the promise that this form of community involvement in science might contribute to reproducible research and democratize artificial intelligence. However, while the distinction between organizers and participants may mitigate certain biases, there exists a risk that biases in data selection, scoring metrics, and other experimental design elements could compromise the integrity of the outcomes and amplify the influence of randomness. In extreme cases, the results could range from being useless to detrimental for the scientific community and, ultimately, society at large. Our objective is to structure challenge organization within a rigorous framework and offer the community insightful guidelines. In conjunction with the tools of challenge organization that we are developing as part of the CodaLab project, we aim to provide a valuable contribution to the community. This thesis includes theoretical fundamental contributions drawing on experimental design, statistics and game theory, and practical empirical findings resulting from the analysis of data from previous challenges.

Acknowledgements

I would like to express my deepest gratitude to the esteemed members of my jury : Bertrand Thirion, Ioannis Tsamardinos, Mihaela Van Der Schaar, Kristin Bennett, and Magali Richard. Their insightful perspectives and stimulating discussions during my PhD defense were invaluable.

A special acknowledgment goes to my supervisor, Isabelle Guyon, whose guidance and opportunities have been fundamental to my growth and development.

I am deeply grateful to Sylvie Pommier, Caroline Appert, Alain Denise, Stéphanie Druetta, and Anne Vilnat for their support, their contributions to the doctorate school, and for granting me the authorization to defend my PhD thesis.

My sincere thanks to my colleagues and friends Romain Egele, Dinh-Tuan Tran, and Marine Djaffardjy for their assistance in preparing and conducting the PhD defense.

I am also thankful to the interns who played a significant role in my journey : Louis Hernandez, Michael Vaccaro, Gaëtan Serré and Aleksandra Kruchinina.

To the CodaLab team : Anne-Catherine Letournel, Laurent Darré, Ihsan Ullah, Benjamin Bearce, Tristan Mary, Tyler Thomas, and Eric Carmichael. Thank you for your contributions.

I extend my gratitude to our industry partners : Stéphane Nachar, Fabrice Lebeau, Thibault Perdrix (from Dassault-Aviation), Antoine Marot, Benjamin Donnot and Laure Crochepierre (from RTE). Your contributions have enriched my research experience.

A special mention to all the researchers who participated in the writing of the book "*AI Competitions and Benchmarks : The Science Behind the Contests*" : Yuna Blum, Harald Carlens, Dustin Carrión, Phil Culliton, Hugo Jair Escalante, Sergio Escalera, Justin Guinney, Julio Jacques Junior, David Rousseau, Gustavo Stolovitzky, Sébastien Treguer, Wei-Wei Tu, Andrey Ustyuzhanin, Jan N. van Rijn, Joaquin Vanschoren and Evelyne Viegas.

Acknowledging Samy Jousset, from Paris Region, for the opportunity to work as a research engineer.

I thank Michèle Sebag and Marc Schoenauer, and more generally all members of the TAU team, for providing such an enriching and caring work environment.

Gratitude is also extended to my friends and colleagues : Bruna Lopes, Anaclara Alvez, Nicolas Atienza, Laurent Basara, Nicolas Béreux, Eva Boguslawsky, Barbara Hajdarević, Diviyan Kalainathan, Thanh Gia Hieu Khuong, Alice Lacan, Armand Lacombe, Zhengying Liu, Emmanuel Menier, Matthieu Nastorg, Solal Nathan, Francesco Saverio Pezzicoli, Audrey Poinsot, Benedictus Kent Rachmat, Cyriaque Rousselot, Théophile Sanchez, Lisheng Sun, Manon Verbockhaven and Zhen Xu. Your friendship and support have been a great source of strength.

I'd like to warmly thank Nicolas Thiery, Dominique Quadri, and Alexandre Allauzen for sparking my passion for computer science, believing in me, and providing numerous opportunities throughout my academic journey.

I thank Steven Lepoutre for his support and insightful advice.

Thank you, Rayan Elalamy, for choosing to help me with mathematical proofs instead of sleeping.

My heartfelt thanks to Eléonore Bartenlian for her unparalleled support, and for her meticulous proofreading of all my papers and of my thesis.

I am grateful to Marie Kovaleff, an exceptional mathematics professor, whose rigor and passion were the foundations of my academic success.

Lastly, to my friends and family, your endless encouragement and support have made my achievements possible.

Thank you all.

Table des matières

1 - Introduction

1.1 . Background

In the evolving landscape of computer science, machine learning (data-driven artificial intelligence) emerges as a rapidly advancing and influential domain, offering endless possibilities. Despite the robust mathematical foundations underlying machine learning methods, they often demand empirical scrutiny to validate their effectiveness and reliability (Langley, 1988; Drummond, 2006; Langley and Kibler, 1991). This trend intensifies as the complexity of the methods increases, especially with the recent advent and widespread application of deep neural networks. Characterized by their intricate architectures and vast parameter spaces, these models perform, by nature, a large number of operations and are particularly hard to explain, analyze and interpret (Gilpin et al., 2018; Ribeiro et al., 2016). This dependence on empirical evaluation originates not only from the inherent complexities of the algorithms but also from the unpredictable and random nature of data. Consequently, experimental benchmarks become indispensable for researchers aiming to compare models and understand their behaviors. The perspective of studying artificial intelligence as an experimental science isn't novel, and is well put by Cohen (1995) :

> *Our subject is empirical methods for studying AI programs, methods that involve running programs and recording their behaviors. Unlike other scientists, who study chemical reactions, processes in cells, bridges under stress, animals in mazes, and so on, we study computer programs that perform tasks in environments. It shouldn't be difficult : Compared with biological systems, AI systems are simple ; compared with human cognition, their tasks are rudimentary ; and compared with everyday physical environments, those in which our programs operate are extremely reduced. Yet programs are in many ways like chemical, biological, mechanical, and psychological processes. For starters, we don't know how they work. We generally cannot say how long they will take to run, when they will fail, how much knowledge is required to attain a particular error rate, how many nodes of a search tree must be examined, and so on. [...] Studying AI systems is not very different from studying moderately intelligent animals such as rats. One obliges the agent (rat or program) to perform a task according to an experimental protocol, observing and analyzing the macro- and micro-structure of its behavior.*

In recent years, the use of **scientific competitions** has made it possible to systematize large-scale experiments and have shown the effectiveness of "crowds" in finding satisfying solutions to difficult problems. The field of data science witnesses an increasing number of such competitions each year, all aiming to address either scientific or industrial dilemmas. Mendrik and Aylward (2019) defines a challenge as an online competition that uses data, truth and metrics to evaluate the performance of automatic algorithms, submitted by participants, with respect to a research problem. A prevalent competition framework prompts participants to develop algorithms addressing a supervised learning problem, offering a price to the best performing participants according to a predetermined measure of success.

However, such a design only scratches the surface of the diversity of problems that challenges can address. Some notable examples of impactful past competitions include the *DARPA Grand Challenge* (Thrun et al., 2006), the *ImageNet Large Scale Visual Recognition Challenge* (Russakovsky et al., 2015; Krizhevsky et al., 2012), the *Netflix Prize* (Bennett and Lanning, 2007) and the *Higgs Boson Challenge* (Adam-Bourdarios et al., 2014, 2016). These competitions have lead to major breakthroughs in their respective fields, and the use of competitions is a growing practice. The *DARPA Grand Challenge* led to significant advances in the development of autonomous vehicles. The *ImageNet LSVRC*, as an annual computer vision competition, contributed in the development of large image datasets and to the emergence of deep neural networks, particularly convolutional neural networks (CNNs), in all industrial and academic fields involving computer vision. The *Higgs Boson Challenge* in high-energy particle physics has played a significant role in enhancing the methods used to detect Higgs boson signals within the extensive datasets produced by the Large Hadron Collider (LHC). This highlights the valuable cross-disciplinary applications of machine learning technologies.

A significant benefit of this benchmarking protocol lies in its ability to create a uniform evaluation process for all candidate models, regardless of their authors. This method ensures unbiased and fair benchmarking, effectively preventing any "inventor-evaluator bias", where the problem could be manipulated to favor a specific solution. Furthermore, this approach leverages the power of crowdsourcing while sparing participants from the burden of data preparation and task formulation.

In this thesis, we propose to develop and study a systematic and unified methodology to organize and use scientific challenges in research, particularly in the domain of machine learning. While the use of challenges might contribute to reproducible research and democratize artificial intelligence, there exists a risk that biases in data selection, scoring metrics, and other experimental design elements could compromise the integrity of the outcomes and amplify the influence of randomness. Our objective is to frame challenge or-

ganization within a rigorous framework and offer the community insightful guidelines. In conjunction with the tools of challenge organization that we are developing as part of the CodaLab project, we aim to provide a valuable contribution to the community. More generally, we study how to compare machine learning models, how to design competitions depending on the problem and what are the tools needed to improve the scientific quality of competitions. This thesis includes theoretical fundamental contributions drawing on experimental design, statistics, social choice theory and game theory, and practical empirical findings resulting from the analysis of data from previous challenges.

Figure 1.1 gives an overview of the life cycle of the resolution of a scientific question as a challenge, the process of designing such a challenge, and the underlying problems we tackled. The genesis of a challenge is a **scientific question**. A scientific question is a question that can be investigated through experiments, observations, or other data collection methods, and is grounded in a testable hypothesis. Countless scientific questions can be studied through challenges, for instance : How to detect cancer at an early stage ? How to optimize energy consumption in cities ? Can a pre-trained neural network be pruned to reduce its size without impacting its performances ? The whole process of designing the challenge is connected to its initial scientific question. This is why each problem requires a **problem specific protocol**. Cancer diagnosis requires specific data. Studying cities energy consumption may require simulated environments. Comparing pruning methods for neural networks requires yet another protocol, where participants are supplied with pre-trained models, and both size and performances are evaluated. Examples of problem specific design are detailed in Chapter 7.

Then, the **data collection** is a crucial step. Data collection includes the gathering of data by any means, its transformation, cleaning, preparation, and labelling. While it is an important step, the data collection is not addressed in this work, as this thesis is focused on the comparison of algorithms.

A challenge serves as an experiment to identify the optimal model for a specific task. Organizing such competitions involves task design and determining a ranking methodology. This includes scoring metrics, ranking functions, significance evaluation, and qualitative expert analysis. In today's data science landscape, metrics must address societal issues, such as algorithm fairness, energy efficiency, and decision interpretability. We suggest metrics to address these concerns. Scoring metrics, presented in Chapter 4, refer to how we quantify the models' performance, while the ranking functions, studied in Chapter 5, are functions used to produce the order of participants (or leaderboard). The ranking is at the same time a *ranking of the participants*, to incentivize participation, select winners and possibly distribute prizes, and most importantly, *a ranking of the methods*, in order to select the best algorithm to

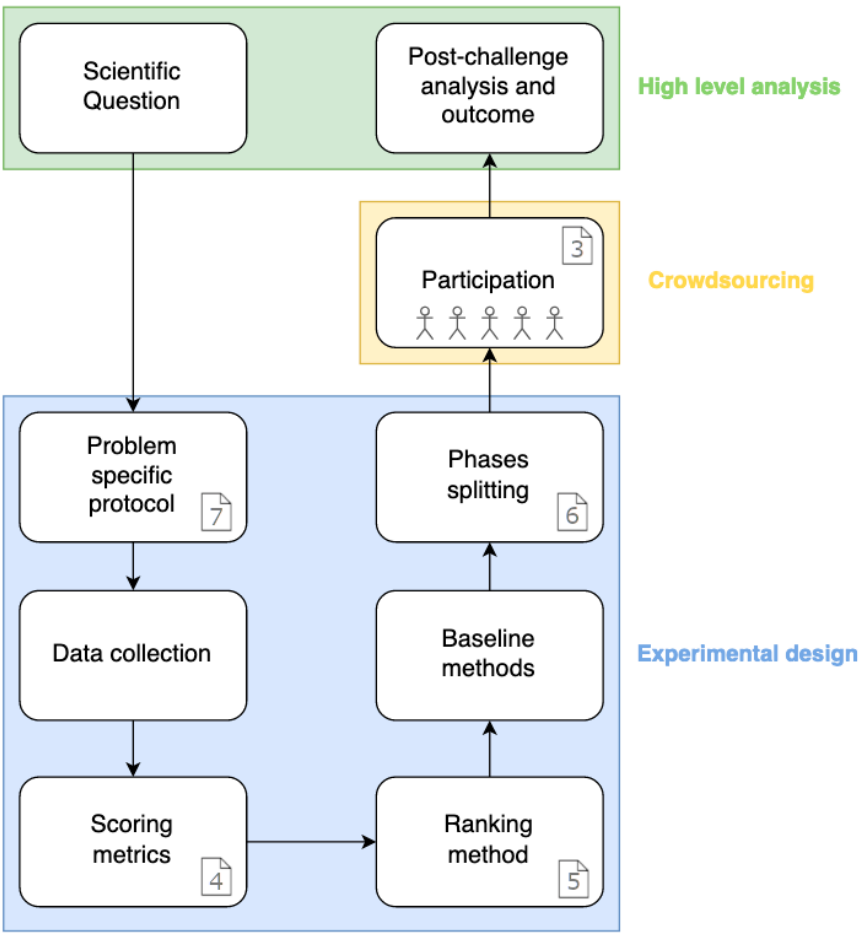


Figure 1.1 – Life cycle of the resolution of a scientific question as a challenge, and the process of designing such a challenge. The corner icons indicate the relevant chapters for each topic : examples of problem specific protocols are given in Chapter 7, scoring metrics are studied in Chapter 4, ranking methods in Chapter 5 and phases splitting in Chapter 6. Participation is made possible by appropriate platforms; we give insights about the online competition platform we manage in Chapter 3.

solve the task under study.

Baseline methods serve as a reference or starting point for participants. These models or techniques provide a benchmark against which more models can be compared. The baseline models offer several key benefits : they give an initial understanding of the problem, act as a sanity check to ensure the more complex models are learning from the data, and set a performance threshold for competition participants.

In challenge design, **structuring the competition into phases** is also crucial. It ensures robust and generalizable model performances. By introducing staged competitions, organizers can effectively mitigate the risk of overfitting. Typically, participants train their models during an initial phase, after which a final evaluation is conducted on unseen data. This process closely mirrors the classic training and testing paradigm in machine learning, reinforcing the idea that competitions are more than just contests — they can be interpreted as meta-learning frameworks that emphasize the criticality of both learning and validation phases. The initial phase can also serve as a filtering mechanism, retaining only those participants who demonstrate strong performance. This approach enhances the winner selection. These ideas are further explored in Chapter 6.

The **participation** is the crucial period when the challenge becomes accessible to contenders and is in progress. The timeline can vary greatly, extending anywhere from a few hours to multiple years. This is where the "magic happens". Indeed, the efficiency of data science competitions resides in their ability to gather a multitude of diverse, innovative approaches from a global pool of participants. Participation is made possible by appropriate platforms; we give insights about the online competition platform we develop and manage, and on the efficiency of the crowdsourcing in Chapter 3.

The final step of this journey is the **post-challenge analysis**. This is where we conclude the high level analysis and finally answer the scientific question. We analyze competitions we have organized in Chapter 8.

It should be noted that this thesis primarily focuses on the underlying **methodology** and the approaches to utilizing crowdsourcing for addressing scientific questions, rather than the specific solutions and scientific advancements achieved through competitions and benchmarks. In the words of Ray Solomonoff, "*In science, my interest was more in how things were discovered than in the contents of the discoveries. The Golden Egg was not as exciting as the goose that laid it*" (Solomonoff, 2011).

1.2 . Outline and contributions

The goal of this thesis is to provide a rigorous foundation for the use of scientific competitions in machine learning. The primary advantage of this pro-

tocol is its ability to offer a standardized evaluation procedure for all candidate models, independent of their authors. This feature ensures a fair and impartial benchmarking of the problem being addressed. Moreover, this approach facilitates efficient problem-solving through crowdsourcing while alleviating participants from the responsibility of data and task preparation.

Our research employed both theoretical and empirical analyses, utilizing metadata from past benchmarks to investigate and refine the comparison of ranking methods. The framework we propose allows a comprehensive evaluation of algorithms, considering various criteria beyond performance, such as ethical and social impact, as well as energy consumption. We also introduced a new empirical framework for studying ranking functions. These ranking functions play a role in ranking candidate algorithms based on a set of scores, typically from a set of tasks. By applying these methods to data from machine learning benchmarks, we found that the "*average rank*" method is effective, displaying favorable theoretical properties and good generalization and stability performance. Additionally, we highlight that filtering participants through a qualification phase statistically improves the generalization of the selected winner, adding to the validity and reliability of the competition outcomes. Finally, we present various competition protocols centered around different tasks, such as metalearning or the automation of machine learning (AutoML).

This thesis primarily focuses on applied research, conducted in conjunction with our Research Engineer position at LISN, and we have applied these methodologies in practice by organizing numerous competitions. Implementing our approach within competitions contributes to advancing scientific rigor in evaluating machine learning models. We advocate for the use of free and open-source tools to promote research accessibility and reproducibility. We also recommend code submission, where participants submit their algorithms instead of predictions, to enhance evaluation robustness and transparency.

As part of the contributions of this thesis work, we have co-authored and co-edited a seminal book titled "*AI Competitions and Benchmarks : The Science Behind the Contests*" (Pavao et al., 2023b) with Evelyne Viegas and Isabelle Guyon, along with multiple other contributors. This book serves as a comprehensive guide to the scientific foundations, methodologies, and practical insights related to challenges in artificial intelligence. We are in the process of submitting the book to the newly established journal Data-centric Machine Learning Research (DMLR), further extending its reach and impact within the academic community.

The list of published papers is given below :

- Judging competitions and benchmarks : a candidate election approach, *ESANN 2021* (Pavao et al., 2021a).

In this work, we study the problem of ranking machine learning models on a set of tasks. To that end, we review theoretical properties of ranking functions, and propose novel empirical measures to characterize them. We conclude that the methods called *average rank*, *Copeland's method* and *relative difference* are great candidate to solve this problem.

- Filtering participants improves generalization in competitions and benchmarks, *ESANN 2022* (Pavao et al., 2022b).
  In this paper, we highlight that filtering the number of candidates during the first stage of a competition increases the probability of selecting a general winner in the final phase. The generalization ability is thought of as the probability of winning subsequent post-final phases. We show this statistical properties on simulated experiments, both on artificial and real data.
- CodaLab Competitions : An open source platform to organize scientific challenges, *JMLR MLOSS* (Pavao et al., 2023a).
  This contribution to the *Machine Learning Open-Source Software* track consists in the code of the competitions platform *CodaLab Competitions*, and a paper highlighting the key concepts behind its design : result and code submissions, modularity of computational resources, containerized environments for reproducibility, and more. The platform regroups more than 100,000 users and 800 public challenges.
- Aircraft numerical "twin" : A time series regression competition, *ICMLA 2021* (Pavao et al., 2021b).
  This paper summarizes the design and the analysis of the AI4Industry Challenge, a multi-variate time series regression competition in the field of Aeronautics, organized in partnership with Dassault-Aviation. *Gradient boosting machines (GBM)* were the best performers on this task. Several methodological points are raised, such as the estimation of variance in scores for different granularity levels, and the use of confidential data.
- Design and Analysis of Experiments : A Challenge Approach in Teaching, *CiML Workshop 2019* (Pavao et al., 2019).
  In this abstract, we present the way we used machine learning challenges in teaching at the University. The graduate students were competition designers and organizers, while the under-graduate students were participants. We show how it is an efficient learning tool, as well as having a positive impact on the community.

Publications as co-author or papers published prior to the commencement of our thesis work :

- Winning solutions and post-challenge analyses of the ChaLearn AutoDL challenge 2019, *IEEE TPAMI* (Liu et al., 2021a).
  In this paper, we present the design and the analysis of the Automa-

ted Deep Learning Challenge, an ambitious competition involving blind-testing of algorithms on a wide variety of classification tasks. AutoDL Challenge involved a complex evaluation protocol and we gathered many insightful metadata from it. The winning solution was fast to train and reliable on all data domains tackled by the competition : image, video, audio, time series and tabular.

- Codabench : Flexible, easy-to-use, and reproducible meta-benchmark platform, *Patterns 2022* (Xu et al., 2022).
  This paper introduces *Codabench*, a modernized version of *CodaLab Competitions* with a emphasis on benchmarks. *Codabench* implements all key features from *CodaLab Competitions*, as well as new useful features such as the transparency of evaluation methods, real-time feedback to participants, and fact sheets forms.

1.3 . Synthèse en français / Summary in French

L'objectif de cette thèse est de fournir une base rigoureuse pour l'utilisation des compétitions scientifiques en apprentissage automatique. Le principal avantage de ce protocole est sa capacité à offrir une procédure d'évaluation standardisée pour tous les modèles candidats, indépendamment de leurs auteurs. Cette caractéristique garantit une évaluation juste et impartiale du problème traité. De plus, cette approche facilite la résolution efficace des problèmes par le biais du crowdsourcing, tout en soulageant les participants de la responsabilité de la préparation des données et des tâches.

Notre recherche a employé des analyses à la fois théoriques et empiriques, en utilisant les métadonnées des benchmarks passés pour étudier et affiner la comparaison des méthodes de classement. Le cadre que nous proposons permet une évaluation complète des algorithmes, en considérant divers critères au-delà de la performance, tels que l'impact éthique et social, ainsi que la consommation d'énergie. Nous avons également proposé un nouveau cadre d'étude empirique des fonctions de classement. Ces fonctions de classement entrent en jeu dans le classement d'algorithmes candidats à partir d'un ensemble de scores, venant typiquement d'un ensemble de tâches. En appliquant ces méthodes à des données issues de benchmarks en apprentissage automatique, nous concluons que la méthode du "*rang moyen*" est efficace, montrant des propriétés théoriques favorables et de bonnes performance de généralisation et de stabilité. De plus, nous mettons en lumière le fait que le filtrage des participants par une phase de qualification améliore statistiquement la généralisation du gagnant sélectionné, contribuant à la validité et à la fiabilité des résultats de la compétition. Finalement, nous présentons divers protocols de mise en place de compétitions autour de divers tâches, telles que le metalearning ou l'automatisation du machine learning

(AutoML).

La mise en oeuvre de notre approche au sein des compétitions contribue à renforcer la rigueur scientifique dans l'évaluation des modèles d'apprentissage automatique. Nous préconisons l'utilisation d'outils libres et open-source pour promouvoir l'accessibilité et la reproductibilité de la recherche. Nous recommandons également l'utilisation de soumission de code, où les participants transmettent leurs algorithmes au lieu de simples prédictions, afin de renforcer la robustesse et la transparence des évaluations.

À mesure que l'intelligence artificielle occupe une place de plus en plus prépondérante dans notre société, il devient essentiel d'évaluer rigoureusement les programmes, en tenant compte de l'aspect expérimental, voire comportemental, de cette évaluation, afin d'assurer un progrès scientifique intègre et rigoureux.

2 - Competitions and benchmarks

The advancement of machine learning (ML) research is significantly dependent on benchmarking algorithms. Challenges can be viewed as a method for gathering the collective efforts of the community to solve complex AI problems and create a standardized benchmark for numerous researchers and ML practitioners. This chapter is a brief overview of the state of competitions and benchmarks, situating them historically, and putting them in the broader framework of crowdsourcing.

2.1 . Brief history

Challenges have played a crucial role in the evolution of AI and predictive modeling, involving a stated problem, competitors, public dataset, and scoring methodology (Liberman, 2010; Donoho, 2017). In her NeurIPS 2022 keynote talk [1], Isabelle Guyon presents the rich history of competitive machine learning, serving as a primary inspiration for the historical overview that follows. The idea of leveraging a community of experts and non-experts to solve a scientific problem has been around for hundreds of years. The long history of scientific challenges begin with the brachistochrone problem, presented by Johann Bernoulli in 1696 as a challenge to the best mathematicians (Herrera, 1994; Bernoulli, 1696). The problem seeks to find the curve of quickest descent, where an object under gravity travels between two points in the shortest possible time. This optimal curve is called the "brachistochrone" curve. Five prominent mathematicians submitted solutions to the challenge : Johann Bernoulli himself (anonymously), Jakob Bernoulli (his brother), Isaac Newton, Gottfried Wilhelm Leibniz and Guillaume de l'Hôpital, although there may have been others who attempted to solve the problem. Leibniz proposed all the received solutions (Leibniz, 1697). The brachistochrone problem was related to mathematics, physics, and engineering. Similarly, machine learning (ML) challenges often cut across multiple domains, and solutions may require knowledge and expertise from various fields. This interdisciplinary approach can lead to more robust and innovative solutions. Another early example of crowd-sourced challenge is the 1714 British Board of Longitude Prize, which was to be awarded to the person who could solve arguably the most important technological problem of the time : to determine a ship's longitude at sea (Sobel, 2005). After challenging many established scientists of the time, the prize was awarded to John Harrison for his invention of the marine chronometer. There are two important take home messages from the Longitude

1. https://nips.cc/virtual/2022/invited-talk/56158

Prize example. One is the fact that the winner of the prize was John Harrison, an unknown carpenter and clock-maker, and not a more recognized scientist of that era. The second key idea is that the problem was posed as an open participation challenge, what we refer to today as crowdsourcing. When it comes to novel problems, it is possible that the breakthroughs do not come from other the most established experts in a field.

Obviously, the earliest challenges do not include machine learning benchmarks, since this discipline had yet to come into existence, and its birth occurred in the twentieth century. Before the 1950s, researchers had discovered and refined statistical techniques, establishing the foundation for future developments. In 1950, Alan Turing proposed the *Turing Test*, a measure of a machine's ability to exhibit intelligent behavior indistinguishable from that of a human. While not a machine learning challenge in the modern sense, it set an early standard for evaluating artificial intelligence. In the 1950s, pioneering machine learning research was carried out using light algorithms, and by the 1960s, Bayesian methods were introduced for probabilistic inference within the field. However, the 1970s marked an "AI winter", characterized by pessimism regarding the effectiveness of machine learning. This period of stagnation was followed by a revival in the 1980s, triggered by the discovery of the modern backpropagation algorithm.

Before the 1990s, datasets were exceedingly scarce, often consisting of what we now refer to as "toy data". It was common to primarily demonstrate new algorithms using synthetic data or small datasets. One such dataset remains commonly used in many introductory machine learning tutorials. This dataset is the Iris dataset (Fisher, 1936), which presents a 3-class classification challenge involving three distinct types of Iris flowers. Remarkably, these classes can be effectively separated using a linear discriminant classifier, commonly known as Fisher's linear discriminant, employing just four features : petal length and width, as well as sepal length and width. Each class in the dataset comprises 50 examples.

The 1990's marked a shift from knowledge-driven to data-driven approaches in machine learning, with a focus on analyzing large datasets. This era witnessed the rise of support-vector machines (SVMs), recurrent neural networks (RNNs), and an increase in computational complexity through neural networks. An important landmark in ML dataset availability was the creation of the UCI ML repository in 1987 by David Aha and his students. The initial datasets were also rather small in size, with the number of examples ranging from 50 to a few hundred and the number of features not exceeding 100. The initial datasets were all tabular dataset, that is tables with samples in lines and features in columns. Raw data, like images, sound, text, video, appeared only later. These datasets were widely used in NeurIPS papers during the 1990's and as you can imagine people started overfitting them. Even more

concerning, people started reporting results only on the subset of datasets that made their algorithms shine. This was denounced with humor in a joke paper submitted to NeurIPS in 2002, "Data Set Selection", pretending to give a theoretical backing to the bad habit of dataset selection (LaLoudouana and Tarare, 2002). The 1990's were marked also by the appearance of systematic benchmarks like the EU project Statlog (King et al., 1995). A consortium of 13 institutions worked together to compare a large number of methods on a large number of datasets, mostly coming from the UCI ML repository. They produced a ranking of algorithms for each dataset. The algorithms included classical statistics (linear and quadratic discriminant, logistic regression, KNN, BN), machine learning (decision tree, rule-based) and neural networks (MLP).

Interestingly, they made a methodology statement :

> *"The Project laid down strict guidelines for the testing procedure. First an agreed data format was established, algorithms were "deposited" at one site, with appropriate instructions [...]. Each dataset was then divided into a training set and a testing set, and any parameters in an algorithm could be "tuned" or estimated only by reference to the training set. Once a rule had been determined, it was then applied to the test data. This procedure was validated at another site by another (more naive) user. This ensured that the guidelines for parameter selection were not violated, and also gave some information on the ease-of-use for a non-expert in the domain."* – Michie et al. (1994)

The methodology described by the Statlog authors encapsulates crucial principles of machine learning. They separate the data into training and testing sets, which allows for the evaluation of a model's performance on unseen data. They ensure that model parameters are tuned only with reference to the training set, preventing overfitting. They also apply replication and validation procedures, enhancing the robustness of their findings and preventing overoptimization. By storing algorithms at a single site, they enable standardization and comparison, contributing to the development of robust and reproducible machine learning research and practices. In 1997, the inaugural KKD Cup was organized (Parsa, 1997; Kohavi et al., 2000). Hosted by the Association for Computing Machinery (ACM), this annual competition in Knowledge Discovery and Data Mining is often regarded as the pioneering modern machine learning contest. It continues to be held annually to this day. In the first edition, participants were tasked with classifying anonymous Microsoft web page users. Over the years, the KDD Cup has tackled a variety of problems from different domains [2]. Challenges have ranged from predicting protein structures, to network intrusion detection, to recommendation systems, and more. Similarly, the National Institute of Standards and Technology (NIST) held their first

2. https://kdd.org/kdd-cup

Speaker Recognition Challenge in 1996 (Martin and Przybocki, 2001). This series of challenges, focused on automatic speaker recognition technology, has received significant appreciation from the research community and continues to be held to this day [3] (S. et al., 2020).

As the field entered the 2000's, we observed a rise in popularity of techniques like support-vector machines, kernel methods, and unsupervised learning. In these years, people from the NeurIPS community started focusing more on raw data, as opposed to data preprocessed as nice tables. In the realm of image recognition, many datasets relating to Optical Character Recognition (OCR) were collected. OCR is the technology used to convert different types of documents, such as scanned paper documents, PDF files or images captured by a digital camera, into editable and searchable data. There was also an increased interest in datasets for face and object recognition, along with video datasets designed to recognize human actions. In the field of speech recognition, the Linguistic Data Consortium popularized various datasets (Cieri and Liberman, 2000; Maeda and Strassel, 2004). This organization also provided several extensive text corpora for language study, significantly contributing to language research and natural language processing. Researchers also started to explore other forms of sensor data, including Electroencephalogram (EEG) data, which captures electrical activity in the brain. Additionally, the first datasets representing graph data began to emerge, opening new possibilities for research and algorithm development in the field of graph theory and network analysis.

The 2000's were marked by the notable Netflix Prize (Bennett and Lanning, 2007). Announced in 2002 and launched in 2006, this competition continued with various rounds until 2009 [4]. Hosted by Netflix, an online movie rental service, its aim was to enhance their movie recommendation algorithm. A grand prize of $1,000,000 awaited the team or individual capable of achieving a 10% improvement over Netflix's existing recommendation system. This decades also witnessed the DARPA Challenge, a series of autonomous vehicle competitions organized by the Defense Advanced Research Projects Agency (DARPA). These competitions, initiated in 2004, were designed to promote the development of autonomous ground vehicles to navigate complex terrains without human intervention. The challenges varied in complexity, from navigating desert terrains to urban settings, pushing the boundaries of robotics, machine learning, and computer vision. The advancements from these challenges have significantly impacted the automotive industry, paving the way for the recent growth in autonomous vehicle research (Thrun et al., 2006).

The 2010's marked the rise of deep learning, which made machine learning an essential component of various software services and applications

3. https://www.nist.gov/itl/iad/mig/speaker-recognition
4. https://en.wikipedia.org/wiki/Netflix_Prize

used widely across industries. At the dawn of our millennium, the euphoria of "big data" led industry leaders to believe that all problems could eventually be solved by adding in more data. Peter Norvig, Director of Research at Google, is often quoted for having said in 2011 that "We don't have better algorithms, we just have more data". However, this simplistic idea has back-fired, with several embarrassing failures of algorithms making racist or sexist decisions (Zou and Schiebinger, 2018). This prompted Peter Norvig to revise his claim in 2017 to "More data beats clever algorithms, but better data beats more data". These are also the years where the first systematic competitions platform appeared, as *Kaggle* (Goldbloom and Hamner, 2010) was launched in 2010 and *CodaLab Competitions* (Pavao et al., 2023a) in 2013. There was a noticeable trend of an increasing number of challenges being organized in conjunction with scientific conferences. ChaLearn, a non-profit organization focused on organizing machine learning challenge, was founded in 2011. At that time, the researchers behind ChaLearn had already organized many challenges, for instances the Feature Selection Challenge at NIPS 2003 (Guyon et al., 2004), the Performance Prediction Challenge at WCCI 2006 (Guyon et al., 2006a), and the Active Learning Challenge at AISTATS 2010 (Guyon et al., 2011). They pursue this efforts by hosting a variety of challenges, covering diverse areas like computer vision through the "Looking at People" series, neurology, causal discovery, automated machine learning and physics [5].

This period also saw a rise in interest in the area of ethics in AI and algorithmic fairness, as the awareness of the societal implications of AI grew. More recently, there has been a growing emphasis on the need for explainable and interpretable AI, particularly given the opaque nature of many deep learning models. Despite the challenges, the progress made in these years set the stage for many exciting developments in machine learning that we are witnessing today.

And there we are now in the 2020's, hopefully at the beginning of maturity in Machine Learning. The center piece is going to be peer review of datasets and benchmarks, which should become a standard and that NeurIPS is strongly encouraging by establishing the NeurIPS D&B track. This decade also brings with it a stronger focus on the robustness and reproducibility of machine learning models, underlining the importance of understanding and documenting every step of the machine learning pipeline from data collection to model deployment. The growing popularity of initiatives like PapersWithCode and HuggingFace underscores this point. These platforms facilitate the reuse of state-of-the-art machine learning algorithms, promoting greater accessibility and efficiency in the field. Competitions continue to demonstrate their efficiency in driving innovation and solving long-standing problems, as illustrated by the CASP14 competition, where DeepMind's AlphaFold 2 achieved

5. http://www.chalearn.org/challenges.html

breakthrough accuracy in protein structure prediction (Jumper et al., 2021b,a).

2022 was a big year for competitive machine learning, with a total prize pool of more than $5,000,000 across all platforms, as highlighted by Carlens (2023). Hundreds of competitions were organized, and the range of machine learning problem studied was wider than ever : computer vision, natural language processing, sequential decision-making problems, robotics, graph learning, optimization, automated machine learning, audio processing, security, meta-learning, causal inference, time-series forecasting, and more, with applications in all fields. In today's landscape, there are dozens of platforms, as discussed in Chapter 3.

2.2 . Machine learning as an experimental science

With a more comprehensive understanding of the history of competitions and benchmarks, let's explore their connection with experimental science. We'll also discuss potential future trajectories, especially as they shift towards behavioral science and holistic evaluation methods. Many of the fundamental techniques and ideas in AI, particularly in the domains of machine learning and deep learning, have been known for a long time. However, their recent resurgence and practical success can be attributed to the joining of several factors, such as the evolution of processing power, the abundance of data, and the development of new tools and frameworks. Over the past few decades, there has been a tremendous growth in computing capabilities. Graphics processing units (GPUs) and tensor processing units (TPUs) have played a key role in making deep learning models more efficient and scalable, as these specialized processors enable parallel processing. This has significantly reduced the time required to train complex models, allowing researchers to experiment with larger and deeper neural networks that were previously computationally out of reach. In addition to the advances in computing power, the digital revolution has led to an explosion of data across various domains. This vast quantity of data provides an ideal environment for training data-hungry deep learning models.

The availability of large-scale datasets has been instrumental in achieving state-of-the-art results in numerous applications, such as computer vision, natural language processing, and speech recognition. In essence, the abundance of data has enabled AI techniques to truly demonstrate their potential and transform the way we approach problem-solving. A survey conducted on 786 papers published at NeurIPS2019 and ICLR2020 shows that around 90% of these papers are experimental research, as opposed to theoretical research (Bouthillier and Varoquaux, 2020). Another significant factor that has contributed to the practical success of AI is the development of open-source tools and frameworks, which have democratized access to AI technologies. Libra-

ries and platforms such as PyTorch (Paszke et al., 2019), Keras (Chollet et al., 2015) and Scikit-learn (Pedregosa et al., 2011) have made it easier for researchers and practitioners to design, implement, and experiment with various deep learning models. By lowering the barrier to entry, these tools have encouraged a broader community to contribute to the advancement of AI.

The evolution of deep neural networks calls for a shift towards a **behavioral science approach** for evaluating their performance. This requires a comprehensive method that starts with defining the desired behaviors for specific tasks, such as ethical alignment and explainability, and then establishing behavioral benchmarks that measure the models' cognitive skills like reasoning and generalization. Ethical considerations are crucial, as we need to examine whether the model exhibits biased behavior or generates harmful content. Observational evaluation, which involves analyzing the model's behavior in real-world scenarios or through simulations, also becomes essential to identify any unexpected or undesirable outcomes. By incorporating user feedback, developers can gain an understanding of the models' strengths, weaknesses, and areas for improvement. Combining these elements, a behavioral science approach to evaluating machine learning models can provide a more holistic understanding of their behavior, leading to better-performing and more responsible AI systems. Following this idea, a clear example is how the Generative Pre-trained Transformers (GPT) (OpenAI, 2023), the famous large language models, was tested using psychology tests (Uludag and Tong, 2023; Li et al., 2022), high-school tests (de Winter, 2023) and mathematics tests (Frieder et al., 2023). GPT was also compared to human annotators (Huang et al., 2023) and has even demonstrated the ability to outperform Amazon Mechanical Turk (ATM) workers (Gilardi et al., 2023) [6]. The Beyond the Imitation Game Benchmark (BIG-bench) (Gur-Ari et al., 2022) makes a step forward in this direction, proposing an extensive benchmark for large language models including more than 200 tasks of understanding, reasoning, summarization or even alignment and self-awareness. In a similar fashion, the "Holistic Evaluation of Language Models" (HELM) benchmark (Liang et al., 2022) grouped numerous natural language processing tasks, and measured the performances of language models using a multi-metric approach, measuring accuracy, calibration, robustness, fairness, bias, toxicity, and efficiency. Based on the findings of Maslej et al. (2023), while the frontiers of artificial intelligence are consistently advancing, the year-over-year improvement on many benchmarks is marginal. Moreover, the speed at which benchmark saturation is being reached is increasing. The rise of more comprehensive benchmarking suites such as BIG-bench and HELM offers potential solutions to this trend. Evaluating the behaviors of deep neural networks through a behavioral science lens can provide a more com-

6. This could speed up the developments of future AI models, given that AMT is widely used to label training data.

prehensive understanding of their strengths and weaknesses. While this approach may be the future of the evaluation methods for heavy deep learning models, many machine learning models in-use for precise tasks are not large and opaque models, and won't necessarily benefit from this approach.

2.3 . Motivations and state-of-the-art

Having discussed the historical landscape and projected future trends of competitions and benchmarks, it is important to understand the incentives behind their organization. What motivates such crowdsourced benchmarks in the domain artificial intelligence ? Why invest time and resources in these events ? We mentioned earlier that the main motivations for the use of scientific competition is the lack of inventor-evaluator bias and an improved reproducibility.

Crowdsourced challenges offer unbiased evaluation through the openness to external contributors, avoiding the inventor-evaluator bias, or "self-assessment trap" (Norel et al., 2011). We can consider that a benchmark is not a fair comparison of machine learning methods if the number of models included in the study is low. Bouthillier and Varoquaux (2020) conducted a survey on experimental methods of around 780 papers published at NeurIPS2019 and ICLR2020. The majority of papers (73%) that included experiments had between 1 and 5 baselines to compare with the model they proposed. More specifically, 17% of papers had between 5 and 10 baselines, and only 5% had more than 10 baselines. Moreover, nothing indicates that the researchers did their best to provide competitive baselines that may out-perform their own method.

The other benefit of competitions we highlight is the improved reproducibility. According to Baker (2016), on more than 1,500 researchers surveyed, 52% claim that there is a reproducibility crisis in scientific research, and only 3% claim that there is no crisis. Raff (2019) tried to reproduce the results of 255 papers published from 1984 to 2017 [7]. Their method consisted in rewriting their own code in order to reproduce the experiments found in the papers. They were able to successfully reproduce the results of 162 papers out of the 255, which is a reproducibility rate of 63.5%. Even if this result is not completely satisfying, it is far more optimistic than the reproducibility rate of 26% reported by Gundersen and Kjensmo (2018). This second study include 400 papers from the conference series IJCAI and AAAI, and defines reproducibility with a metric based on documentation (grouped into three factors : experiment, data and method) instead of trying to reproduce the experiments manually. According to Raff (2019), the reproducibility of papers did not improve

7. The papers publication date were not equally distributed, as most papers included in this study were published between 2000 and 2017.

in recent years, thus one can argue either that there is no reproducibility crisis, or it has been going on for decades. In the case of online competitions and benchmarks, as we show in Chapter 3, we can efficiently store the whole experimental procedure (including the data, the models and the evaluation program) in order to replicate the results after the end of the challenge.

It is also a way to stimulate the community to solve new hard problems and foster fair comparisons between methods. High visibility success stories include the Netflix prize of movie recommendation, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), the CERN Higgs Boson challenge, the Learning to Run reinforcement learning challenge, the SemEval NLP series, the DREAM bioinformatics challenges, to name a few. Over time, competitions have gained in sophistication. Single phase challenges with a simple training/test data split have been replaced by 2-phase challenges, allowing participants during a "development phase" (or "feedback phase") to practice with performance feedback on a validation set, before being evaluated during a "final phase" with just one submission on a final test set. Additionally, AutoML competition protocols have been introduced, using complete blind testing of code submissions on different tasks in the various phases (Guyon et al., 2019b; Liu et al., 2021b), and then meta-learning challenges (Baz et al., 2022).

Ultimately, one strong motivation is to weed out papers, some of which are not worth reading because they are plagued by inventor-evaluator bias. While at the onset of machine learning in the 1980's it was possible to keep up with the literature, the rapid growth of the number of publications quickly made this task daunting. Thus, competitions have become a resourceful means of recommending algorithms.

Crowdsourced challenges offer an alternative way to conduct scientific research and problem-solving by engaging diverse solvers and providing valuable data. As data generation outpaces our ability to analyze it, it is useful to explore different research methods. These challenges can enhance academic education and research, serving as learning modules or opportunities for interdisciplinary collaboration. Crowdsourced challenges can produce rigorously benchmarked data and methods, with results representing the state-of-the-art in their fields, as in the Sage-BCC DREAM Challenge (Margolin et al., 2013). However, the increasing number of potential challenges presents difficulties in selecting the most impactful ones. A potential solution may involve crowdsourcing the challenge selection process, letting the community decide which problems to address, as participation is crucial to the success of these challenges.

2.4 . Types of crowdsourcing

While this thesis is focused on competitions and benchmarks, other related types of crowdsourcing should be mentioned. Introduced by Jeff Howe in an article in Wired Magazine (Howe, 2006), crowdsourcing is "*the act of taking a job traditionally performed by a designated agent (usually an employee) and outsourcing it to an undefined, generally large group of people in the form of an open call*". Crowdsourcing has been used in many contexts such as business (Boudreau and Lakhani, 2013), journalism (for the collection of information), and peer-review. Here, we are interested in the application of crowdsourcing to computational problems in science and technology.

Crowdsourcing can involve the crowd providing data or actively working to solve problems (Good and Su, 2013). Active crowdsourcing has various forms, including volunteer-based crowdsourcing, ideation challenges, human intelligent tasks, benchmarks, competitions and coopetitions. Another appraoch consists in dividing a problem into smaller tasks, such as in bioinformatics (Ansari et al., 2013; Kutmon et al., 2016; Thiele et al., 2013; Vashisht et al., 2012; Mortensen et al., 2015).

Volunteer-based crowdsourcing

Volunteer-based crowdsourcing occurs when individuals contribute their time or resources out of interest, such as in Wikipedia, Foldit (Cooper et al., 2010), Folding@home (Larson et al., 2009), and rosetta@home (Das et al., 2007). Gamification has proven successful in increasing engagement, as seen in Foldit and EteRNA (Treuille and Das, 2014; Lee et al., 2014; Cooper et al., 2010; Andreasson et al., 2022). Other games like EVE Online (Sullivan et al., 2018) and Borderlands Science (Waldispühl et al., 2020) also harness crowdsourcing to solve problems.

Ideation challenges

Ideation challenges collect new ideas or directions, such as the Longitude Prize 2014 (Rees, 2014) and the XPRIZE Qualcomm Tricoder prize (Chandler, 2014). Combining crowdsourcing with benchmarking activities enables rapid development of state-of-the-art solutions. Participants work on provided data, and organizers evaluate solutions against Gold Standard data. This encourages collaboration and idea exchange.

Human intelligent tasks

Human Intelligent Tasks (HITs) refer to tasks that require human intelligence, which are typically distributed among a crowd of people in a crowdsourcing platform, such as Amazon's Mechanical Turk (Goodman et al., 2013). This is a labor-focused type of crowdsourcing, where anyone can take up a job. These tasks can be difficult for machines to perform autonomously or require a human's nuanced understanding and problem-solving abilities. HITs may include tasks such as image annotation, natural language processing, or senti-

ment analysis. In machine learning, HITs often serve as a means to generate labeled training data, validate algorithms, or evaluate model performance.

Benchmarks

A benchmark is a standardized set of tasks, problems, or datasets for evaluating the performance of machine learning models or algorithms. Benchmarks enable researchers and developers to compare the performance of different models and techniques objectively, providing insights into the strengths and weaknesses of each approach. Benchmarks are crucial for tracking progress in the field and promoting the development of more efficient and effective machine learning solutions. Examples of widely-used benchmarks include ImageNet for image classification and SQuAD (Rajpurkar et al., 2016, 2018) for question-answering.

Competitions

Competitions are organized events or challenges in which teams or individuals compete to develop the best possible solution to a specific problem or task. Competitions typically provide a problem statement, a dataset or set of tasks, and evaluation metrics to measure the performance of submitted solutions. Competitions encourage innovation and collaboration among researchers and practitioners, often resulting in significant advancements in the field. Well-known examples include the platforms Kaggle (Goldbloom and Hamner, 2010) and CodaLab Competitions (Pavao et al., 2023a), hosting machine learning competitions covering diverse tasks and domains of application.

Coopetitions

Collaborative competitions, or coopetitions, have become popular among research scientists in recent years. These competitions utilize leaderboards to enable participants to track their performance and receive real-time feedback, while also offering incentives such as monetary prizes and co-authorship opportunities. Factors that contribute to successful challenges include low entry barriers, continuous organizer engagement, small milestones, high-impact problems, and opportunities for recognition or job offers.

The wisdom of the crowd emerges in these contexts, as aggregating solutions from independent teams often yields more accurate results than individual solutions (Surowiecki, 2005). Collaborative competitions are used in various scientific and technological fields, such as machine learning (Kaggle, CodaLab (Pavao et al., 2022a)), structural biology (CASP (Kryshtafovych et al., 2021), CAPRI33), human genetic variation interpretation (CAGI (Hoskins et al., 2017)), and experimental technology assessments (BioCreative (Liu et al., 2019), MICCAI (Schnabel et al., 2019)). These challenges also evaluate software pipelines for processing different data types, like RGASP (Engström et al., 2013),

and have expanded in scope over time, such as the DREAM Challenges (Stolovitzky et al., 2007; Saez-Rodriguez et al., 2016). NeurIPS has also started hosting competitions covering a broad range of topics.

2.5 . Methodology and challenge design

After providing an overview of competitions, benchmarks, and their underlying motivations, we examine into the methodological considerations that must be addressed to design successful challenges. Several key aspects of ML challenge organization must be addressed to ensure a successful and engaging competition. Ensuring fairness and equal chances for all participants is important, and can be achieved by clearly defining the allowable techniques and resource limits. Objectivity is equally important, guaranteeing unbiased and impartial results through carefully crafted rules for data splitting, training and test sets, and final score calculations. Moreover, the reliability of competition results relies on reproducibility, which can be achieved by documenting data preprocessing, cleaning methods, and evaluation procedures. Transparency is essential, as it allows for easy verification and understanding of the results by others. This can be accomplished by explicitly stating how the results will be reported, communicating with participants, and making the competition's code and data accessible. Additionally, reducing the role of luck and randomness is a vital aspect of organizing a successful ML challenge. This can be achieved by designing evaluation methods and data splits that minimize the impact of chance and provide a more accurate reflection of the participants' skills and models performance.

Organizing crowd-sourced competitions requires expertise in science, technology, legal, and ethical aspects. Challenges typically arise from complex datasets, new scientific questions, or the need for objective algorithm evaluation (Green et al., 2015; Abdallah et al., 2015; Atassi et al., 2014; Bansal et al., 2014; Boutros et al., 2014). The process starts with defining an impactful scientific question that can be addressed in a competition setting, often involving domain experts. Data collection and processing must be thorough, with unpublished data used as a gold standard dataset for scoring submissions. As more practical concerns, providing a clear documentation and a "starting kit" with background information and example submissions is beneficial for reducing participation barriers. Incentives such as co-authoring papers, speaking invitations, monetary awards, and job offers can be used to attract participants. An effective communication campaign is necessary to attract participation and make a successful challenge. The challenge process includes an open phase for algorithm improvement and submission, followed by a final assessment phase to evaluate the best performers. Post-challenge activities include analyzing results, writing a paper, organizing conferences, and curating databases

for future use.

2.6 . Conclusion

In conclusion, scientific competitions have a rich history, and are getting more attention in recent years. As of 2022, hundreds of competitions are being organized annually, featuring a cumulative prize pool exceeding $5,000,000. This trend aligns with the evolution of the field of machine learning towards an experimental science. One of the main motivations behind crowdsourced benchmarks is the systematic comparison of a multitude of models. Given the reproducibility crisis that the machine learning field is facing, challenges offer an alternative way for conducting efficient and transparent research. Competitions and benchmarks represent one type of crowdsourcing. Other types include volunteer-based crowdsourcing, ideation challenges, human intelligent tasks, and coopetitions. For any competition to achieve its intended objectives, several methodological aspects must be addressed. This includes rigorous documentation, minimizing randomness in evaluation, and conducting high-level post-challenge analyses.

While theoretical mathematics can be studied with just a pencil and a piece of paper, or even the mind alone, experimental science requires tools. Crowdsourcing the resolution of complex machine learning tasks demands a tool that can gather many programs, inspect and evaluate them automatically, and returns a ranking. Ideally, such tool would be flexible, scalable, and able to manage a continuous flow of inputs and outputs. This is exactly what *CodaLab Competitions* is.

3 - An open-source web platform to host challenges

In this chapter, we present the platforms *CodaLab Competitions* (Pavao et al., 2023a) (Section 3.1) and *Codabench* (Xu et al., 2022) (Section 3.2), open-source web platforms, managed by the Laboratoire Interdisciplinaire des Science du Numérique (LISN) from Université Paris-Saclay. The former, *CodaLab Competitions*, is a competition platform. The latter *Codabench*, is a benchmark platform, also capable of hosting competitions.

We contributed to these projects in parallel and in conjunction with our PhD studies, as an engineer at LISN. Our contributions cover multiple aspects of platform development and enhancement. We managed the development process, conceptualized and implemented innovative features, and participated in the platforms' documentation and dissemination. The features we introduced include, among others, the ability to store data directly inside the compute workers, improved transparency between organizers and participants, and enhancement of the user interface and monitoring tools. Beyond these, we also refined the existing documentation, published the platforms as scientific papers, and created templates for competitions. We also organized various competitions. Furthermore, we provided general maintenance and support to address diverse queries and issues, ensuring good user experience and operational continuity.

In the following, we discuss the key features needed to organize rigorous challenges, and analyze the scientific impact of competitions. To that end, we provide a statistical analysis of the past challenges organized on *CodaLab Competitions* (Section 3.3) : it enumerates the quantity of competitions and participants, and the wide array of application fields, showcasing the platform's versatility in driving scientific progress. In order to study the utility of crowdsourcing and the difficulty of machine learning tasks, we propose statistical measures, described in appendix (Section 10.1).

3.1 . CodaLab Competitions

Given the relevance and potential impact of competitions, the availability of off-the-shelf platforms allowing anyone to create and run their own competition is a catalyzer for making rapid progress while ensuring inclusiveness and building community. In this context, we are involved in the development, administration, and use, as competition organizers and as participants, of our own competition platform, *CodaLab Competitions*.

Criteria	AICrowd	**CodaLab Competitions**	CrowdAnalytiX	EvalAI	Kaggle	RAMP	Tianchi
Code-sharing		✓		✓	✓	✓	✓
Code submission	✓	✓		✓	✓	✓	✓
Active community	☆☆	☆☆☆	☆	☆☆	☆☆☆	☆	☆☆☆
Custom metrics	✓	✓	✓	✓	✓	✓	?
Staged challenge	✓	✓		✓		✓	
Private evaluation		✓		✓			
Open-source	✓	✓		✓		✓	
Human evaluation				✓			
RL-friendly	✓	✓		✓			
Run for free		✓		✓	✓		?

Table 3.1 – Comparison of competition platforms, borrowed from Rousseau and Ustyuzhanin (2020). References for EvalAI : (Yadav et al., 2019), AICrowd : (Mohanty et al., 2016), CrowdAnalytiX (Mishra, 2012), Kaggle : (Goldbloom and Hamner, 2010), RAMP : (Kégl et al., 2018) and Tianchi : (Group, 2014). (Isdahl and Gundersen, 2019) have compare platforms more extensively. Other platforms are presented in (Neo, 2021).

CodaLab Competitions [1] is an open source platform for running data science competitions that has been used in hundreds of challenges associated to physics, machine learning, computer vision, natural language processing, among many other fields. *CodaLab Competitions* has unique features that make it an excellent open source option for organizing competitions. Table 3.1 shows a comparison of *CodaLab Competitions* with other popular platforms. The reader is referred to the project website where the code and complete documentation are found. The code is released under an Apache 2.0 License.

3.1.1 . Key concepts and features

The subsequent overview of the key concepts and features closely follows the paper Pavao et al. (2023a) published in the Open-Source software issue of the Journal of Machine Learning Research.

Competition bundle

A competition bundle is a ZIP file containing all necessary elements to create a competition : data, documentation, scoring program, and configuration settings. A new competition is created by uploading a bundle to the platform. The settings can be updated from the platform web interface via an editor. After editing, updated competition bundles can be re-downloaded from the platform for archival and sharing purposes. There is an option to download a light version of the bundle, without datasets and/or programs; the configuration file then directly points to database entries for such resources. This facilitates cloning competitions without having to re-upload data or code. The use of a competition bundle make the process of creating a competition both simple and highly flexible.

1. https://codalab.lisn.fr/

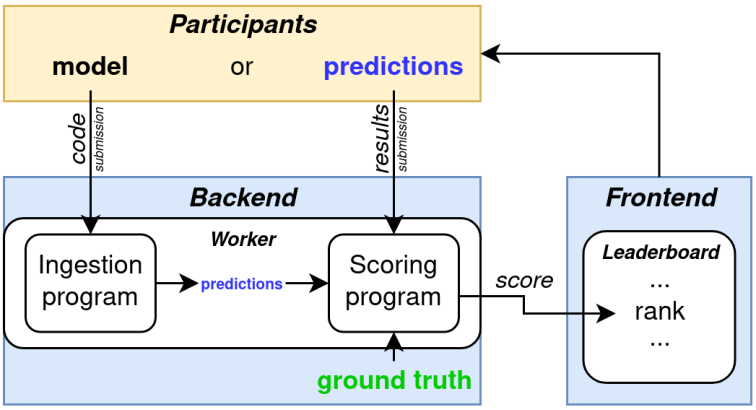


Figure 3.1 – General competition workflow.

Results and code submission

CodaLab Competitions supports the submission of models' predictions to be scored; and also supports code submission, allowing to train and test the methods in a dedicated environment on the server side. Competitions with only results submissions are easy to organize and usually need few computational resources. On the other hand, code submission has exhibited many advantages, such as providing potentially powerful machines to the candidates (shared in a fair way among them) and improving reproducibility. The logic of scoring of a competition (Figure 3.1) is coded by the organizers in any programming language (typically Python) : *CodaLab Competitions* requires at least a **scoring program** to score participants' submissions (e.g. by comparing results and ground truth), and optionally an **ingestion program** to execute a code submission in a controlled manner, with an organizer-defined API, e.g. calling a "fit" method and a "predict" method. Thus organizers have the flexibility of implementing any challenge protocol, with any data format or even data generating models. This versatility enables them to organize challenges in areas such as reinforcement learning, among others.

In determining whether to allow results (predictions) submissions or code submissions in competitions, organizers must weigh the pros and cons of each method. With the goal in mind of establishing a robust benchmark for a given problem, we advice for allowing only **code submission** for the reasons outlined below.

The primary advantage of code submissions is enhanced **reproducibility**. With results submission, you can only re-score based on the provided predictions and ground-truth. This is quite limited, in comparison to code submissions, enabling the re-execution of the entire pipeline : model training, predictions generation, and scoring. Having the actual code makes a huge difference. Without it, understanding participants' methods for generating their predictions becomes reliant on their explanations. Even with detailed descriptions,

there is an inherent reliance on trust. For instance, it is conceivable that submitted predictions could be manually produced by humans. For the integrity of experimental science, it is required to have access to the algorithms' code and the being able to re-run all experiments on the same environment.

Another significant advantage of code submissions is the possibility to conduct **complex evaluation procedures**. Code submission allows for methods like cross-validation, multiple trials, and testing the model under different conditions or on hidden datasets. In contrast, with results submissions, the possibilities are constrained, for instance to techniques like re-sampling (bootstrap) for calculating error bars.

It makes it easier to **control cheating**. By reviewing participant-submitted code, organizers can review the code and ensure that no techniques forbidden by the challenge rules are being utilized.

A related advantage of code submissions is that the **data are hidden** for real. When allowing result submissions, you necessarily give access to X_{test}, the test data given as input to the models to generate the predictions $\hat{y}_{test}$. This opens the door for them to employ semi-supervised learning techniques or even to manually analyze the data in ways that might not adhere to the spirit or rules of the competition. More generally, code submission is less prone to data manipulation, and even allows the use confidential data in the evaluation.

The final argument in favor of code submission is its inherent **equitability** for participants. It ensures a consistent environment and standardized computational resources for all candidates during the evaluation, and facilitates the monitoring of training and evaluation time. In contrast, with result submission, participants face fewer constraints. While unrestricted freedom can be beneficial, as it lets participants utilize all available resources to enhance their score, it may also introduce inequities based on available resources.

The main disadvantage of code submission is its demand on the organizers to supply potentially considerable computational resources. This intensifies with an increasing number of participants and the complexity of the task. Result submission, on the other hand, decentralizes computations across the devices of all participants. This trade-off is significant when considering the advantages of code submission mentioned above.

For all the reasons mentioned above, we consider code submission as the gold standard for scientific competitions. A common misconception about code submission is that it is tricky to setup in practice. Using *CodaLab Competitions*, it is actually extremely simple [2]. This discussion is summed up in the table 3.2.

2. https://github.com/codalab/codalab-competitions/wiki/Create-your-first-competition

Criteria	Result submission	Code submission
Information about the model		✓
Reproducible training		✓
Complex evaluation procedure		✓
Control cheating		✓
Hidden data		✓
Equal resources		✓
Light computation	✓	
Easy to put in place	✓	✓

Table 3.2 – Main advantages and disadvantages of result submissions and code submissions.

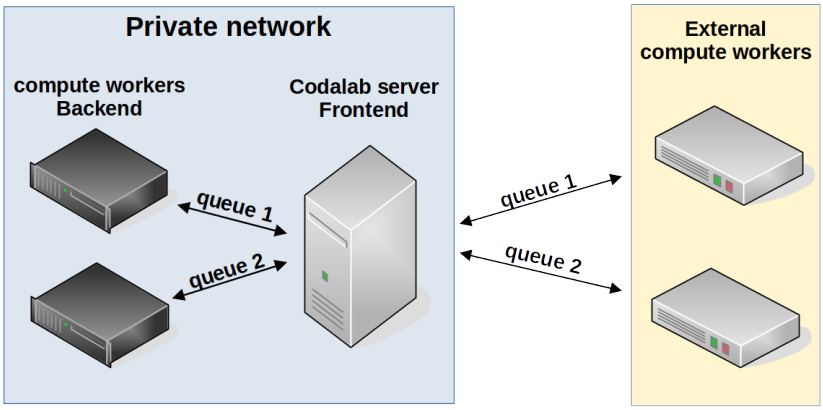


Figure 3.2 – Compute worker connections.

Compute workers (add external computational resources)

The public instance of *CodaLab Competitions* offers compute workers to process the submissions. Yet, for competitions with high computational demands, organizers can create custom queues and attach their own CPU or GPU compute workers, whether from physical hardware or virtual machines available on any cloud service (Figure 3.2). This modular architecture of *CodaLab Competitions* has been a key ingredient in its success and user base growth. This approach ensures that the institution hosting the main instance isn't burdened with all computational costs. The architecture of compute workers is shown in figure 3.3.

Another advantage of this feature is that algorithms can be trained and tested on confidential data securely, preventing any data leakage, by storing the data directly inside the compute workers. This is particularly useful for medical research, challenges organized by industries, and in other restricted domains. More details about the use of confidential data in competitions are given in Chapter 7.

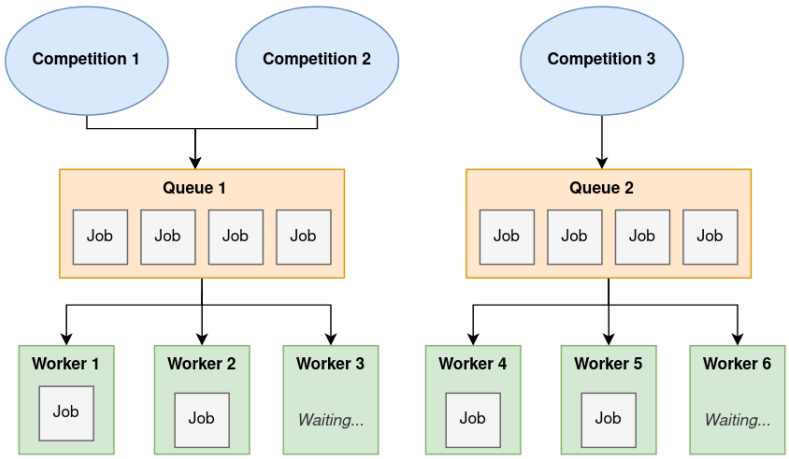


Figure 3.3 – Internal and external compute workers can be linked to CodaLab competitions. The queues dispatch the jobs between the compute workers. Note that a queue can receive jobs (submissions) from several competitions, and can send them to several compute workers.

Competition Docker environment

The execution of participants' code and scoring are performed inside Docker containers, preventing damage to the servers and allowing organizers to define a custom controlled environment for their competitions.

Docker (Merkel, 2014) is an open-source platform utilizing containerization to automate application deployment. It encapsulates applications and dependencies into isolated units called containers, enabling them to operate uniformly across different environments.

Organizers can configure a specific environment, selecting libraries and programming languages for participant submissions. This environment can be integrated into a competition effortlessly by using its corresponding Docker Hub name and tag [3]. This ensures that every participant's submission is tested in the exact same conditions. This eliminates variations due to different software versions, system configurations, or other environmental factors that could affect the results. Additionally, the use of Docker also contributes to the longevity and adaptability of the competition. Because Docker containers encapsulate everything needed to run the software, competitions do not become deprecated or obsolete due to changes in the underlying system environment. Furthermore, organizers can easily add new libraries or update the experimental environment in a transparent manner. This not only ensures consistency in the experimental setup over time but also facilitates replica-

3. Docker Hub is a cloud-based repository where users can create, store, and distribute containerized applications. It serves as a library for Docker images, similar to how GitHub functions for source code.

tion of the competition setup for future editions or by other researchers for related studies.

Phases

Competitions can be structured into multiple phases, each with its own settings, including distinct dates, data, and scoring programs. For instance, an initial *feedback phase* with minimal submission constraints can be followed by a *final test phase* using a fresh test set or entirely new datasets. This structure can be compared to "public" and "private" leaderboards seen on other platforms. The last code submitted in one phase can be automatically forwarded to the next phase. By segmenting the competition into phases, organizers can maintain participant engagement over extended duration, and design complex competition protocols.

Segmenting competitions into independent phases is a crucial methodological consideration, as it mitigates overfitting. A detailed discussion on this approach can be found in Chapter 6.

Multiple scores

The leaderboard is highly customizable, capable of handling multiple scoring functions simultaneously. This facilitates a comprehensive evaluation of models, as discussed in Chapter 4. Participants can be ranked based on the average rank they achieve across all sub-scores. The advantages of computing the average rank, along with an in-depth comparison of ranking functions, are provided in Chapter 5.

Documentation

The documentation is organized according to stakeholders categories *participants*, *organizers*, *administrators*, and *contributors* directly on the first page of the documentation.

- As a *participant*, you will find explanation on how to register, use the GUI to submit code or results. Mail communication with the organizers and a challenge forum are available. You will receive feedback on your submissions : scores, running time, plots, logs and predictions.
- As an *organizer*, you are accompanied with many competition templates, from simple to elaborate, to ease the technical aspects of building a competition, and to let you concentrate on scientific aspects of the competition. You can optionally propose a starting kit to the participants to lower the barrier of entry of beginners. You learn how to configure your competition using a YAML file, where you specify your logo, HTML pages, computer language, datasets organization, participant resource limitations, dates, and phase durations.
- As an *administrator* of your own instance of *CodaLab Competitions*, each piece of the infrastructure is configurable and offered as a docker com-

ponent. You can deploy your instance in a very flexible way concerning the sizing of your project thanks to deployment guide hints.
- As an occasional *contributor*, you can discuss with the main developers via the GitHub issues and suggest pull requests to solve some of the issues you have encountered. As a regular contributor, some project management rules are used to facilitate any external contribution.

3.1.2 . Technical aspects

CodaLab Competitions is implemented in Python's Django framework, one of the most extended web application framework in Python, with constant maintenance and security updates and a huge list of plugins. The system is divided in three main blocks : Front-end, API, and workers. Front-end is implemented using the default Django template system, providing all the views users interact with. For the API, the Django Rest Framework is used. Front-end and API constitute the core of *CodaLab Competitions*, and are in charge of managing all the data entities of the system, such as users, competitions, and submissions. For data management, a PostgreSQL database is used, pointing to MinIO for file storage. Submissions are evaluated in an asynchronous manner, using computation workers. This process follows a classical producer-consumer approach, using RabbitMQ as a queue manager, and a Celery client for message management. This approach is also used for periodic tasks happening on the core system, such as opening or closing competition phases or system checks. Another key technology used is Docker. Although *CodaLab Competitions* itself can be deployed as a Docker, the main advantage of containerization is related to the computation workers. All selected technologies are open source and can be deployed in a self-hosted manner and have equivalent services on most common cloud providers, to organize competitions at scale. Project management is organized in an agile way for bug correction and development of new features, leveraging GitHub issues, labeling process, feature-branch creation, and merge. We welcome new contributors.

3.2 . Codabench

The primary focus of this chapter is *CodaLab Competitions*, due to its widespread use and the large quantity of hosted challenges it offers to learn from. In addition, we are developing *Codabench* (Xu et al., 2022), a modernized version of *CodaLab Competitions* that emphasizes on benchmarks. *Codabench* was officially launched in August 2023, and already hosted several serious competitions and benchmarks, demonstrating a smooth user experience and reliable performance. The platform implements all key features from *CodaLab Competitions*, such as result and code submissions, modularity of computational resources, and containerized environments for reproducibility. Moreover, it

also has new useful features such as the transparency of evaluation methods and competition settings, real-time feedback to participants, and fact sheets forms.

One significant enhancement we've made to *Codabench* is improving **transparency** between challenge organizers and participants. Unlike the original *CodaLab Competitions*, where code availability (both ingestion and scoring programs) and input data were often shared via documentation, *Codabench* directly integrates the possibility to share these elements. As such, participants can download programs or data, knowing they're accessing the exact versions used in the back-end operations. Similarly, the platform transparently provides the Docker image tag used for submission execution. To further promote clear scientific communication, we've incorporated an option for linking competition reports or papers directly to challenges. Moreover, fact sheet forms are now embedded within the submission interface, allowing organizers to gather useful details for post-challenge analysis.

With *Codabench* emphasizing benchmarks, we've also introduced a novel paradigm of **dataset submissions**. Instead of the traditional approach of using static datasets and submitting algorithms, it is now possible to fix algorithms and submit datasets. This enables a systematic evaluation of model performance across varied tasks and datasets.

In future developments, organizers will have the option to provide a "meta-scoring program", which defines a specific ranking function. This function can be employed to aggregate scores across different tasks by comparing the participants together. An in-depth discussion on ranking functions can be found in Chapter 5. This discussion underscores the fact that no single ranking function universally outperforms the others. Therefore, giving organizers the possibility to supply their own code for ranking is beneficial.

3.3 . Scientific impact of past challenges

Competitions are nowadays a key component of academic events, as they comprise effective means for making rapid progress in specific topics. By posing a challenge to the academic community, competition organizers contribute to pushing the state of the art in specific subjects and/or to solve problems of practical importance. Challenges, in essence, provide a playground for verifying and reproducing experimental results.

We conducted a statistical analysis on 830 challenges hosted on *CodaLab Competitions*. Our selection criteria focused on "serious" challenges — only public competitions with a minimum of 5 participants and at least 10 submissions were considered, while excluding teaching materials. The data spread from January 2013 to August 2022. Additionally, we explore the concept of *crowdsourcing utility*, to identify the key factors that contribute to a challenge's success.

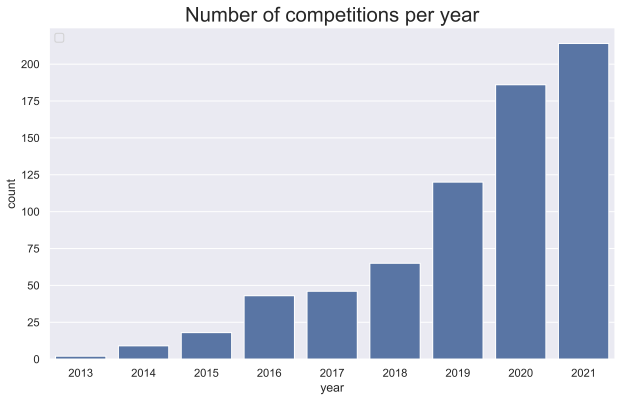


Figure 3.4 – Evolution of the number of competitions hosted on *CodaLab Competitions* each year.

In practice, we can distinguish two types of competitions : those associated to industry or aiming at solving a practical problem, and those that are associated to a research question. While *CodaLab Competitions*'s community is mainly focused on scientific challenges, both can be found on the platform. In this context, competitions have been organized in a number of fields like natural language processing (Harman, 1993), machine learning (Guyon et al., 2004) and knowledge discovery in databases [4]. Their spread and impact has considerably increased during the last decade. Indeed, the platform is growing and host numerous competitions each year, as shown in Figure 3.4. In 2021, more than 200 competitions were organized on *CodaLab Competitions*. A more selective count, proposed by Carlens (2023), which includes only high-impact competitions — those either accepted at major conferences or featuring substantial prize pools — suggests that 23 competitions were hosted on *CodaLab Competitions* in 2022. This ranking positions *CodaLab Competitions* third in terms of the number of competitions organized, after Tianchi and Kaggle. *CodaLab Competitions* has indeed established itself as a key player in this field, having a user base of over 120,000 registered members prior to 2023 [5].

The majority of challenges are computer vision (45%) or natural language processing (37.8%) tasks. Other machine learning domains of interest are signal processing (3.3%), automated machine learning (2.8%) and reinforcement learning (2.2%), as shown in Figure 3.5. Multi-domain tasks (e.g. image captioning) represents 8.9% of the competitions. The fields of applications are diverse, including not surprisingly multimedia and linguistics, due to the large proportion of computer vision and NLP competitions, and also medi-

4. https://www.kdd.org/kdd-cup/view/kdd-cup-1997
5. https://github.com/codalab/codalab-competitions/wiki/Statistics-of-the-public-servers

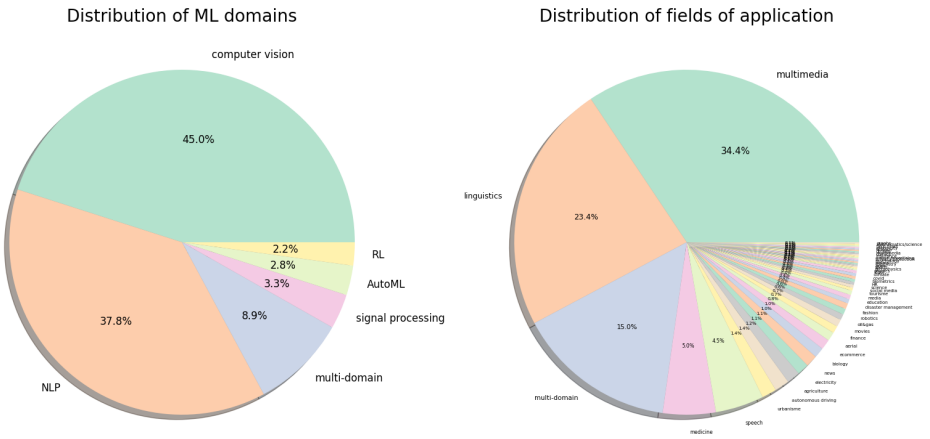


Figure 3.5 – Distribution of machine learning domains (left) and fields of applications (right) of competitions hosted on *CodaLab Competitions*. Competitions involving multiple domains or multiple fields were simply marked as "multi-domain".

cine, speech recognition, urbanism, autonomous driving, agriculture, electricity, news, biology, e-commerce, finance, robotics and more. 15% of the competitions are in the interface of multiple fields.

In our study, we found that only approximately 12% (102 out of 829) of the competitions offer prize money. This observation can be attributed to two key factors. First, *CodaLab Competitions* frequently serves as a platform for academic and research-based competitions, where it is common to conduct events without financial incentives for the winners. Second, some competitions may not have disclosed their prize pools through the interface, leading to the absence of this data in our study. The average prize amount for these competitions offering a reward is approximately $7,052$ USD, with the maximum prize reaching $40,000$ USD. It should be noted that all currency conversions to USD, when needed, were performed at the time of this study, and there may be slight imprecisions due to fluctuations in exchange rates.

Note surprisingly, the value of the prize pool impacts participation levels in a challenge. Figure 3.6 displays a scatter plot correlating the number of participants with the prize money for each competition. While the Pearson correlation coefficient of 0.29 suggests a positive relationship, it is not particularly strong. Some competitions with modest prize pools still attracted a considerable number of participants, indicating that other factors also contribute to high participation rates. Additionally, a large number of registered participants does not necessarily ensure high-quality solutions to the problem.

In section 3.1.1, we argue that code submission competitions are the gold standard of methodology of scientific competitions. In total, 16% of the compe-

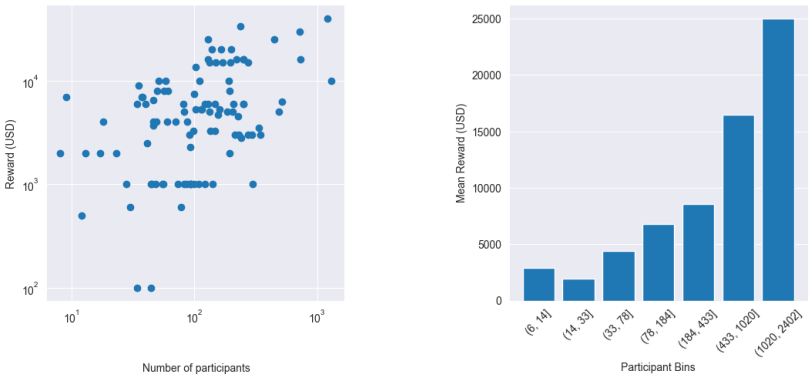


Figure 3.6 – Evolution of the number of participants in relation to the reward. In the scatter plot (left), each point represents competition, with both reward and participation displayed on a logarithmic scale. In the histogram on the right, competitions are grouped into bins with logarithmic scaling to emphasize the correlation between reward and participation.

titions are using code submission protocols. This statistics may be somehow disappointing, but the trend is clearly growing, as shown by the Figure 3.7. In some domains, almost all competitions are code submission (RL, AutoML), due to the requirements of their experimental protocols.

To analyze the success of the challenges, and the role played by crowdsourcing, we compute the *utility* score for each challenge. It is basically the ratio between the best score obtained s_{best} and the average score s_{mean} of the group of candidates $\mathcal{C}$, on the final phase of competition :

$$U_C = \frac{s_{best}}{s_{mean}}$$

More details about this measure and its interpretation are given in Section 10.1.

Across all competitions studied, the average U_C is around 1.73, meaning that, on average, the competition winner's score is 1.73 times better than the participants' average score. This statistics tend to show the usefulness of crowdsourced competitions. But how does it relate with other statistics, such as the number of submissions per participant, or the monetary reward of the competition ?

A Mean Decrease in Impurity (MDI) analysis, conducted using a random forest regressor, points that the most important features to predict utility are, in order, the number of participants, the reward, and the number of submissions per participant (Figure 3.9). The feature importance values add up to 1, serving as a percentage representation of each predictor's impact. The duration of challenge, and the use of code submissions are also predictors of the utility, however their predictive value are relatively low (around 11% and

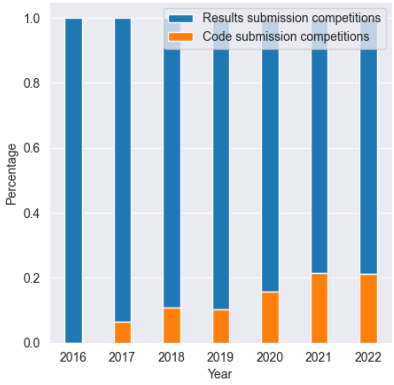


Figure 3.7 – Evolution of the proportion of code submission competitions. Years before 2016 are not shown because too few competitions were organized.

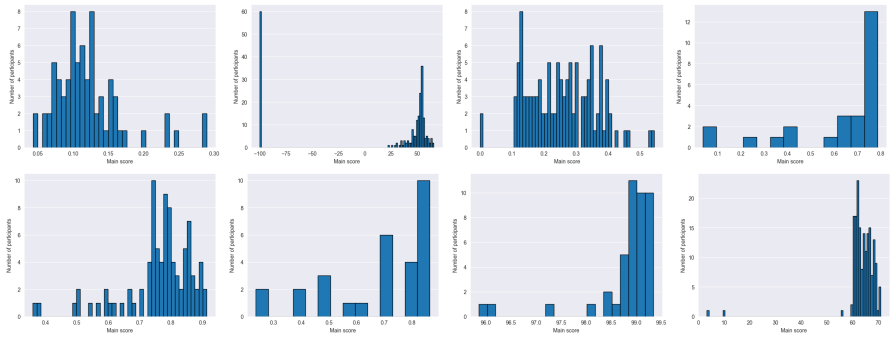


Figure 3.8 – Examples of histograms of score distribution in leaderboards. The score shown is the main score of the competition (the one used for the final ranking) and n is the number of participants in the bin of score. Phases with more than 25 participants randomly selected. Number of bins is $\frac{n}{2}$.

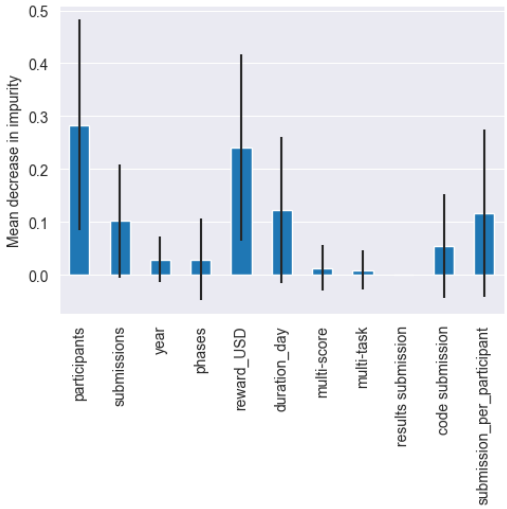


Figure 3.9 – Mean Decrease in Impurity (MDI) feature importance when predicting utility U_C with a Random Forest.

5%). This observation aligns with the phenomenon often seen in project deadlines, where participation tends to be concentrated towards the end, irrespective of the overall duration. While we advocate for the use of code submissions, this setting may not strongly influence the utility, as the utility is a purely performance-based metric, not taking into account the equitability and the reproducibility of the experiments. Note that the '*results submissions*' variable registers a zero importance, likely because competitions with code submissions are also considered as results competitions, rendering only the '*code submission*' variable informative. Interestingly, the year of the competition and the number of phases hold minimal importance. Overall, utility emerges as a complex, multi-factorial attribute that cannot be easily predicted by a single variable.

3.4 . Conclusion

Crowdsourced competitions and benchmarks require powerful tools to be used in practice and made efficient. During our PhD studies at LISN, we extensively contributed to the development of *CodaLab Competitions* and *Codabench* platforms, introducing innovative features, refining documentation, publishing scientific papers, organizing competitions, and ensuring overall maintenance and user support. These platforms are free, open-source, and offers many useful features to organize scientifically rigorous and high-impact competitions. For instance, they support the definition of Docker environments to score competition submissions, ensuring the critical aspect of reproducibility. They also enable code submissions, and the use of external computatio-

nal resources, to provide a solid ground for fair evaluations of complex and ambitious benchmarks. Moreover, *CodaLab Competitions* and *Codabench* facilitate staging competitions in multiple phases, a crucial element of competition design, leveraging overfitting and helping in selecting the most general solutions, as shown in Chapter 6. For challenge organizers, the platforms are highly flexible, as the scoring programs are completely customizable to meet any requirements.

These features, along with the fact that the platform is free and open-source, likely played a role in *CodaLab Competitions* success and wide adoption by the scientific community. Numerous competitions hosted on *CodaLab Competitions* were accepted by top conferences, and more than 800 competitions were organized on the platform since its creation. Its user base has been growing rapidly, with 50,000 users in 2020 and scaling up to 120,000 in 2022. We observed a wide diversity of research fields, from physics and biology to autonomous driving and finance. The vast majority of past competitions studied tasks pertaining to computer vision or natural language processing.

Our statistical analysis shed light on the dynamics of participant engagement. While the monetary rewards do influence participation rates, the correlation between the number of participants and the reward is counter-intuitively low, suggesting that other factors are important. When measuring the success of a competition — defined as the ratio between the highest and average scores among participants — our data underscores that the number of participants is the most important factor.

4 - Metrics and significance

Machine learning competitions, in many ways, resemble sport events. From the perspective of the organizers, much like in sports, there is a pursuit to rank participants fairly based on their skills, in a specific set of rules. However, one of the primary objectives in scientific competitions is not just to select a winner, but to address a particular problem or answer a scientific question. The emphasis lies in judging the participants equitably and, even more importantly, in judging the quality of their proposed solutions — the models — to make a significant progress in the problem-solving. Therefore, tasks must be designed and evaluated in a way that promotes meaningful improvements, rather than a way that can be exploited and tricked, intentionally or not.

Beyond the goal of generating significant, reproducible, and universal results, there is a legal aspect to consider. In many jurisdictions, gambling is strictly regulated. As competition organizers, it is essential to distinguish these contests from games of chance. Often, the competition rules explicitly state : "this is a skill-based contest in which chance plays no role". However, this isn't always entirely accurate. Indeed, the cash prize might be awarded to a winner whose score is not significantly distinct from other competitors. This means that random fluctuations in the test data can influence the selection of the winner, as illustrated by the Wheat Detection Challenge (David et al., 2020).

In the pursuit of designing a meaningful competition, the importance of the choice of the evaluation metric cannot be overstated. The evaluation metric is, obviously, the core of a challenge : it produces the value that participants aim to optimize. It was observed that the final ranking of methods is often highly sensitive to the metric choice (Caruana and Niculescu-Mizil, 2004). Nonetheless, seemingly intuitive choices of metrics can yield unintended outcomes. For instance, when an algorithm designed to maximize playtime in the game *Tetris* simply paused the game indefinitely, it technically maximized the chosen metric but deviated entirely from the intended behavior (Murph, 2013). Another example : computing accuracy in imbalanced binary classification can misleadingly reward a model that over-predicts the majority class. These examples underscore the importance of selecting metrics that align with the true objectives of the problem at hand. Given the vast variety of metrics in the literature, selecting the right one can be challenging; however, we outline the most commonly used metrics in various machine learning domains, discussing their strengths and limitations, with general overviews available in surveys (Raschka, 2018; Hernández-Orallo et al., 2012).

In this chapter, we discuss the choice of the scoring metric (Sections 4.1 to 4.4) and the statistical significance of the results (Section 4.5). This chapter

draws inspiration from my chapter titled “How to judge a competition : Fairly judging a competition or assessing benchmark results” in the book “AI Competitions and Benchmarks : The Science Behind the Contests” (Pavao et al., 2023b). Its primary aim is to present a comprehensive taxonomy of metrics, offer guidance on selecting the appropriate metrics, and provide insights into effectively judging competitions and ensuring meaningful evaluations. In the following, we distinguish between performance metrics (e.g. accuracy), ethical and societal impact metrics (e.g. measure of fairness), resources consumption metrics (e.g. time consumption) and evaluator-centric metrics (e.g. human evaluation). Note that many classification, regression and clustering metrics are implemented [1] in the famous machine learning Python package Scikit-Learn (Pedregosa et al., 2011). For clarity, the structure of this chapter is illustrated by the Figure 4.1.

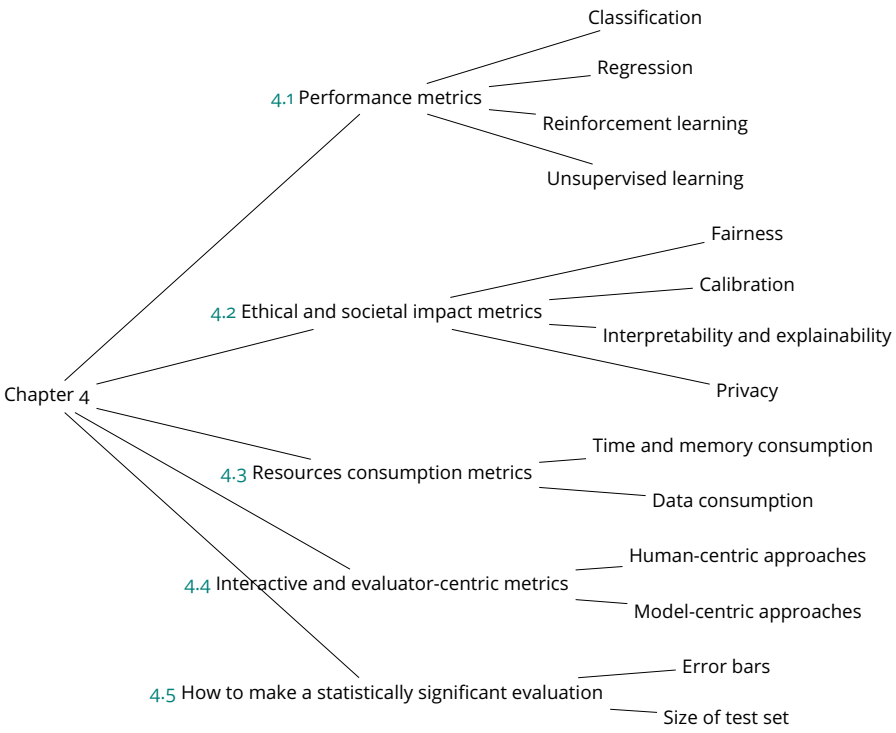


Figure 4.1 – Structure of Chapter 4.

4.1 . Performance metrics

In this section, we describe metrics commonly used as primary objective in classical machine learning problems : classification, regression, reinforcement learning and unsupervised learning.

1. https://scikit-learn.org/stable/modules/model_evaluation.html

4.1.1 . Classification

A prediction task is called a *classification problem* when the possible outcomes to predict are grouped in different classes (Grandini et al., 2020). The simplest setting involves only two classes (binary classification, reviewed by Berrar (2019); Canbek et al. (2017)) ; classification tasks involving more than two classes are called *multi-class classification*. For instance, the classical problem of handwritten digits recognition (LeCun et al., 1998) is a multi-class classification. In *multi-label classification*, each data point can be classified into several classes at the same time. The goal is to use available data called X to obtain the best prediction $\hat{Y}$ of the outcome variable Y. In multi-class classification, Y and $\hat{Y}$ can be seen as two discrete random variables that assume values in $\{1, \dots, K\}$ where each number represents a different class, and K is the number of distinct classes.

When classification algorithms output the probability that a sample from X belongs to a given class; a classification rule is then employed to assign a single class. In binary classification, a threshold can be used to decide the predicted class. In the multi-class case, there are various possibilities, the most employed technique being selecting the highest probability value, commonly computed using the softmax function (Grandini et al., 2020).

To give a general overview of the resemblance between classification metrics, we conducted an experiment to empirically evaluate the correlation between common classification scoring metrics : we computed rankings of the final models from AutoDL Challenge (presented in Section 8), independently on all 66 datasets formatted for this competitions, and compared these rankings using Euclidean distance. The results are presented graphically in Figure 4.2, scaled in a two-dimensional plot. *Jaccard score* and *F1-score* are confirmed to be very close [2]. The *SAR* metric (Caruana and Niculescu-Mizil, 2004), an average between *accuracy*, *AUC* and *root mean squared error* (using $1 - RMSE$), is well centralized, as it was designed to be. Interestingly, *balanced accuracy* seems to be centralized between *accuracy* and *AUC*. Loss functions, such as *mean absolute error* (MAE) and *mean squared error* (MSE), are clearly distinct from classification metrics.

Selecting the right scoring metric for a classification task is important as it directly impacts the evaluation and interpretation of the model's performance. The main points we identified as relevant to guide the selection process are the type of problem, the class balance and the real world objectives.

Problem type : Consider the type of problem. Most classification metrics are defined as binary classification metrics; however, they can be used to score multi-class problems, by breaking the problem down into multiple binary problems. The problems can be broken down using either *One-vs-One*

2. https://stats.stackexchange.com/questions/511395/are-jaccard-score-and-f1-score-monotonically-related/512378#512378

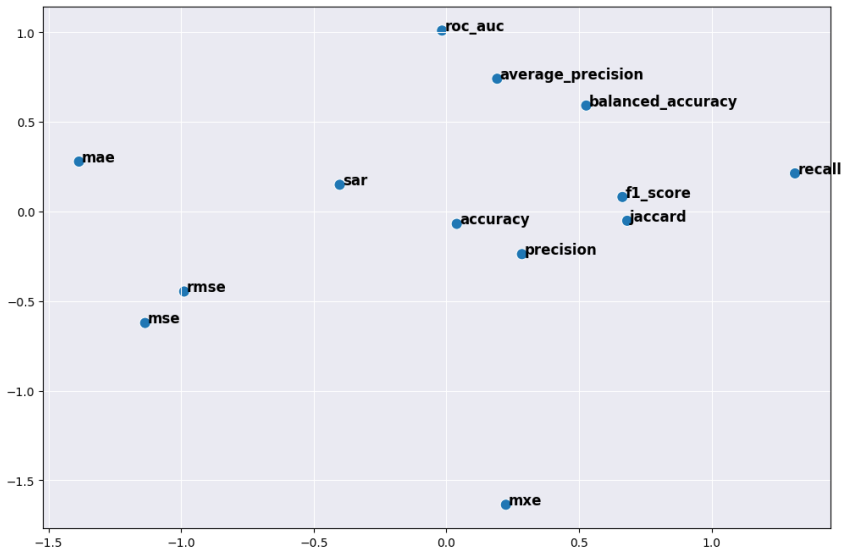


Figure 4.2 – Multidimensional scale (MDS) plot illustrating the degree of correlation between scoring metrics in a 2D space. The metrics are compared by computing Euclidean distance between the rankings they produced. This experiment was performed on the classification tasks and models from AutoDL Challenge (presented in Section 8.1).

(OVO) or *One-vs-Rest* (OVR) approaches. In the OVO approach, the pairwise score of all pairs of classes is computed. In the OVR approach, the scores for each class is computed separately, treating each class as the positive class and all other classes as the negative class. In both cases, the problem is broken down into a series of binary problems, and the final score is obtained by averaging all the scores, either using a simple average, or a weighted average. The OVO approach is computationally expensive, as the number of scores to compute is $\frac{K(K-1)}{2}$ with K being the number of different classes. For this reason, the OVR approach is mostly used in practice, needing only K computations, one for each class. We consider only the OVR approach for the rest of this section. Beyond its computational benefits, the OVR approach is favored due to its simplicity, making it more interpretable for non-experts. It scales linearly as the number of classes grows, contrasting with the exponential complexity of the OVO method. Furthermore, its prominence in modern machine learning tools, given the optimized implementations in popular frameworks, emphasizes its relevance for real-world applications.

The simplest idea when it comes to score classifiers is to use *rates of success*. *Precision*, *recall* and *accuracy* are metrics that simply count the successes and failures of the classifier. Intuitively, *accuracy* is the likelihood that a randomly chosen sample will be correctly classified by the model. The fundamental component of this metric is the individual units in the dataset, each

of which holds equal weight and contributes equally to the score. However, when considering classes instead of individuals, some classes may have a high number of units while others have only a few. In such cases, the larger classes will carry more weight compared to the smaller ones. When the dataset is imbalanced, meaning that most units belong to one particular class, *accuracy* may overlook significant classification errors for classes with fewer units as they are less significant compared to the larger classes.

Class imbalance : Consider the distribution of classes in the dataset. If there is a class imbalance, using accuracy as a metric might not provide an accurate picture of the model's performance. In such cases, metrics like *balanced accuracy* or *area under ROC curve* (AUC) are more appropriate. *Balanced accuracy* addresses this issue by giving each class equal impact on the score. This is simply done using a weighted average of each class accuracy, weighted by the proportion of the class in the test set. Despite having fewer units, smaller classes may have a disproportionately larger influence on the formula. When the dataset is relatively balanced, meaning the class sizes are similar, *accuracy* and *balanced accuracy* tend to produce similar results. The main distinction between the two metrics becomes apparent when the dataset exhibits an imbalanced distribution of classes.

Real world objective : Consider the real world impact of the problem and the cost of false positive and false negatives. For instance, in medical diagnosis, a false positive (e.g. a healthy person is diagnosed with a disease) can lead to unnecessary medical procedures and treatments, causing harm to the patient. On the other hand, a false negative (e.g. a sick person is not diagnosed) can result in a delay in treatment and a potentially fatal outcome. In this case, *recall* (the ability of the model to identify all positive cases) is a relevant metric. Another example : in the finance sector, false positives can lead to substantial revenue loss from incorrect trades or transactions, while false negatives might result in missed opportunities. In such contexts, *precision* (the model's capability to correctly identify positive instances) and *F1 score* (a harmonized metric between precision and recall) could be considered better choices for assessing the model's efficiency.

4.1.2 . Regression

The prediction task is called a regression problem when the outcome is a continuous numeric value, in contrast to a classification problem where the variable to predict falls into discrete categories. For instance, predicting temperatures or road traffic from support variables are regression tasks. A review of metrics for regression is given by (Botchkarev, 2018). Another survey concerns evaluation methods for time series forecasting (Cerqueira et al., 2020).

The main points we identified as relevant to guide the selection process are the type of problem, the scale of the target variable and the real world

objectives.

Problem type : Consider the type of problem you are trying to solve. For example, for time-series forecasting problems, metrics like *mean absolute error* (MAE), *mean squared error* (MSE), and *root mean squared error* (RMSE) are relevant. For problems involving predicting counts, metrics like *mean absolute percentage error* (MAPE) and *symmetric mean absolute percentage error* (SMAPE) might be more appropriate.

Scale of the target variable : Consider the scale of the target variable. If the target variable is large, the absolute difference between the actual and predicted values will also be large, making *MAE*, *MSE* and *RMSE* less interpretable. In such cases *R-squared* (coefficient of determination) might be more appropriate metrics. *R-squared* has no units, can be compared among different tasks, and has an intuitive interpretation.

Real world objective : Once again, consider the real world problems. For instance, if the problem involves predicting stock prices, the magnitude of the error is more important than the direction of the error. In such cases, *RMSE* or *MSE* might be appropriate metrics. Another example : in the field of climate modeling, the right choice of scoring metric vary depending on the problem. If the goal is to predict global temperatures, metrics like *mean absolute error* (MAE) could be used to evaluate the performance of the model. However, in predicting regional precipitation patterns, metrics like *R-squared* or *explained variance score* might be used to evaluate the ability of the model to capture the complex spatial patterns of precipitation.

4.1.3 . Reinforcement learning

Reinforcement learning (RL) consists, for an autonomous agent (e.g. robot), in learning what actions to take, based on experiences, in order to optimize a quantitative reward over time. The agent is immersed in an environment and makes its decisions based on its current state. In return, the environment provides the agent with a reward, which can be either positive or negative. The agent seeks, through iterated experiments, an optimal strategy or *policy*, which is a function that associates the current state with the action to be performed, to maximize the sum of rewards over time. This setting makes the scoring procedure particularly problem-specific and prone to design flaws.

In reinforcement learning, overfitting occurs when the agent becomes too specialized to the conditions of the training environment and is unable to generalize to unseen situations. This is a common problem in RL because the training and the evaluation processes are usually conducted on the same environment, rather than in two separate environments. This creates a situation where the agent can memorize the optimal actions in the training environment without truly understanding the underlying dynamics. As a result, the

agent's performance may appear to be much better than it actually is, leading to a biased evaluation. This can be a serious issue in RL, as it undermines the validity of the evaluation process and may result in the selection of suboptimal algorithms or policies. To mitigate this problem, it is important to separate the training and testing environments and use different metrics and simulators for evaluation. This helps to unbias the evaluation process and to ensure that the results accurately reflect the true performance of the agent.

Some research studying the evaluation of RL algorithms show that behavioral metrics play a crucial role in determining the quality of a state representation, and in learning an optimal representation (Jordan et al., 2020; Lan et al., 2021). Existing methods, such as *approximate abstractions* and *equivalence relations*, aiming at reducing the size of the state or action space by aggregating similar states, are not effective for continuous-state reinforcement learning problems, due to their inability to maintain the continuity of common RL functions and their tendency to generate overly detailed representations that lack generalization. A behavioral metric in reinforcement learning is a measure of an agent's performance in an environment, based on its actions and observed rewards. It can be used to evaluate and compare different reinforcement learning algorithms or policies. Examples include *average reward per episode*, *success rate*, and *convergence speed*.

According to Henderson et al. (2018), multiple trials with different random seeds are necessary to compare performance, due to high variance. To ensure reproducibility, it is crucial to report all hyperparameters, implementation details, experimental setup, and evaluation methods for both baseline comparisons and novel work.

4.1.4 . Unsupervised learning

Numerous tasks in machine learning lack a definitive ground truth for evaluating solutions. Termed as unsupervised learning, this category includes a diverse range of tasks, from clustering and dimensionality reduction to data modeling, generation, and feature extraction. It also covers areas with a more subjective nature, like automatic music composition, identifying molecules that bind to COVID-19, and text summarization.

It is particularly challenging to design competitions for such unsupervised learning tasks, due to the following reasons : the lack of ground truth, the diversity of solutions, and the subjectivity in evaluation. Indeed, unlike supervised learning, unsupervised learning often lacks clear ground truth, making it difficult to evaluate the results objectively. Also, unsupervised learning models can often produce diverse solutions that are equally valid, making it challenging to select a single best solution. Finally, the evaluation of unsupervised learning solutions can be subjective as it depends on the understanding and interpretation of the evaluators.

In the case of distribution modelling and clustering, some clear performance metrics can be identified. Prior research investigate the evaluation metrics for unsupervised learning (Palacio-Niño and Berzal, 2019) and clustering (Ben-Hur et al., 2002; von Luxburg et al., 2012). Typically, when the goal in unsupervised learning is to **learn the underlying data distribution**, various loss functions can be employed depending on the specific type of model :

- *Maximum Likelihood Estimation* (MLE) : In probabilistic models, the goal is often to maximize the likelihood of the observed data.
- *Kullback-Leibler (KL) Divergence* (Kullback and Leibler, 1951) : Measures the difference between two probability distributions. It is often used with Variational Autoencoders (VAEs) (Kingma and Welling, 2019) where one tries to minimize the divergence between the learned distribution and the true data distribution.
- *Reconstruction Loss* : In models like autoencoders (Liou et al., 2014), the objective is to reconstruct the input data from a compressed representation. The loss measures the difference between the original input and its reconstruction.
- *Wasserstein Distance* (Earth Mover's Distance) (Villani, 2009) : Used in Wasserstein Generative Adversarial Networks (WGANs) (Arjovsky et al., 2017) to measure the distance between the generated distribution and the true data distribution.

To assess the performance of **clustering** methods, when no ground truth is available, the following metrics and techniques can be used :

- *Inertia* (Sum of Squared Errors) : Inertia measures the sum of squared distances between each data point and its closest cluster center. Lower inertia values indicate tighter clusters and better performance.
- *Silhouette Score* (Rousseeuw, 1987) : This metric computes the average silhouette value for all data points, measuring how similar a data point is to its own cluster compared to other clusters. A silhouette score close to 1 indicates a well-partitioned dataset, whereas a score close to -1 indicates poor clustering.
- *Davies-Bouldin Index* (Davies and Bouldin, 1979) : This index measures the ratio of within-cluster scatter to between-cluster separation. Lower values of Davies-Bouldin Index indicate better clustering performance.
- *Calinski-Harabasz Index* (Caliński and Harabasz, 1974) : This index evaluates clustering by comparing the ratio of between-cluster dispersion to within-cluster dispersion. Higher values of the Calinski-Harabasz Index suggest better clustering performance.

Although these metrics are valuable for tasks like distribution learning and clustering, they fall short in assessing the performance of machine learning models in areas like text summarization or artistic creation. In these scenarios, it is challenging to score the success. Interesting and effective methods

include human evaluation, employing other machine learning models as evaluators, and using interactive adversarial frameworks.

Indeed, **human evaluation** can play a crucial role in assessing the performance of unsupervised learning algorithms when traditional quantitative metrics may not fully capture the desired outcomes, or when ground truth labels are not available. Human evaluation can provide valuable qualitative insights and help ensure that the resulting patterns or structures discovered by the algorithms align with human intuition and domain knowledge. To perform human evaluation effectively, it is essential to establish clear guidelines for the evaluators, provide proper training, and, when possible, recruit multiple evaluators to increase the reliability of the assessment. Human evaluation can be time-consuming and resource-intensive, so it is often used in combination with quantitative metrics to balance the efficiency and quality of evaluation. This approach is further discussed in Section 4.4.1.

Another interesting technique, that may become more popular in the future, is to use a **model used as a metric**. For instance, a classifier trained on discriminating between two distributions (e.g. fake and real) can be used to evaluate the performance of generative models. This can be compared to the functioning of Generative Adversarial Networks (GAN) (Goodfellow et al., 2014), where the output of a binary classifier is used to guide the learning of a generative model. When using another machine learning model as a metric for unsupervised learning, it is important to remember that the evaluation depends on the performance of the supervised model and the quality of the labeled data. Therefore, the results should be interpreted with caution, as the evaluation might be biased or limited by the chosen supervised model or the available data. This approach is discussed in Section 4.4.2.

Finally, adversarial challenges, where the solutions proposed by the participants are then used as input data in the next phase, is potentially a good way of organizing challenges on unsupervised learning tasks. Adversarial challenges design are explored in details in Section 7.7.

More generally, to overcome these challenges and organize unsupervised learning competitions and benchmarks, it is essential to articulate the problem and describe the data comprehensively. Evaluation methods must be defined considering the missing ground truth. Promoting collaboration among participants can diversify the solutions. Involving domain experts in the evaluation can enhance the result objectivity. Keeping participants informed about competition progress and being open to adjustments can optimize the process. Lastly, providing evaluations using diverse metrics can help participants gauge the pros and cons of their methods. By following these steps, competitions can be designed to effectively evaluate the solutions to an unsupervised learning task while overcoming the challenges of lack of ground truth, diversity of solutions, and subjectivity in evaluation.

4.2 . Ethical and societal impact metrics

Traditional evaluation metrics like *accuracy*, *precision*, and *recall* have long held the spotlight, but there is so much more to consider when measuring a model's real-world impact. In this section, we dive into the lesser-known, unconventional metrics that are reshaping the way we assess machine learning models. From fairness and privacy to interpretability and calibration, these innovative evaluation techniques change how we think about model performance and pave the way for a more responsible, holistic approach to machine learning. Indeed, the performance metrics we have reviewed in this chapter so far reflect only one aspect of the performance of the models : their predictive abilities. Yet, in many applications, our concerns extend to other aspects, like the trustworthiness of the algorithms. This is particularly true in sensitive applications, where an algorithmic decision could mean life or death.

4.2.1 . Fairness

Even if the mathematical definition of machine learning models does not necessarily contains unfair or biased elements, trained models can be unfair, depending on the quality of their input data or their training procedure. A model trained on biased data may not only lead to unfair and inaccurate predictions, but also significantly disadvantage certain subgroups, and lead to unfairness. In other words, the notion of fairness of models describes the fact that models can behave differently on some subgroups of the data. The issue is especially significant when it pertains to demographic groups, typically defined by factors such as gender, age, ethnicity, or religious beliefs. As machine learning is increasingly applied in society, this problem is getting more attention and research, and is subject to debate (Benz et al., 2020; Vasileva, 2020; Chouldechova and Roth, 2018; Chen et al., 2018; Boratto et al., 2021). Some interesting ways to quantify fairness include :

Demographic Parity : This measure checks if the positive classification rate is equal across different demographic groups. The formula is as follows :

$$Demographic\,Parity\ :\ P(\hat{Y}=1|A=0)=P(\hat{Y}=1|A=1)$$

where A is a protected attribute (such as race or gender), Y is the target variable (such as approval or denial) and where $\hat{Y}$ is the predicted value of Y. Demographic parity is a condition to be achieved : the predictions should be statistically independent of the protected attributes.

Statistical Parity Difference : Related to the demographic parity, it measures the difference between positive classification rate across different demographic groups. The formula is :

$$StatisticalParityDifference = P(\hat{Y} = 1|A = 0) - P(\hat{Y} = 1|A = 1)$$

Disparate impact : It calculates the ratio of the positive classification rate for a protected group to the positive classification rate for another group. It is similar to the *statistical parity difference*, but it is a ratio instead of a difference :

$$DisparateImpact = \frac{P(\hat{Y} = 1|A = 0)}{P(\hat{Y} = 1|A = 1)}$$

A value of 1 indicates that the positive classification rate is the same for both groups, suggesting fairness. A value greater than 1 indicates a higher positive classification rate for the group with $A = 0$, while a value less than 1 suggests a higher positive classification rate for the group with $A = 1$. However, it is important to note that disparate impact is a limited measure of fairness and should not be used on its own. There may be cases where a higher positive classification rate for one group is justifiable, for example if the group is underrepresented in the training data. Additionally, disparate impact does not consider other factors such as false positive and false negative rates, which may provide a more comprehensive view of fairness.

Equal Opportunity : This metric checks if the true positive rate is equal across different demographic groups. The formula is :

$$EqualOpportunity : P(\hat{Y} = 1|Y = 1,\ A = 0) = P(\hat{Y} = 1|Y = 1,\ A = 1)$$

As for *demographic parity*, it is a condition to be achieved.

Equal Opportunity Difference : This metric measures if the true positive rate is equal across different demographic groups. The formula is :

$$EqualOpportunityDifference = P(\hat{Y} = 1|Y = 1,\ A = 0) - P(\hat{Y} = 1|Y = 1,\ A = 1)$$

The same idea can be applied to false positive rates.

These are just a few of the metrics that can be used to quantify fairness in classification tasks. It is important to note that fairness is a complex issue, and these metrics should not be used in isolation. Instead, they should be considered in the context of the specific problem and the desired outcome.

4.2.2 . Calibration

Classifiers usually return probabilities to indicate the confidence levels across different classes. However, the question arises whether these confidence levels accurately reflect the classifier's actual performance. As defined

by Naeini et al. (2015); Guo et al. (2017), the notion of miscalibration represents the difference in expectation between the confidence level (or probability) returned by the algorithm, and the actual performance obtained. In other words, calibration measurement answers the following question : is the confidence of the algorithm about its own predictions correct ? Promoting well calibrated models is important in potentially dangerous decision making problems, such as disease detection or mushroom identification. The importance of calibration measurement lies in the fact that it is essential to have a clear understanding of the confidence level that the algorithm has in its own predictions. A well-calibrated algorithm will produce confidence levels that accurately reflect the likelihood of a prediction being correct. In contrast, a miscalibrated algorithm will either over or under estimate its confidence in its predictions, leading to incorrect or unreliable outcomes. In applications where the consequences of incorrect decisions can be severe, it is of utmost importance to have a well-calibrated algorithm. Misclassification of a disease can lead to incorrect medical treatment and harm to the patient. Similarly, misidentification of a mushroom can result in serious health consequences. In these scenarios, well-calibrated models can help ensure that the decisions are made based on reliable predictions.

The calibration can be estimated using the **Expected Calibration Error** (ECE) : this score measures the difference between the average predicted probability and the accuracy (i.e., the proportion of positive samples) in bins of predicted probability. The formula for the ECE is given by :

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} |acc_m - conf_m|$$

where M is the number of bins, B_m is the set of samples in the m^{th} bin, n is the total number of samples in the test data, acc_m is the accuracy of the m^{th} bin, and $conf_m$ is the average predicted probability in the m-th bin.

When computing the calibration, we derive the performance prediction directly from the model's output. Another interesting possibility is to ask the participants to provide an estimation of the generalization score of their method. Indeed, we can make a connection between the calibration and the prediction of generalization error, more commonly estimated by a separated method. The Performance Prediction Challenge (Guyon et al., 2006a) focused on this problem. Competition protocols for the performance prediction problem are further discussed in Chapter 8.

4.2.3 . Interpretability and explainability

Given the complexities of machine learning models, the assessment of their interpretability and explainability emerges as a considerable challenge. While both concepts are crucial for ensuring trust and understanding in model

predictions, especially in critical applications, measuring them accurately is difficult, especially when models vary widely in their structures and underlying mechanisms.

Interpretability and explainability are related but distinct concepts in machine learning.

Interpretability refers to the degree to which a human can understand the cause of a model's predictions. It refers to the ability to understand the internal workings of the model and how it arrived at its decisions.

Explainability refers to the ability to provide a human-understandable explanation of the model's decision making process. It is concerned with the presentation of the reasons behind the predictions to humans in a understandable form, e.g. through feature importance.

In summary, interpretability focuses on the transparency of the model itself, while explainability focuses on the communication of the model's behavior to a human audience. A wide survey on interpretability is proposed by Carvalho et al. (2019). They stressed out how interpretability is greatly valuable in one hand, but hard to define in the other hand. Another way to explain algorithms, automatically, is the sensitivity analysis (Iooss et al., 2022). Sensitivity analysis is a technique used to determine how changes in input variables of a model or system affect the output or outcomes of interest.

Past competitions have been exploring the development and evaluation of explainable models, such as the *Job Candidate Screening Challenge* (Escalante et al., 2017, 2018). It is a challenge of first impressions and apparent personality analysis, on audio-visual data. The candidate models have to predict apparent traits of people [3] (e.g. friendly or reserved, imaginative or practical) from short videos, with a focus on the explanatory power of techniques : solutions have to "explain" why a given decision was made. To this end, participants had to provide a textual description that explains the decision (i.e. the prediction) made. Optionally, participants could also submit a visual description to enrich and improve clarity and explainability. Performance was evaluated in terms of the creativity of participants and the explanatory effectiveness of the descriptions. For this evaluation, a set of experts in the fields of psychological behavior analysis, recruitment, machine learning and computer vision was invited. We can note that, this way, the explainability component of the challenge requires qualitative evaluations and hence human effort.

It is worth noting that some models are interpretable by nature, such as logistic regression or decision tree. Some researchers make the point that there is a trade-off between interpretability and models' performance, especially for complex tasks, that seem to be requiring blackbox models – huge deep learning neural networks. However, Rudin (2019) argues that this "accuracy-interpretability trade-off" is an unfounded myth.

3. The data was labeled by around 2500 annotators.

4.2.4 . Privacy

Privacy must typically be measured when the candidate algorithms are generative models, modelling a distribution of potentially confidential data. The goal in such a case is to use the generative models in order to create artificial data that reassemble sufficiently the real data to be used in actual applications, but not too much for private information to be leaked. A metric to estimate this trade-off is the **adversarial accuracy**, that we introduced in Yale et al. (2019b). Here is its definition :

$$AA_{TS} = \frac{1}{2}\left(\frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\left(d_{TS}(i) > d_{TT}(i)\right) + \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}\left(d_{ST}(i) > d_{SS}(i)\right)\right)$$

where the indicator function $\mathbf{1}$ takes the value 1 if its argument is true and 0 otherwise, T and S are true and synthetic data respectively. d is an arbitrary chosen distance function, such as the Euclidean distance. $d_{TS}(i)$ represents the distance between the i^{th} point of T and its closest neighbor from S. $d_{TT}(i)$ is the distance between this i^{th} point and its closest neighbor in T. Subsequently, $d_{ST}(i)$ and $d_{SS}(i)$ compare the i^{th} point of S to its closest neighbors in T and in S.

It is basically the accuracy of a 1-nearest-neighbor classifier, but the ideal score is not 1 (perfect classification accuracy) but 0.5. Indeed, a perfect score means that each generated data point has its closest neighbor in the real data, which means that the two distributions are overly similar. A score of 0 would mean that the two distributions are too different, thus the practical utility is low. Hence, a 0.5 score, where the closest neighbor of each data point can either be fake or real with the same probability, is what guarantees a good privacy. These principles are illustrated with a toy example in Figure 4.3. Alaa et al. (2022) proposes a similar approach.

One limitation of this method is that a proper measure of distance is needed. This is also a strength because it means that the method is general and can be applied in different fields, by selecting an adequate distance measure.

In the study of privacy, *differential privacy* and *membership inference attacks* are core concepts.

Differential privacy provides a robust framework to ensure that a trained model does not get substantially influenced by the inclusion or exclusion of a single data sample from the dataset. It employs a parameter ϵ to quantify the privacy, with smaller ϵ values meaning more privacy guarantees. It relates to the metric we proposed, which estimates the proximity between generated data and training data. While our approach evaluate the privacy preservation at the scale of the dataset, differential privacy focuses on individual-level privacy.

Inference attacks represent methods by which attackers deduce sensitive

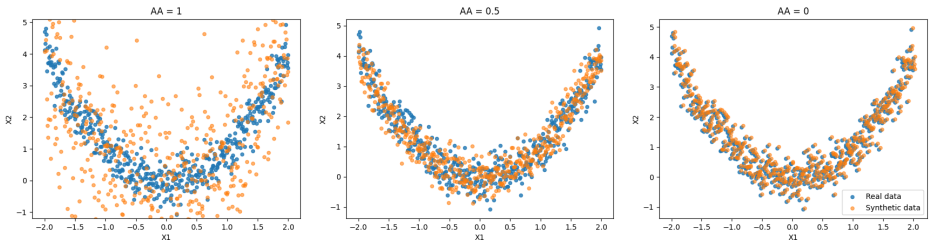


Figure 4.3 – Adversarial Accuracy (AA) is the performance of an nearest neighbor classifier that distinguishes between real data vs synthetic data. The ideal value is $AA = 0.5$ (represented by the plot in the middle). This is a bi-variate ($X1$ and $X2$) illustrative example.

information using the models' output (or predictions). The adversarial accuracy metric intends, in some way, to measure the sensitivity of the model to these inference attacks. If generated data were nearly identical to training data, adversaries might be able to get sensitive information. By ensuring a degree of distance between original and generated data, the resistance against such attacks is improved.

Additionally, privacy concerns are not limited to synthetic data or generative models. Predictors too can be the target of privacy attacks, as highlighted by Pedersen et al. (2022).

4.3 . Resources consumption metrics

Resource consumption metrics quantify the energy, time, and data utilized by models, thereby providing insights into their efficiency and sustainability.

4.3.1 . Time and memory consumption

It is useful to include the consumption of time, memory and energy of ML models in their evaluation and comparison. There are two main approaches to take these into account : **limit the resources** and **track the use of resources**. Both may imply in practice the use of code submission, as opposed to results submission. In code submission competitions, the participants submit their models which then get trained and tested on the servers, while in results submission competitions, the participants work locally and upload their predictions to the platform. Code submission is therefore advised if limiting or tracking the resource usage is part of the competition design.

The **training and inference time**, the **size of the model**, the **memory used** during the process or even the **energy consumption** are variables that can be limited by design or measured and shown on the leaderboard. Obviously, using the same hardware and evaluation conditions for all partici-

pants is needed in order to have a fair evaluation. The number of lines of code, or the number of characters, can also be used as an indicator of the **simplicity and practicability** of the solution. However, obviously, this indicator can be easily tricked by calling external packages and may need a manual review. The simplest models that solve the task are preferable, for being less harmful for the environment, less costly, deployable in weaker devices and easier to interpret.

A model that can produce the same results in less time is more desirable, as it reduces the computational resources required and can lead to cost savings. This is especially important in light of the current ecological crisis, as reducing energy consumption in computing can have a significant impact on reducing the carbon footprint of technology. Additionally, models that are faster to train and make predictions are more scalable and can be deployed in real-time applications, further enhancing their utility. Thus, optimizing time consumption is a key factor in the development of efficient and environmentally sustainable machine learning models.

Anytime learning

Anytime learning refers to a learning paradigm in which a machine learning model or algorithm incrementally improves its performance as it receives more data or training time. The key aspect of anytime learning is that the model can produce meaningful results at any point during the learning process, with its performance generally improving as it acquires more data or spends more time on training. Anytime learning algorithms are particularly valuable in situations where resources, such as time or computation power, are limited, or when it is essential to provide real-time or near-real-time insights. These algorithms can be employed in various machine learning settings, including classification, regression, and reinforcement learning tasks.

To evaluate the score in this framework, one can compute the Area under the Learning Curve (ALC) :

$$ALC = \int_{t_0}^{t_f} s(t)dt$$

where $s(t)$ is the performance score (obtained form a metric depending on the task) at timestamp t. t_0 and t_f refers to the first and last timestamps, and should be fixed to allow a fair evaluation and comparison of the scores.

The time can be linear, or transformed at any scale :

$$ALC = \int_{t_0}^{t_f} s(t)d\tilde{t}(t)$$

where $\tilde{t}$ is the transform function. For instance, a logarithmic scale transform can be used, in order to give more importance to the first steps of training :

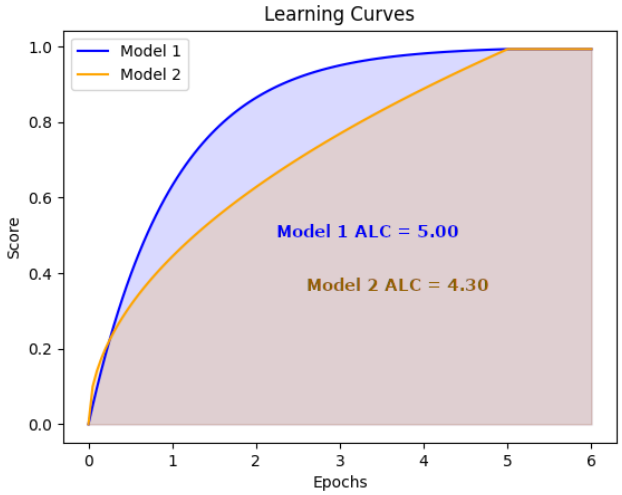


Figure 4.4 – Example of learning curves for two models. While both *Model 1* and *Model 2* converge to the same score, *Model 1* boasts a larger area under the learning curve (ALC). Thus, in an anytime learning context, *Model 1* is the preferred choice.

$$\tilde{t}(t) = \frac{\log(1+t)}{\log(1+t_f)}$$

The metric is depicted in Figure 4.4. Although both *Model 1* and *Model 2* converge to the same score, *Model 1* is favored in an anytime learning context due to its rapid performance improvement across epochs. This advantage is evident from the larger area under its learning curve.

Any-time learning can be linked to multi-fidelity. Multi-fidelity methods in machine learning refer to strategies that use various "fidelities" or qualities of data or models to speed up the learning process. For instance, one might use a simpler, faster-to-evaluate but less accurate model (low fidelity) to guide the learning process in the early stages, and a more complex, accurate but computationally intensive model (high fidelity) later on. The idea is to use less expensive resources to get an initial idea, and then refine it with more accurate but costlier methods. One could combine anytime learning and multi-fidelity methods to create a learning process that is both time-efficient and increasingly accurate. For example, in the early stages of an anytime learning algorithm, one might employ low-fidelity models or data to quickly get a "good enough" model. As time allows, the algorithm could then switch to using high-fidelity models or data to refine its understanding further. This way, one gets the best of both worlds : a quickly usable model, while aiming for high accuracy given more time. This combination could be particularly useful in scenarios where computational resources or time are constrained but where the model quality

also needs to be maximized, such as real-time analytics, robotics, or complex simulations.

4.3.2 . Data consumption

In machine learning, data is a key resource, and examining its consumption is relevant. The data input of models can be divided into two dimensions : the number of samples, and the number of features.

Number of training samples

The amount of training data required is an interesting metric to consider when comparing different models. As the saying goes, "data is the new oil", but not every situation allows for the luxury of vast datasets. In many real-world applications, gathering sufficient labeled data can be time-consuming, expensive, or even impossible. Tracking and limiting data consumption is, therefore, an essential aspect of model evaluation.

Monitoring data consumption can help identify algorithms that perform well with limited data, making them more suitable for scenarios with small datasets. On the other hand, constraining the quantity of available training data can encourage the development of models that are more efficient in learning. This is typically called few-shots learning, where meta-learning techniques, such as the k-shot n-way approach, are used. In this method, models are trained to quickly adapt to new tasks using only a limited number of examples k from each class n. The k-shot n-way design, used in MetaDL competition, is described in Section 7.3. By intentionally limiting data consumption, meta-learning promotes the development of models capable of generalizing better from smaller datasets, ultimately enhancing their utility and adaptability in diverse situations.

Feature selection

Feature selection is the process of selecting a subset of relevant features, or variables, for use in model construction. Even if recent trends in deep learning tend to use the raw data without further preparation, feature selection is an essential aspect of machine learning that aims at selecting the most informative features for a given model. Proper feature selection can lead to simpler, more interpretable, and faster-performing models that may also have improved generalization.

Two primary methods exist to assess feature selection : **evaluation metrics** and **intrinsic metrics**. Evaluation metrics, also called the *wrapper approach*, assess a model's performance after feature selection, based on the assumption that effective model performances signify well-chosen features. The specific metrics used vary depending on the nature of the problem, such as classification or regression, as elaborated previously in this chapter. On the other hand, intrinsic metrics, also called the *filter approach*, evaluate the

inherent quality or relevance of features without necessarily training a model. They act as heuristics of the fundamental information contained within each feature. Intrinsic metrics evaluate the significance of individual features without necessitating a full model. These include the *correlation coefficient*, which gauges the linear relationship between a feature and the target ; *mutual information*, indicating the relevance of a feature based on how much it informs about another variable ; *feature importance* from tree-based models such as decision trees and random forests ; and *variance threshold*, where low-variance features, assumed less informative, might be discarded. While Kohavi and John (1997) advocates for the wrapper approach (evaluating trained algorithms), Tsamardinos and Aliferis (2003) argue that neither approach is inherently better, and that both the learner and the evaluation metric should be considered. Hybrid approaches, such as maximizing a performance score while minimizing the number of features, are efficient in balancing between model accuracy and model simplicity.

In the NIPS Feature Selection Challenge (Guyon et al., 2004, 2006b), participants were ranked on the test set results using a score combining *Balanced Error Rate* (BER), the fraction of features selected F_{feat}, and the fraction of probes found in the feature set selected F_{probe}. The aim of feature selection is often to reduce the feature set's size without a significant loss in predictive performance. Hence, a lower F_{feat} could be seen as favorable if the model's performance remains strong. "Probes" in the context of this challenge refer to "dummy" or "non-informative" features that were intentionally added to the dataset. These features don't have any relation or correlation with the target variable and are essentially noise. Thus, a good feature selection algorithm should ideally avoid selecting these probes, minimizing F_{probes}.

Briefly : they used the McNemar test (McNemar, 1947) to determine whether classifier A is better than classifier B according to the BER with 5% risk yielding to a score of 1 (better), 0 (don't know) or -1 (worse). Ties (zero score) were broken with F_{feat} (if the relative difference in F_{feat} was larger than 5%). Remaining ties were broken with F_{probe}. The overall score for each of the five datasets was the sum of the pairwise comparison scores.

These methods of feature selection, from intrinsic metrics to challenge-specific criteria such as those in the NIPS Feature Selection Challenge, play a role in optimizing machine learning models simplicity, performance and interpretability, and reducing their data consumption.

4.4 . Interactive and evaluator-centric metrics

Evaluator-centric metrics represent a paradigm in machine learning evaluation where the benchmarking process actively involves specific evaluators, be they humans or models. Within this category, there is a distinction :

human-centric approaches primarily leverage human judgment and perspectives, while model-centric approaches utilize predefined algorithms or models for evaluation.

4.4.1 . Human-centric approaches

In addition to the quantitative evaluation metrics discussed previously, it is essential to consider more "human" evaluation techniques when assessing machine learning models. These approaches place emphasis on qualitative aspects and subjective interpretation, bringing a human touch to the evaluation process. For instance, in the case of text-to-image algorithms, manual inspection of generated images can help determine whether the outcomes are visually appealing, coherent, and contextually relevant. More generally, models that produce art, such as automatic music generators, benefit from manual evaluation. AI art competitions typically involve human evaluation through voting, such as the *Deep Art* competition of NeurIPS 2017 [4]. IEEE CEC also hosts regular art competitions [5]. Similarly, large language models can be subjected to psychological or behavioral tests, where human evaluators rate the model's responses based on factors such as coherence, empathy, and ethical considerations. Such human-centric evaluation methods can reveal insights that purely numerical metrics might overlook, providing a more nuanced understanding of a model's strengths and weaknesses. It is important to distinguish between human evaluation and the comparison to human performance. While both are human-centric approaches, the latter specifically uses human abilities as a baseline for performance comparisons. As highlighted in Chapter 2, a clear example of this approach is how the Generative Pre-trained Transformers (GPT) (OpenAI, 2023), the famous large language models, was tested using psychology tests (Uludag and Tong, 2023; Li et al., 2022), high-school tests (de Winter, 2023) and mathematics tests (Frieder et al., 2023). By integrating these human-oriented techniques into our evaluation toolbox, we can ensure that our machine learning models are not only effective in solving problems but also resonate with the multifaceted nature of human experiences.

4.4.2 . Model-centric approaches

A model can be used as a metric to evaluate the performance of other models, offering a dynamic and specialized approach to scoring complex tasks. This approach can be called model-centric, as opposed to human-centric approaches discussed in the previous section.

Typically, discriminative models can be used to assess the performance of generative models. Examples of this were given with the use of k-nearest neighbor adversarial accuracy to compute privacy (Section 4.2.4), and the use

4. https://nips.cc/Conferences/2017
5. https://sites.google.com/view/ieeecec2021ecmac/

of a classifier to score an image generation task (Section 4.1.4). More generally, in this case, the discriminant is trained to distinguish between real data from the target distribution and artificially generated data, thereby judging how "realistic" the generated data appears to be. This approach is similar to the learning framework of generative adversarial networks (Goodfellow et al., 2014). However, as underscored by the metric of adversarial accuracy for privacy, a generative model that completely deceives the discriminative model – in the sense that the discriminant gets an extremely low score – is an indication of privacy leakage of the training data. In other words, if the generative model performs exceptionally well within this adversarial framework, it raises concerns about its ability to generate general and original data. To avoid this, an "originality" or "privacy" metric should be invoked to measure the similarity, as mentioned in the Section 4.2.4. One distinct advantage of employing a discriminative model for performance measurement is its ability to output numerical values. Consequently, this form of performance assessment can easily be incorporated into a ranking system or any quantitative evaluation framework. An illustration of this protocol can be found in the Dog Image Generation challenge (Kan et al., 2019).

Another, more qualitative, way of measuring performance using models emerges through the use of language models. Large Language Models (LLMs) can serve as evaluators on various Natural Language Processing (NLP) tasks. For instance, in text summarization, an LLM can be employed to measure the semantic coherence and relevance of generated summaries by comparing them with the original text. The LLM could produce a likelihood score or even generate textual critiques to indicate how well the summary captures the essence of the source material. When it comes to explainability, an LLM can analyze the output explanations of complex models to assess their clarity and coherence. Naturally, as with any model-based metric, the initial prerequisite is to have confidence in the reliability of the evaluating model. More broadly, LLMs can act as judges for smaller models in a variety of NLP tasks. They can evaluate the quality of machine-generated translations, assess the sentiment consistency in chatbot dialogues, or even measure the relevance of answers generated by a question-answering system. These ideas were explored by the innovative competition *Auto-Survey Challenge 2023* [6] (Khuong and Rachmat, 2023). In this challenge, the participants propose AI agents capable of composing scientific survey papers and reviewing them. Such AI agents thus operate either as authors or reviewers. API calls to chatGPT were used to output the scores of *conclusion* (how well the conclusion highlights the main findings in the text) and *contribution* (relevance of the paper). To bridge between qualitative and quantitative outputs, the organizers asked the LLM to provide a number in Likert scale (Likert, 1932), for better differentiation bet-

6. https://www.codabench.org/competitions/1145/

ween good and bad results (for instance, *1 - Strongly Disagree*, *2 - Disagree*, *3 - Neutral*, *4 - Agree*, *5 - Strongly Agree*). They also made clear and complete prompts, detailing how the characteristics should be evaluated.

Using a model as a metric offers many advantages and may even be essential for certain applications, but it also comes with its notable drawbacks and challenges. One of the immediate concerns is the additional layer of complexity and computational cost involved in using one model to evaluate another, which becomes particularly problematic when computational resources or time are limited. This issue is closely followed by questions regarding the reliability and consistency of the metric model itself. If the model employed as a metric has inherent weaknesses or biases, these could be transferred to the evaluation of the models being evaluated. The method's potential unreliability, complexity, and sensitivity to data and hyperparameters can result in difficult interpretations and risk misleading evaluations, especially in critical fields like healthcare and legal decision-making. This vulnerability introduces the risk of "circular reasoning", particularly when the metric model is trained on similar data or shares architectural components with the model being evaluated, potentially leading to overly optimistic results. Typically, the organizers of the *Auto-Survey Challenge 2023* reported that chatGPT was usually overly optimistic and tended to grade its own work better than actual human work.

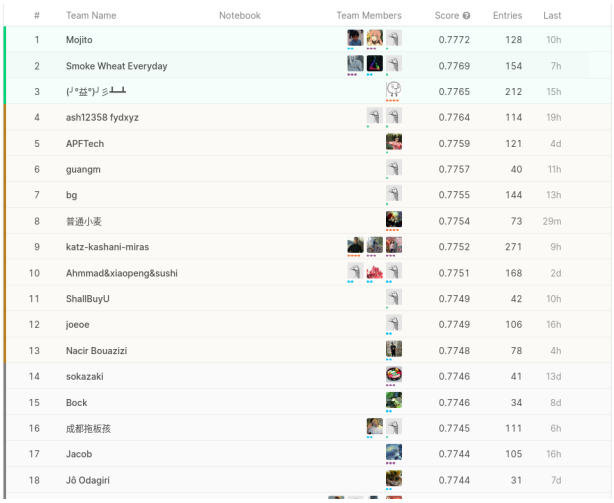
Despite these various challenges, using models as metrics can provide nuanced and context-specific insights that are hard to capture with traditional evaluation methods. This approach should be applied with caution and rigorous methodology to ensure the most reliable and informative results.

4.5 . How to make a statistically significant evaluation

We stressed out that the selection of appropriate evaluation metrics is critical. Equally critical is ensuring that the evaluation is statistically significant. For a robust evaluation, a sufficiently large test set is essential, along with the computation of score error bars. The figure 4.5 shows an example of a tight leaderboard from a past competition. In this particular competition, the third place candidate, qualified for a prize, only has a 0.0001 difference in score with the fourth place candidate. This thin margin between the third and fourth place highlights potential concerns regarding the evaluation methodology. This section presents methods for computing error bars, and addresses questions such as the ideal size of the test set. The goal is to provide methods to minimize the influence of randomness in competitions.

4.5.1 . Error bars

Error bars are the representation of the uncertainty of a measurement, allowing to distinguish score estimations between candidate models. There



#	Team Name	Notebook	Team Members	Score	Entries	Last
1	Mojito			0.7772	128	10h
2	Smoke Wheat Everyday			0.7769	154	7h
3	(ﾉ°益°)ﾉ彡┻━┻			0.7765	212	15h
4	ash12358 fydxyz			0.7764	114	19h
5	APFTech			0.7759	121	4d
6	guangm			0.7757	40	11h
7	bg			0.7755	144	13h
8	普通小麦			0.7754	73	29m
9	katz-kashani-miras			0.7752	271	9h
10	Ahmmad&xiaopeng&sushi			0.7751	168	2d
11	ShallBuyU			0.7749	42	10h
12	joeoe			0.7749	106	16h
13	Nacir Bouazizi			0.7748	78	4h
14	sokazaki			0.7746	41	13d
15	Bock			0.7746	34	8d
16	成都拖板孩			0.7745	111	6h
17	Jacob			0.7744	105	16h
18	Jô Odagiri			0.7744	31	7d

Figure 4.5 – The (very tight) leaderboard from the Global Wheat Detection challenge (David et al., 2020) on Kaggle (Goldbloom and Hamner, 2010). Even if the scores are close, only the top-3 candidates share the $15,000 cash prize.

are three common types of error bars : *standard deviation* (STD), *standard error of the mean* (SEM) and *confidence interval* (CI) (Krzywinski and Altman, 2013).

The **standard deviation** consists in the average distance between each sample and the mean :

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}}$$

The use of $n-1$ in the denominator when computing the sample standard deviation is a result of what's called Bessel's correction (Bishop, 2006; Murphy, 2012). The main reason for using $n-1$ instead of n is to provide an unbiased estimate of the population variance and standard deviation when computed from a sample.

The **standard error of the mean**, if the n observations are statistically independents, is the standard deviation divided by the square root of the sample size :

$$SEM = \frac{\sigma}{\sqrt{n}}$$

The **confidence interval** is calculated by using the standard deviation to create a range of values likely – to a given probability (commonly 95%) – to contain the true population mean. This technique requires the computation of approximations in practice, and is not commonly used to analyze competitions and benchmarks in machine learning.

Given this context, what are the sources of variability in our area of interest? Specifically, in supervised learning, performance estimation often involves comparing model predictions with a test set's ground truth. Here, **the**

variability comes from the model in one hand, and the data in the other hand.

The model can be stochastic during three different processes : the initialization, the learning process and the prediction process. Note that each of these processes is not necessarily stochastic in nature, and many models are completely deterministic. Even models that involve random initialization, such as neural networks, can be made deterministic by fixing a random seed [7]. This raises a question : should we impose to participants to fix a seed in order to reduce the variability of their methods ? The main benefit of fixing the random seeds being improving the reproducibility of the methods. In general, having high variation of the results due to initialization can be a sign of low generalization capabilities. However, fixing the seed is controversial as it overlooks variability factors. In previous contests, we executed participants' code multiple times, choosing their poorest performance to motivate variance reduction. Although averaging multiple runs decreases variance, organizers shouldn't do this as it may favor high-variance methods ; participants should ensure their methods have low variance.

The data is inherently stochastic, originating from real-world observations that can be considered as samples from unknown distributions. This randomness is compounded by variations in labeling quality, splitting into training and test sets, and other factors. One effective ways to mitigate these sources of variance is through high quantities of data. While it is often stated that the quality of a machine learning model is largely determined by the quantity of available data, it is at least safe to say that abundant data enhances the reliability of model evaluations, as will be explored in subsequent sections.

The authors of Bouthillier et al. (2021) shows that most evaluations in deep learning focus on the impact of random weight initialization, which is only a small source of variance, comparable to residual fluctuations from hyperparameter optimization. However, this variance is much lower compared to the variance caused by splitting the data into training and test sets.

Statistical hypothesis testing

Statistical hypothesis tests are used to decide whether the data at hand sufficiently support a particular hypothesis. In our area of interest, hypotheses often involve comparisons, such as whether algorithm A outperforms algorithm B or if the performance of various algorithms aligns with that of the baseline method. We mostly make multiple comparisons of multiple algorithms, multiple comparisons between two algorithms, or comparison between algorithms to a control (the baseline). The tests used for different scena-

7. A random seed is a number used to initialize a pseudo-random number generator, which is then used to generates the weights. A fixed random seed means that the number generator will always returns the same values.

rios are detailed in Japkowicz and Shah (2011). Even if the classical null hypothesis statistical tests (NHSTs) are widely used, recent research advocate for the use of Bayesian analysis instead (Benavoli et al., 2017). The authors present the Bayesian correlated t-test, the Bayesian signed rank test and a Bayesian hierarchical model that can be used for comparing the performance of classifiers, arguing that it solves the drawbacks of the frequentist tests. One of the drawbacks underlined is the fact that NHST computes the probability of getting the observed (or a larger) difference between classifiers if the null hypothesis of equivalence was true, which is not the probability of one classifier being more accurate than another, given the observed empirical results. Another common problem is that the common usage of NHST relies on the wrong assumptions that the p-value is a reasonable proxy for the probability of the null hypothesis (Demăr, 2008). Other areas of science are also moving from NHSTs to Bayseian approaches, as evidenced by the journal *Basic and Applied Social Psychology*, which in 2015 banned the use of NHSTs and related statistical procedures (Trafimow and Marks, 2015).

In practice, in competitions, the statistical testing boils down to ranking participants and declaring ties. In the following, we examine the use of bootstrap and cross-validation as estimators of the variance in models performance.

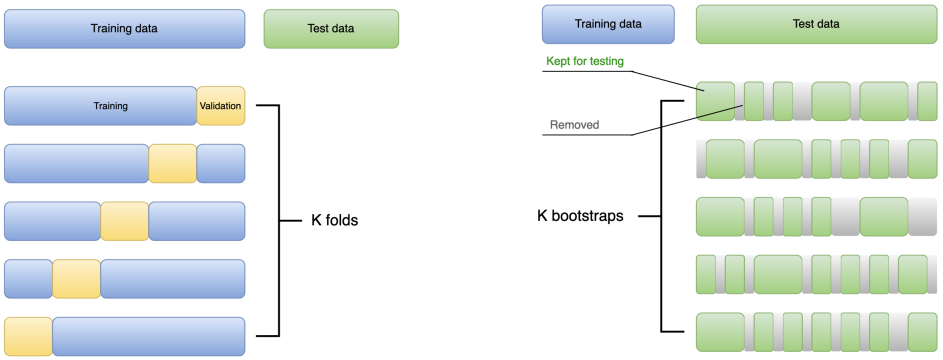
Bootstrap and cross-validation

In the field of machine learning, *cross-validation* and *bootstrapping* are widely used ways of computing the bias and variance of models performances. These two methods are inherently different, as cross-validation involves retraining the model from scratch several times, while bootstrapping only uses re-shuffling of the test samples and predictions, making it quicker to compute. Validation methods help to prevent overfitting, a common issue in machine learning where a model performs well on the training data but poorly on new unseen data.

The **K-fold cross-validation** (CV) (Hastie et al., 2009) involves dividing the original data into several parts (folds), where one part is used for testing and the rest for training. This process is repeated multiple times, each time with a different part used for testing, and the performance metrics are averaged across all iterations to get a final evaluation of the model.

Bootstrapping (Efron and Tibshirani, 1993a) involves generating multiple subsets of the original data set, using sampling with replacement, each having the same size as the original set. The algorithm is then evaluated on each subset (known as a "bootstrap sample"). The performance metrics are averaged across all bootstrap samples to get a more robust evaluation of the algorithm.

K-fold CV and bootstrapping are illustrated and compared in the Figure 4.6. In both cases, the variance can be computed on the set of scores obtai-



(a) K-fold cross-validation (b) Bootstrapping

Figure 4.6 – Schema of K-fold cross-validation (left) and bootstrapping (right). The cross-validation implies training on subsets of data with size determined by K. Bootstrapping involves sampling with replacement the test data, thus implying duplicated and missing samples in each evaluation. There is no limit to the number of re-sampling (bootstraps) that can be performed.

ned. Note that, the bootstraps and the folds not being independents, we can't compute the standard error (dividing by the square root of the number of scores), as mentioned in Section 4.5.1.

The question of determining the best methods for estimating a model's generalization error received substantial attention in the literature, as evidenced by numerous studies (Nadeau and Bengio, 2003; Bengio and Grandvalet, 2004b; Markatou et al., 2005; Kohavi, 1995a; Zhang and Yang, 2015a; Dietterich, 1998; Tsamardinos et al., 2018; Esbensen and Geladi, 2010; Molinaro et al., 2005; Langford, 2005b,a; Forman and Scholz, 2010). This problem is deep and the suitability of an estimator appears to depend both on the evaluated models and the data they are evaluated on. Some empirical evaluations of generalization error estimators have been conducted (Kohavi, 1995b; Zhang and Yang, 2015b), and advise for a 10-fold CV. Top participants of the Performance Prediction Challenge (Guyon et al., 2006a) used various cross-validation techniques to minimize average guess errors. The top performer employed virtual leave-one-out (VLOO) cross-validation for kernel classifiers, optimizing loss function through intensive cross-validation and using fresh data splits for re-estimation. While many preferred standard 10-fold cross-validation for hyper-parameter tuning, others experimented with methods like bagging with bootstrap re-sampling or using challenge validation sets for predictions. Bengio and Grandvalet (2004a) demonstrate that, when dealing with simple cases, neglecting the dependencies between test errors can result in a bias that is roughly equal to the variance. These experiments highlight that one must exercise caution when interpreting the significance of differences in cross-validation

scores.

Bootstrapping is more practical due to its computational efficiency and can be applied directly to results, eliminating the need to access the underlying algorithms (e.g. when evaluating results submissions). Therefore, bootstrapping is highly valuable in the context of competitions and benchmarks, as it enables variance calculations without the need for computationally intensive operation.

Within group and between group variance

It is common to have multiple levels of granularity when computing scores and error bars. The highest granularity level is the level of data points, or samples. Samples can be grouped in lower granularity levels, such as tasks or datasets. Indeed, each task has a unique test set, leading to a distribution of scores for each algorithm on each task. This can also be defined as *within group variance*, the variance between the samples of a dataset, and *between group variance*, the variance between the tasks.

In this situation, the variance can be studied by invoking the law of total variance. The law of total variance (Weiss, 2005), also known as Eve's law, decomposes the variance of a random variable into two parts : the expected value of the variances conditioned on another random variable, and the variance of the expected values conditioned on that same random variable. Formally, let X and Y be two random variables. The law of total variance states :

$$\text{Var}(X) = \mathbb{E}[\text{Var}(X|Y)] + \text{Var}(\mathbb{E}[X|Y])$$

It naturally resonates with the concept of multi-granularity variance in the context of machine learning and algorithm evaluation. The overall variance in the performance of the algorithms (without considering tasks) is represented by $\text{Var}(X)$. The expected variance within each task (given the task) is similar to $\mathbb{E}[\text{Var}(X|Y)]$, where Y denotes the specific task. The variance in the average performance of the algorithms across tasks is represented by $\text{Var}(\mathbb{E}[X|Y])$. It dissects the total variance of algorithm performance into parts : one due to inherent variability within each task, and another due to the variability in the algorithm's relative performance across different tasks. This view allows researchers and practitioners to understand the robustness of an algorithm across tasks and the variability of performance within specific tasks.

To dive more in-depth into this multi-level granularity scenario, we conducted experiments on the models' predictions from the AutoML (Guyon et al., 2019a) and AutoDL (Liu et al., 2021a) Challenges benchmarks (presented in Chapter 8). For each model, we estimated the within group and between group variance and compared the results. Our empirical experiments suggest that the highest granularity level exhibits lower variance of scores. Figure 4.7 shows the standard deviation of the ranks obtained by each participant of the

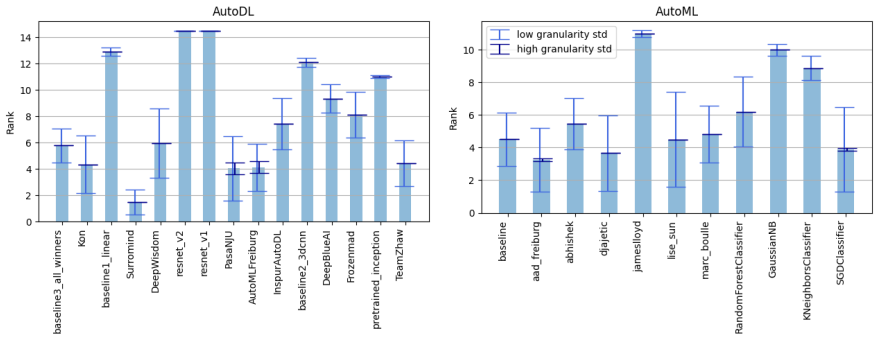


Figure 4.7 – Average and standard deviation of ranks of AutoDL (left) and AutoML (right) candidates. The deviation is computed on bootstraps of data samples (high granularity) and bootstraps of datasets (low granularity).

AutoML and AutoDL challenges. In these challenges, the candidates are evaluated on a set of datasets, with a test set for each dataset. We can therefore compute the standard deviation of the ranks of each candidate, by varying the samples (high granularity) or the datasets (low granularity). This computation was performed using bootstraps, and highlights the difference in deviation depending on the granularity.

4.5.2 . Size of test set

A crucial consideration is determining the test set size that provides a reliable error rate estimation. This should be the number one worry of the organizer : having enough test data to enable a robust judgement of the candidates. In the case of classification, Guyon et al. (1998b) suggest as a rule of thumb to use $n = \frac{100}{p}$, where n is the test set size and p is the error rate of the best classifier, as estimated, for instance, by the human error rate. Hence, the better your classifier, the bigger your test set needs to be in order for you to compute precisely the error rates. In the case of imbalanced classes, one can base this analysis on the size of the smallest group, or even regroup the least represented classes. More generally, outside of classification, the absolute precision on the scores or means can be used to separate the participants.

Recently, Guyon introduced a refined formula, which holds for all additive losses. The formula gives the sample size n required to get a given precision ν and confidence k :

$$n = \frac{\sigma^2}{\mu^2} \times k^2 \times 10^{2\nu}$$

Where μ represents the mean error of the model evaluated, and σ represents the standard deviation of the error rates. Interestingly, to increase the precision ν by one decimal, 100 times more test examples are needed. k, μ

and σ are squared, also indicating that the number of samples needed grows quickly under the influence of the error rate, the variance and the targeted confidence.

Another interesting point to take into account is the difficulty of the tasks. It is important to run baseline methods and to adjust the tasks : too easy or too hard tasks do not allow organizers to separate the participants ; the outcomes are then left to chance. The metrics for *intrinsic difficulty* and *modelling difficulty* of tasks are described in appendix (Section 10.1).

We highlight the importance of the number of test samples in an experiment conducted on tasks from the AutoDL Challenge (presented in Section 8.1). Only the datasets having less than 50,000 test samples were kept, for improved readability of the results. The experiment, done independently on each dataset, consist in increasing gradually the number of test samples used to compute the metrics and rank the candidate models. The size of the test set is therefore increased between 1 and m, m being the total number of test samples of the dataset. For each intermediate value i, we sample with replacement (bootstrap) i samples from the test set, and compute a ranking of the algorithms according to the scores obtained on this test set of i samples. We perform $t = 5000$ trials of this procedure, resulting in t different rankings, on which we compute the ranking stability using the Kendall W concordance measure (defined in Chapter 5). The number of candidates n is fixed in this experiment. n should not have an impact on the value of the stability itself, but only the variance of this value.

The results, given in Figure 4.8, indicate that, unsurprisingly, the stability increase with the number of test samples. A value of stability of 1 means that the ranking of all methods does not change when bootstrapping the test samples. In this experiment, most datasets converge into a stability near 1 when the number of test samples reaches 10^4. This indicates that the proposed models are well separated by the tasks. In the presence of ties, the stability converges to a value below 1. Under 1,000 samples, the rankings are unstable, meaning that there is an insufficient number of samples to significantly rank the candidates by performance in the case of this benchmark. For benchmarks where the models are numerous and harder to separate, more than 10^4 may be required in order to obtain a significant final ranking.

4.6 . Conclusion

In the design of competitions and benchmarks, the choice of scoring metrics stands out as a core decision. It not only shapes the goals of algorithmic development but also reflects our priorities and concerns in a given problem. In this chapter, we gave an overview and a comprehensive taxonomy of machine learning metrics, grouping them in the following categories : per-

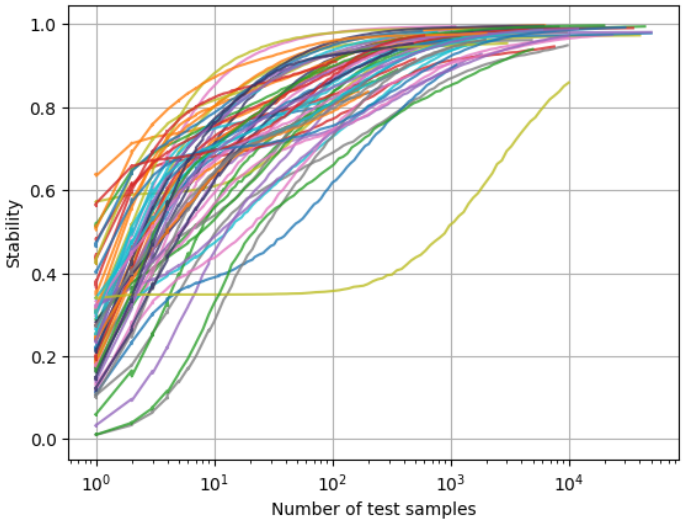


Figure 4.8 – Evolution of the ranking stability depending on the number of test samples used to score the candidates. The stability is the Kendall W measure computed on repeated trials. Each line is an independent dataset, and colors are displayed for readability.

formance metrics, ethical and societal impact metrics, resources consumption metrics and evaluator-centric metrics (using human or models as evaluators). While performance metrics are required in evaluating the proficiency of models on specific tasks, societal impact and resource consumption metrics are also crucial, ensuring that models align with ethical considerations and operate efficiently within real-world constraints. They help leveraging the reality gap between competitions and production environments. In competitions, the main aim is to beat other participants, which can lead to solutions that prioritize a small boost in performance, even if it means taking on a lot of technical debt. In contrast, real-world production requires stable and easily maintained solutions. Incorporating real-world constraints into the evaluation procedures can therefore solve the potential pitfalls of winning models and help bridge the gap between competition and production. Lastly, we presented alternatives for the problems where no clear ground truth and metrics can be used to select a model : evaluator-centric metrics, emphasizing the evaluation of humans or evaluator models.

We also stressed out the importance of computing statistical significance and error bars, in order to reduce the role of randomness and ensuring the credibility of benchmarks. We advocate for the use of bootstrapping to compute error bars, given its computational simplicity and effectiveness. Furthermore, it is imperative to underscore the importance of a sufficiently large test dataset, to ensure statistical significance and reliable evaluation. The required

number of test set samples, $n = \frac{\sigma^2}{\mu^2} \times k^2 \times 10^{2\nu}$, grows quadratically with the mean error μ, the standard deviation of error rates σ and the confidence k, while it grows exponentially with the targeted precision level ν.

To conclude, the taxonomy presented should serve researchers and practitioners, casting light on a new venue for machine learning models evaluation. It is not just about achieving high scores but ensuring that the metrics, and by extension the goals, align with broader aspirations of efficiency, equity, sustainability, and relevance.

Having explored the details of scoring metrics, it is clear that these metrics provide insights into individual performances. Yet, when faced with multiple scores, a mechanism to systematically rank and compare them is necessary. In the following chapter, we will turn our attention to ranking functions, designed specifically to rank based on multiple scores.

5 - Ranking n candidates from m judges

Building on the previous chapter where we provided a comprehensive taxonomy of scoring metrics, this chapter addresses a consequential problem : how to fuse multiple scores from multiple tasks, datasets, or from multiple metrics, into one cohesive ranking. The principle of ranking lies at the heart of the notion of competition. It is essentially a way to arrange or order candidates, from the most successful to the least successful. While ordering according to a single measure is somehow trivial, it becomes trickier when trying to order candidates on the basis of multiple scores. In this chapter, building on our previous study "Judging competitions and benchmarks : a candidate election approach" (Pavao et al., 2021a), we explore the problem of declaring a winner, or ranking "candidate" algorithms, based on results obtained by "judges" (scores on various tasks). Inspired by social science and game theory on fair elections, we compare various ranking functions, ranging from simple score averaging to Condorcet methods. We devise novel empirical criteria to assess the quality of ranking functions, including the generalization to new tasks and the stability under judge or candidate perturbation. We conduct an empirical comparison on the results of 5 competitions and benchmarks (one artificially generated). While prior theoretical analyses indicate that no single ranking function satisfies all desired properties, our empirical study reveals that the classical average rank method fares well. Some pairwise comparison methods also shows positive empirical abilities.

It should be noted that, in comparison with multi-objective optimization, the concept of Pareto fronts is not directly applicable in our context. This is because we do not actively explore the space with multiple objectives; instead, we receive the submissions provided by the participants. Additionally, the judges are not limited to being just metrics; they can also be different datasets or tasks.

5.1 . The hard problem of ranking

The problem of aggregating individual preferences into one global ranking is encountered in many application domains : politics, economics, sports, web pages ranking, and more. We are interested in a specific instance of this problem, where *candidates* are machine learning (ML) algorithms and *judges* are test performances on tasks (e.g. classification, regression, or reinforcement learning problems). Organizers of competitions, or benchmarks, usually simply average scores or competitor ranks over the various tasks, to obtain a global ranking. Nonetheless, theory has been developed around the problem of

ranking in the fields of social choice theory, economics and game theory, characterizing the properties satisfied or not by different ranking functions. As highlighted in Section 5.1.3, some desirable properties are theoretically irreconcilable, leading us to name this problem as "the hard problem of ranking".

5.1.1 . Problem setting

We consider a list of candidates $\mathcal{C} = (\mathbf{c_1}, ..., \mathbf{c_n})$ representing models (or algorithms) to be evaluated, and a list of judges $\mathcal{J} = (\mathbf{j_1}, ..., \mathbf{j_m})$ representing the tasks to be solved by the candidates. A score matrix M of size $n \times m$ is obtained by scoring the performance of each algorithm on each task ($\mathcal{C}$ is the list of rows of M and $\mathcal{J}$ is the list of columns of M). It can be thought of as a competition leaderboard in a multi-task competition. The scoring procedure (including choice of metrics, data split, the use of cross-validation, etc.) may vary from task to task, but all algorithms considered are scored using the same scoring procedure for a given task. The problem we are addressing is to obtain a single ranking of candidates $\mathbf{r} = \texttt{rank}(f(M))$ from the score matrix, using a ranking function $f : \mathbb{R}^{n \times m} \to \mathbb{R}^n$.

The function $\texttt{rank} : \mathbb{R}^n \to \mathbb{R}_+^n$ is defined as follows :

$$\forall i \in \{1, ..., n\},\ \texttt{rank}(\mathbf{v})_i = 1 + \sum_{j \neq i} \mathbb{1}_{\mathbf{v}_j > \mathbf{v}_i} + \frac{1}{2} \sum_{j \neq i} \mathbb{1}_{\mathbf{v}_j = \mathbf{v}_i}$$

We are looking for a ranking function f that performs well according to given criteria.

5.1.2 . Ranking functions

Ranking functions associate a global score to each candidate based on an aggregation of the columns of the score matrix M. We can then derive a ranking from such global scores. In this work, the ranking functions under study are : *mean*, *median*, *average rank*, and three methods based on *pairwise comparisons* of candidates. We also present the *random dictator* as a baseline.

Random Dictator

A straightforward approach to deriving a ranking from a matrix of scores is to uniformly select a judge at random, and then adopting their judgment as the definitive ranking. This method is referred to as the *random ballot* or the *random dictator*. While it may initially seem counter-intuitive or even absurdly incorrect, this method is actually prevalent in reality. In essence, the **random dictator is omnipresent**. Whenever we don't tackle a ranking problem head-on, but rather rely on a singular score to rank objects, we effectively permit the outcome to be governed by chance. This isolated score is typically drawn from a "mother distribution" and is consequently chosen at random. Examples of such scenarios include ranking models without re-runs, lack of bootstrap re-sampling, or cases focusing on just one task. In such situations, the influence

of the random dictator becomes evident.

Mean and Median

Mean and *Median* are average judges, obtained by either taking the mean (average value) or the median (middle value) over all judges, for each candidate. These two approach are fairly simple and very common in practice, especially the *mean*.

A potential issue with the *mean* is its sensitivity to extreme values, meaning that all judges don't have an equal impact on the outcome, especially if the scores are not normalized on the same scale, or are of different nature. The median leverage a bit this bias. On the other hand, if the scores are of similar nature, i.e. independently sampled from the same distribution, for examples several re-runs of the same experiment, then the *mean* naturally computes and converges to the expected value as the number of judges m increases, as stated by the central limit theorem (Anderson, 2010) and the law of large numbers (Evans and J.S.Rosenthal, 2004).

Average Rank

Average rank, or *Borda count*, is defined as follows :

$$f(M) = \frac{1}{m} \sum_{\mathbf{j} \in \mathcal{J}} \texttt{rank}(\mathbf{j})$$

It has the interesting property of computing a ranking which minimizes the sum of the Spearman distance with all the input judges, as shown by Kendall and Gibbons (1990a).

Pairwise comparisons

Pairwise comparisons methods give scores based on comparisons of all pairs of candidates :

$$f(M) = \left(\frac{1}{(n-1)} \sum_{j \neq i} w(\mathbf{c}_i, \mathbf{c}_j) \right)_{1 \leq i \leq n}$$

where $w(\mathbf{c}_i, \mathbf{c}_j)$ represents the performance of $\mathbf{c}_i$ against $\mathbf{c}_j$.

We can define different pairwise methods by designing different w functions :

- *Success rate* : $w(\mathbf{u}, \mathbf{v}) = \frac{1}{m} \sum_{k=1}^{m} \mathbb{1}_{u_k > v_k}$.
- *Relative Difference* : $w(\mathbf{u}, \mathbf{v}) = \frac{1}{m} \sum_{k=1}^{m} \frac{u_k - v_k}{u_k + v_k}$.
- *Copeland's method* : $w(\mathbf{u}, \mathbf{v}) = 1$ if the candidate $\mathbf{u}$ is more frequently better than the candidate $\mathbf{v}$ across all judges, 0.5 in case of a tie, and 0 otherwise.

Optimal Rank Aggregation

Optimal rank aggregation (ORA) methods are a family of ranking methods that consist in proposing a distance function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$ and finding a ranking r which minimizes the following objective function :

$$l(\mathbf{r}) = \sum_{\mathbf{j} \in \mathcal{J}} d(\mathbf{r}, \mathbf{j})$$

Some well-known distance functions that can be used are Kendall's τ distance, Spearman's distance or the Euclidean distance. The definitions of these metrics are given in appendix (Section 10.2).

The ORA using Kendall's τ as a distance function is known as the *Kemeny-Young* method. It has interesting properties such as being a Condorcet method and satisfying Local IIA; however, its computation is NP-Hard. The high complexity of the *Kemeny-Young* method prevented us from including it in the experiments.

The ORA using the Spearman distance also has interesting properties and is computationally linear as it produces the same ranking as the *average rank* method (Kendall and Gibbons, 1990a), as mentioned earlier.

In practice, the optimization can be performed using differential evolution (Storn and Price, 1997). A good overview of ORA and rank distance functions is given in Heiser and D'Ambrosio (2013).

5.1.3 . Impossibility theorems

Arrow (1950) and later Gibbard (1973) have shown that the problem of ranking n candidates from m judges is not trivial and that no aggregation method can satisfy all the desired properties. Such properties include that a candidate ranked first by a majority of judges must be declared winner, and that the ranking should be stable under perturbations of the set of judges or the set of candidates.

There is a connection between the impossibility theorems in Social Choice Theory and the Decision-Making Paradox, and it involves the difficulties and contradictions that can arise when trying to aggregate individual preferences or decisions into a preference ranking or collective decision. In Social Choice Theory, Arrow's Impossibility Theorem essentially states that no voting system can convert the ranked preferences of individuals into a community-wide (or collective) ranking while simultaneously satisfying a certain set of seemingly reasonable conditions. In decision-making (especially in the context of multi-criteria decision-making), a paradox can arise when different methods for aggregating decision criteria lead to different results. This is sometimes referred to as the Decision-Making Paradox (Triantaphyllou and Mann, 1989; Triantaphyllou, 2000). The paradox points out that different aggregation methods can yield different rankings of alternatives, raising the question of which me-

thod is the "best" or most accurate. In both cases, in Social Choice Theory and in Multi-Criteria Decision Analysis, the challenge lies in the process of aggregation and the complexities that arise when trying to satisfy multiple conditions or take into account multiple criteria.

Before stating the theorems, some desirable ranking functions properties (whether they can be satisfied or not) need to be defined.

Definition 1. ***Non-dictatorship*** *There does not exist a distinguished judge that can choose the winner.*

Definition 2. ***Non-imposition*** *The choice is imposed if a fixed candidate* $\mathbf{u}$ *is the winner for all preference profile of judges.*

Definition 3. ***Straightforward*** *Once the judge* i *has identified its own preferences* P_i*, it can choose a strategy* $s_i^*(P_i)$ *that best defends its preferences, with no need to know or to guess the strategies chosen by the other judges.*

Definition 4. ***Unanimity*** *If every judge prefers a certain option to another, then so must the resulting global preference order* $f(X)$*.*

Definition 5. ***Independence of irrelevant alternatives (IIA)*** *: The final ranking between candidates* $\mathbf{u}$ *and* $\mathbf{v}$ *depends only on the individual preferences between* $\mathbf{u}$ *and* $\mathbf{v}$ *(as opposed to depending on the preferences of other candidates as well).*

All the criteria defined above are desirable. Typically, non-dictatorship and non-imposition are absolutely required by any serious ranking procedure as, if these criteria are not satisfied, the preference profiles of the judges are not taken into account and the whole procedure is useless. The latter criteria, the independence of irrelevant alternatives, basically states that adding or removing candidates from the bottom of the leaderboard should not influence the order of the candidates at the top of the leaderboard. This property is very interesting in the case of machine learning competitions, where the set of candidates is not fixed and can vary a lot. However, this property may be hard to satisfy in practice.

Now that these properties are defined, we can reveal the definitions of the impossibility theorems.

Theorem 1. ***Arrow's theorem*** *(Arrow, 1950). Whenever the set* C *of possible alternatives has more than 2 elements, then the following three conditions become incompatible : unanimity, non-dictatorship and IIA.*

Theorem 2. ***Gibbard-Satterhwaite theorem*** *(Satterthwaite, 1975). If a voting rule has at least 3 possible outcomes and if it is non-manipulable, then it is dictatorial.*

Arrow's theorem and Gibbard-Satterthwaite theorem only apply to ordinal judgement. The theorem that applies to cardinal judgement is Gibbard's theorem. Gibbard's theorem is itself generalized by Gibbard's 1978 theorem and Hylland's theorem, which extend these results to non-deterministic processes.

Theorem 3. ***Gibbard's theorem*** *(Gibbard, 1973). If a ranking function is straightforward and has at least 3 possible outcomes, then it is dictatorial.*

The first thing can be noted regarding these theorems, is that they concern the case where there are **more than 2 candidates**. If there are only two candidates to rank, then any ranking function will work just fine ; in this case, it is trivial to take into account all judges' preferences. But the problem arises as soon as we have 3 candidates. We are then in a situation where the process is either dictatorial (one judge can choose the winner) or not straightforward (judges do not always defend their "interests" or preferences). Earlier, we stated that non-dictatorship is absolutely required. In this case, the conclusion is that judges can't always defend their interests.

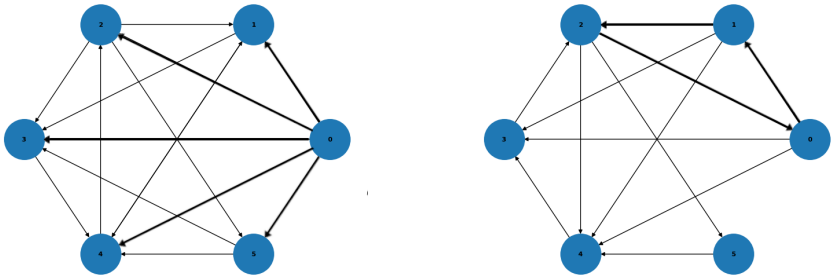
5.2 . Theoretical analysis

We have seen the main incompatibilities between the theoretical properties of ranking functions. More criteria can be defined in order to characterize the ranking functions, and decide which one is the best to rank machine learning competitions or any kind of ranking problem based on preference matrices. We summarize in Table 5.1 the main theoretical results on ranking functions under study, defined below. The first two relate to properties of the winning solution ; consistency and participation criteria relate to resilience against judge perturbation ; and the last two relate to resilience against candidate perturbation.

Definition 6. ***Majority criterion*** *(Rothe, 2015) : If one candidate is ranked first by a majority (more than 50%) of judges, then that candidate must win.*

Definition 7. ***Condorcet criterion*** *(Gaertner, 2006) : The Condorcet winner is always ranked first if one exists. The Condorcet winner is the candidate that would obtain the majority against each of the others when every pair of candidates is compared. The Condorcet criterion is stronger than the Majority criterion.*

If the Condorcet criterion is satisfied, then the Majority criterion is satisfied too. It is interesting to note that the majority and the Condorcet criteria are incompatible with the independence of irrelevant alternatives criterion, as a direct consequence of Arrow's theorem. This is because the majority preferences can be cyclic, thus exhibiting what is called a Condorcet paradox (Gehrlein, 1997). This property can be illustrated using Condorcet graphs, a graphical representation of pairwise comparisons between candidates. An arrow is



(a) *Candidate 0* is the Condorcet winner. (b) There is no Condorcet winner.

Figure 5.1 – Condorcet graphs where vertices represents candidates, and where there is an arrow between candidates **u** and **v** if **u** is more frequently better than **v** according to the judges' preferences. In the left example, *Candidate 0* is a clear Condorcet winner, beating all other candidates. The right example exhibit a Condorcet paradox, as *Candidate 0* beats *Candidate 1*, *Candidate 1* beats *Candidate 2* and *Candidate 2* beats *Candidate 0*, resulting in a cycle. Bold arrows are highlighted for clarity.

drawn from one candidate to another when it performs better. Examples of such graphs, with a clear Condorcet winner, and exhibiting a cycle, are given in Figure 5.1.

Definition 8. ***Consistency*** *(Young, 1975b), also called separability (Smith, 1973) or convexity (Woodall, 1994a) : Whenever the set of judges is divided (arbitrarily) into several parts and rankings in those parts garner the same result, then a ranking on the entire judge set also garners that result.*

Definition 9. ***Participation criterion*** *(Woodall, 1994b) : The removal of a judge from an existing score matrix, where candidate* **u** *is strictly preferred to candidate* **v***, should only improve the final position of* **v** *relatively to* **u***.*

Methods that do not satisfy the participation criterion are said to exhibit the no show paradox (Fishburn and Brams, 1983) : in some cases, the absence of a judge can help its preferred candidates.

Definition 10. ***Local IIA (LIIA)*** *: If candidates in a subset are in consecutive positions in the final ranking, then their relative order must not change if all other candidates get removed.*

If the IIA criterion, defined earlier, is satisfied, then the LIIA criterion is satisfied too.

Definition 11. ***Independence of clones****, or clone-proof (Tideman, 1987) : Removing or adding clones of candidates must not change the final ranking between all the other candidates.*

The independence of clones is extremely relevant in the context of machine learning competitions, as it is an environment potentially full of clones : participants submitting the same baseline, a previous winning solution, or a public model.

	Winner		Judge		Candidate		
	Majority	Condorcet	Consistency	Participation	IIA	LIIA	Clone-proof
Random dictator			✓	✓	✓	✓	✓
Mean			✓	✓	✓	✓	✓
Median	Rated[1]				✓	✓	✓
Average Rank			✓	✓			
Success Rate			✓	✓			
Relative Difference			✓	✓			
Copeland's method	✓	✓					
Kemeny-Young	✓	✓				✓	

Table 5.1 – Main theoretical criteria satisfied or not by the ranking functions. Proofs of the theoretical properties of "Success Rate" and "Relative Difference" can be found in appendix (Section 10.2).

5.3 . Empirical analysis

Our goal is to determine whether, in spite of pessimistic theoretical predictions, some ranking functions offer a good compromise between all criteria on specific data. To that end, we devise empirical quantitative equivalents of the theoretical criteria, and estimate them using bootstrapping techniques (Efron and Tibshirani, 1993b). In line with Brazdil and Soares (2000), we make a connection with meta-learning and include meta-generalization as part of our criteria, to evaluate whether ranking functions identify algorithms that fare well on *future* tasks of a similar nature. This setting arises in particular in AutoML competitions (e.g. Guyon et al. (2019a); Liu et al. (2021a)). We perform empirical evaluations on five competitions and benchmarks, contrast the results with theoretical predictions, and provide guidelines to select methods according to circumstances.

5.3.1 . Empirical criteria

In order to compare ranking functions, we must specify desirable properties, with respect to our end goals. Theoretical properties are strictly binary in nature, either satisfied or not. However, a method which does not satisfy a property could, in practice, satisfy it in most cases. To remedy this problem and have a more thorough comparison of methods, we propose these empirical criteria.

1. The median does satisfy the majority criterion only if the input matrix is matrix of scores, as opposed to ranks.

The **average rank of the winner** is the average rank across all input judges of the candidate ranked first in $f(M)$. To obtain a score between 0 and 1 to be maximized, we normalize it using the following formula : $1 - \frac{\texttt{average rank} - 1}{m-1}$.

The **Condorcet rate** is the rate of ranking the Condorcet winner first when one exists. This rate can be evaluated on a set of score matrices and is the direct empirical equivalent of the Condorcet criterion.

The **generalization** is the ability for a ranking function to predict the ranking of the candidates on new unknown tasks, which are not part of the set used for evaluation (of the benchmark or competition).

$$\texttt{generalization}(f) = \sum_{\mathbf{j} \in \mathcal{J}^{valid}} \frac{1}{m} \sigma(f(\mathcal{J}^{train}), \texttt{rank}(\mathbf{j}))$$

where σ is the chosen rank correlation coefficient, and $\mathcal{J}^{train}$ and $\mathcal{J}^{valid}$ are two disjoint sets of judges taken from M.

There exist many ways of computing the rank correlation, i.e. the degree of consensus between two rankings. In our research, we tested Spearman's ρ (Daniel, 1990) and Kendall's τ (Kumar and Vassilvitskii, 2010; Nelsen, 2001) (more precisely Kendall's τ_b, accounting for ties (Agresti, 2010)), and decided to stick to Spearman's ρ as the results were similar in both cases.

The **stability** of a ranking method f is the concordance of the output rankings it produces under some variability of the input. The stability against perturbations on $\mathcal{J}$ and the stability against perturbations on $\mathcal{C}$ can be estimated separately, by performing variations either across the judge axis or the candidate axis respectively. To measure the overall agreement between a set of q rankings resulting from perturbations, we compute an index of concordance by averaging correlations between $\binom{q}{2}$ pairs of ranking function outputs (typically using Spearman's ρ or Kendall's τ) :

$$\texttt{stability}(f) = \frac{1}{m(m-1)} \sum_{i \neq j} \sigma(X_i, X_j)$$

where X is a matrix whose columns are the rankings $f(M')$ produced on several variation M' of the score matrix M, and X_i is the i^{th} column of X. When perturbing candidates, σ is restricted to the subset of matching candidates.

5.3.2 . Experimental results

Data used for experiments are performance matrices of ML algorithms on a set of tasks in several benchmarks (Table 5.2), from past benchmarks as well as from a new benchmark we put in place involving 66 datasets and 13 algorithms from the AutoDL Challenge (Liu et al., 2021a). In this new benchmark, the datasets represent a wide variety of classification tasks on images, videos,

time series, text and tabular data. The algorithms are automated pipelines designed to perform as best as possible on any classification task, and most of the approaches are based on deep learning. The benchmark was computed in a systematic way with the same experimental environment for each training and testing. To estimate the values of the empirical criteria, we use bootstrapping. The generalization score, the average rank of winner, and the Condorcet rate are estimated on $10,000$ repeated trials. Each trial is based on a new version of the matrix M sampled with replacement both on the candidate axis and on the judge axis (in the case of generalization, the validation set is constituted of the out-of-bag judges). For the stability criteria, we consider two separate cases : one where bootstrap is done on the candidate axis and one on the judge axis. We use $q = 100$ bootstraps yielding each a new version of the matrix M (and a corresponding ranking), thus yielding $q(q-1)/2$ comparisons of pairs of rankings. The procedure is repeated 10 times. To compare the ranking functions, we compute their respective scores on the same bootstraps of the score matrix M.

The code of the experiments is public [2] and based on the Python package Ranky [3] (Pavao, 2020). Ranky is a package we developed which includes many ranking functions, as well as statistical measures such as distances and correlations.

	# Datasets	# Algorithms	Metric	W	Norm	Source
AutoDL-AUC	66	13	AUC	0.38	No	AutoDL (Liu et al., 2021a)
AutoDL-ALC	66	13	ALC	0.60	No	
AutoML	30	17	BAC or R^2	0.27	Yes	AutoML (Guyon et al., 2019a)
Artificial	50	20	*None*	0.00	Yes	Authors of (Sun-Hosoya et al., 2018)
OpenML	76	292	Accuracy	0.32	Yes	Alors (Misir and Sebag, 2017) website
Statlog	22	24	Error rate	0.27	Yes	Statlog in UCI repository

Table 5.2 – Datasets-Algorithms (DA) matrices used in the experiments. ALC refers to the Area under the Learning Curve, where each point is a ROC AUC over time. The first DA matrices only count as one in standard error calculations, because they come from the same benchmark. W is the concordance between datasets (as judges) within a benchmark, evaluating their degree of agreement. "Norm" means that the matrix was globally standardized.

The multidimensional scale plot and the rank correlation matrix between ranking functions suggest that all the methods yield different results, but we can see that the *Average rank*, *Success rate* and *Copeland's method* are closely related (Figure 5.2).

2. https://github.com/didayolo/ranking-esann-2021
3. https://github.com/didayolo/ranky

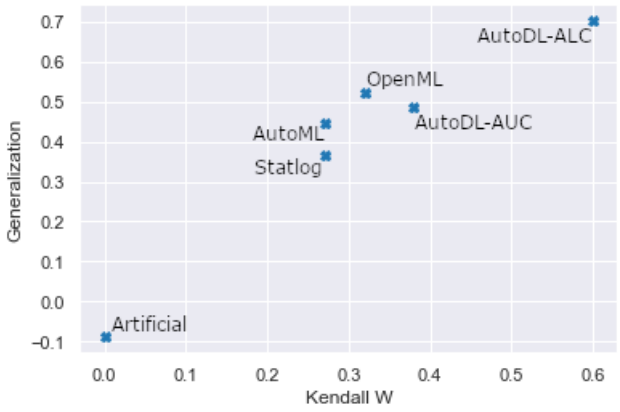
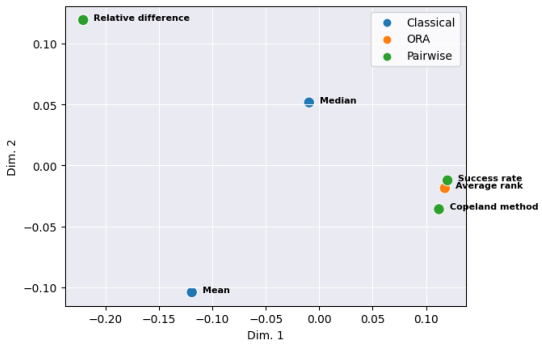


Figure 5.3 – Concordance, measured by Kendall W, of the score matrices (Datasets-Algorithms matrices) versus the mean generalization score obtained by the ranking functions. Each point is a benchmark.



	Mean	Median	Av. rank	Suc. rate	Rel. diff.	Copel.
Mean	1.00	0.81	0.75	0.75	0.75	0.76
Median	0.81	1.00	0.84	0.84	0.75	0.85
Average rank	0.75	0.84	1.00	1.00	0.65	0.98
Success rate	0.75	0.84	1.00	1.00	0.65	0.98
Relative difference	0.75	0.75	0.65	0.65	1.00	0.64
Copeland	0.76	0.85	0.98	0.98	0.64	1.00

Figure 5.2 – Multidimensional scale (MDS) plot (left) and correlation matrix (right). Both compare the rankings produced by each ranking method on the various benchmark matrices. The metric used for the MDS is the Spearman distance, and the Spearman correlation coefficient for the correlation matrix, averaged on all benchmarks. This gives an idea of the similarities between the methods.

Intuitively, the average generalization score obtained by the ranking functions should be correlated to the concordance between the judges for each score matrix. Indeed, the stronger is the concordance, or "agreement", between the judges, the easier is the task of producing a general ranking. This can be observed in the Figure 5.3, in which each point is a score matrix, whose location depends on the average generalization score (y axis) and the concordance between the judges (x axis).

Table 5.3, show that the empirical results are more nuanced than the theoretical results. The following observations can be made :

Mean and *Median* are, by nature, insensitive to candidate-wise perturba-

	Theoretical properties							Empirical properties				
	Winner		Judge		Candidate			Winner		Judge		Candidate
	Maj.	Condorcet	Consist.	Particip.	IIA	LIIA	Clone-proof	Winner rank	Condorcet rate	Generalization	Stability (judge)	Stability (candidate)
Mean	0	0	1	1	1	1	1	0.68	0.4	0.36	0.753	1.000
Median	0	0	0	0	1	1	1	0.70	0.5	0.37	0.702	1.000
Average rank	0	0	1	1	0	0	0	0.74	0.8	0.41	0.780	0.954
Success rate	0	0	1	1	0	0	0	0.73	0.8	0.40	0.777	0.839
Relative diff.	0	0	1	1	0	0	0	0.73	0.8	0.41	0.884	0.941
Copeland	1	1	0	0	0	0	0	0.73	1.0	0.41	0.771	0.965

Table 5.3 – Theoretical properties of ranking functions (left), estimated values of empirical properties (right). The empirical results are averaged over bootstraps and all $b = 5$ benchmarks. Error bars computed as $std(bootstraps)/\sqrt{b}$ govern the number of significant digits. The theoretical properties, presented in Section 5.2, are displayed as a reminder and comparison with empirical properties.

tions, as their calculations do not involve comparisons between candidates. Accordingly, they satisfy IIA, LIIA, and clone-proof criteria, and they are empirically perfectly stable against candidate perturbations. However, their winner rates and generalization scores are poor in comparison to the other ranking functions which make candidate comparisons, and which are robust to non-normalized scores.

Average rank performs better than *mean* and *median* in every respect, except candidate stability, but does not surpass *Copeland's method* and *relative difference*. It may be criticized for not taking actual differences between candidates into account and for generating ties. It remains a very attractive method, for being simple, easy to understand and to interpret, and for having a strong generalization score. *Average rank* also satisfies desired theoretical properties, notably *consistency* (Young, 1975a), ensuring that subdividing datasets doesn't alter average rankings. This implies notably that a top-ranked algorithm on a specific dataset won't benefit from the removal of that dataset. Furthermore, Brazdil and Soares (2000) empirically found that the average rank performs well in terms of generalization score when compared to other ranking methods. It is also flexible, as it can efficiently be tweaked into the *weighted average rank* function, by arbitrarily defining weights for each judge, thus adapting the method to different needs.

The *weighted average rank* function can be expressed as following, with $\mathbf{w}$ the list of weights :

$$f(M) = \frac{1}{m} \sum_{\mathbf{j} \in \mathcal{J}} w_j \times \texttt{rank}(\mathbf{j})$$

One can indeed choose coefficients of importance for each judges, for

instance to take into account secondary objective metrics such as *computation time* or *calibration* (see Chapter 4). Shah and Wainwright (2017) also advocates for the *Average Rank* ranking function for being simple, robust and optimal.

Success rate should (in non pathological cases) provide the same ranking as *average rank*. However, unlike *average rank*, it is sensitive to introducing clones and gets a very bad candidate stability score. We see no advantage to it.

Copeland's method, *relative difference* and *average rank* outplay the others. By design, *Copeland's method* has a perfect Condorcet rate and it gets the best empirical candidate stability (even though it is not theoretically IIA and clone-proof), as shown in Figure 5.4. Overall, these three functions exhibit a good candidate stability, which is good news in practice given the importance of robustness to clones and to candidate perturbations in the context of machine learning challenges. *Relative difference* closely approaches the performances of *Copeland's method* on these two criteria. However, *relative difference* is much better than all other ranking functions in terms of judge-wise perturbations. We performed side experiments on toy examples to understand the advantage of *relative difference* over other methods. One limit case is instructive : consider two pairs of identical judges, the second two providing rankings in the exact inverse order as the first two. Furthermore, the scores equate the rankings. All methods (except *relative difference*) return an n-way tie. *Relative difference* orders candidates to either favor little variance or large variance in judges' opinions. The former occurs when higher scores are better and the latter when lower scores are better. If one of the judges is suppressed, all methods (except *relative difference*) drastically change their opinion in favor of the majority of judges (two vs. one having now the same opinion). *Relative difference* also changes its opinion, but it remains biased with respect to judge variance, hence staying more faithful to its previous opinion, thus enjoying more judge-wise perturbation stability. *Relative difference* can therefore be voluntarily biased towards "generalist" or "specialist" algorithms. On a side note, the *Copeland method* can be used in complement, as in the case it identifies a Condorcet winner, it is insightful to give credibility to the winner.

In Figure 5.5, stability is observed as the number of datasets varies. The logarithmic-shaped curves clearly indicate increased stability with a rising number of judge datasets. For the final evaluation of AutoDL, 10 datasets were employed to rank participants, resulting in a stability score of approximately 0.85 for the average rank on ALC[4]. Meanwhile, the AutoML Challenge utilized 5 datasets in its final phase, corresponding to a stability score of roughly 0.65. To ensure a stable evaluation, with a stability score over 0.9, at least 20 test datasets would have been required for AutoDL, and at least 30 for

4. The ALC was used as the challenge metric. The AUC achieves an average rank stability of about 0.7.

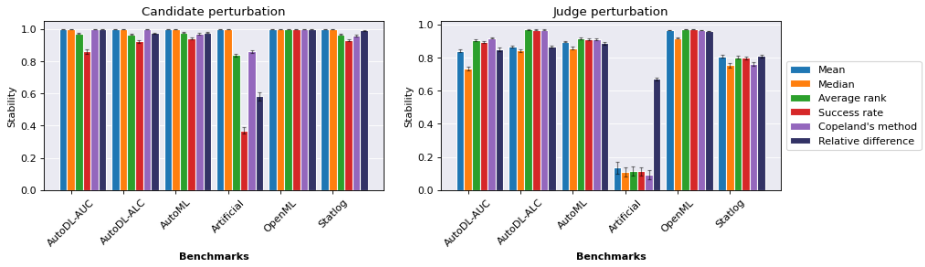


Figure 5.4 – **Stability against candidate axis perturbation (left) and judge axis perturbation (right)** of the ranking functions on each benchmark. The error bars represents the standard deviation across the scores obtained on all 10,000 repeat trials.

AutoML. Ideally, we recommend using more than 50 evaluation datasets for these tasks when possible. However, collecting such a number of datasets can be challenging, and the evaluation process can be computationally intensive.

It is interesting to note that the ALC metric, representing the area under a learning curve constructed from AUC points, exhibits greater stability than the AUC metric alone. It's also important to note that in the AutoDL challenge, scores aren't normalized. This lack of normalization explains the strong performance of *average rank* and *Copeland method*, as they inherently involve normalization procedures (like ranking). In comparison, benchmarks such as AutoML achieve higher stability, with less judges, particularly with functions like *mean*, which greatly benefits from the pre-normalization of the scores.

Median is less stable on odd numbers of judges, causing sawtooth-shaped curves. This is due to the fact that the computation of the median is different for an even or an odd number of values : for an odd number of values, the median is the number that is exactly in the middle of the ordered set, while for an even number of values, the median is the average of the two middle numbers. This averaging between the two middle values seems to increase the stability of the ranking produced by the median.

5.4 . Conclusion

The problem of ranking, which involves fusing scores given by judges to produce a definitive ranking of candidates, is prevalent across various application domains. In machine learning, it naturally emerges when ranking algorithms based on their performance across multiple tasks, test samples, or evaluation metrics. While this problem seems straightforward, it is inherently complex due to the theoretical incompatibilities of desired ranking function properties. Specifically, when applied to more than two candidates, a ranking function is either dictatorial or has no guarantee to reflect the judges prefe-

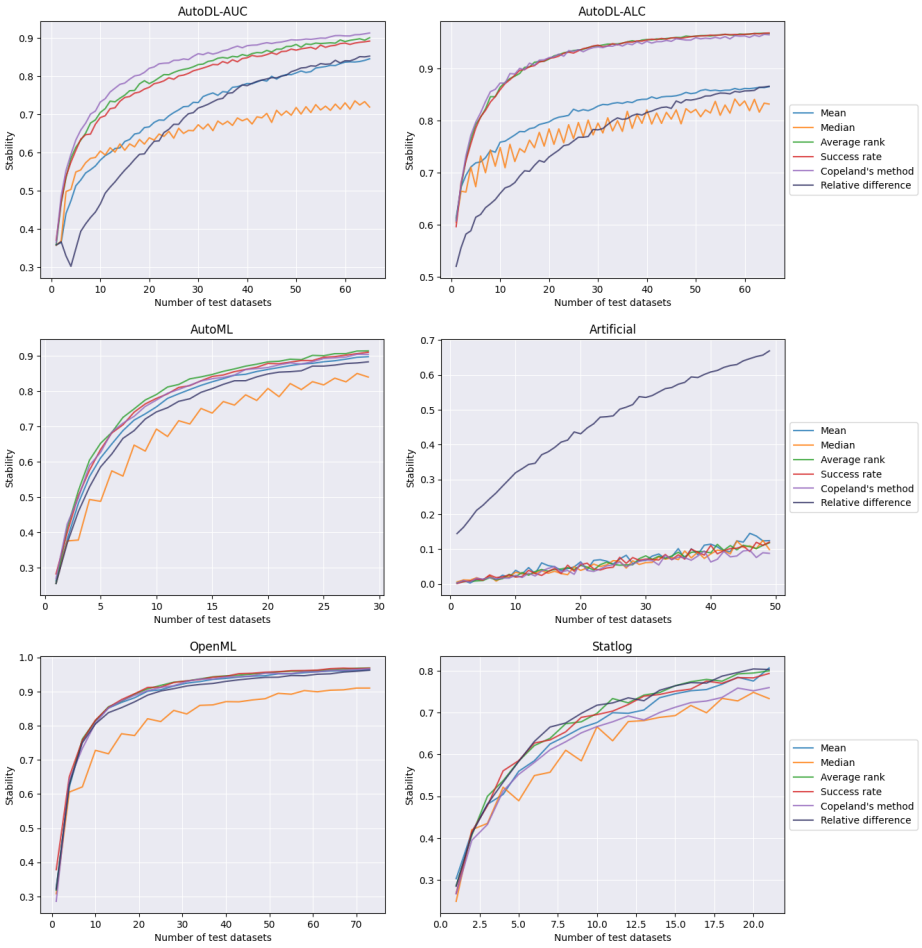


Figure 5.5 – Stability against judges perturbation (Kendall W) of the ranking when increasing the number of datasets used in the evaluation.

rences.

Analyzing the properties of different ranking functions offers valuable insights. One key observation is the distinct difference between functions that make direct candidate comparisons and those that assess each candidate individually. Notably, this distinction conflicts with the ability to identify a Condorcet winner, that is a candidate that surpasses all others in pairwise comparisons.

The theoretical assessment, while insightful, doesn't fully address the actual generalization and stability of ranking functions. To bridge this gap, we introduced an experimental framework to estimate the empirical characteristics of these functions. This framework can be employed in diverse scenarios to compare different evaluation methodologies. The process involves randomized repeated estimations of rankings produced by the functions, comparing these with independent judges or other ranking systems. Our empirical study, conducted on several machine learning benchmarks, reveals that when ranking machine learning models across multiple tasks, the *mean* and *median* functions are not satisfying, especially if scores aren't pre-normalized. *Average rank*, in the other hand, offers stability and better generalization properties, while remaining straightforward to compute and interpret, and can be adjusted using *weighted average rank*. The Condorcet-based *Copeland's method* is promising in selecting a strong winner, though the presence of a Condorcet winner is not guaranteed in every score matrix.

Moreover, similar to the importance of having a sufficient number of samples when evaluating models, it is crucial to have a large pool of judges to ensure a reliable and stable evaluation. To estimate stability, repeated bootstrap trials can be employed, varying either the candidate axis, the judge axis, or both. By calculating the concordance between the resulting rankings, the stability of the evaluation protocol can be measured. It is recommended to conduct such analysis on baseline models to assess the stability of the evaluation before running the benchmark or competition.

Having examined the methods of deriving rankings from a set of judges, we now turn our attention to analyzing pre-existing rankings. Specifically, we focus on rankings that are organized in two phases, and we study the methodology behind selecting a winner from these two-phase rankings.

6 - Judging competitions as a meta-learning problem

Continuing our earlier discussions on the selection of appropriate metrics and the consolidation of multiple scores into a singular ranking, we address in this chapter the issue of leaderboard overfitting in competitive settings. Particularly in competitions with extended duration or those permitting multiple submissions, participants might adapt to the tasks, effectively learning from the test set and leading to skewed results. Two-phase challenges aim to solve this problem. Moreover, we propose to consider competitions as systems of algorithm recommendation, and consequently as a meta-learning problem, and to study the problem of selecting a winning algorithm in such context. While evaluations of algorithms carried out by third party organizers eliminate the inventor-evaluator bias, little attention has been paid to the risk of overfitting the winner's selection by the organizers. By casting the problem of judging machine learning competitions as a meta-learning problem, we provide an analysis which outlines the danger of such overfitting. We propose a simple scheme to choose the winner, which alleviates this problem, in two-phase competitions having an identical number of similar tasks : the **top-k algorithm**. This algorithm selects the winner in the *final phase* among the best k participants in the *development phase* (used as prior knowledge). To evidence this phenomenon, we carry out an empirical evaluation using the results of several challenges and benchmarks. We show that a heuristic commonly used by organizers consisting of pre-filtering participants using a trial run, or qualification phase, reduces overfitting. We formalize this method and derive a semi-empirical formula to determine the optimal number of top k participants to retain from the trial run.

6.1 . Participant overfitting and organizer overfitting

Competitions are now part of most major conferences and their protocols are peer reviewed, to increase the rigor of evaluations and check, for instance, sufficient test set sizes and possible bias in data. The code of the winners is generally open-sourced, to maximize their impact. The ambition of competitions is generally to *recommend algorithms* that could perform well on new tasks resembling that of the competition. Thus competitions are a problem in which the *organizers* perform a learning task : from the task(s) of the challenge, they select an algorithm that should perform well on new future tasks.

From the machine learning point of view, **algorithm recommendation can be thought of as a meta-learning problem**. In a regular ML problem,

a dataset consists of data samples drawn independently and identically from the same unknown distribution, and split into a training set and a test set. In a meta-learning setup, what plays the role of data samples are datasets drawn from a meta-distribution of datasets from which a meta-training set and a meta-test set are drawn. The goal of meta-learning is to select a good algorithm that will perform well when *trained and tested* on new datasets.

Although meta-learning competitions have recently been organized (Baz et al., 2022), we take the stance that any competition is a meta-learning experiment, since their charter is to meta-learn from the results of the competition and perform algorithm recommendation. This raises the question of the validity of recommendations made by competitions, i.e. their capability to meta-generalize without overfitting the task(s) of the challenge. In this context, we introduce two types of overfitting :

- **Participant overfitting** may arise when participants perform model-selection based on a leaderboard's feedback. In practice however, as indicated by a large meta-study (Roelofs et al., 2019a), such participant overfitting is not severe, at least from the point of view of correctly ranking participants : the rankings in both phases are strongly correlated. However, prior to introducing 2-phase competitions, participants did strongly overfit, hence we believe that this is not advisable.
- **Organizer overfitting** occurs when the number of participants is large and rankings are noisy, increasing the chance of poorly selecting a winner. Competition organizers face a sad paradox : the larger the number of participants, the more "successful" their competition, but also the greater the risk to overfit.

A heuristic often employed in sports, chess, and other types of competition is to use eliminatory trial runs to filter participants for the final competition phase. We investigate this strategy in the context of 2-phase machine learning competitions. To alleviate overfitting the challenge setting, we propose to use the *development phase* to filter participants before entering the *final phase*. We extend the European Symposium on Artificial Neural Networks (ESANN) 2022 conference paper "Filtering participants improves generalization in competitions and benchmarks" (Pavao et al., 2022b) to include more background and a more complete description and analysis of the proposed methodology.

6.2 . General considerations about splitting

6.2.1 . Avoid overfitting using staged evaluation

In order to do a reliable evaluation, the data must be divided in **at least** three sets : train, validation, and test. Note that validation and test sets are

also commonly named "development and final phases"[1] or "public and private leaderboard". In some competitions, these sets contain distinct tasks or datasets; the validity of the argument still holds in these cases.

The **training set** is fundamental for model building. It includes both the features and the labels (i.e., the ground truth), which enable the model to learn the underlying patterns in the data. The **validation set** serves as an immediate check for how well the model is generalizing to unseen data. Although the ground truth is hidden, participants can get feedback on their performance. This enables them to tweak their models for improvement. It acts as a "sandbox" for understanding how the model performs on data it hasn't seen before but could potentially overfit to if used improperly. The **test set** is the ultimate arbiter of a model's generalization capability. No feedback on performance is provided, preventing any last-minute tweaking that could artificially inflate the model's evaluation metrics. This is summarized in Table 6.1. Ideally, these sets should share a similar data distribution, unless concept drift or shift is an inherent aspect of the problem being addressed.

The test set plays a critical role in this ecosystem by serving as a "firewall" against overfitting, since participants don't get feedback on their test set performance until the competition concludes. Indeed, receiving a repeated feedback from the leaderboard after each submission can lead participants to overfit their models to the validation data. The purpose of the test set is to evaluate performance on entirely new, unseen data, thereby ensuring that the winning solution is general rather than only excelling on the validation data. On a positive note, an empirical study conducted on 120 Kaggle competitions suggests that the overfitting between development and final phases (public and private leaderboards) is not common (Roelofs et al., 2019b). This could either mean that most participants are adhering to best practices or that the dataset sizes and complexities are sufficient to mitigate the risks of overfitting.

The importance of splitting data into at least three sets – train, validation, and test – cannot be overstated for ensuring both the reliability and generalizability of machine learning models. This is even more crucial in competitive settings, where the temptation to fine-tune models based on leaderboard performance can potentially lead to overfitting. Some past competitions allowed participants to fine-tune their models during the final phase, which is generally not a good practice. It blurs the lines between validation and testing, compromising the integrity of the evaluation process. A well-structured competition should aim to measure a model's ability to generalize to new, unseen data, and letting participants fine-tune their models based on test set performance undermines this objective.

1. Other synonyms of development phase include feedback phase and practice phase.

	Train	Validation	Test
Can participants access ground truth?	YES	NO	NO
Can participants obtain a score on it?	YES	YES	NO
Can organizers obtain a score on it?	YES	YES	YES

Table 6.1 – Train, validation and test sets. Here, the **validation set** refers to evaluation data whose ground truth is hidden from participants; not to be confused with the validation procedure they can perform on the **train set**. The **test set** is for the final evaluation, avoiding leaderboard overfitting.

6.2.2 . The dangers of splitting

We have seen that splitting the data or tasks for training, validation and testing is necessary in order to evaluate the participants fairly and avoid overfitting. A common practice is to completely shuffle randomly all data samples before splitting. This strategy assumes that data samples are independently and identically distributed across sets, avoiding to bias the evaluation towards one set or another. However, this approach can often yield misleading results, particularly in specialized domains where data naturally clusters into groups.

Consider a dataset composed of n microscopic images of cells collected from m different patients. These images aim to train a model for automated diagnosis, designed to generalize to new, previously unseen patients. If data splitting is executed at the image level, rather than the patient level, there is a high risk of overestimating model performance. In such cases, the model's apparent success in validation may not translate into effective generalization. This misleading effect is sometimes referred to as "voodoo machine learning" (Saeb et al., 2016). To better evaluate model performance in such scenarios, it is crucial to perform data splitting at the patient level, as illustrated in Figure 6.1, ensuring that all images from the same patient are grouped together in one of the training, validation, or testing sets.

The First Impressions Dataset (Escalante et al., 2018), utilized in various challenges, presents a similar data-splitting dilemma. The dataset consists in different videos of the same individuals, captured at different time intervals. To group all videos of the same individual into a single set, be it training, validation, or testing, allows for a more accurate measure of the model's ability to generalize to new, unseen individuals. This approach reduces the risk of overfitting and better prepares the model for real-world applications.

More generally, all applications where the data are stratified in two or more levels must be split in a manner that respects these hierarchical structures to ensure accurate evaluation and robust generalization.

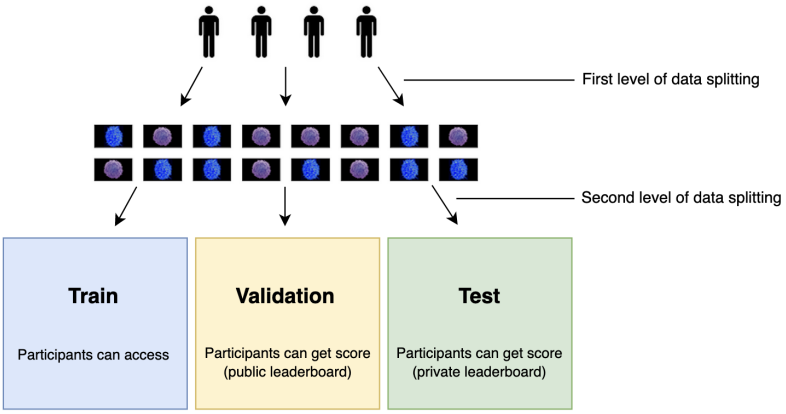


Figure 6.1 – Illustration of several possible levels of data splitting. Here, the samples coming from the same source (the same person or element of the first level of splitting) should be kept in the same subsets of data to avoid overfitting.

6.3 . Related problems and related work

The algorithm selection problem can be traced back at least as far as 1976 (Rice, 1976). Its relationship with the problem of meta-learning and the risk of overfitting at the meta-level has been pointed out (Liu, 2021). However, to the best of our knowledge, this is the first work that considers *judging competitions* or conducting a *benchmark* as an algorithm selection problem, and considers the risk of organizer overfitting. This chapter addresses the problem of determining *a single winner from rankings already obtained in two-phase competitions*.

The upstream problem of generating a good ranking has been addressed from various viewpoints. Briefly : the test set size needed to get good error rate estimates has been studied for single tasks (Guyon et al., 1998a) (discussed in Chapter 4). Recent work outlining issues with unbalanced data and fairness have proposed to consider worst group accuracy (Sagawa et al., 2019). Considering score distributions for algorithms having stochastic components has been advocated (Reimers and Gurevych, 2017). In the context of competitions, it has been suggested to run algorithms multiple times and select the worst run, to reduce the chance factor from winning (Baz et al., 2022). The problem of fusing scores from multiple "judges" (multiple tasks and/or multiple metrics) can be treated in various ways, ranking from simple averaging to solving an optimization problem (Pavao et al., 2021a) (presented in Chapter 5).

While we address the problem of optimally selecting the winner giving provided rankings in two-phase challenges, we do not address that of evaluating to what extent this choice is reliable or stable nor what the error bar

on the winner's solution is or whether there is a tie or near tie with other participants. Such issues are left for further work (see Section 6.6). The stability of competition ranking under perturbations of judges (representing e.g. multiple tasks or multiple objectives) and/or participants (removing or adding participants, or adding clones) has been studied by Pavao et al. (2021a). Bootstrap re-sampling has been used as a means to generate multiple competition rankings and report their frequency rather than a single ranking (Turner et al., 2021). Other issues of interest include computing error bars on the solutions of the participants. Recently, quantifying uncertainty has received a lot of interest, including in a setting similar to that of meta-learning : dataset shift (Ovadia et al., 2019). While generally competition settings consider single dataset splits into e.g. training/validation/test sets or simply training/test sets, practitioners usually attempt to reduce their error bars by averaging results over multiple splits (popular methods include leave-one-out, k-fold cross-validation, and bootstrap re-sampling (Kohavi et al., 1995)). For supervised learning competitions with result submission, this would not be possible without revealing to the participants the target values to be predicted. However, this is possible (though computationally expensive) using the competition protocol of *code submission*. Using multiple-split procedures is particularly advisable in meta-learning challenges, because of the scarcity of examples (since one example is a full task/dataset in meta-learning problems).

A question related to the problem we address is that of "out of domain generalization", which has been addressed from a variety of standpoints (although not in the context of judging competitions). For example, recent work has been warning about the danger that meta-learning algorithms may overfit the set of meta-training tasks (Shen et al., 2021). Propositions to reduce overfitting at the "meta-level" include algorithm-specific regularization methods like dropout (Tseng et al., 2020), Bayesian mechanisms (Yoon et al., 2018), and meta-augmentation of datasets (Rajendran et al., 2020) (which could easily be applied to challenges). Learning-theoretic bounds on the meta-generalization gap have also been proposed (Amit and Meir, 2018), introducing prior knowledge through setting an experience-dependent prior for novel task. This relates to the algorithm we proposed : we use the *development phase* as a prior to select the winner in the *final phase*.

6.4 . Proposed algorithm : top-k

In the sequel, we consider multiple phase competitions or benchmarks, in which participants are ranked multiple times. We assume that such rankings are drawn *i.i.d.* from a distribution of rankings. This abstracts from the need of describing more precisely how such rankings are obtained, which may widely vary from one competition to another. We revisit the *i.i.d.* assumption in

Section 6.6.

In a two-phase competition, the *development phase* is the "trial run" during which participant are filtered, and the subsequent *final phase* is used to determine the winner. The proposed algorithm (Algorithm 1) retains the top k participants in the *development phase*, and then selects the winner among them on the basis of the best rank in the *final phase*.

Algorithm 1: top-k method

Let :

$n \in \mathbb{N}, n \geq 1$, be the number of participants,

$k \in \mathbb{N}, 1 \leq k \leq n$, be a chosen quota,

$rank D(i)$ be the rank of participant i in the *development phase*,

$rank F(i)$ be the rank of participant i in the *final phase*, $i = 1 \cdots n$.

Select the winner as : $i^* = \text{argmin}_{rank D(i) \leq k} \, rank F(i)$

Clearly, keeping only a few participants in the *final phase* may save time and resources and may be logistically necessary if the *final phase* is a live competition. The question addressed in the remainder is the following : does the top-k algorithm yield as good or better meta-generalization than the "vanilla" method of simply selecting the winner in the *final phase*, without pre-filtering? If so, is there an optimal value of k?

6.5 . Empirical results

Although, from the participants' point of view, their algorithms are being tested in the *final phase*, **from the organizers' point of view the *final phase* is used for meta-training** [2]. Hence we need a "*post-challenge phase*" for meta-testing, to evaluate meta-generalization, if we do not have direct access to the distribution of rankings (which is only possible for synthetic data). We plot meta-learning curves as a function of k, the number of participants pre-selected in the top-k of the *development phase* :

- **Meta-training error** : score in the *final phase* of the declared winner.
- **Meta-test error** : score in the *post-challenge phase* of the declared winner.
- **Generalization gap** : difference between meta-test and meta-training error.

6.5.1 . Experiments on real data

We ran the top-k algorithm on 4 real meta-datasets coming from previously run challenges or benchmarks : AutoDL, provided by us, and Au-

2. The *development phase* during which the participants "practice" (i.e. perform a form of training) is used as a prior by the organizers.

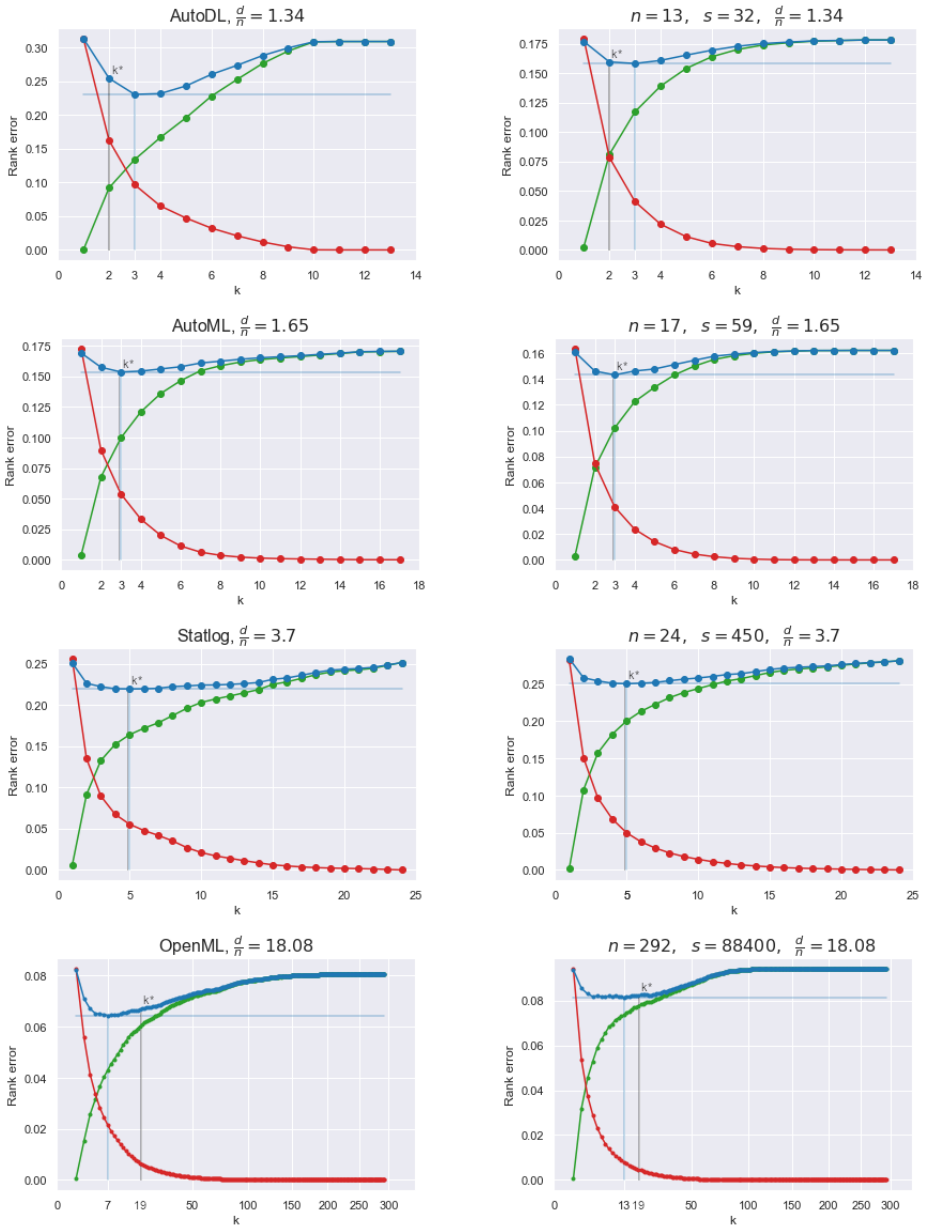
toML, OpenML and Statlog, provided by Sun-Hosoya et al. (2018). We simulate phases by averaging the scores on subsets of tasks. For the purpose of having comparable scores, regardless of the metric used in the challenge or benchmark, we use as score the average rank r of the participants on all tasks, in any given phase, normalized by $\frac{(r-1)}{(n-1)}$. The learning curves shown in Figure 6.2 were obtained by averaging over 10,000 data-split trials. We first observe that the meta-training error monotonically decreases as a function of k. This is to be expected since the meta-training error is the minimum score in the *final phase* of the top-k in the *development phase* : as k increases, one should get smaller and smaller scores by taking the minimum over a larger subset of values. The meta-training error basically varies with $\frac{1}{k}$ (Gordon et al., 2006). We also observe that **the meta-test error goes through a minimum**, following the desired behavior. The last point represents the vanilla method consisting in selecting the candidate with the smallest score in the *final phase*. We remark that the first and last point have similar values. This is not surprising since the first point is analogous to the vanilla method, but using the *development phase* to select the winner instead of the *final phase*. This prompted us to study the behavior of learning curves on synthetic data, to better understand under what conditions the top-k method is better than the vanilla method and which value of k is optimal.

6.5.2 . Experiments on synthetic data

To gain more insight into the problem, and understand under which condition the top-k method allows us to meta-generalize better than simply selecting the winner in the *final phase*, we generated synthetic data, as follows :

Consider a competition with $n \in \mathbb{N}$ participants. Assume that there is an ideal true (but unknown) ranking of participants g (g is for "generalization"). Without loss of generality, the ideal ranking is assumed to be $g = [1, 2, \cdots, n]$, i.e. the ideal participant ranks are their IDs. The rank of the true winner is 1. A synthetic empirical ranking obtained in a challenge phase is generated from g by repeated permutations of pairs of neighbors : a position i is drawn at random from $\{1, \cdots, n-1\}$ and the participants i and $i+1$ are exchanged. We repeat this operation s times. The smaller s, the more the empirical rankings will be correlated to the true ranking g (and to one another). We generate in this way 3 fake participant rankings, one for each phase : D, F, and P. We can then perform similar experiments as with real data and compute meta-training error and meta-test error of the top-k method. A theoretical analysis of this problem is given in appendix (Section 10.3).

We show in Figure 6.2 the results of experiments on synthetic data next to those on real data. On each row, we took care of matching the distance d between phase rankings (as measured by the Kendall τ). We are pleased to see that, qualitatively, the real and synthetic data curves remarkably resemble



(a) REAL data (b) SYNTHETIC data

Figure 6.2 – **Learning curves with top-k method** (averaged over 10000 trials) : Meta-training error, Meta-test error, Generalization gap. Rank error is the rank r of the selected winner in F and P, normalized by $(r-1)/(n-1)$. The blue vertical bar marks the optimum, s is the number of random swaps, d is the average Kendall τ distance (Kumar and Vassilvitskii, 2010) between phase rankings, and the grey vertical bar marks $k^* \simeq 1 + \frac{d}{n}$, an estimation of the optimal value of k.

one another : monotonic decrease of meta-training curve; meta-test curve going through an optimum; first and last point of meta-test curve nearly identical (performance of vanilla method). We observe that the optimum value of k is relatively small, even for large numbers of participants (last row). It increases with d, the distance between rankings. While s (the number of swaps applied to the ideal ranking, in synthetic data) can go to infinity, the expected value of d is bounded by half the Kendall τ distance between a ranking and its reverse, i.e. $\frac{n(n-1)}{4}$. Noting that the random process to create artificial rankings is essentially a random walk of the winner, we propose a semi-empirical formula for the optimum of k [3] :

$$k^* \simeq 1 + \frac{d}{n}$$

In Figure 6.3, we validate the formula with simulations [4] using synthetic data and position the optimal k observed in real data, for comparison. We observe that, on synthetic data, the formula fares well, but there is quite a bit of variance. On real data, for all meta-datasets, except OpenML, the true optimum is between $1 + \frac{d}{n}$ and $1 + \frac{2d}{n}$. For OpenML, the optimum is obtained for a smaller value of k. These results suggest that choosing $k^* \simeq 1 + \frac{d}{n}$ should provide the best meta-generalization, on average, but with a risk of biasing results, due to the large variance. Hence, a more conservative choice of k such as $1 + \frac{2d}{n}$ would be safer.

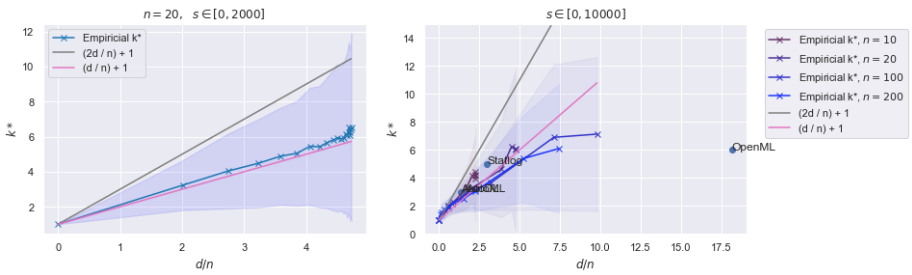


Figure 6.3 – Experimental estimation of k^* (mean and std shown). The pink curve shows the value predicted by the formula $k^* \simeq 1 + d/n$. Artificial swap data (left) and real data (right).

6.6 . Conclusion

3. Although regular random walk calculations are simple, the boundary conditions imposed in this problem, as well as the inter-dependence between the ranks of all candidates, increase the difficulty of obtaining a simple exact formula.

4. The code to reproduce all experiments and supplemental material can be found at https://github.com/didayolo/competition-metageneralization.

We presented the top-k method to alleviate overfitting in challenge winner selection, applicable in particular to ML contests. The top-k method simply selects the winner in the *final phase*, from a pre-selection of top k participants in the *development phase*. Empirical results show an advantage of top-k over the *vanilla* method of selecting the best participant in the *final phase*, thus improving meta-generalization (as described in this chapter). We confirm the results using a synthetic example, where competition phase rankings are altered versions of an ideal ranking, determining a true winner.

The remarkable resemblance between real and synthetic curves when the Kendall τ distance d between phase rankings is matched allowed us to propose a semi-empirical formula to predict the optimal value of the number of challenge participants k selected during the "trial run" : $k^* \simeq 1 + \frac{d}{n}$. However, this requires knowing d *a priori* or estimating it. A more conservative choice of k may anyway be preferable in practice, because when rankings are "noisy" (d large), there is a lot of variance (not visible in our experimental results, which are averaged over 10000 trials). So, instead of using k^*, we advocate **eliminating the participants who do not outperform the baseline methods** provided by the organizers with the "starting kit" (which may include well performing methods from previous challenges). This should be more acceptable to the participants than setting a hard threshold on the number of entrants of the *final phase*, and will at least eliminate the least serious participants who just submit the "starting kit". The validity of our paradigm rests on the assumptions that rankings in various phases are drawn *i.i.d.* from a distribution of rankings, and without ties. This assumption may be violated if, for instance, the tasks of the *development phase* and *final phase* are of different nature or difficulty, or if the participants overfit the tasks of the *development phase* (e.g. by making many submissions and seeking feedback from a leaderboard). This last problem does not seem to be severe in practice (Roelofs et al., 2019a).

The risk associated with the method is that it introduces bias if the *development phase* does not include tasks distributed similarly as the *final phase*, if the candidates (algorithms/participants) performances are not distributed uniformly (and thus swaps between the ideal ranking and the observed rankings are not distributed uniformly, as assumed), or if the participants use various means of biasing the evaluation, e.g. by having participants of the same team make submissions under different identifiers, to increase the chances that their team would be selected in the top-k and exclude other teams. In practice, we have been facing both problems and have been quite conservative, i.e. eliminated from consideration in the *final phase* only teams that did not perform better than baseline methods. The rules also forbid making submissions from the same team under different accounts and, since the winners must disclose their identity to win the prizes, this form of cheating is rare and easily caught. We also consider the expected values of meta-generalization

results over many repeated trials and do not address the variance of the meta-generalization, which, however, can be important enough that the meta-generalization benefit of using the top-k method is not significant. However, the computational benefit remains. The computational advantage can be significant if the *final phase* involves a large number of tasks, large datasets, and/or many repeated experiments, in an effort to reduce the variance in the evaluation. Additionally, results in the *final phase* must often be produced under time constraints.

After discussing model scoring, ranking models based on multiple scores, and selecting a winner from two-phase rankings, we will now review specific competition protocols.

7 - Specific protocols and design

In the previous chapters, the discussion focused on the analysis of machine learning challenges and benchmarks, addressing elements such as scoring metrics, ranking functions, or competition phases. As we move forward, in the two last chapters, we investigate the relationship between the previously discussed methodologies and the practical implementation of machine learning challenges. The need for particular experimental design is highlighted, given the wide variety of machine learning algorithms and applications, to ensure fairness, transparency, and relevance to specific scientific questions.

Indeed, machine learning is an expansive field offering a rich diversity of algorithms, each developed to solve specific tasks. These algorithms are commonly grouped into three main categories : supervised learning, unsupervised learning, and reinforcement learning algorithms. Beyond the algorithms themselves, the possibilities are further augmented by the diversity of data and domains of applications. Depending on the nature of the data, its source, shape, quantity and patterns, different approaches are required. The applications of machine learning are virtually limitless, covering medicine, physics, natural language processing, economics, and more. To be able to capture this complexity and diversity in competitions and benchmarks, innovative experimental design is required.

In this chapter, we explore the methodological considerations needed for designing machine learning competitions in specific scenarios of application. We cover supervised learning in Section 7.1, automated machine learning in Section 7.2, meta-learning in Section 7.3, time series analysis in Section 7.4, reinforcement learning in Section 7.5, handling confidential data in Section 7.6 and adversarial challenges in Section 7.7.

7.1 . Supervised learning

Supervised learning is a foundational paradigm in machine learning where models are trained using labeled data. In this paradigm, for each input instance in the dataset, there is an associated correct output, commonly referred to as *label* or *ground truth*. The primary goal of supervised learning is to construct a model capable of making accurate predictions for unseen instances based on this training.

In a classic supervised learning competition, participants evaluate their models on a given task using a dataset split into training and test sets. Typically, as depicted in Figure 7.1, participants are provided with a training dataset

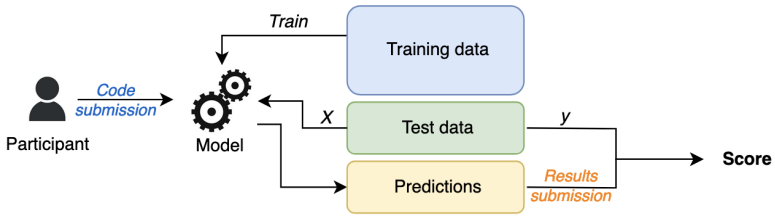


Figure 7.1 – Supervised learning evaluation workflow. Models are trained and subsequently evaluated on the withheld test set. The two possible protocols, *code submission* and *results submission*, are illustrated. X represents the features, and y the ground truth, of the test set.

to develop their models, and the evaluation is conducted on the withheld test set. Each competition phase must feature a different test set to prevent overfitting. Importantly, participants should not have access to the labels of the test set.

The choice of evaluation metrics in supervised learning challenges typically depends on the nature of the task — be it regression, classification, or others. The underlying goal is to objectively measure the performance of submitted models in terms of *accuracy*, *precision*, or other relevant metrics. Further insights into evaluation metrics are provided in Chapter 4.

7.2 . Automated machine learning

Automated machine learning (AutoML) is a field of study that focuses on developing methods and systems that can automate the process of building machine learning models. The goal of AutoML is to make it easier to build accurate and effective machine learning models without requiring extensive human intervention. By nature, AutoML methods are built to be able to solve a wide variety of tasks. Examples of such competitions include the AutoML Challenge Series (Guyon et al., 2019a), the AutoDL Challenge Series (Liu et al., 2021a), the AutoML Decathlon (Roberts et al., 2022) and the AutoML Cup (Roberts et al., 2023) based on the NAS-Bench-360 benchmark (Tu et al., 2023).

The general competition protocol consists in evaluating the candidate algorithms on a set of m tasks. For each of these tasks, the model is trained from scratch and evaluated on a hold-out test set, as demonstrated by the diagram in Figure 7.2. The m scores that result from evaluating the algorithm across the tasks are subsequently fed into to a master scoring and ranking process. Although the scores obtained on various tasks could simply be averaged, we suggest computing the average of the ranks achieved by comparing all candidates across the given tasks. This approach ensures a more robust ranking

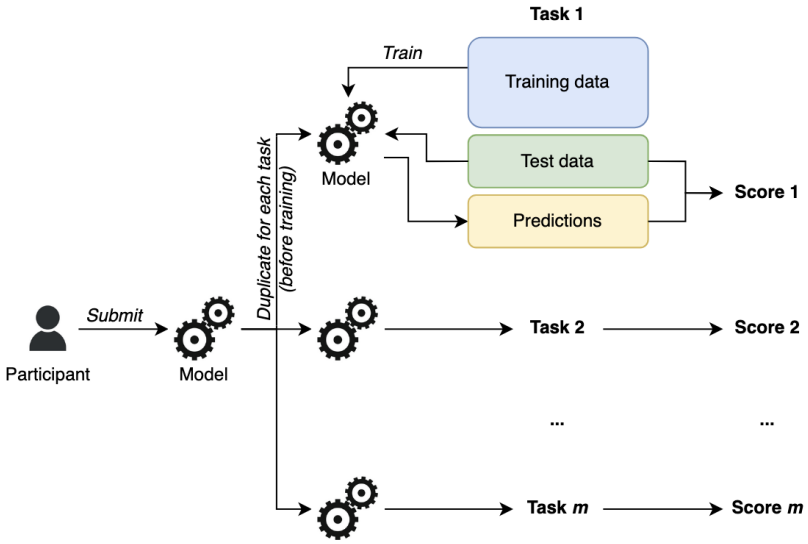


Figure 7.2 – Automated machine learning evaluation workflow. The submitted model is trained and tested from scratch on a set of independent tasks.

that accurately reflects the aim of a competition in automated machine learning. This point is explained in details in Chapter 5, and illustrated in Section 8.1.4.

A key aspect of the experimental design of automated machine learning competitions and benchmarks is the **blind testing**. To accurately assess a model's capability to solve diverse and unrelated tasks, participants must not have access to the test data. While some example training datasets can be made available to help participants in developing their models, the feedback and final evaluation stages must be conducted blindly, typically through the submission of code rather than direct interaction with the test data.

The selection of datasets and evaluation metrics is flexible and intrinsically tied to the specific objectives of the challenge. The main principle is that greater diversity in datasets is likely to yield a winning solution with more general applicability. Reciprocally, using similar datasets and metrics is more likely to produce an algorithm specialized in a particular domain or task. Having a large number of different tasks, while computationally expensive, enhances the overall diversity of the model's capabilities. This topic is discussed in Chapter 5.

While most AutoML challenges focus on supervised learning tasks, classification and regression, this experimental design can be used to organize crowd-sourced competitions or benchmarks on the automation of other machine learning tasks, such as data processing, clustering, content recommendation and more. The pre-requisite is to have a scoring metric defining the objective of the problem. Judging unsupervised learning competitions is dis-

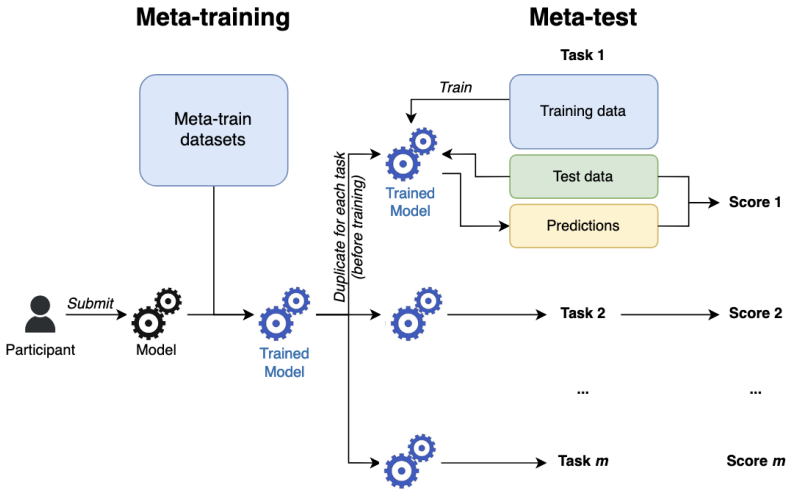


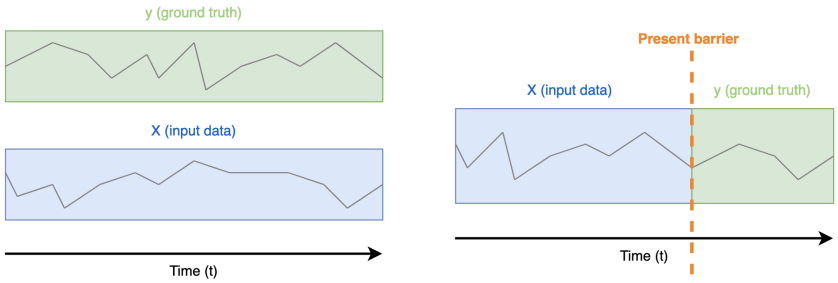
Figure 7.3 – Meta-learning evaluation workflow. The submitted model is trained on meta-train datasets, and then it is tested on a set of meta-test tasks.

cussed in Section 4.1.4.

7.3 . Meta-learning

Meta-learning is a sub-problem of AutoML. In its general definition, AutoML is a process of automating the machine learning process, including tasks such as data preprocessing, feature engineering, model selection, and hyperparameter tuning. AutoML techniques use algorithms to search for the best machine learning pipeline automatically. On the other hand, meta-learning is focused on learning how to learn. Meta-learning algorithms learn from experience to adapt their learning strategies for different tasks and domains (Brazdil et al., 2022). In essence, while AutoML automates the process of finding the best machine learning pipeline for a specific task, meta-learning takes a step further and automates the process of improving the learning algorithm's generalization capability across multiple tasks.

In the meta-learning challenge protocol proposed by El Baz et al. (2021); Baz et al. (2021) for the Cross-domain MetaDL Challenge, the evaluation of the candidate algorithms is divided in two sequential phases : the meta-training and the meta-test. During the meta-training, the submitted algorithm is trained on a set of datasets. The trained model is then forwarded to the meta-test, where it will be trained and evaluated separately on a new set of tasks. The whole process is illustrated by the diagram in Figure 7.3. The set of scores produced is then used to compare the model with other candidate models. As for the AutoML Challenge, the entire process is conducted blindly, preven-



(a) Time series regression (b) Time series forecasting

Figure 7.4 – Schematic view of time series regression (left) and time series forecasting (right). In regression, predictors can use past, present, or future values, while in forecasting, the task is to predict future values based on historical data and trends.

ting the participants from adapting their approaches to the specific datasets used. The main difference with the AutoML protocol (Section 7.2) is the use of a controlled meta-training phase, which implies that all candidate algorithms are pre-trained on the same data.

7.4 . Time series analysis

Time series analysis includes a wide variety of tasks, such as anomaly detection, sequence-to-sequence problems, or survival analysis, each presenting unique specificity. In the this section, our discussion centers around two central time series tasks : *time series regression* and *time series forecasting* (or prediction). While time series regression (Section 7.4.1) involves modeling the relationship between a dependent time-indexed variable and one or more independent variables, aiming to understand or predict the dependent variable's variations over time, time series forecasting (Section 7.4.2), on the other hand, is primarily concerned with predicting future values of a series based on its own past values and inherent patterns. This distinction is highlighted by Figure 8.9. The key distinction lies in the fact that regression models are more general and can be applied to predict values at any point in time, not strictly in the future, whereas forecasting is explicitly future-oriented, leveraging the temporal order of data to make predictions. Non sequential meta-data may also be available.

7.4.1 . Time series regression

Time series regression is essentially a supervised learning task with a temporal dimension, where the goal is to predict a continuous target variable based on historical data. While it shares similarities with classical regression in terms of learning from input-output pairs and minimizing prediction error,

the time component introduces dependencies between observations, necessitating consideration of the order and timing of data points in the modeling process. Time series regression can be multivariate, meaning that multiple variables must be predicted, as it is the case in the *Paris Region AI Challenge 2020* (PRAIC) (Pavao et al., 2021b) presented in Chapter 8. In this case, the performance can be measured using an average score (weighted or not) across the output variables, or using any ranking function. Another example of time series regression competition is the *AutoSeries Challenge* (Xu et al., 2021), which happens to be also an AutoML competition. This competition confirmed the efficiency of Gradient-Boosting Machines (GMB) to tackle time series regression tasks, as well as random search hyper-parameter tuning to tackle the AutoML part of the problem.

7.4.2 . Time series forecasting

"*A Brief History of Time Series Forecasting Competitions*" by Hyndman (2023) traces the transformative impact of forecasting competitions from the *Makridakis Competitions* series, organized by Spyros Makridakis and spanning from 1980 to today (Makridakis et al., 1982; Makridakis and Hibon, 2000; Makridakis et al., 2018), highlighting their role in shaping forecasting methodologies across diverse data types. The paper emphasizes the consistent success of combination forecasts, encouraging the use of ensemble methods, and points out the balance needed between automated forecasting and domain-specific expertise. Other exemplary time series prediction competitions include the competitions organized at the Santa Fe Institute (Weigend and Gershenfeld, 1993) which contributed to our understanding of time series prediction in a variety of contexts.

Time series forecasting competitions can be designed in both interactive or non-interactive settings, depending on the objectives and constraints of the challenge. In a **non-interactive** format, participants are provided with a complete dataset up to a certain point in time, and they are required to make predictions for future data points. The models are then evaluated based on their accuracy in predicting these unseen data points. This format is straightforward but may not fully capture the dynamic nature of real-world time series forecasting, where new data continuously become available, and models need to be updated accordingly. On the other hand, **interactive competitions** aim to mimic these real-world conditions by releasing data in stages. Participants make predictions based on available data, and as the competition progresses, new data are released, which can be used to update and improve the models. This format encourages the development of adaptive models that can respond to changes in data patterns over time. This design was typically found in the *COVID-19 Global Forecasting*[1] challenge on Kaggle, where

1. https://www.kaggle.com/c/covid19-global-forecasting-week-1

participants were tasked with predicting the spread of COVID-19 disease. Subsequently, the initially unknown ground truth was revealed and added to the training data on a weekly basis.

7.5 . Reinforcement learning

Reinforcement Learning (RL) is a subset of machine learning where an agent learns to make decisions by interacting with an environment. The agent receives feedback in the form of rewards or penalties, guiding it to optimize its behavior to maximize cumulative rewards over time, as illustrated by Figure 7.5. RL has been successfully applied in various domains, including robotics, game playing, and autonomous vehicles. Organizers of such challenges must choose suitable problem simulations, balance environmental complexity with computational demands, and set objective evaluation criteria that consider efficiency, adaptability, and robustness of the agent's performance.

Designing challenges for RL is an inherently complex task. One of the primary difficulties is the requirement for a simulated or real-world environment where participants' algorithms can interact, learn, and be evaluated. Ensuring the stability, reliability, and realism of these environments is crucial, as inconsistencies or inaccuracies can lead to misleading results and prevent the learning process. Furthermore, RL algorithms typically require a substantial amount of interactions with the environment to learn effectively, making the computational cost a significant consideration. Additionally, there is no one-size-fits-all metric for assessing the performance of RL algorithms across various tasks and environments, necessitating the careful selection and design of evaluation criteria that accurately reflect the objectives of the specific competition.

RL challenges can be designed following different protocols, primarily distinguished by the availability of pre-collected data. In challenges **without** pre-collected data, the algorithms proposed by participants engage directly with the environment, allowing data acquisition and learning concurrently. On the other hand, challenges **with** pre-collected data enable participants to refine and train their algorithms in an offline manner. The setting without pre-collected data is often referred to as **online learning**, and typically occurs in *OpenAI Gym Competitions* such as the *Retro Contest* (Nichol et al., 2018) where agents interacts with video games environment without prior knowledge. This approached is opposed to **offline learning**, which uses pre-collected data, such as the Atari Grand Challenge dataset (Bellemare et al., 2013). This particular dataset comprises a collection of human demonstrations across various of Atari games. In offline learning scenarios, the algorithm learns exclusively from this existing data, without the opportunity for real-time interaction or data acquisition, as seen in online learning settings. Regardless of whether

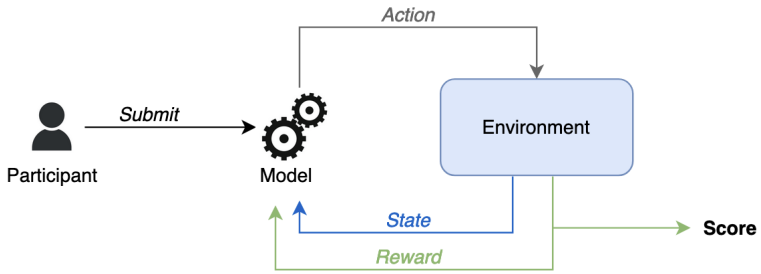


Figure 7.5 – Reinforcement learning competition workflow. The submitted agent interacts with the environment, with the objective of maximizing the reward over time.

algorithms are trained through online or offline learning protocols, their performance must ultimately be evaluated by interacting to live environments.

It is common to use a cumulative reward over time as a primary metric for determining the final score of participants' models. In such settings, the manner in which time is quantified plays a crucial role in the evaluation process, introducing a potential challenge in ensuring equitable and unbiased benchmarking. To mitigate inconsistencies that may arise from hardware disparities, it is advisable to standardize the measurement of total time in terms of the **number of environmental steps** taken, rather than relying on real-time duration measured in seconds. Adopting this approach ensures a consistent and equitable evaluation framework, as it remains invariant across different hardware configurations, thereby enhancing the fairness and reliability of the competition results.

Moreover, in RL challenges, having well-defined and expert-crafted metrics is crucial for evaluating the performance of participating algorithms accurately and fairly. These metrics need to capture not just the immediate rewards but also the long-term impact of decisions made by the RL agents. The design of these metrics requires a deep understanding of the specific domain, the goals of the RL task, and the potential trade-offs between different objectives. To gain a more detailed insight into how these metrics are crafted and their importance in RL challenges, refer to Section 4.1.3.

Instances of RL competitions include those in the domain of biomechanics. Notable examples in this category are the *Learning to Run* challenge (Kidzinski et al., 2018) and the *AI for Prosthetics* competition (Łukasz Kidziński et al., 2018). These events enabled progress in simulating and understanding complex biomechanical processes. In addition to biomechanics, RL competitions also extend to video game environments, offering unique challenges that require agents to navigate and interact within virtual worlds. The *MineRL Competition* (Guss et al., 2019) and the *Procgen Competition* (Mohanty et al., 2021) typically test the ability of RL algorithms to adapt and perform across procedurally ge-

nerated environments. Furthermore, competitions such as *Metalearning from Learning Curves* (Nguyen et al., 2022) explore general aspects of machine learning. This challenge study the meta-analysis of learning processes, encouraging the development of algorithms that can learn effectively from existing learning trajectories.

In conclusion, designing challenges for RL competitions is a nuanced task that requires careful consideration of the learning environment, computational resources, and evaluation metrics.

7.6 . Use of confidential data

Confidential data may include sensitive information such as personal data, financial data, or trade secrets, and it is important to ensure that this data is handled in a secure and ethical manner. There are several concerns associated with using confidential data in machine learning competitions, including the need to protect the data from unauthorized access, and the need to comply with relevant laws and regulations. Confidential data holds a great importance in many applications, both in scientific and industrial contexts. Some examples include finance, healthcare, and human resources. Using confidential data in crowd-sourced benchmarks is a challenge in itself, but can be highly beneficial by enabling innovation in critical fields.

In this section, we present two different protocols for handling confidential data : **replacing the data by synthetic data**, and **running the participants' models blindly on the real data**. These two protocols, with their advantages and drawbacks, make it possible to crowd-source research on private data without compromising confidentiality.

7.6.1 . Synthetic data

In order to propose a task based on confidential data to the participants without exposing the private data, one approach is to train a generative model to replicate the dataset. Synthetic data can then be generated from the model, and used to simulate the task without disclosing the actual dataset. This approach raises two antagonistic issues : in one hand, the synthetic data must resemble the original data to ensure the problem remains relevant and connected to the real world ; on the other hand, the generative model must not leak any real data points. We have developed metrics to evaluate generators *utility* and *privacy* (Yale et al., 2019a, 2020) (presented in Section 4.2.4).

The limitation of using synthetic data is the potential trade-off between *privacy* and *utility*. The *utility* of artificial data can be evaluated by deploying it in real-world scenarios and verifying that model outcomes are consistent with those achieved using real data.

We applied this concept in "To be or not to be", referenced in Pavao et al.

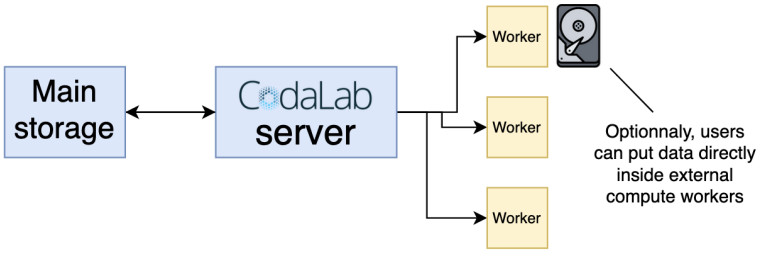


Figure 7.6 – Confidential data can be put directly inside organizers' compute workers, externally from the main servers of the platform.

(2019), a challenge designed to instruct health students. The task is to predict the survival or decease of patients in intensive healthcare units, based on tabular medical records. The source of the data is the MIMIC-III dataset, which consists in both numerical and categorical variables describing thousands of patients, such as age and blood pressure. Given the inherent confidential and sensitive nature of this data, it is subject to access restrictions. We generated a synthetic dataset using a Wasserstein Generative Adversarial Network (WGAN) (Goodfellow et al., 2014; Arjovsky et al., 2017) model. The resultant challenge continues to be used in Rensselaer Polytechnique Institute to train health students [2].

7.6.2 . Blind access to the data

The second approach for utilizing private data is to blindly execute participants' models on the real data. Two mechanisms are in play to benchmark the participants' solutions despite the private nature of the data : **code submission**, and **storing the data inside the compute workers**, as exposed in Figure 7.6. This way, only the uploaded models can read the data, it remains completely hidden from the participants. We implemented this feature to *CodaLab Competitions*. This is particularly interesting since, as shown in Chapter 3, organizers can link their own machines to the platform as external compute workers, ensuring a complete control over the data security. We employed this approach in the Paris Region AI Challenge 2020 (Pavao et al., 2021b), presented in Section 8.2.

A sample of artificial data, as well as documentation and baseline methods, should be provided to help the participants building their methods despite the constraints associated with not being able to access the dataset directly. Having extensive output logs can also helps the participants to navigate through the problem despite of the blind testing. However, it is advised to limit the size of the output logs to avoid the leakage of the sensitive data.

2. https://codalab.lisn.upsaclay.fr/competitions/3073

The security of external workers is ensured because the computer workers are owned by the organizers, and *CodaLab Competitions* platform cannot read them. The main limitation of this approach is that it is harder for the participants to work without direct access to the data, making it more difficult to reach the same performance level.

7.7 . Adversarial challenges

Adversarial challenges promote the development and enhancement of machine learning algorithms through an adversarial process, either designed using multiple iterative tracks, or simultaneous tracks. The premise of an adversarial challenge is intriguing : competitors in the initial track might work on a specific problem, such as audio generation or synthetic private data creation. The outcome of this first track serves as the foundation for the subsequent track, where the competitors are tasked to challenge the results produced from the prior track. This could mean developing algorithms that distinguish between real and fake audio, or conducting membership attacks on the synthetically generated data. Adversarial challenges can be compared to the architecture of Generative Adversarial Networks (GAN) (Goodfellow et al., 2014), with a generator and a discriminator. Just like in GANs, the initial phase of these challenges could be likened to the role of the generator, producing an "artifact" such as a synthetic audio file or private data. The subsequent phase corresponds to the discriminator, where the goal is to analyze and challenge the product of the first phase, be it through discerning real from fake audio or detecting privacy leaks. However, it is important to note that, while adversarial challenges can be compared to GANs, the scope of adversarial challenges in machine learning competitions can be broader and more diverse, including a wide range of problems and domains, and not solely limited to the generation of synthetic data and its evaluation.

In this section, we discuss adversarial challenges mechanisms, and review existing examples of such challenges, providing insights from our experience in helping to implement these challenges in our competition platforms. We make the distinction between the sequential design, in which the adversarial processes are separated in two distinct phases, and the simultaneous design, involving interactive processes. In reference to the game "Hide and Seek", we will call the generators "hiders" and the identifiers "seekers".

7.7.1 . Sequential design

In sequential adversarial challenges, the competition unfolds in distinct stages or phases. Typically, the initial phase may involve tasks such as generating synthetic data or producing certain algorithmic outputs. These outputs are then passed on to the next phase of the competition. In this second phase,

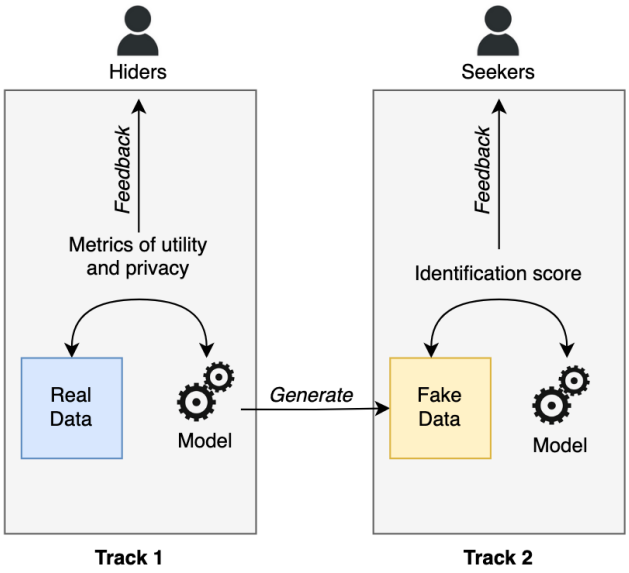


Figure 7.7 – Overview of a sequential adversarial challenge design. The data produced during the first phase are used in the second phase.

participants work on the problem of evaluating or challenging the results of the first phase, for instance, discriminating between real and generated data, or identifying vulnerabilities in the proposed algorithms. This approach provides a clear separation between the sequential stages of the competition, allowing competitors to focus specifically on the problem at hand for each phase. Figure 7.7 shows a schematic overview of such a challenge design. The first phase is used to generate synthetic data (participants are hiders), and this fake data is then used in the second phase to be identified (participants are seekers).

The ASVSpoof Challenge (Yamagishi et al., 2021; Liu et al., 2023) tackles the security and protection of automatic speaker verification systems. The 2023 edition introduces a two phases adversarial protocol. In phase 1, the aim is to collect deepfake attacks data from external data contributors who provide a spoofing data generation protocol and the access to automatic speaker verification systems. In phase 2, participants submit deepfake detection models, run on the data from phase 1. Deepfake attacks are expected to be more adversarial than in previous editions of ASVspoof and to fool automatic speaker verification, making the task harder and leading potentially to progress in the domain of speaker verification.

The Data Anonymization and Re-identification Challenge (DARC) (Boutet et al., 2020) is a sequential adversarial challenge of private data anonymization and re-identification.

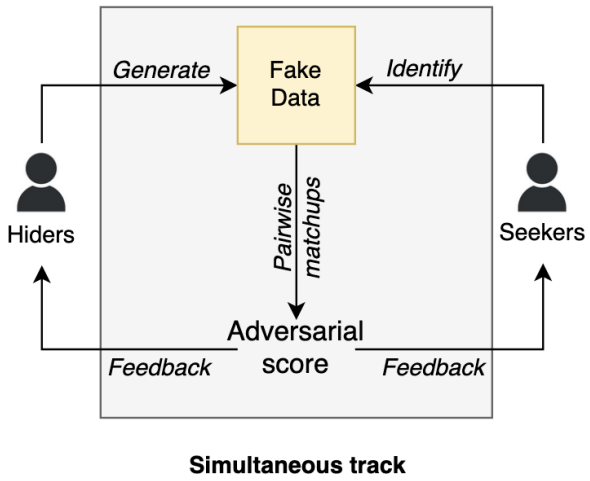


Figure 7.8 – Overview of a simultaneous adversarial challenge. Both hiders (participants generating data) and seekers (participants identifying data) proposes their methods during the same tracks. Pairwise matchups are performed to give feedback to all teams.

7.7.2 . Simultaneous design

Simultaneous adversarial challenges run both phases concurrently. While the participants are still working on generating outputs, the attackers are already attempting to challenge the preliminary results. This structure can add an additional layer of complexity as changes and improvements are made in real-time, creating a highly dynamic environment that demands agility and adaptability from the participants. Figure 7.8 gives an overview of such a challenge design. The attack score obtained by the identifiers (the seekers) against the generators (the hiders) is used as a feedback for both sides.

In Section 7.7.1, we briefly mentioned an adversarial anonymization challenge. Other adversarial anonymization challenges include the Hide-and-Seek Privacy Challenge (Jordon et al., 2020) and the Privacy Workshop Cup (Murakami et al., 2023). In these challenges, the two tracks of *synthetic data generation* and *patient re-identification* are run simultaneously. The synthetic data are attacked by the re-identification algorithms in head-to-head match-ups, resulting in feedback for both teams.

7.8 . Conclusion

In this chapter, we focused on examining the design specificity inherent in competitions and benchmarks in machine learning. We illustrated various experimental designs : supervised learning, AutoML and metalearning, time

Protocol	Data	Multiple tasks	Code submission	Interactive design
Supervised learning	✓			
AutoML	✓	✓	✓	
Metalearning	✓	✓	✓	
Time series	✓			*depends*
Reinforcement Learning	*depends*		✓	✓
Confidential data	✓		*depends*	
Adversarial challenges	*depends*		*depends*	✓

Table 7.1 – Characterization of the challenge protocols presented in the chapter, indicating the specific criteria that are mandatory (✓), and highlighting those that are possible depending on the design of the challenge (*depends*).

series analysis, reinforcement learning, the use of confidential data and adversarial challenges. The main characteristics and differences between these designs are outlined in Table 7.1. A common thread of most of these protocols is the necessity for participants to submit their model's code to the platform for evaluation. This resonates with Chapter 3 where we advocate for the use of code submissions, both allowing complex evaluation procedures and improving the reproducibility and the validity of the evaluation. Interactive designs are at play in reinforcement learning, where algorithms interact with a dynamic environment; in adversarial challenges, where competing algorithms engage with one another; and occasionally in time series prediction tasks, where datasets are regularly augmented with new observations, allowing previously used testing data to become part of the training set for future iterations.

It is also interesting to note that artificial data holds potential utility in certain challenge designs. For tasks where the ground truth is almost exclusively artificial data, or when emulating real data that is confidential, synthetic datasets are beneficial. The latter can also be addressed with real data, by employing blind-testing methods, ensuring participants cannot access confidential datasets. More generally, synthesizing artificial datasets can be beneficial for tasks lacking a ground truth, such as in unsupervised learning, since the synthesis rules can be precisely known by the organizers. Indeed, synthetic data in machine learning competitions can offer a controlled environment to evaluate algorithms, with the advantage of generating diverse and challenging scenarios that resemble complex real-world data distributions. Moreover, using artificial data, organizers can control the task difficulty, generate large datasets, address data imbalance, and reduce data collection cost. Additionally, in scenarios like reinforcement learning, where agents must learn from interaction within an environment, synthetic data provides an endless landscape of tasks for testing the robustness and adaptability of algorithms, as seen in competitions such as the *AI Driving Olympics* (Zilly et al., 2019). The main drawback of this approach is the potential *reality gap* between artificial and real data.

Adversarial challenges, where models are developed through adversarial processes, are particularly interesting, as they allow to evaluate models in complex tasks, on which sometimes no well-defined metrics can be applied. In such scenarios, the submitted models can act as model-based metrics, as discussed in Section 4.4.2.

Given the diverse and rapidly evolving nature of the field of machine learning, a comprehensive enumeration of all possible design features and evaluation criteria is impossible. For instance, competitions centered on one-shot learning might evaluate the ability of models to generalize from minimal data, while those focusing on fairness could prioritize unbiased predictions across diverse demographic groups. In the domain of real-time processing, the emphasis might shift to algorithmic speed and responsiveness. Tasks involving multi-modal learning demand the integration of information from varied data sources like text, images, and audio. Meanwhile, resource-constrained competitions challenge participants to optimize the model performance with tight computational or memory budgets. However, the methodologies and approaches outlined here can serve as references for future competitions, particularly those in emerging paradigms such as automated machine learning or adversarial challenges.

8 - Illustrative challenges

The broad reach and complexity of machine learning offer fertile ground for innovation, but they also make the task of designing meaningful competitions challenging. Given the richness of possible experimental design, each competition presents its own set of challenges and opportunities for learning. In this chapter, we draw upon our own experiences and the machine learning competitions we've organized to provide real-world examples that illustrate a variety of experimental designs, methodologies, and lessons learned.

We present the *Automated Deep Learning Challenge* in Section 8.1, the *Paris Region AI Challenge 2020* in Section 8.2 and the *Learning to Run a Power Network Challenge 2022* and *2023* in Section 8.3. We played a significant role in their organizations, contributing to their overall structure and execution. In the AutoDL challenge, my involvement was primarily as a consultant, where we took the lead in preparing datasets and deliberating on the design aspects of the competition. For the latter two challenges, our engagement was more profound due to our position as a Research Engineer at Université Paris-Saclay, funded by Paris region Ile-de-France, where we managed various parts of the competitions to ensure their successful execution.

Importantly, in the following, we analyze these illustrative challenges through the lens of the methodological frameworks and considerations discussed in earlier chapters, thereby offering a grounded understanding of how theory translates into practice. For each of these challenges, we review the following methodological questions :

1. **General presentation.** What problem is addressed by the competition? What is the specificity of the protocol?
2. **Scoring metric.** Is the scoring metric relevant given the task of the challenge?
3. **Size of test set and significance.** Is the evaluation grounded on a sufficient number of test examples and an estimation of variance to ensure a robust assessment?
4. **Ranking function.** Is the choice of ranking function relevant? (if applicable)
5. **Computation.** Is the competition using code submissions, and allocating equitable and sufficient computation resources for all participants?
6. **Splitting into phases.** Is the competition divided in multiple phases? If so, are the performances stable from one phase to the other?
7. **Participation and filtering of candidates.** Are there enough participants for a meaningful crowdsourcing? Are the participant filtered

through a qualification phase?

8. **Conclusive results.** Does the challenge provide a conclusive output? Are there creative solutions? Is the difficulty of the challenge well balanced?

In the subsequent, green checkmarks (✔) indicate positive answers, orange checkmarks (✔) partial answers, and red crosses (✘) negative answer.

8.1 . Automated Deep Learning Challenge

8.1.1 . General presentation

In this section, we present the design and analysis of the *Automated Deep Learning* (AutoDL) Challenge series (Liu et al., 2021a) that we organized in partnership with ChaLearn and Google. Following the AutoML Challenge series (Guyon et al., 2019a), an ambitious challenge that led to the development of auto-sklearn (Feurer et al., 2019), AutoDL is a research challenge that aims to develop automated machine learning methods to build high-performing deep learning models for classification tasks in diverse data modalities. The goal of the challenge is to develop methods that can handle a wide range of datasets and perform well without requiring extensive human intervention. The datasets collected for this challenge feature classification tasks [1], and are grouped in **five type of data : image, video, text, time series and tabular data**. No methods were imposed to the participants, however, in practice, all of them used deep learning. The competition was organized in five separated rounds : AutoCV (image and video), AutoNLP (text), AutoSpeech (time series), AutoWSL (tabular), and finally, AutoDL, involving all types of data [2]. The results shown later in the chapter concern the AutoDL round.

A hallmark of the AutoDL challenge series is that the code of the participants is blind tested, without any human intervention, in uniform conditions imposing restrictions on training and test time and memory resources, to push the state-of-the-art in automated machine learning. Each submission is trained and tested independently on all the datasets associated to the phase receiving the submission, following the framework described in Section 7.2.

In the AutoDL challenges, participants are given raw data in various formats such as images, videos, audio, and text, all uniformly formatted as TFRecords. While this format is native to TensorFlow, a data reader is also provided for PyTorch conversion. Images in formats like JPEG, BMP, and GIF use raw bytes directly, and are decoded on-the-fly to a 4D tensor. Video files in MP4 and AVI formats are treated similarly. Text datasets use integer indices repre-

1. The few regression tasks were transformed into classification tasks using binning and thermometer encoding methods.
2. https://autodl.chalearn.org/

senting words or characters, based on an included vocabulary. Speech and time series data are sequences of floats denoting amplitude over time, akin to the WAV format. Lastly, tabular data fits into the 4D tensor representation as a special case.

All tasks are supervised multi-label classification problems, i.e. data samples are provided in pairs (X, Y), X being an input 4D tensor and Y a target binary vector.

8.1.2 . Scoring metric ✔

In the previous AutoML Challenge series, the scoring was done after the training, on a test set, using r^2 score for regression tasks and either *balanced accuracy*, ROC AUC, *F1-score* or *probabilistic accuracy* for classification tasks. This is the common approach for scoring such tasks – setting a time limit for the training and scoring the models after the training.

On the other hand, the AutoDL challenges encouraged learning in a short time period both by imposing a small time budget of 20 minutes per task and by using an "any-time learning" metric. Specifically, within the time budget, the algorithm can make as many predictions as wanted, along the whole execution. This approach enabled the use of the Area under the Learning Curve (ALC), detailed in Section 4.3.1, to serve as the performance metric. Each point of the learning curve was scored by the *Normalized* Area Under ROC Curve (NAUC) averaged over all classes :

$$NAUC = 2 \times AUC - 1$$

The participants train in increments of a chosen duration, not necessarily fixed, to progressively improve performance, until the time limit is attained. This way, we encourage participants to develop techniques that improve performance rapidly at the beginning of the training process. The ROC AUC metric proved suitable as it is robust to varying scales of the target variables and is applicable to multi-label classification. Since all tasks were classification tasks, employing the same scoring metric across them contributed in clearer and more interpretable results.

The use of the area under the learning curve served its role well, promoting the fastest learning algorithms. A drawback of this approach is that it favored too much the most optimized learning pipelines, regardless of the final machine learning models, especially since the logarithmic time transformation emphasizes on the beginning of the learning curve. Indeed, the winner reported great improvements in scores after using engineering tricks, accelerating the data loading and the initialization of the model. While these are legitimate improvements, they do not align well with the initial objectives of the challenge. Moreover, the winners could obtain greater ALC without necessarily ending on top in terms of after-training performance.

As pointed out by Hutter (2019), a more segmented competition design that evaluates improvements in modular components individually could mitigate this risk. The different modules could for instance include pre-processing, hyperparameter tuning, or neural architecture search. Adopting such a modular competition framework could enable a more detailed evaluation, distinguishing essential scientific progress from engineering efficiency.

8.1.3 . Size of test set and significance ✔

For practical reasons, each dataset was kept under 2.5 GB, requiring some decrease in image resolution, cropping, and/or downsampling videos. We made sure to include application domains in which the scales varied a lot. We formatted around 100 datasets in total and used 66 of them for AutoDL challenges : 17 image, 10 video, 16 text, 16 speech and 7 tabular. The distribution of domain and size is visualized in Figure 8.1. All the datasets marked public can be downloaded on the corresponding challenge websites [3] and the metadata of all the datasets can be found on the "Benchmark" page [4] of our website. We have carefully selected the datasets out of the 100 possibilities using two criteria : (1) having a high variance in the scores obtained by different baselines, ensuring an interesting modelling difficulty (see Section 10.1) and (2) having a relatively large number of test examples to ensure reasonable error bars (see Section 4.5).

The datasets were dispatched into the different rounds of competition as public datasets, feedback phase datasets or final phase datasets. In the *final phase* of the last round of competition, **10 datasets** were used to evaluate the candidates, 2 for each data modality. As indicated by an experiment presented in Chapter 5, the stability of the average rank function is around 0.85 (Kendall W) on this competition datasets and models, with ALC metric. Ideally, 20 or more datasets should have been used in the final evaluation, for a stability above 0.9.

To obtain the final results, shown in Figure 8.3, the entry of the top 6 teams were re-run from scratch 9 times, and 3 times for other teams.

Overall, key factors enhancing the robustness include diverse data representations, strategic dataset selection based on challenging modeling scenarios and statistical reliability, and multiple run trials for reliability. These combined efforts provide a comprehensive and consistent evaluation, though there is always room for further refinement – typically including more datasets in the evaluation, and having larger test sets.

8.1.4 . Ranking function ✔

3. https://autodl.lri.fr/competitions/162
4. https://autodl.chalearn.org/benchmark

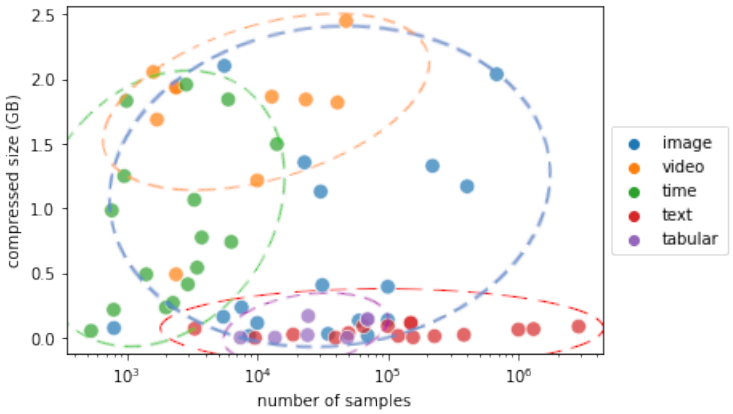


Figure 8.1 – **Distribution of AutoDL challenge dataset domains** with respect to compressed storage size in gigabytes and total number of examples for all 66 AutoDL datasets. We see that the text domain varies a lot in terms of number of examples but remains small in storage size. The image domain varies a lot in both directions. Video datasets are large in storage size in general, without surprise. Speech and time series datasets have fewer number of examples in general. Tabular datasets are concentrated and are small in storage size.

In each phase, an overall ranking of the participants is derived by averaging their ALC ranks obtained on each individual dataset. The **average rank** function was used in the final phase to determine the ranking and select the winner. As shown in Chapter 5, the use of the average rank offers many advantages. It allows us to fuse scores of different scales, it satisfies desirable theoretical properties, and its generalization abilities (rank correlation with unseen tasks) was empirically validated by Brazdil and Soares (2000) and later Pavao et al. (2021a). In light of Chapter 5, *average rank* was a good choice. This method selected a clear winner, that happened to be a Condorcet winner, meaning that it surpassed all other participants in pairwise majority comparisons. It is noteworthy that the winning approach relies on sub-models specialized for each data modality. The master algorithm acts as a crossroads, the selecting the right sub-model after analyzing the shape of the input data to recognize between images, videos, texts, time series and tabular data. While the *average rank* function aims at selecting a general approach, the emerging solution of the challenge can be interpreted as a group of specialized models.

8.1.5 . Computation ✔

To handle the intensive computational demand, we utilized on cloud virtual machines, facilitated by a donation of $100,000 cloud credits from Google. Participants' computations were executed on virtual machines equipped with

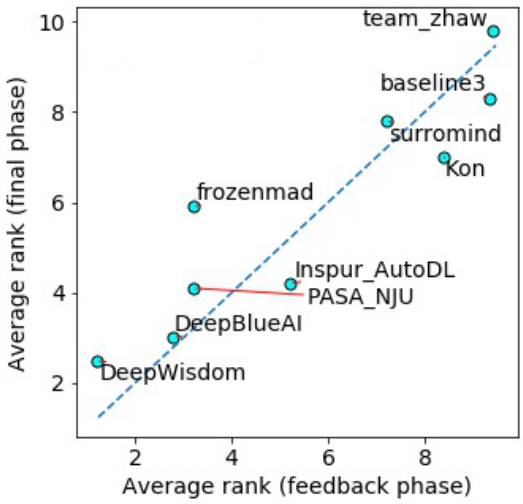


Figure 8.2 – Comparison of performance in the *feedback* phase and *final phase* of AutoDL. The scores shown are the *average rank*. The phases are positively correlated, indicating little overfitting of the *feedback phase*.

NVIDIA Tesla P100 GPUs, 4 vCPUs, 26 GB of memory and 100 GB disk storage. As part of the experimental design, the execution time was limited to 20 minutes for each dataset.

The use of code submission, blind-testing, containerized environment, and execution time limit ensured that the evaluation process was equitable among participants. Despite this, the allowance of pre-training was relevant for the task, potentially giving an advantage to participants with access to abundant data and significant computing power. The computational constraints naturally encouraged the development of efficient and rapid learning algorithms.

8.1.6 . Splitting into phases ✔

The challenge followed the classical staging into two phases : a *feedback phase* during which methods were trained and tested on five practice datasets and a *final phase* using ten fresh datasets. Only one final code submission was allowed in that phase. Since this was a complete blind evaluation during both phases, we provided additional "public" datasets for practice purposes and to encourage meta-learning.

The average ranks of top methods in both phases are shown in Figure 8.2, with a Pearson correlation $\rho_{X,Y} = 0.91$ and p-value $p = 5.8 \times 10^{-4}$. This means that the correlation is statistically significant and no leaderboard overfitting is observed. Thus the winning solutions can indeed generalize to unseen datasets.

8.1.7 . Participation and filtering of candidates ✔

Each round featured a **$4000** prize pool : $2000 for the 1^{st} place, $1500 for the 2^{nd} place and $500 for the 3^{rd} place. In total, **twenty teams** participated in the last AutoDL round. While the number of participating teams was not vast, these teams were comprised of highly competent and cutting-edge researchers, making the competition very interesting and competitive.

In order to optimize computational resources, only participants who outperformed the baseline were allowed to compete in the competition's final phase. This baseline, established from previous competition rounds, was fairly strong, resulting in only 9 out of 20 participants able to access the stage. Consequently, we were able to re-run the final evaluation 9 times for the top-6 teams and 3 times for the remaining teams, ensuring more statistically significant results. While the initial intent of filtering the participants to the final phase was to save resources, we demonstrate in Pavao et al. (2022b) (detailed in Chapter 6), that this filtering based on the feedback phase actually helps in statistically strengthening the final ranking, increasing the probability of selecting the most general winner (most general in the sense that it would rank well in new post-final phases). In was therefore, in retrospect, a good choice of competition design.

8.1.8 . Conclusive results ✔

Overall, the outcomes of this challenge were impactful, with a definitive winner emerging. The winner achieved exceptionally strong performance across a diverse range of tasks, as shown in Figure 8.3. The task difficulty was high, yet well balanced, evidenced by several teams surpassing the baseline with a diverse range of outcomes. Building the competition in multiple rounds was efficient, as the winning solution for each round (each data modality) could be re-used in the last AutoDL round. The emerging solution did not manifest as a singular, elegant method applicable across all data types; instead, it took the form of a collection of specialized algorithms adapted for specific data categories.

8.2 . Aircraft Numerical Twin

8.2.1 . General presentation

The *Paris Region AI Challenge* (PRAIC) 2020, organized in partnership with Dassault-Aviation and the Ile-de-France region, was a contest between 10 startup companies, taking place in 2021. The 10 participating teams (startup companies and academics) were pre-selected upon submitting an application file. The teams had to tackle a multivariate time-series regression task, in the field of aeronautics. Dassault-Aviation, a French aircraft manufacturer, provided a confidential dataset consisting of aircraft sensory inputs during flights. The

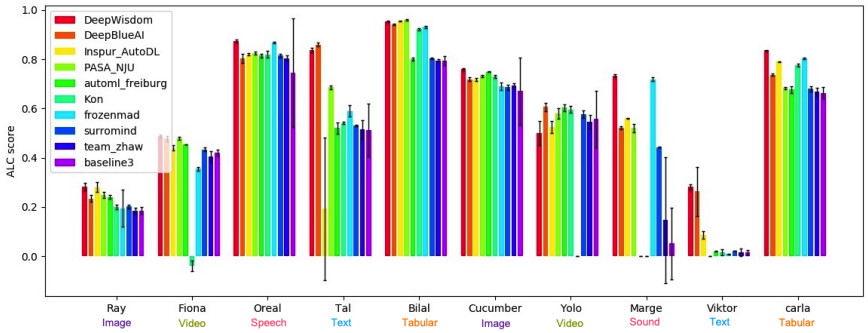


Figure 8.3 – ALC scores of top 9 teams, and the baseline, in AutoDL final phase averaged over repeated evaluations. The entry of top 6 teams were re-run 9 times and 3 times for other teams. Error bars are shown with (half) length corresponding to the standard deviation from these runs. The team ordering follows that of their average rank in the final phase. It is worth noting that the baseline is a robust model, having been derived from the winning solutions of previous competition rounds. Only participants out-performing this baseline in the feedback phase were allowed to enter the final phase.

design and analysis of the challenge are presented in Pavao et al. (2021b). Given a number of input parameters (sensor data) recorded in sequence during the flight, the participants' model had to predict output values (strain gauges), also recorded sequentially on test aircraft, but not recorded on service aircraft. In the application domain considered, collecting strain gauge data on service aircraft is costly or infeasible. The objective is to create a predictive model of the strain gauges measurements from the data of test aircraft, to obtain "virtual" strain gauges for service aircraft, hence the concept of "Aircraft Numerical Twin". The main goals are to optimize scheduled maintenance and to improve the design of future aircraft.

Time series regression addresses the problem of predicting certain unavailable *output* time series from other given *input* time series, as explained in Section 7.4.

In this application, input time series (data routinely recorded) and output time series (strain gauge data recorded on test aircraft only) are multivariate time series, **irregularly sampled** or sampled at different frequencies, but with given time stamps. We refer to this problem as "asynchronous time series" regression. The problem is cast into a supervised learning problem.

Formally, both input $X = [x_{t,j}], t = 0 \cdots t_{max}, j = 1 \cdots n_{sensors}$ and output $Y = [y_{t,j}], t = 0 \cdots t_{max}, j = 1 \cdots n_{gauges}$ are multivariate time series. The index t indicates time and the index j the various sensors, called *sensors* for the inputs and *gauges* for the outputs. A typical method is to regress a window of inputs centered at time t to predict $\mathbf{y}_t$.

In PRAIC 2020, the protocol follows the usual workflow of code submission

competitions organized on *CodaLab Competitions*, as presented in Chapters 3 and 7 (Section 3.1.1 and Section 7.1). At training time, both X and Y are available, while at test time, only X is available and Y must be predicted. The goal is to predict missing output values in the test time series, given the time stamps, as the time series are asynchronous. A typical way to proceed is to learn to regress a window of input data centered at time t to the output at time t.

Due to the industrial sensitivity of the information, **the entire dataset had to be kept strictly confidential**. To ensure the security of the data while still enabling meaningful competition, we employed the blind testing approach presented in Section 7.6. In this setting, participants submit their models through code submissions. Both training and testing of the models was done on the platform in both phases, hence the participant had no direct access to the data, except for some sample data provided with the starting kit, for illustrative purposes. In addition, the data were stored directly inside the compute workers, to avoid any data transmission between different parts of the platform. Finally, the quantity of output logs returned by the platform were capped to avoid any malicious data leakage. This way, the data remained secure within the boundaries of the compute workers assigned for the challenge. This not only safeguarded the proprietary information but also allowed participants to engage in the competition under controlled and secure conditions. By utilizing this blind testing methodology, we were able to maintain data integrity and confidentiality without compromising the quality or the objectives of the competition. Such a challenge would not have been possible without this blind-testing procedure.

8.2.2 . Scoring metric ✔

The predictions were evaluated using the Mean Absolute Error (MAE). Given that $\mathbf{y}$ is a ground truth output variable and $\hat{\mathbf{y}}$ a prediction, MAE is defined as :

$$MAE = \frac{1}{n}\Sigma_{i=1}^{n}|\mathbf{y}_i - \hat{\mathbf{y}}_i|$$

Performances were averaged over all data points in all test sequences and over all gauge outputs. The MAE metric is simple to compute and to interpret, and its choice was validated by the experts in the field of aeronautics from Dassault-Aviation who proposed the task and the data.

8.2.3 . Size of test set and significance ✔

The input time series included between 124 and 130 input parameters (sensors) depending on the flights. The output time series included between 62 and 67 output parameters (gauges) to be predicted. During the *feedback phase*, models were trained on 117 complete flight records and tested on 4,805

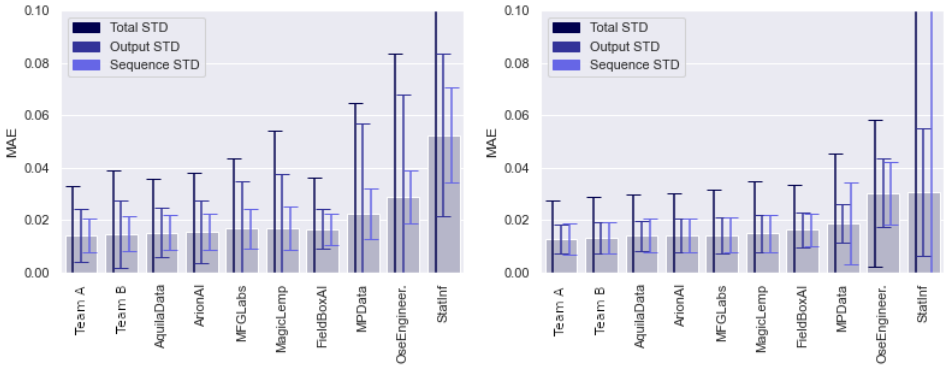


Figure 8.4 – **Average MAE** on all sequences and all output variables, on *feedback phase* (left) and *final phase* (right). Standard deviations shown are **Total STD** for STD computed on all scores (all pairs (*output, sequence*)); **Output STD** for average scores grouped by output variables; **Sequence STD** for average scores grouped by flight sequences.

flight sequences from 186 flight records. During the *final test*, they were trained on the same data, but tested on 7,398 other test flight sequences from the same flights used for testing in the feedback phase. With an average sequence length of 49, it resulted in around 230,000 data points in the feedback evaluation and around 350,000 data points in the final evaluation, for each variable. Altogether, there were $\simeq 70GB$ of data. The volume of data presented a major difficulty to be handled by the participants. Typically, it was not possible to load all the data at once in live memory (only 16 GB RAM available). Such volume of data guaranteed a robust evaluation.

The final score was computed as the average across all output variables and test sequences. For a more nuanced analysis and to determine error bars, we calculated the standard deviation over these different factors. Figure 8.4 displays these results. The variance across flight sequences is less than the variance across output variables. This aligns with the findings detailed in Chapter 4, considering that sequences is a higher granularity. As anticipated by the law of total variance, the overall variance is the most pronounced.

8.2.4 . Ranking function ✔

As mentioned previously, the leaderboard scores were obtained by averaging the MAE across all output variables and flight sequences. However, to refine the analysis and try to separate the participants' performance, we applied the *relative difference* method, presented in Chapter 5. Interestingly, in regression tasks, ranking functions can be applied by considering each test sample as a judge.

Relative difference is a pairwise ranking function, were two candidates i and j are compared across {output, sequence} pair k (judge) using the following

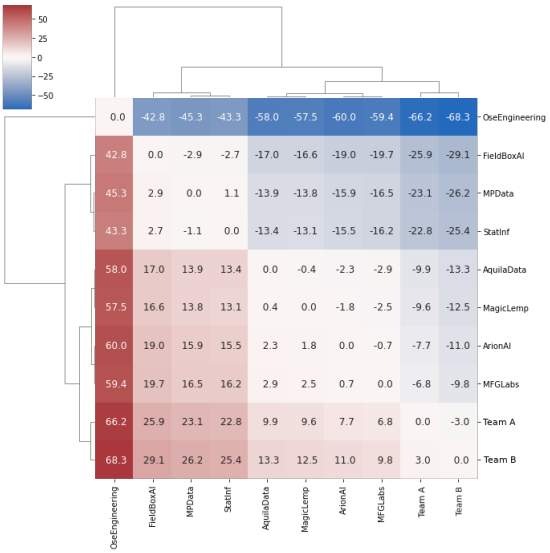


Figure 8.5 – Relative differences in MAE w_{ij} over all 471033 {output, sequence} pairs in phase 2 (in percent). Positive values mean that method i is better than method j on average (i is the line index and j the column index). Lines and columns have been grouped using a hierarchical clustering algorithm.

formula, computing w_{ij} for all pairs of candidates :

$$w_{ij} = \sum_{k \in Y} 2 \frac{MAE(j,k) - MAE(i,k)}{MAE(j,k) + MAE(i,k)}$$

We show the result table in (Figure 8.5), where i is the line index and j the column index. We distinguish three clusters and a singleton (*Ose Engineering*). We notice a pair of very similar methods *Team A* and *Team B* performing best and a nearby a cluster of 4 other gradient boosted trees. We have another cluster of 3 neural networks performing worse than all gradient boosted trees, and in last position, *Ose Engineering*. More information about the methods used by each team is given in Table 8.1.

8.2.5 . Computation ✔

The competition was employing code submission coupled with containerized environments. A crucial aspect of the setup was allow long training times and avoid limitations due to computational resources.

To this end, 10 compute workers were provisioned, one designated for each team. Each of these compute workers was robustly equipped with an NVIDIA RTX 2080Ti GPU, 4 vCPUs operating at 2.60 GHz, and 16 GB of DDR4 RAM. While participants were restricted to making only one submission at a time, there was no time limit imposed on the duration or processing of that submission. This provided them the flexibility to fine-tune their models and approaches without the pressure of a ticking clock. On the other hand, the

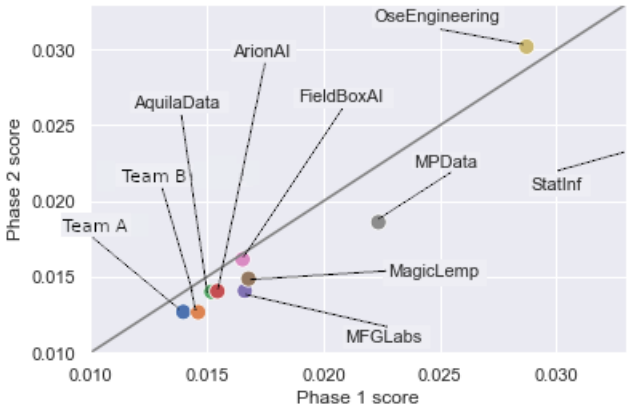


Figure 8.6 – Comparison of performance in the *feedback* phase and *final phase* of PRAIC 2020. The scores shown are *mean absolute error*. The phases are well correlated, indicating no sign of overfitting.

memory was quite constrained, as the training dataset could not fit the 16GB RAM, inherently forcing participants to manage the memory usage.

We can see in the results, Table 8.1, that the total duration vary a lot from methods, ranging from around 3 hours to almost a day. The speed and simplicity of the solutions may have been a determinant factor in for the jury evaluating the submissions.

8.2.6 . Splitting into phases ✔

As usual, the competition was staged into two phases : the *feedback phase* and the *final phase*. The objective of the *feedback phase* was to let participants develop their model, and the objective of the *final test phase* was to evaluate the code submitted by each participant on fresh data, not used to develop their solution. Only one final submission per participant was allowed.

Figure 8.6 compares the performances of the participants in the *feedback phase* and the *final phase*. There is no sign of overfitting the *feedback phase*, as the participants even obtained slightly better scores in the *final phase*. This is coherent with the claim made in Chapter 6 that overfitting the *feedback phase* is not common in practice.

8.2.7 . Participation and filtering of candidates ✘

Ten teams were pre-selected to participate in the competition. Eligibility required that these teams either be research organizations or companies (preferably startups), already established in the Paris region or with plans to relocate there. The reward for the winning team was a research grant amounting to **$500,000**, coupled with an opportunity to continue the project's development in collaboration with Dassault-Aviation. Beyond the results displayed on the leaderboard, the selection of the winner was also determined by a jury,

who evaluated a scientific dossier submitted by the candidates.

The substantial reward, coupled with the coaching from Dassault-Aviation engineers, provided significant motivation for teams, encouraging them to participate with seriousness and efficiency. These incentives ensured the quality and commitment of participants. On the other hand, the exclusive nature of the competition inherently limited the number of participants. Given its closed format, a broad crowdsourcing approach involving a larger pool of candidates was not possible. Furthermore, due to the already limited number of participants, there was no scope to apply a meaningful filtering of the candidates, and all teams entered the final phase.

8.2.8 . Conclusive results ✔

The results are presented in Table 8.1, in order of performance in the final phase, best comes first. The participants tested three types of methods :

- **Linear models** inspired by ARIMAX models (Hyndman and Athanasopoulos, 2021).
- **Deep-learning** feed-forward Convolutional Neural Networks (CNN), regular Multi-Layer Perceptrons (MLP), and Recurrent Neural Networks (RNN) (typically Long Short-Term Memory (LSTM)) (Sarker, 2021).
- **Gradient Boosted Tree** (GBT) regressors (CatBoost, XGBoost, LightGBM), incorporating the time component via feature engineering (Friedman, 2001; Chen and Guestrin, 2016).

We indicate in boldface in Table 8.1 the preferred method of the participants. All six top ranking teams all used gradient boosted tree (GBT) methods in their final solution. These models therefore appear to be best suited for time series regression, as confirmed by Xu et al. (2021). If we assumed that teams were selecting at random GBT *vs.* other methods, there would be $1/2^6 = 1.5\%$ chance to get all 6 first teams to select GBT methods, therefore this result is significant.

As it can be seen in Table 8.1 and Figure 8.4, the first seven teams got a similar performance, impossible to significantly distinguish. The final winner was selected by a jury, taking into account scientific methodology and future plans described by each team in a report. Another way to break the ties could have been to rank the solutions on their training and prediction time (see Section 4.3.1). Although the top-ranked team's code required 23 hours to execute, the team in third place needed just 2.65 hours. This efficiency suggests that the third team's solution might be more appropriate for production environments, a crucial consideration for applications in embedded computing.

The task difficulty was within reach of the participants, but the performances of the participants were very close, thus the methods could not be well differentiated. When participants in a challenge achieve scores that fall within the same error bar range, it points to a potential limitation in the chal-

Team	Score phase 1	Score phase 2	Main methods	Duration (seconds)	Duration (hours)
Team A	0.01_{399}	0.01_{269}	**CatBoost**, multiple regression	83045	23.07
Team B	0.01_{462}	0.01_{274}	**XGBoost**	70031	19.45
Aquila-Supméca	0.01_{519}	0.01_{402}	**LightGBM**, RF, Ridge, MLP	9551	2.65
Arion AI	0.01_{545}	0.01_{406}	**XGBoost**, regular regression	30243	8.40
MFG Labs	0.01_{662}	0.01_{408}	**CatBoost**, Tabnet (attention), TCN	58699	16.30
Magic LEMP	0.01_{678}	0.01_{488}	Cascade **XGBoost**, LSTM, CNN	64392	17.89
FieldBox.ai	0.01_{653}	0.01_{616}	**MLP** CNN	12335	3.43
MP Data	0.02_{236}	0.01_{861}	CNN, GNN, attention, LSTM	59312	16.47
Ose Engineering	0.02_{871}	0.03_{019}	CNN, LSTM	21250	5.90
Statinf	0.05_{243}	0.03_{053}	**MLP**, LSTM, RF	38119	10.59

Table 8.1 – **MAE performances in both phases.** The order is given by final phase scores, best is first. The score differences between participants being not statistically significant, the ranking is based upon digits of lesser significance, indicated as subscripts. We indicate also the approaches used by each team and the duration of execution of their final entry, which is the duration of training and prediction. All runs are under 1 day of calculations, way under the time budget limitation of 5 days.

lenge's design or the nature of the task itself. While it is encouraging that the task was accessible and within the capabilities of the participants, it limits the insights and utility of the challenge.

The closed nature of the challenge, restricted to only 10 pre-selected teams, also introduced limitations. Such a constraint potentially limits the diversity of approaches and methodologies. Open challenges, being publicly available, usually attract a broader range of participants, leading to a wide array of unique perspectives and techniques. Pre-selection can inadvertently introduce biases, which might skew the results and not genuinely represent the broader capabilities of the community. On the bright side, the challenge strongly confirmed the efficiency of GBT methods in multivariate time series regression, even with the specificity of the task, such as the asynchronous nature of the time series.

8.3 . Learning to Run a Power Network

8.3.1 . General presentation

In this section, we propose a comparative analysis of two distinct editions of the *Learning to Run a Power Network* (L2RPN) challenge, namely : *L2RPN 2022 - Energies of the Future and Carbon Neutrality* and *L2RPN 2023 - The Paris Region AI Challenge for Energy Transition*. The former, an open competition accepted at the World Congress on Computational Intelligence (WCCI), set the stage for the latter, which took place as a closed competition organized in collaboration with Réseau de Transport d'Electricité (RTE) and the Paris Region Ile-de-France. The L2RPN 2023 challenge escalates in ambition, introducing a larger dataset, and integrating features such as online agent retraining. Additionally, it necessitates the creation of "assistant" models by the participants, aiming

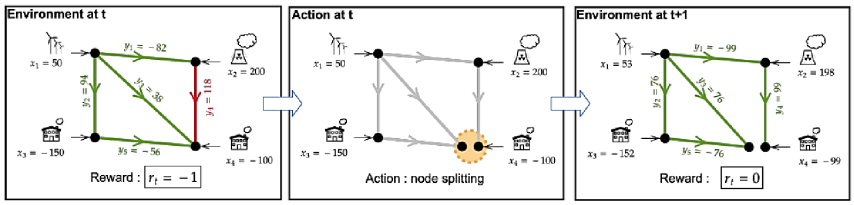


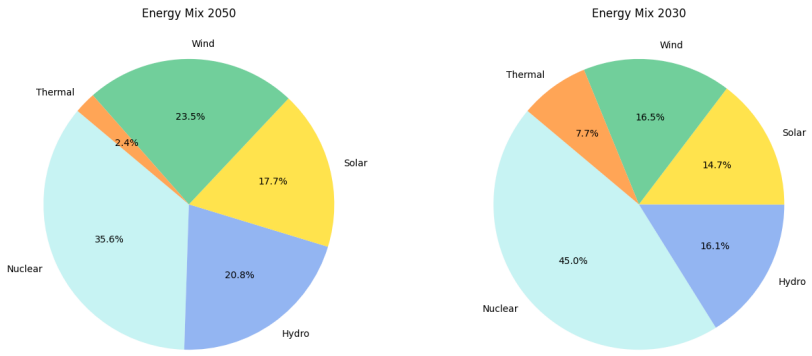
Figure 8.7 – Power system operation : The task of dispatchers is to monitor the power network and make eventual changes to ensure safe network operations with no line overflow. If in the environment at time t (left) a line is overflowing (indicated in red), a corrective action may be taken (center), such as a "node splitting", resulting in restored "power network safety" in the environment at time $t + 1$ (right). Borrowed from (Marot et al., 2021).

at improving the explainability of the employed techniques.

L2RPN (Marot et al., 2019) is a series of research challenges focused on developing machine learning algorithms that can optimize the operation of a power network, organized in partnership with Réseau de Transport d'Electricité (RTE). The goal of the L2RPN challenge is to develop algorithms that can learn how to operate a power network in a way that is efficient, reliable, and resilient. In the L2RPN challenge, participants are given a simulation of a power network and are asked to build an algorithm that can control the operation of the network in order to meet specified objectives. These objectives may include, for example, minimizing the cost of operating the network, maximizing the reliability of the network, or minimizing the environmental impact of the network. Figure 8.7 gives an overview of the agent's task. The L2RPN series can be characterized as reinforcement learning (RL) challenges, though the suggested algorithms are not strictly limited to RL approaches. RL challenge design is discussed in Section 7.5.

In the 2022 edition, *Energies of the Future and Carbon Neutrality*, the simulated energy grid features mainly renewable and nuclear energies as mean of production, in accordance to scenarios proposed by RTE (France, 2021) to reach carbon neutrality by 2050. Indeed, the Paris region, Ile-de-France, has expressed particularly strong concern regarding the means of energy production, and proposed as milestones to reach 100% renewable and zero carbon emissions by 2050 (Pécresse, Valérie and Conseil régional d'Ile-de-France, 2018). The energy mix used in this new edition of the competition features less than 3% of electricity being generated by fossil fuels. This competition targets the real-world problem of ensuring the safety of power networks, considering a large proportion of renewable energies and several batteries, with a focus on real-time operations. The challenge design and the results of this edition are detailed in the competition report [5].

5. https://medium.com/@adrienpavao/learning-to-run-a-power-network-



(a) L2RPN 2022 (b) L2RPN 2023

Figure 8.8 – Energy mix over a year. *L2RPN 2022* aims at simulating a year 2050 energy mix while *L2RPN 2023* aims at simulating a year 2030 energy mix. "Thermal" represents the proportion of fossil fuels.

The 2023 edition follows a similar idea and intends to simulate a year 2023 energy mix. The energy mixes of the two competitions are compared in Figure 8.8.

The L2RPN competition requires a library that can simulate a power system within a reinforcement learning framework. To meet this requirement, RTE has developed Grid2Op (Donnot, 2020), a Python module that converts the operational decision-making process into a Markov Decision Process (S, A, P_a, R_a) (Bellman, 1957). This module discretizes the time of a scenario into a list of states, each corresponding to a 5-minute time step. For instance, a one-day long scenario would be divided into $\frac{24\times60}{5} = 288$ time steps. Using this system, given the current state $s_t \in S$ and action $a_t \in A$, Grid2Op can calculate the power flow (the amount of electricity flowing on each power line) at the next time step s_{t+1}. For this, the data of energy production and consumption at time t are needed.

Indeed, this RL challenge protocol is not free of data; in order to make Grid2Op work, time series describing the electricity injections into the power network, referred to as *chronics*, must be generated. These chronics take into account the amount of electricity injected into the network by generators, loads, and batteries. In order to generate these chronics, data about the architecture of the power network, weather, consumption, and generators (such as their types and maximum production) are required. The Chronix2Grid library (Marot et al., 2020), created by RTE, uses this data to generate chronics,

`with-renewable-energies-a-challenge-design-and-analysis-e4ffbe05f22`

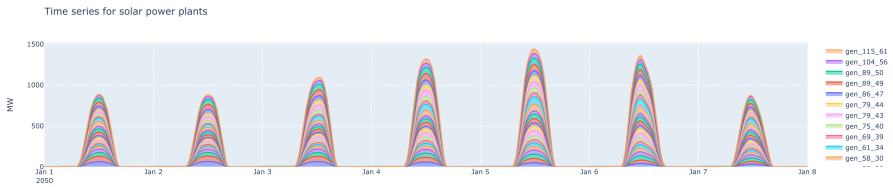


Figure 8.9 – Example of time series representing the energy produced by each solar power plant at each time step. Each day is characterized by an energy peak. The colors represent the different generators.

an example of which can be seen in Figure 8.9.

8.3.2 . Scoring metric ✔

To rank the participants, a score function is needed to quantify their agent's performance. In L2RPN 2022, the score function was designed as the average of three cost functions, calculated over test scenarios :

- **Energy Losses Cost** : determined by multiplying the electricity lost due to the Joule effect by the current price per MWh.
- **Operation Cost** : the sum of expenses incurred by the agent's actions. Changes in electricity production have a cost that varies based on the energy market, while using batteries has a fixed cost per MWh.
- **Blackout Cost** : in cases where the agent failed to manage the power network until the end of the scenario, this cost is calculated by multiplying the remaining electricity to be supplied by the current price per MWh.

The cost of a blackout, as expected, is significantly higher than the other two costs. This means that an agent who successfully completes a scenario will always have a higher score than one who does not, even if their actions are less expensive. Additionally, our score function is normalized to be maximized and falls within the range of $[-100, 100]$. A score of 0 indicates an agent who takes no actions at each time step, while a positive score is considered a good result.

L2RPN 2023 uses a similar approach to measure performance, however the function is balanced with two additional scores :

- **Low Carbon Score** : the percentage of non-curtailed renewable energy. This score favors renewable energy utilization and lower carbon emissions.
- **Assistant Score** : accuracy in predicting and alerting potential attacks. Indeed, the agents are asked to raise alerts when they predict disruptions, such as grid instabilities or natural disasters. The assistant score measure how well the agent predicts such events.

By introducing these supplementary scores, the L2RPN 2023 challenge adds layers of complexity to the task, aligning with the objectives of promoting

renewable energy usage and enhancing grid stability. Interestingly, a connection can be made between the assistant score and the calibration score presented in Section 4.2.2, as both measure the accuracy of the model's level of confidence in its own abilities.

It should be noted that, in reinforcement learning, there is no universal, one-size-fits-all metric for evaluating performance. Rather, the evaluation metric must be carefully crafted to suit the unique characteristics and objectives of each specific problem. This objective metric was carefully designed by experts from RTE.

8.3.3 . Size of test set and significance ✔

The test data for the L2RPN 2022 challenge represents one year (52 scenarios), while L2RPN 2023 extend it to four years (208 scenarios). Each scenario in these challenges represents a week-long period with a time interval of five minutes between each step, resulting in a total of 2016 timesteps per scenario. The significant increase in the number of scenarios from L2RPN 2022 to L2RPN 2023 enhances the robustness of the evaluation process. The standard deviation across the different scenarios can be visualized as error bars in Figure 8.10.

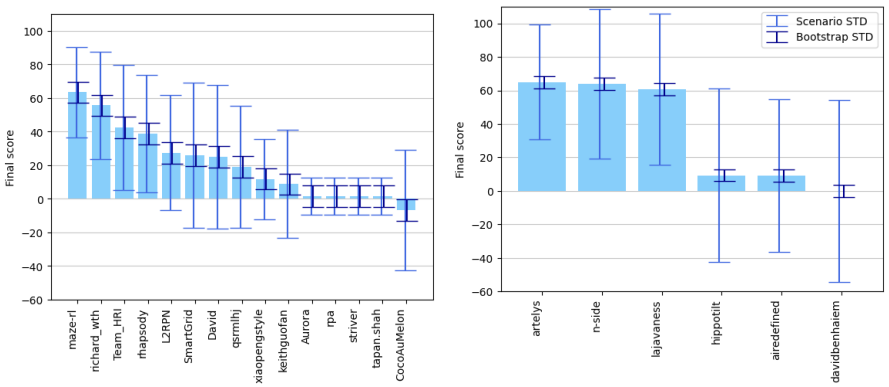
In order to explore the variance of the scores, two distinct types of error bars. The "scenario STD" represents the standard deviation of the scores across different test scenarios, highlighting the variation in performance from one scenario to another. On the other hand, the "bootstrap STD" is calculated from the standard deviation of scores obtained by performing bootstrap sampling of the set of scenarios, offering an insight into the inherent variability of the total score.

Notably, as the number of scenarios increases, as seen in the transition from L2RPN 2022 to L2RPN 2023, the "bootstrap STD" tends to decrease. This trend underscores the benefit of having a larger set of scenarios for a more stable and reliable evaluation of the participants' performance. In the other hand, the "scenario STD" is very high in both competitions.

Despite the more robust evaluation, it is observed that the top 3 participants in L2RPN 2023 were not distinctly differentiated. This intense competitiveness among the leading teams could be attributed to the substantial reward at stake, motivating the participants to push the boundaries of their performance. This led to a tight competition, showcasing the participants' skills and the effectiveness of their approaches in handling the complex challenges posed by the L2RPN competitions.

8.3.4 . Ranking function ✔

To aggregate the scores achieved across different scenarios and obtain the leaderboard, a straightforward arithmetic mean is employed. However, the application of the *average rank* function instead of the *mean* to rank the



(a) L2RPN 2022 (b) L2RPN 2023

Figure 8.10 – Main *final phase* results of L2RPN 2022 (left) and L2RPN 2023 (right). "L2RPN" is the baseline provided by the organizers. scenario STD is the deviation computed between all scenarios and bootstrap STD is computed on repeated bootstraps of scenarios.

participants across scenarios yields different outcomes, as outlined in Figure 8.11.

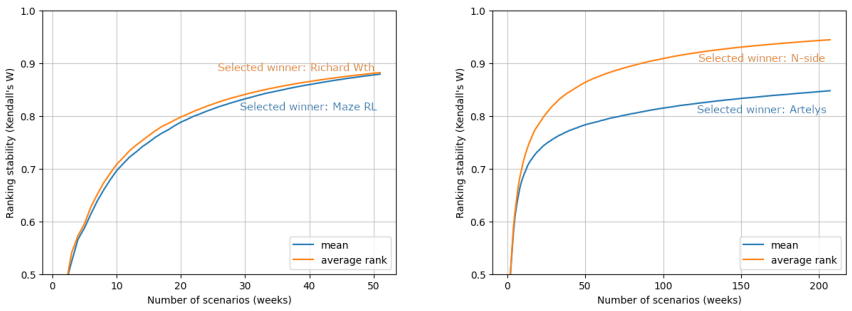
In the context of L2RPN 2022, *Richard Wth*, originally positioned third, ascends to the top spot when using *average rank*, surpassing *Maze RL*. Similarly, for L2RPN 2023, *N-Side* team, initially in second place, is ranked first by *average rank*. This shift in rankings implies that, despite possessing a lower mean score, these agents consistently outperform the winning team across a majority of individual scenarios.

It is also worth noting that the *average rank* method exhibits greater stability compared to the *mean* approach, corroborating the findings from experiments discussed in Chapter 5. Interestingly, this enhanced stability is not evident in the L2RPN 2022 results. The disparities observed could potentially be attributed to factors such as the number of participants and the diversity of scenarios presented in each competition. Specifically, the L2RPN 2023 competition, with a higher number of scenarios, showcases an improvement in ranking stability, underscoring the impact of these factors on the robustness of the evaluation methodology.

8.3.5 . Computation ✔

Both competitions were evaluated using code submissions within containerized environments, ensuring consistency and reproducibility. The competitions allowed and actively promoted the pre-training of models. This was facilitated by a downloadable validation environment made available to the participants.

In the L2RPN 2022 competition, the rules permitted participants to make



(a) L2RPN 2022 (b) L2RPN 2023

Figure 8.11 – Evolution of the ranking stability in relation to the number of scenarios used in the evaluation. In L2RPN 2023, *average rank* is more stable than *mean*. These two ranking functions do not select the same teams as winners.

a maximum of 20 submissions per day, with each submission subjected to an execution time limit of approximately 2 hours. To accommodate the computational demands, a pool of workers was made available, with each worker configured with 8 vCPUs and 16GB of RAM.

The L2RPN 2023 competition imposed a restriction of one parallel submission per participating team, with no specific time constraint. Essentially, this allocation meant that each team had access to a dedicated compute worker for their exclusive use. The specifications for these workers included 4 vCPUs, 16GB of RAM, and a NVIDIA Tesla T4 GPU.

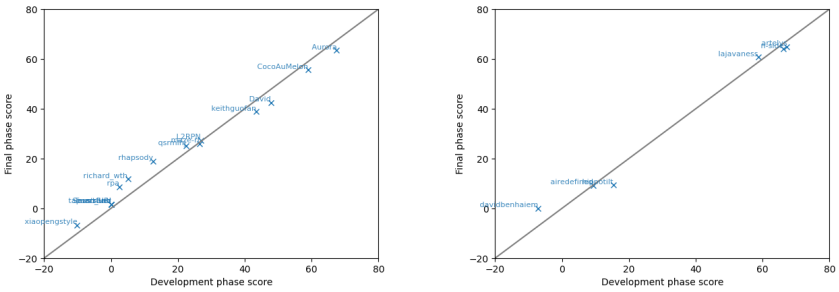
Despite the computational intensity of the competition, the provided resources were sufficient, especially given the limited number of participants. No team reported a lack of computing power, ensuring a smooth and efficient evaluation process for all involved.

8.3.6 . Splitting into phases ✔

As mentioned earlier, the operation of the environment requires the input of time series data, which detail the electricity injections into the power network. Consequently, we can generate different years of chronics – time series data of energy production and consumption – that are identically and independently distributed. These chronics can then be employed in distinct phases of competition, typically a *feedback phase* and a *final phase*. The results of the participants on the two phases were remarkably correlated, with no sign of overfitting the *development phase*, as shown in Figure 8.12, underscoring the effectiveness of the competition design in evaluating the generalizability and robustness of the participants' solutions.

8.3.7 . Participation and filtering of candidates ✘

The L2RPN 2022 challenge attracted a total of **16 candidates**, competing for a prize pool of **5000 EUR** where the first place was rewarded with 2500



(a) L2RPN 2022 (b) L2RPN 2023

Figure 8.12 – Comparison of performance in the *feedback* phase and *final phase*. The scores shown are the custom reward function. The phases are well correlated, indicating no sign of overfitting.

EUR, the second place with 1500 EUR, and the third place with 1000 EUR.

Moving forward to the L2RPN 2023 edition, the competition saw a significant increase in the grand prize, amounting to a **500 000 EUR** grant for the winner, a substantial incentive provided with the support of the Paris region [6]. This edition had **7 participating teams**, pre-selected, and required to be either operating or planning to initiate operations in the Paris region, to qualify for the grand prize.

It is noteworthy that there was no filtering process between the *feedback phase* and the *final phase* in both editions, allowing all participants to progress through the competition without elimination. While the number of participants did not allow for a meaningful filtering of participants, it helped to provide enough computing power to process all candidates' submissions.

8.3.8 . Conclusive results ✔

Only the last submission of each participant, the submissions available on the leaderboard, are taken into account in this analysis. The scores of the final phases of both competitions are summed up in the Figure 8.10.

The top-performing teams in L2RPN 2022 employed diverse methodologies and agents, showcasing the complexity and variety of solutions for this problem. The winning team, *Maze RL*, used an AlphaZero-based approach combined with a contingency-aware controller, benefiting from their prior domain knowledge in congestion management for the energy sector (Dorfer et al., 2022). The second-ranked team, *Richard Wth*, opted for a brute-force search and optimization method, making use of a single-step agent and solving the DC Optimal Power Flow problem, despite having no prior domain knowledge. *Team HRI*, ranking third, utilized a simpler approach by drawing

6. https://www.iledefrance.fr/toutes-les-actualites/entreprises-et-chercheurs-participez-au-challenge-ia-pour-la-transition-energetique

random actions and redirecting currents when necessary, building on their previous experience in energy management systems. From this analysis, it can be concluded that the winning teams used different approaches and methodologies to solve this problem of automatic management of the electricity grid. Furthermore, it's interesting to see that there's a diverse range of techniques like AlphaZero, brute force search and optimization, and reinforcement learning were used by the top-performing teams, showing the complexity and diversity of the problem.

In L2RPN 2023, the winner has yet to be announced at the time of this writing. The selection process extends beyond the leaderboard rankings, as in PRAIC 2020. An independent jury is set to deliberate and select the winner after a thorough evaluation of the scientific dossiers submitted by each participating team. Three teams have emerged with remarkable performances, statistically indistinguishable from one another : *Artelys*, *N-side*, and *La Javaness*. Interestingly, a common thread among these leading teams is their adoption of a multi-agent or modular approach to solve the challenge. This methodology demonstrates a strategic decomposition of the complex problem into manageable sub-tasks, each handled by specialized agents or modules. *Artelys* has crafted their agent through an union of a topological agent, a redispatch agent, and a suite of heuristics. Similarly, *La Javaness* solution uses optimization techniques and heuristic strategies. Their agent comprises modules like *Dynamic Topology Optimization* and *Alert Module*. *N-side* has deployed a diverse ensemble of agents, each designed to address specific tasks crucial for the optimal operation of the power grid. This ensemble includes agents such as the *Line Reconnection Agent*, *Node Reconfiguration Agent*, and various others.

The parallel between these teams' strategies and their successful performances underscores the efficacy of a modular approach in tackling the intricate task of power grid operation. It highlights the importance of specialization and coordination among different components of the agent to achieve optimal results. These results will be further analyzed in a near future to produce a challenge analysis.

8.4 . Conclusion

In this chapter, we presented and analyzed the design of three distinct competitions, AutoDL, Paris Region AI 2020, and L2RPN 2022 and 2023, underscoring various methodological considerations. An overview of the answers to those questions is given in Table 8.2. Each challenge presented its own unique outcomes, offering valuable lessons that enrich our understanding of effective competition design. A notable strength common to all three was the use of code submission, which added a layer of rigor to evaluation process, allowing blind testing and enhancing reproducibility. A common thread is the

Question	AutoDL	PRAIC 2020	L2RPN 2022 and 2023
1. Specificity	Blind-testing Multiple data types	Confidential data Asynchronous data	Reinforcement learning Chronics data
2. Scoring metric	✔	✔	✔
3. Significant evaluation	✔	✔	✔
4. Ranking function	✔	✔	✔
5. Computation	✔	✔	✔
6. Splitting into phases	✔	✔	✔
7. Participation and filtering	✔	✘	✘
8. Conclusive results	✔	✔	✔

Table 8.2 – **General answers to the 8 methodological questions.** Green checkmarks indicate positive answers, orange checkmarks partial answers, and red crosses negative answer. For details, refer to the corresponding sections.

design of metrics by domain experts, underscoring the importance of designing evaluation metrics in relation the specificity of each domain. Overally, we confirmed the statement that participant overfitting in the *feedback phase* (public leaderboard) is not common. Nonetheless, maintaining a two-phase design is recommended to continue promoting solutions that generalize without overfitting (Blum and Hardt, 2015). However, this analysis also acknowledged certain shortcomings, such as the consequences of Paris Region AI Challenge's limited level of difficulty and lower number of participants. With these lessons in mind, we are better equipped to design challenges that are not only engaging and innovative but also methodologically sound.

9 - Discussions and conclusion

Machine learning competitions gained interest as a mean for crowdsourcing problems. Such settings allow for a diverse set of strategies to be proposed, enabling the most effective method to naturally emerge from the group, rather than simply settling for a single approach. Furthermore, competitions also provide a clear separation between the individuals who present the problems and those who develop the solutions, which helps to prevent any inherent bias that may influence the problem-solving and evaluation processes. Studying and improving the methodology of machine learning competitions is essential due to the field's early stage. Therefore, improving methodologies in machine learning challenges is a crucial step for the field's growth and ethical use.

The main benefit of this protocol is its ability to provide a unified evaluation process for all candidate models, irrespective of their authors, thereby enabling a fair benchmarking of the problem. Additionally, this approach allows for efficient problem-solving through crowdsourcing, while relieving participants of the burden of data and task preparation. Moreover, it is an efficient way to make benchmarks systematic and reproducible.

9.1 . Main contributions

Throughout this research, the methodology of machine learning competitions has been studied, both from a design and evaluative standpoint. The following key conclusions can be made :

Competition hosting tools

In Chapter 3, the emphasis is on the need for open-source platforms that can host competitions efficiently. To that end, we develop and maintain *CodaLab Competitions* (Pavao et al., 2023a) and *Codabench* (Xu et al., 2022). These platforms, by enabling code submissions and providing fixed environments, ensure a standardized evaluation process. Key features are the flexibility in the design of scoring procedures, and the ability to allow staged competitions. Data analysis reveals the wide applicability of competitions across a vast variety of fields, such as linguistics, medicine, biology, agriculture, multimedia, and more. The number of participants is shown to be a key success factor. Interestingly, the reward is not the primary attraction for many participants.

Metrics for evaluation

Chapter 4 offers a categorization of evaluation metrics, covering performance, ethics and societal impact, resource utilization, and evaluator view-

points (human or model based evaluation) (Pavao et al., 2023b). It is clear that real-world constraints must be a part of any evaluation, as they ensure that the results are practical and applicable. To ensure statistically reliable evaluations, it is recommended to have a consequent test set and to use bootstrap methods to assess error bars, given their computational efficiency and accuracy. The required number of test set samples grows quadratically with the mean error, the standard deviation of error rates and the confidence, while it grows exponentially with the targeted precision level.

Techniques for ranking

Chapter 5 brings forward a methodological framework for assessing empirically the stability and generalization of ranking functions on real data. On past benchmarks data, the average rank method showed its efficiency (Pavao et al., 2021a), while remaining simple to compute and to interpret. In competition protocols involving multiple datasets or tasks, the ranking stability can be estimated through repeated trials, introducing perturbations along both the candidate and judge axes. Conducting this analysis is advisable to guarantee the robustness of the evaluation.

Judging from a meta-learning viewpoint

Chapter 6 proposes seeing competitions as meta-learning problems. Having a distinct final phase prevents potential overfitting. Filtering participants accessing the final phase improve the generalization of the winner selection (Pavao et al., 2022b). To enhance the relevance of results, as a rule of thumb, only participants that exceed a predefined baseline should be allowed into this phase.

Specific protocols and design

Chapter 7 details various experimental setups used in machine learning competitions : AutoML, metalearning, confidential data use, causal discovery, performance estimation, and adversarial challenges (Pavao et al., 2023b). The utility of artificial data is highlighted, especially in situations with no clear ground truth or where data confidentiality is a concern. Adversarial challenges also offer a way to conduct competitions on tasks that lack clear metrics or ground truth. The adaptability of competition designs is also showcased, with each challenge requiring its own unique focus.

Insights from organized challenges

Chapter 8 shares experiences from real-world challenges that were organized, namely AutoDL (Liu et al., 2021a), Airplane Numerical Twin (Pavao et al., 2021b) and L2RPN (Marot et al., 2021). The consistent theme is the importance of code submissions, ensuring result reproducibility and a fair evaluation. Another common theme is the need for metrics designed by domain experts. Ho-

wever, it is worth noting potential pitfalls when the challenge difficulty is not calibrated correctly or when participation is low.

In essence, machine learning competitions serve as practical platforms for technological discovery and robust evaluative methodologies. As the machine learning field progresses, ensuring fairness and consistency in competition design and evaluation is vital. The observations from this research aim to inform and guide researchers and practitioners in future machine learning competition and benchmark organizations.

9.2 . Research perspectives

In this section, we describe topics and problems for future research.

Ranking on samples

In Chapter 5, an in-depth analysis of ranking functions is presented, emphasizing how commonly used functions, such as *mean*, *median*, and *average rank*, produce varied outcomes in terms of stability and generalization. This study, primarily focused on algorithm comparisons across tasks, hints at the potential of extending the analysis for cases where judges $\mathcal{J}$ represent individual test sample scores. Traditional scoring metrics like MAE or ROC AUC average over the entire test set, possibly overlooking nuances. Preliminary results indicate that leveraging the relative ranking of algorithms, i.e. considering how often one outperforms another, might enhance ranking stability, especially when there is a high correlation between candidate algorithms. These relative metrics use order as added information, contrasting with simple averages that discard this detail. In scenarios like regression, where detailed scores for data points are available, these methods are particularly promising. The essence is to shift from assessing absolute performance to more comparative evaluations, minimizing the impact of outliers. With predictions or data samples acting as judges, the high-dimensionality of the score matrix suggests that learning-to-rank techniques might outperform traditional statistical methods.

Exploring probabilistic ranking functions

Further work in the exploration of ranking functions holds significant potential, particularly probabilistic ranking functions. In our study, we present deterministic methods such as *mean*, *median*, *average rank*, and pairwise methods, notably the *Copeland's method*. However, a promising direction involves modifying these deterministic methods with probabilistic significance tests. For instance, in Copeland's method, rather than considering a straightforward pairwise comparison, a victory could be attributed only if candidate $\mathbf{u}$ significantly surpasses candidate $\mathbf{v}$ (using null hypothesis statistical testing or Baye-

sian analysis). This approach would effectively prune the Condorcet graph, representing the pairwise wins and losses, likely leading to more consistent and robust ranking outcomes, and minimizing the influence of threshold effects and marginal victories which may not be statistically meaningful.

Relationship between stacking and crowdsourcing

Future research could probe the interplay between stacking and crowdsourcing in machine learning. Stacking, a method of blending multiple model predictions through a "meta-model", is often utilized in competitions to leverage the strengths of various algorithms for superior outcomes. On the other hand, "crowd wisdom" is the idea that collective judgments, free from individual biases, often yield more accurate results, evident in platforms like Wikipedia or Stack Exchange. In machine learning competitions, participants' diverse techniques echo both model stacking and crowd wisdom's aggregated perspectives. Post-competition discussions often lead to solution enhancements by integrating top methods. Just as crowd wisdom offsets individual biases, stacking in machine learning can mitigate overfitting, thus boosting accuracy. However, the utility of stacking is debated due to the complications it introduces, such as increased system complexity and potential data dependencies (Roberts et al., 2014).

Exploring new ways of distributing prizes

Future research should consider alternative methods for prize distribution in competitions. Instead of rewarding only the top performers, an "all-pay contest" model could distribute prizes to any participant who surpasses a certain baseline, potentially increasing participation. However, this might dilute motivation for top performers. Prizes could be uniform or performance-based, with the latter requiring careful consideration of the scaling factor. Issues to address include collaboration among participants, cheating prevention, and logistical distribution concerns. A comparative methodology is crucial to evaluate different prize models, factoring in participation rates and individual effort.

Integrating challenge design and resolution in education

Incorporating challenges in the classroom presents a promising avenue for future educational programs, enriching the learning experience and bridging the gap between theory and practice. The practice of graduate students at University Paris-Saclay in creating challenges end-to-end, and subsequently deploying these challenges for undergraduate students to solve, exemplifies a hands-on approach that encourages practical skills and team collaboration (Pavao et al., 2019). This model not only enhances students' understanding of machine learning concepts but also cultivates a community of future data scientists and challenge organizers, emphasizing the importance of sound

data science methodology and the adoption of best practices.

9.3 . General conclusion

In recent years, the use of data science challenges has grown and has proven to be an effective protocol for addressing complex scientific problems. This approach was employed in countless fields and recognized as a valuable tool in advancing research. The evolution of machine learning leads toward studying the field as an experimental science, implying the need for solid methodology to compare algorithms. Crowd-sourced competitions and benchmarks offer a powerful framework in this regard, conveying multiple advantages, such as avoiding the inventor-evaluator bias.

While theoretical frameworks, such as the no-free-lunch theorem stating that no machine learning algorithm universally outperforms random guessing, and Arrow's impossibility theorem which asserts that no voting system can perfectly capture voters' preferences, often paint a pessimistic picture, our empirical studies provide a more optimistic outlook, suggesting that satisfactory methodologies can indeed be identified.

This thesis, while centered on machine learning, introduces general methods to compare ranking functions, applicable across various evaluation systems and fields. The empirical approach offers a fresh perspective to the traditionally theory-focused Social Choice Theory. Our protocol could potentially reveal the inefficiencies of certain voting systems, such as the uninominal voting method, by highlighting their practical instability. Beyond political elections, the methodologies can be applied to any domain that involves ranking based on score matrices. Similarly, our finding that filtering participants in staged competitions enhances the winner selection is a general result. This concept is universally relevant to fields involving candidate qualification and selection.

Throughout human history, the rise of Artificial Intelligence represents a significant turning point, much like when we harnessed electricity or introduced motorization. Where we once automated physical labor, we are now aiming to automate thought and reasoning. As AI integrates into every field, it is quickly becoming a foundational element of our society, holding immense and powerful potential. However, with such potential comes the need for rigorous scrutiny. Our research into the methodology of machine learning competitions and benchmarks has underscored the intricate complexities of evaluating and comparing algorithms. Particularly, a critical challenge we face for future benchmarks is the quantization of ill-defined and nuanced objectives, such as fairness, ethical alignment, or interpretability. Defining precise metrics, establishing objective standards, and ranking algorithms have emerged as critical components in this journey. Especially in an era where huge mo-

dels, e.g. large language models, challenge the boundaries of conventional evaluation, the importance of systematic and scientific benchmarking cannot be overstated. It is only through this methodical evaluation that we can truly understand the strengths, limitations, and the multifaceted nature of AI models. As AI continues to reshape our world, it becomes of prime importance that we utilize this powerful tool with responsibility, ensuring that our pursuit of progress is always supported by a commitment to integrity, clarity, and scientific rigor.

10 - Appendix

10.1 . Utility and difficulty metrics

We present here two related metrics that can be used to judge challenges : **crowdsourcing utility** and **difficulty**. The utility represents how much the winner of the challenge made a significant improvement in comparison to the crowd (the pool of candidates). The difficulty compares the scores of the baseline, the best performing candidate, and the best theoretically possible score, to estimate the potential range of improvement and the difficulty proposed by a challenge. We and provide a theoretical interpretation on how the utility scales with the number of participants.

10.1.1 . Utility

The efficiency of crowdsourcing comes from the sheer number of contributors ; that is, the crowd. The idea is straightforward : ask one person to solve a problem, and you'll obtain one solution. Ask a hundred persons to solve the same problem, and you'll obtain a hundred of solutions. Assuming these solutions exhibit variable quality, the likelihood of obtaining a high-quality solution significantly increases due to the expanded pool of solutions. This concept can be formalized within a simplified framework where the quality of solutions is quantifiable by a singular value, similar to the use of a single scoring metric in machine learning competitions. Consequently, a competition can be seen as a set of scores, one for each participant. We denote the set of candidates, or the crowd, $\mathcal{C}$. If we assume the scores to follow a normal distribution $N(\mu, \sigma)$, the excepted score of a single participant is μ. This situation of having only a single participant can be thought as the absence of crowdsourcing, the absence of competition. When the participant pool enlarges to n members, the top performer's score is likely to surpass the average. We call the coefficient of augmentation between the mean and the highest score the **crowdsourcing utility**. It can be interpreted as measuring the value added by crowdsourcing, its usefulness, as the mean is the excepted score from a single participant, while the highest represents the score of the competition winner.

Presuming a normal distribution $N(\mu, \sigma)$, and n participants, we can approximate the expected maximum score E_{max} :

$$E_{max} \approx \mu + \sigma\sqrt{2ln(n)}$$

E_{max} is the winner's excepted score. This approximation is based on the properties of the Gumbel distribution [1]. This results come from the extreme

1. https://en.wikipedia.org/wiki/Gumbel_distribution

value theory, which concerns the study of the extreme deviations from the median of probability distributions. The Fisher–Tippett–Gnedenko theorem (or the *extreme value theorem*) shows that the only possible types of limiting distribution for properly normalized maxima of a sequence of i.i.d. random variables are the Gumbel, Frechet, and Weibull families (Basrak, 2011). If the original variables are normally distributed, the appropriate distribution is the Gumbel distribution.

The Gumbel distribution has the following cumulative distribution function :

$$F(x) = \exp\left(-\exp\left(-\frac{x-\mu}{\beta}\right)\right)$$

where μ is the location parameter and β is the scale parameter. It can be shown that for a standard normal distribution, the appropriate values are $\mu = \gamma$ and $\beta = 1$, where γ is the Euler-Mascheroni constant, approximately 0.577. The expected value (mean) of a Gumbel distribution is $\mu + \beta\gamma$. For maxima, these parameters of the Gumbel distribution are effectively approximated by fitting the Gumbel distribution to the normal distribution. The scaling factor $\sqrt{2ln(n)}$ for the standard deviation comes from this approximation. We finally extend this approximation to normal random variables with arbitrary mean and standard deviation. The precision of the approximation depends on the specific parameters of the normal distribution and the sample size.

As defined earlier, the theoretical excepted crowdsourcing utility E_U is then the ratio between E_{max} and the mean of the distribution :

$$E_U = \frac{E_{max}}{\mu}$$

$$E_U \approx 1 + \frac{\sigma\sqrt{2ln(n)}}{\mu}$$

Just as with the standard normal distribution, calculating the exact variance of the maximum of a sample of i.i.d. normal variables is quite difficult and generally does not have a simple closed form solution. We can estimate using simulations, as in Figure 10.1.

In practice, μ is estimated by the average score s_{mean} and E_{max} is the score of the winner s_{best}. We use these values to estimate the empirical crowdsourcing utility U_C of the crowd C. We simply want this measure to represent the coefficient of augmentation between the average and the best score. For scores bounded in $[0, 1]$, such as the accuracy score, we have :

$$U_C = \frac{s_{best}}{s_{mean}}$$

To compute this metric for scores with different bounds, we can use the min-max normalization, using the bounds of the metric s_{min} and s_{max}.

$$U_C = \frac{\frac{s_{best}}{s_{mean}} - s_{min}}{s_{max} - s_{min}}$$

$$U_C = \frac{s_{best}}{s_{mean}(s_{max} - s_{min})} - \frac{s_{min}}{s_{max} - s_{min}}$$

$$U_C = \frac{s_{best} - s_{mean}s_{min}}{s_{mean}(s_{max} - s_{min})}$$

We can see that, if $s_{min} = 0$ and $s_{max} = 1$, we get back the initial formula. However, if the metric is semi-bounded, for instance the Mean Absolute Error (MAE) which ranges in $[0, +\infty]$, we need another trick to normalize and compute this coefficient. A possible solution is to use the best possible score s_{max} as the maximal bound, and the score obtained by a random model s_{random} as the minimal bound. Using this technique, we obtained this formula :

$$U_C = \frac{s_{best} - s_{mean}s_{random}}{s_{mean}(s_{max} - s_{random})}$$

If the score is to be minimized ("lower is better"), as it is for error rates or distances, we can transform s_{best} and s_{mean} in this way before the computation of the utility metric :

$$\tilde{s}_{best} = s_{max} + s_{min} - s_{best}$$

$$\tilde{s}_{mean} = s_{max} + s_{min} - s_{mean}$$

The theoretical approximation of E_U and the artificially simulated U_C are displayed in Figure 10.1, relatively to the number of participants n (the size of the crowd $|C|$) ranging between 1 and 140. The approximation of the expected utility E_U predicted by the formula for a normal distribution $N(0.5, 0.1)$ (blue curve). Such distribution can reasonably represents a competition leaderboard. We then compute empirically the average and standard deviation of the crowdsourcing utility U_C by sampling n points and computing the formula (orange curve).

Unsurprisingly, having only 1 candidate gives an utility of 1, which is the minimum possible crowdsourcing utility, corresponding to no crowdsourcing at all. More candidates directly leads to better solution, hence the curves are monotonically increasing. However, it is a logarithmic scaling law, which results in having a tremendous number of participants (thousands) is not necessary to significantly increase the utility. With 100 participants, we can except to do

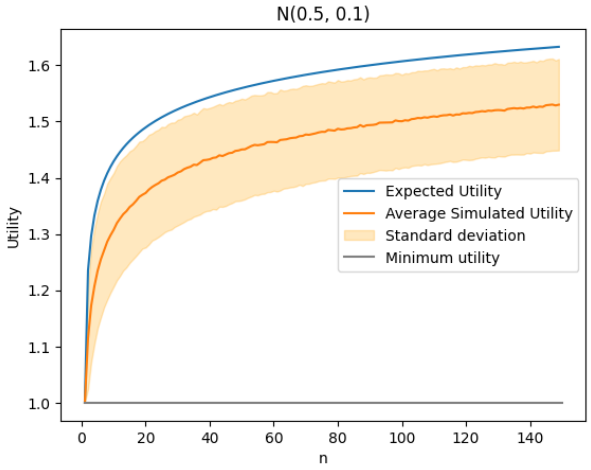


Figure 10.1 – Plot of the expected utility E_U, and the average crowdsourcing utility U_C simulated on $10,000$ trials for n in $[1, 140]$. Both theoretical and empirical approximations are computed given the normal distribution $N(0.5, 0.1)$.

around 1.5 times better than without crowdsourcing with such normal distribution. In practice, the participants performance may follow more complex - and possibly skewed - distributions. Section 3.3 showcases histograms that illustrate the varied distributions of scores observed in actual competitions (see Figure 3.8). Another limitation of this measure, in practice, is that it may behave in different ways depending on the nature of the scoring metrics used to output the scores. Typically, semi-bounded metrics naturally leads to greater utility values then metrics bounded between 0 and 1.

10.1.2 . Difficulty

The notions of intrinsic difficulty and modelling difficulty were introduced by Guyon et al. (2019a), to define the difficulty of a machine learning task to be solved in the context of a challenge.

The **intrinsic difficulty** is the difference between the max possible score s_{max} (the bound of the metric) and the score obtained by the best model discovered s_{best}. As the name suggest, it represents how hard the task is to solve.

$$D_{intrinsic} = |s_{max} - s_{best}|$$

The **modelling difficulty** is the difference between the random baseline score s_{random} and the best model score s_{best}. We interpret the modelling difficulty as the range of potential improvement that can be reached by selecting an efficient model to solve the task.

$$D_{modelling} = |s_{best} - s_{random}|$$

Guyon et al. (2019a) actually proposed to use the score of a baseline, instead of a random model, but the selection of a decent baseline is vague. Using the score of a random model therefore leads to a more simple and robust definition of the modelling difficulty. The metrics $D_{intrinsic}$ and $D_{modeling}$ are defined as absolute differences, as to generalize to both instances of scoring paradigms, where a higher score is more desirable and where a lower score signifies better performance.

A greater modelling difficulty implies a wider difference between the performance of the best achievable model and the most basic one. This means that a greater modelling difficulty leads to more variance in the scores obtained by the candidates, hence increasing the crowdsourcing utility U_C. This increase can be quite substantial, as we previously demonstrated that the variance in the distribution of scores acts as a multiplicative coefficient of the excepted utility. Therefore, it is recommended to estimate these difficulty measures during the organization phase of a competition. Organizers should aim to present a problem that allows for a broad spectrum of possible scores, thereby optimizing the competition's outcomes.

10.2 . Ranking functions and correlation measures

10.2.1 . Proofs

All the theoretical results presented in the paper were found in the literature, with the exception of the criteria associated with the *success rate* and *relative difference* methods. In this section we provide the corresponding proofs for these criteria.

Criteria satisfied by success rate method

(1) Majority : *Success rate* does **not** satisfy majority criterion. Table 10.1 is a counter-example.

	j_1	j_2	j_3
A	1	1	0
B	0.8	0.8	1
C	0.6	0.6	0.6

Table 10.1 – This is a score matrix exposing a counter example, showing that *Success rate* does not satisfy the majority criterion. A is ranked first by a majority of judges but its average success rate is the same as B : $\frac{2}{3}$.

(2) Condorcet : *Success rate* does **not** satisfy Condorcet criterion, as implied by the fact it does not satisfy majority criterion (Majority ∈ Condorcet).

(3) Consistency : *Success rate* meets consistency criterion and **(4) participation criterion**. A ranked system is consistent and meets participation criterion if it is a scoring function (i.e. positional system), as proven by Young (1975b). Success rate is positional as adding a judge improve the score of the candidates according the their position in the ranking of this judge.

(5) LIIA : *Success rate* does **not** satisfy LIIA. Table 10.2 is a counter example.

	j_1	j_2	j_3
A	0.6	0.6	0
B	0.4	0.4	1
C	1	0	0.4

Table 10.2 – This is a score matrix exposing a counter example, showing that *Success rate* does not satisfy the LIIA criterion. If there is only A and B then A wins. If we add C then it is a tie.

(6) IIA : *Success rate* does **not** satisfy IIA, as implied by the fact it is not LIIA (LIIA $\in$ IIA).

(7) Clone-proof : *Success rate* is **not** clone-proof. Table 10.3 is a counter-example.

	A	B	C	Average
A	-	0.6	0.4	0.5
B	0.4	-	0.5	0.45
C	0.6	0.5	-	0.55

Table 10.3 – This is a pairwise success rate table exposing a counter example, showing that *success rate* is not clone-proof. If we repeatedly duplicate the candidate B, A will end up in front of C.

Criteria satisfied by relative difference method

(1) Majority : *Relative difference* does **not** satisfy majority criterion. A counter example was found empirically, with a majority rate $\neq 1$.

(2) Condorcet : *Relative difference* does **not** satisfy Condorcet criterion. A counter example was found empirically, with a Condorcet rate $\neq 1$. It is also implied by the fact that this method does not satisfy majority criterion (Majority $\in$ Condorcet).

(3) Consistency : *Relative difference* meets consistency.

Let's suppose that :

1. $\texttt{rank}\Big(f(X)\Big) = \texttt{rank}\Big(f(Y)\Big)$
2. $\texttt{rank}\Big(f([X\ Y])\Big) \neq \texttt{rank}\Big(f(X)\Big)$

Without loss of generality, let's assume that the candidate ranked first is not the same in $\texttt{rank}\Big(f(X)\Big)$ and $\texttt{rank}\Big(f([X\ Y])\Big)$, respectively at index α and β.

Then :

$$\texttt{rank}\Big(f([X\ Y])\Big)_\alpha > \texttt{rank}\Big(f([X\ Y])\Big)_\beta$$

$$\implies f([X\ Y])_\alpha < f([X\ Y])_\beta$$

From the definition of f :

$$\sum_{j\neq\alpha}\frac{1}{m}\sum_{k=1}^{m}Q_k^{\alpha,j} < \sum_{j\neq\beta}\frac{1}{m}\sum_{k=1}^{m}Q_k^{\beta,j}$$

where $Q_k^{i,j} = \frac{c_{ik}-c_{jk}}{c_{ik}+c_{jk}}$

$$\implies \frac{1}{m}\sum_{k=1}^{m}\widetilde{Q}_k^{\alpha} < \frac{1}{m}\sum_{k=1}^{m}\widetilde{Q}_k^{\beta}$$

where $\widetilde{Q}_k^{i} = \sum_{j\neq i}Q_k^{i,j}$

$$\implies \sum_{k=1}^{m}\widetilde{Q}_k^{\alpha} < \sum_{k=1}^{m}\widetilde{Q}_k^{\beta}$$

$$\sum_{k\in\mathcal{J}_X}\widetilde{Q}_k^{\alpha} + \sum_{k\in\mathcal{J}_Y}\widetilde{Q}_k^{\alpha} < \sum_{k\in\mathcal{J}_X}\widetilde{Q}_k^{\beta} + \sum_{k\in\mathcal{J}_Y}\widetilde{Q}_k^{\beta}$$

This is **impossible** because $\texttt{rank}\Big(f(X)\Big) = \texttt{rank}\Big(f(Y)\Big)$ which implies that $\sum_{k\in\mathcal{J}_X}\widetilde{Q}_k^{\alpha} > \sum_{k\in\mathcal{J}_X}\widetilde{Q}_k^{\beta}$ and $\sum_{k\in\mathcal{J}_Y}\widetilde{Q}_k^{\alpha} > \sum_{k\in\mathcal{J}_Y}\widetilde{Q}_k^{\beta}$ (and $\forall a,b,c,d\in\mathbb{R}$, if $a>c$ and $b>d$, then $(a+b)>(c+d)$).

In conclusion, if $\texttt{rank}\Big(f(X)\Big) = \texttt{rank}\Big(f(Y)\Big)$, then $\texttt{rank}\Big(f([X\ Y])\Big) = \texttt{rank}\Big(f(X)\Big) = \texttt{rank}\Big(f(Y)\Big)$

(4) Participation : *Relative difference* meets participation criterion. The final score given to a candidate can be expressed as the mean of the scores that this candidate obtained on all judges. Suppose we have a judge **j** and a score matrix X, with $f(X)_u > f(X)_v$, and $j_u > j_v$.

By following the same reasoning as for consistency, we have :

$$f([X\ \mathbf{j}]) = \sum_{k\in\mathcal{J}_X}\widetilde{Q}_k + \widetilde{Q}_j$$

Also, we have :

$$\texttt{rank}\Big(f(\mathbf{j})\Big) = \texttt{rank}(\mathbf{j})$$

$$\implies \texttt{rank}\left(\widetilde{Q}_j\right) = \texttt{rank}(\mathbf{j})$$

We can deduce :

$$f([X\ \mathbf{j}])_u > f([X\ \mathbf{j}])_v$$

Therefore, a judge which prefers a candidate $\mathbf{u}$ against another candidate $\mathbf{v}$ can only improve the position of $\mathbf{u}$ relatively to $\mathbf{v}$.

(5) LIIA : *Relative Difference* does **not** satisfy LIIA. Table 10.4 is a counter example (same as for *success rate* above).

	j_1	j_2	j_3
A	0.6	0.6	0
B	0.4	0.4	1
C	1	0	0.4

Table 10.4 – This is a score matrix exposing a counter example. If there is only A and B then A wins. If we add C then it is a tie.

(6) IIA : *Relative difference* does **not** satisfy IIA, as implied by the fact it is not LIIA (LIIA $\in$ IIA).

(7) Clone-proof : *Relative difference* is **not** clone-proof. Table 10.5 is a counter example.

	j_1	j_2	j_3
A	0.6	0.6	0.4
B	0.5	0.4	0.6
C	0.5	0.7	0.5

Table 10.5 – This is a score matrix exposing a counter example. If we repeatedly duplicate the candidate B, A will end up in front of C using the relative difference method.

10.2.2 . Distance, correlation and concordance measures

In this section we define distance, correlation and concordance measures invoked in the chapter.

Kendall's rank correlation coefficient

Intuitively, Kendall's τ measures a correlation linked to the number of "neighbor swaps" needed to transform $\mathbf{j}$ into $\mathbf{j}'$. Its definition involves all possible pair of observations $(\mathbf{j}_c, \mathbf{j}'_c)$ and $(\mathbf{j}_{c'}, \mathbf{j}'_{c'})$. The two pairs are said to be *concordant* if both judges $\mathbf{j}$ and $\mathbf{j}'$ agrees on their order of the candidates c and c', otherwise they are *discordant*. Kendall's τ is defined as following (Kumar and Vassilvitskii, 2010) :

$$\tau = \frac{n_c - n_d}{\binom{n}{2}}$$

where n_c and n_d represent the number of concordant pairs and discordant pairs respectively.

The actual version we use is the τ_b which accounts for ties and have a more complex definition (Agresti, 2010) :

$$\tau_b = \frac{n_c - n_d}{\sqrt{(\binom{n}{2} - n_1)(\binom{n}{2} - n_2)}}$$

where $n_1 = \sum_i \frac{t_i(t_i-1)}{2}$ and $n_2 = \sum_{i'} \frac{u_{i'}(u_{i'}-1)}{2}$ with t_i the number of tied values in the i^{th} group of ties for the first quantity and $u_{i'}$ the number of tied values in the j^{th} group of ties for the second quantity.

In practice, Spearman's ρ and Kendall's τ seem to be correlated.

Kendall's W

To compute the concordance, we proposed to compute the mean Spearman's ρ of all possible pairs of rankings. However, this algorithm's complexity is exponential $O(2^n)$ with the number of judges. To avoid the problem of complexity, we can compute the concordance using Kendall's W statistics (Kendall and Smith, 1939). In practice, we use a more recent version of Kendall's W accounting for ties (Siegel and Castellan, 1988).

$$W(M) = \frac{12 \sum_{i=1}^{n} (R_i - \bar{R})^2}{m^2(n^3 - n)}$$

With R_i being the total rank of candidate i on all judges :

$$R_i = \sum_{j=1}^{m} r_{i,j}$$

And $\bar{R}$ being the mean value of all total ranks :

$$\bar{R} = \frac{1}{n} \sum_{i=1}^{n} R_i$$

It has been shown in Kendall and Gibbons (1990b) that W is linearly related to $\bar{r_s}$, the mean value of the Spearman's rank correlation coefficients between all $\binom{m}{2}$ possible pairs of rankings between judges. Here is the relation between Kendall's W and the average Spearman's ρ between all possible pairs of judges :

$$\bar{r_s} = \frac{mW - 1}{m - 1}$$

We have published our implementation of Kendall's W and its second version accounting for ties in the Python Package Ranky (Pavao, 2020), dedicated to ranking methods and measures.

Spearman distance

The Spearman rank correlation coefficient ρ is defined as following :

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

where d_i is the difference between the ranks of corresponding values in the two data sets and n is the number of data points.

The value $1 - \rho$ can be used as a measure of dissimilarity, or Spearman distance. Therefore, the formula is given by :

$$d_\rho = \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

Euclidean distance

Given two points $\mathbf{x} = (x_1, x_2, \dots, x_n)$ and $\mathbf{y} = (y_1, y_2, \dots, y_n)$ in an n-dimensional space, the Euclidean distance d between them is :

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2}$$

10.2.3 . Datasets-Algorithms matrices

The Datasets-Algorithms matrices used in the experiments are represented using heatmaps with hierarchical clustering in Figure 10.2.

10.3 . Theoretical analysis of top-k algorithm

To formally compare the top-k method with the vanilla method of selecting the winner of the *final phase*, we use our synthetic example obtained by swapping neighbors, starting from an ideal true ranking. We first formalize the problem as maximizing the probability of finding the winner with the top-k method (Section 10.3.1). We then decompose the probability of finding the winner with the top-k method (Section 10.3.2), which we call method "accuracy" $acc(k)$, in two factors playing the role of training accuracy and generalization gap. We compute the accuracy of the vanilla method and show that it is equal to $acc(k = 1) = acc(n)$, for n participants (Section 10.3.3). Finally, we prove that, under the condition that $n = 3$, $acc(k)$ goes through an optimum as $k = 2$ (Section 10.3.4). While the proof may not be universally applicable for all $n \in \mathbb{N}$, the subsequent analysis provides insight into the effect highlighted in Chapter 6.

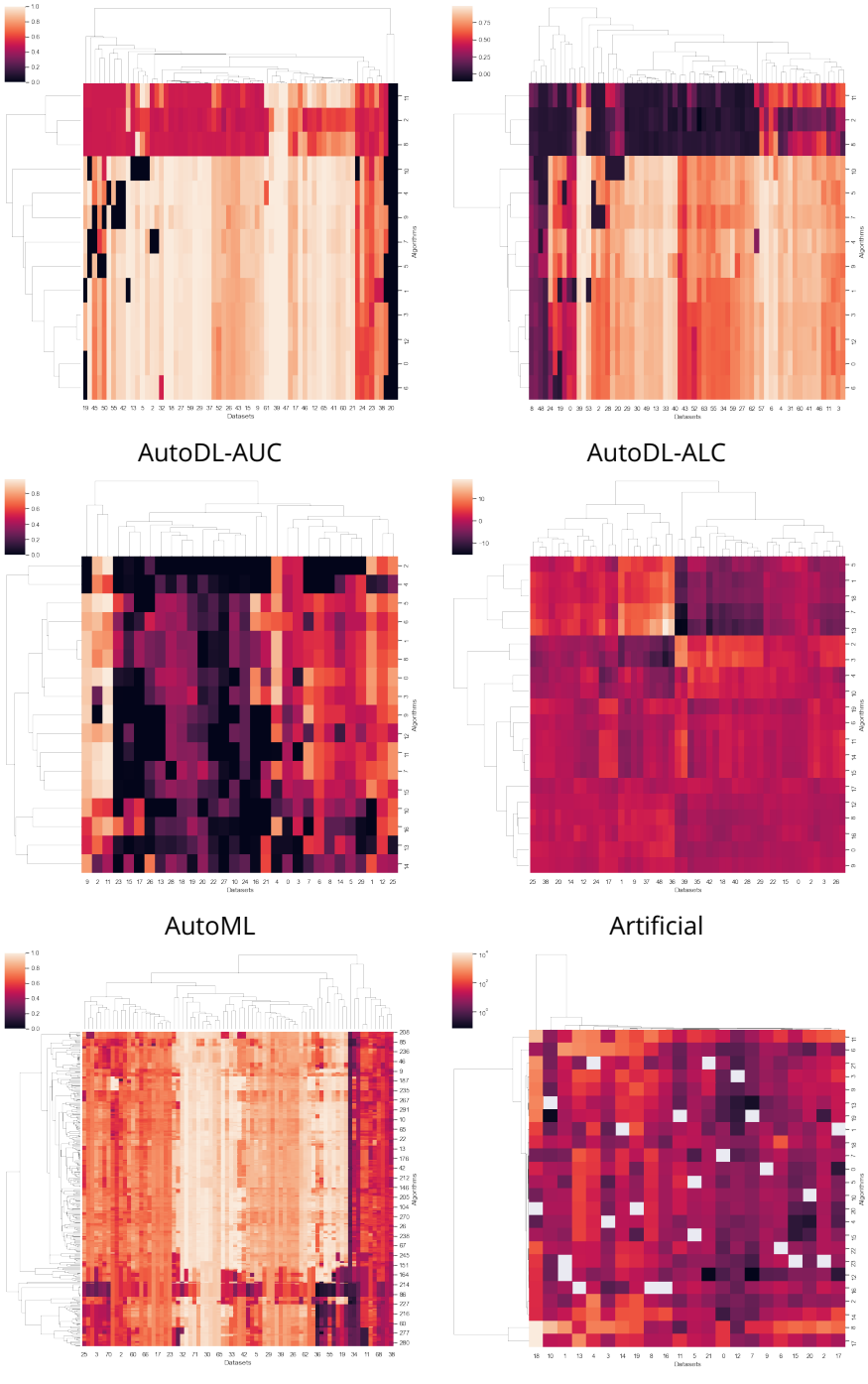


Figure 10.2 – Heatmap with hierarchical clustering for the 6 benchmarks' score matrices.

10.3.1 . Problem definition

We consider a competition with $n \in \mathbb{N}$ participants. We name the n participants as $\{1, 2, ..., n\}$ where their name corresponds to their true but unknown ranking :

$$g = [1, 2, ..., n]$$

An empirical ranking obtained in a competition phase is assumed to be obtained from g by repeated permutations of pairs of neighbors. A position i is drawn at random from $\{1, \cdots, n-1\}$ and the participants i and $i+1$ are inverted. We repeat this operation s times. The smaller s, the more the empirical rankings will be correlated to the true ranking.

We call D the random variable (RV) corresponding to a ranking drawn from the previously described process, for the *development phase* and F the RV corresponding the that of the *final phase*, drawn similarly.

The degree of correlation between D and F is governed by $\phi = \frac{s}{n}$. In practice, we estimate the correlation by computing the Kendall τ distance d between D and F. This approach is interesting as it applicable in real-world scenarios.

The participant i^* selected by the top-k method has rank j^* :

$$j^* = \arg\min_{j \leq k} F^{-1}(D(j)).$$

Using :

$$D(j) = i$$

$$D^{-1}(i) = j$$

we get :

$$i^* = \arg\min_{D^{-1}(i) \leq k} F^{-1}(i).$$

The choice of the top-k method is the true winner *iff* $i^* = 1$.

The objective is to find an optimal value $k*$ that maximize the probability $acc(k)$ that the declared winner is the *real winner*, using the top-$k(D, F)$ method. The problem is formalized as follows :

$$k^* = \arg\max_k acc(k)$$

with :

$$\boxed{acc(k) = \texttt{Proba}[\arg\min_{D^{-1}(i) \leq k} F^{-1}(i) = 1]} \tag{10.1}$$

10.3.2 . Calculation of acc(k=n) (vanilla method)

We first evaluate the value of the first and the last point of the curve $acc(k = 1)$ and $acc(k = n)$, corresponding to the *vanilla* method. If $k = 1$, we select the winner of the *development phase* as our winning candidate. If $k = n$, we select the winner of the *final phase*.

$$acc(k = 1) = \texttt{Proba}[D^{-1}(1) = 1]$$

$$acc(k = n) = \texttt{Proba}[F^{-1}(1) = 1]$$

The probability that this is the true winner is identical in both cases since the processes to generate D and F are identical :

$$acc(k = 1) = acc(k = n) = \texttt{Proba}[D^{-1}(1) = 1] = \texttt{Proba}[F^{-1}(1) = 1]$$

We can notice that $D^{-1}(1) = 1$ occurs if the real winner (that is the participant with first position in g) does not move in the s swapping trials, or if it moves forward then backward to return to its original position.

We can therefore model the movements of the candidate by a Markov chain with each state representing a possible position, and a transition matrix T as defined in Figure 10.3.

$$T = \begin{pmatrix} 1-p & p & 0 & 0 & ... & 0 \\ p & 1-2p & p & 0 & ... & 0 \\ 0 & p & 1-2p & p & ... & 0 \\ ... & ... & ... & ... & ... & p \\ 0 & 0 & 0 & 0 & p & 1-p \end{pmatrix}$$

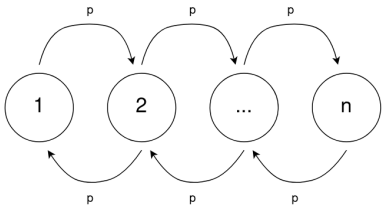


Figure 10.3 – Definition of the transition matrix (left) and schema of the Markov chain (right). $p = \frac{1}{(n-1)}$. The probabilities of staying in the same state are not shown for simplicity.

$P_{ij}(s)$, the probability to reach the position j from the position i after s steps on the Markov chain, can be computed using a matrix power, according to the Chapman-Kolmogorov equation :

$$P_{ij}(s) = (T^s)_{ij}$$

Considering that the probability of being involved in a swap at each time step is $\frac{1}{n-1}$ at the bounds and $\frac{2}{n-1}$ for any other nodes, the probability of transition to a different state is given by $p = \frac{1}{n-1}$. The probability that the winner stays the winner (going from 1 to 1) after s swaps is given by :

$$\boxed{acc(k=1) = acc(k=n) = P_{11}(s)}$$

More generally we can compute the probability of ending in a position j for any candidate i, after s swaps, using :

$$\texttt{Proba}[D^{-1}(i) = j] = P_{ij}(s) \tag{10.2}$$

10.3.3 . Decomposition of acc(k)

We decompose Equation 10.1 into two factors : the probability that the candidate selected is the true winner selected, when we know that the true winner is in the top-k of the *development phase*, and the probability that the the true winner is in the top-k. Finding the winner in the top-k of the *development phase* can be written as $D^{-1}(1) \leq k$, hence :

$$\boxed{\mathbf{acc(k)} = \texttt{Proba}[\arg \min_{\mathbf{D^{-1}(i) \leq k}} \mathbf{F^{-1}(i) = 1 \mid D^{-1}(1) \leq k}] \ \texttt{Proba}[\mathbf{D^{-1}(1) \leq k}]} \tag{10.3}$$

The probabilities given by Equation 10.2 can be added to obtain the probability of ending in a set of positions (e.g. between 0 and k). Thus, the probability $\texttt{Proba}[D^{-1}(1) \leq k]$ that the true winner is in D top-k is given by :

$$\begin{aligned}
P_{topk} &= \texttt{Proba}[D^{-1}(1) \leq k] \\
P_{topk} &= \texttt{Proba}[D^{-1}(1) = 1] + \texttt{Proba}[D^{-1}(1) = 2] + ... + \texttt{Proba}[D^{-1}(1) = k] \\
P_{topk} &= P_{11}(s) + P_{12}(s) + ... + P_{1k}(s) \\
P_{topk} &= \sum_{j=1}^{k} P_{1j}(s)
\end{aligned}$$

To approximate the probability that the true winner is selected when we know it is in the top-k of D, we compute and add together : (1) the probability that the true winner is ends up first (in F); (2) Then, for each other candidates c, their probability of **not** being in D top-k ($1 - \texttt{Proba}[D^{-1}(c) \leq k]$) multiplied by their probability of ending up first ($\texttt{Proba}[D^{-1}(c) = 1]$). As an approximation and for simplicity, the probabilities of these events are computed independently.

$$\begin{aligned}
P_k &= \texttt{Proba}[\arg \min_{D^{-1}(i) \leq k} F^{-1}(i) = 1 \mid D^{-1}(1) \leq k] \\
P_k &\approx P[D^{-1}(1) = 1] + \sum_{i=1}^{n} \Big(1 - P[D^{-1}(i) \leq k]\Big) \times P[D^{-1}(i) = 1] \\
P_k &\approx P_{11}(s) + \sum_{i=1}^{n} \sum_{j=1}^{k} \big(1 - P_{ij}(s)\big) \times P_{i1}(s)
\end{aligned}$$

All together, after factorization :

$$acc(k) \approx \sum_{j=1}^{k} P_{1j}(s)\Big(P_{11}(s) + \sum_{i=1}^{n}\sum_{j=1}^{k}\big(1 - P_{ij}(s)\big) \times P_{i1}(s)\Big)$$

Or, using matrix power :

$$\boxed{P(k) \approx \sum_{j=1}^{k}(T^s)_{1j}\left((T^s)_{11} + \sum_{i=1}^{n}\sum_{j=1}^{k}(1 - (T^s)_{ij}) \times (T^s)_{i1}\right)} \tag{10.4}$$

10.3.4 . Proof that acc(k=2) ≥ acc(k=1) when n = 3

Theorem 4. *For $n = 3$, we have*

$$acc(k = 2) \geq acc(k = 1). \tag{10.5}$$

Démonstration. For $n = 3$.

We can represent the sampling process by the following graph (Figure 10.4), in which each vertex represents a possible order of the elements, and each edge represents a swap between two neighbors. It is undirected because swaps can always be inverted. It is also bipartite, as the states are grouped in two categories : the orders obtained after an even number of swaps from the initial state, and the orders obtained after an odd number of swaps from the initial state. The possible states are A, B, C for even numbers of swaps, D, E, F for odd numbers of swaps. A is the initial state before the swaps.

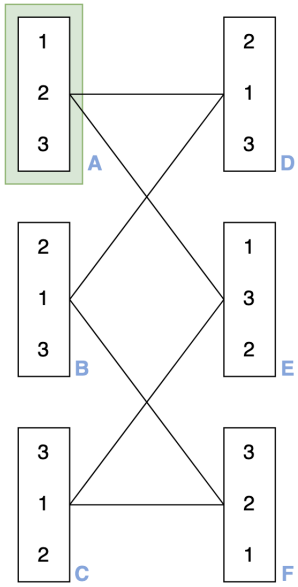


Figure 10.4 – Graph of the possible states of the ranking with $n = 3$. All transitions are bi-directional and have a probability $p = \frac{1}{2}$. The state A, highlighted in green, is the initial state.

We can associate a transition matrix T to this graph, in which all edges coming from one state are equiprobable, with probability $p = \frac{1}{n-1}$.

In our problem, we have a pair of independent rankings (D, F), representing the *development* and *final* phases. For each possible pair, the top-k method can either select the *real winner* or make a mistake. To show that $acc(k = 2) > acc(k = n)$, we need to show that the top-k method select the right winner in more cases with $k = 2$ than with $k = n$.

We can already established that show that $acc(k = 1) = acc(k = n)$ (Section 10.3.2).

With $n = 3$, top-1 and top-2 algorithms have the same outcome (both winning or both losing) for most pairs of rankings. In only four cases their outcomes are different : AB, CA, DE and EF. In the cases AB and EF, top-1 algorithm is performing better, while in the cases CA and DE it is top-2 which performs better.

As, in our problem definition, the rankings D and F are produced by the exact same number of swaps, we can consider separately the case with even number of swaps and the case with odd number of swaps. This means that we can compare separately the probabilities of AB and CA, and the probabilities of DE and EF.

$$P(AB) \leq P(CA)$$

$$P(DE) \geq P(EF)$$

Once again, D and F are independent. As a result, $P(XY) = P(X) \times P(Y)$, therefore we can simplify :

$$P(C) \geq P(B)$$

$$P(D) \geq P(F)$$

Using the Chapman-Kolmogorov equation :

$$P(X) = T_{AX}^{s}$$

with s the number of swaps and T_{AX} the probability of transitioning from the state A to the state X.

To solve this, we consider two sub-graph. On with the states {A, B, C} and another with the states {D, E, F}. Each have the following transition matrix :

$$T' = \begin{pmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{pmatrix}$$

We can apply the egein-decomposition of the matrix :

$$T' = QdQ^{-1}$$

$$T'^{s} = Qd^{s}Q^{-1}$$

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{pmatrix}^{s} = \begin{pmatrix} 1 & -1 & -1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{1}{4^s} & 0 \\ 0 & 0 & \frac{1}{4^s} \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} \end{pmatrix} \tag{10.6}$$

$$= \begin{pmatrix} 1 & -\frac{1}{4^s} & -\frac{1}{4^s} \\ 1 & \frac{1}{4^s} & 0 \\ 1 & 0 & \frac{1}{4^s} \end{pmatrix} \cdot \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{2}{3} \end{pmatrix} \tag{10.7}$$

$$= \begin{pmatrix} \frac{1}{3} + \frac{2 \times 4^{-s}}{3} & \frac{1}{3} - \frac{4^{-s}}{3} & \frac{1}{3} - \frac{4^{-s}}{3} \\ \frac{1}{3} - \frac{4^{-s}}{3} & \frac{1}{3} + \frac{2 \times 4^{-s}}{3} & \frac{1}{3} - \frac{4^{-s}}{3} \\ \frac{1}{3} - \frac{4^{-s}}{3} & \frac{1}{3} - \frac{4^{-s}}{3} & \frac{1}{3} + \frac{2 \times 4^{-s}}{3} \end{pmatrix} \tag{10.8}$$

We can see that :

$$P(B) = P(C)$$

So, for even number of swaps, $acc(k = 1) = acc(k = 2)$ (for $n = 3$).

To compute the actual transition matrix of the "odd" states $\{D, E, F\}$, taking into account the fact that the initial state is A, we need to dot product T' with the transition vector $[\frac{1}{2}, \frac{1}{2}, 0]$, corresponding to the probability of reaching D, E and F from the initial state A.

Therefore :

$$P(D) = \frac{1}{2}(\frac{1}{3} + \frac{2 \times 4^{-s}}{3} + \frac{1}{3} - \frac{4^{-s}}{3}) \tag{10.9}$$

$$= \frac{1}{2}(\frac{2}{3} + \frac{4^{-s}}{3}) \tag{10.10}$$

$$= \frac{1}{3} + \frac{2}{3}P(F) \qquad = \frac{1}{3} - \frac{4^{-s}}{3} \tag{10.11}$$

$$P(D) > P(F)$$

As a result, $acc(k = 2) \geq acc(k = 1)$.

□

Bibliographie

Kald Abdallah, Charles Hugh-Jones, Thea Norman, Stephen Friend, and Gustavo Stolovitzky. The prostate cancer dream challenge : a community-wide effort to use open clinical trial data for the quantitative prediction of outcomes in metastatic prostate cancer. *The oncologist*, 20(5) :459, 2015.

Claire Adam-Bourdarios, Glen Cowan, Cécile Germain, Isabelle Guyon, Balázs Kégl, and David Rousseau. The higgs boson machine learning challenge. In *Workshop on High-energy Physics and Machine Learning, HEPML 2014, held at NIPS 2014, Montreal, Quebec, Canada, December 8-13, 2014*, volume 42 of *JMLR Workshop and Conference Proceedings*, pages 19–55. JMLR.org, 2014. URL http://proceedings.mlr.press/v42/cowa14.html.

Claire Adam-Bourdarios, Glen Cowan, Cécile Germain, Isabelle Guyon, Balázs Kégl, and David Rousseau. How machine learning won the higgs boson challenge. In *24th European Symposium on Artificial Neural Networks, ESANN 2016, Bruges, Belgium, April 27-29, 2016*, 2016. URL http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2016-181.pdf.

A. Agresti. *Analysis of Ordinal Categorical Data*. New York : John Wiley & Sons, 2010. ISBN 978-0-470-08289-8.

Ahmed M. Alaa, Boris van Breugel, Evgeny S. Saveliev, and Mihaela van der Schaar. How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 290–306. PMLR, 2022. URL https://proceedings.mlr.press/v162/alaa22a.html.

Ron Amit and Ron Meir. Meta-learning by adjusting priors based on extended pac-bayes theory. In *International Conference on Machine Learning*, pages 205–214. PMLR, 2018.

Carolyn J. Anderson. *Central Limit Theorem*, pages 1–2. John Wiley & Sons, Ltd, 2010. ISBN 9780470479216. doi : https://doi.org/10.1002/9780470479216.corpsy0160. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470479216.corpsy0160.

Johan OL Andreasson, Michael R Gotrik, Michelle J Wu, Hannah K Wayment-Steele, Wipapat Kladwang, Fernando Portela, Roger Wellington-Oguri,

Eterna Participants, Rhiju Das, and William J Greenleaf. Crowdsourced rna design discovers diverse, reversible, efficient, self-contained molecular switches. *Proceedings of the National Academy of Sciences*, 119(18) : e2112979119, 2022.

Sam Ansari, Jean Binder, Stephanie Boue, Anselmo Di Fabio, William Hayes, Julia Hoeng, Anita Iskandar, Robin Kleiman, Raquel Norel, Bruce O'neel, et al. On crowd-verification of biological networks. *Bioinformatics and biology insights*, 7 :BBI–S12932, 2013.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017.

Kenneth J. Arrow. A difficulty in the concept of social welfare. *Journal of Political Economy*, 1950.

Nazem Atassi, James Berry, Amy Shui, Neta Zach, Alexander Sherman, Ervin Sinani, Jason Walker, Igor Katsovskiy, David Schoenfeld, Merit Cudkowicz, et al. The pro-act database : design, initial analyses, and predictive features. *Neurology*, 83(19) :1719–1725, 2014.

Monya Baker. 1,500 scientists lift the lid on reproducibility. *Nature*, 533(7604) : 452–454, 5 2016. ISSN 1476-4687. doi : 10.1038/533452a. URL https://doi.org/10.1038/533452a.

Mukesh Bansal, Jichen Yang, Charles Karan, Michael P Menden, James C Costello, Hao Tang, Guanghua Xiao, Yajuan Li, Jeffrey Allen, Rui Zhong, et al. A community computational challenge to predict the activity of pairs of compounds. *Nature biotechnology*, 32(12) :1213–1222, 2014.

Bojan Basrak. *Fisher-Tippett Theorem*, pages 525–526. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-04898-2. doi : 10.1007/978-3-642-04898-2_254. URL https://doi.org/10.1007/978-3-642-04898-2_254.

Adrian El Baz, Isabelle Guyon, Zhengying Liu, Jan N. van Rijn, Sébastien Treguer, and Joaquin Vanschoren. Advances in metadl : AAAI 2021 challenge and workshop. In Isabelle Guyon, Jan N. van Rijn, Sébastien Treguer, and Joaquin Vanschoren, editors, *AAAI Workshop on Meta-Learning and MetaDL Challenge, MetaDL@AAAI 2021, virtual, February 9, 2021*, volume 140 of *Proceedings of Machine Learning Research*, pages 1–16. PMLR, 2021. URL https://proceedings.mlr.press/v140/el-baz21a.html.

Adrian El Baz, Isabelle Guyon, Zhengying Liu, Jan van Rijn, Sebastien Treguer, and Joaquin Vanschoren. Advances in metadl : Aaai 2021 challenge and workshop. *arXiv preprint arXiv :2202.01890*, 2022.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment : An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47 :253–279, 2013.

Richard Bellman. A markovian decision process. *Indiana Univ. Math. J.*, 6 : 679–684, 1957. ISSN 0022-2518.

Asa Ben-Hur, André Elisseeff, and Isabelle Guyon. A stability based method for discovering structure in clustered data. In Russ B. Altman, A. Keith Dunker, Lawrence Hunter, and Teri E. Klein, editors, *Proceedings of the 7th Pacific Symposium on Biocomputing, PSB 2002, Lihue, Hawaii, USA, January 3-7, 2002*, pages 6–17, 2002. URL http://psb.stanford.edu/psb-online/proceedings/psb02/benhur.pdf.

Alessio Benavoli, Giorgio Corani, Janez Demsar, and Marco Zaffalon. Time for a change : a tutorial for comparing multiple classifiers through bayesian analysis. *J. Mach. Learn. Res.*, 18 :77 :1–77 :36, 2017. URL http://jmlr.org/papers/v18/16-305.html.

Yoshua Bengio and Yves Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *J. Mach. Learn. Res.*, 5 :1089–1105, 2004a. URL http://jmlr.org/papers/volume5/grandvalet04a/grandvalet04a.pdf.

Yoshua Bengio and Yves Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *J. Mach. Learn. Res.*, 5 :1089–1105, 2004b. URL http://www.jmlr.org/papers/volume5/grandvalet04a/grandvalet04a.pdf.

James Bennett and Stan Lanning. The netflix prize. 2007. URL https://api.semanticscholar.org/CorpusID:9528522.

Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. Robustness may be at odds with fairness : An empirical study on class-wise accuracy. *CoRR*, abs/2010.13365, 2020. URL https://arxiv.org/abs/2010.13365.

J Bernoulli. A new problem to whose solution mathematicians are invited. *Acta Eruditorum*, 18 :269, 1696.

Daniel Berrar. Performance measures for binary classification. In Shoba Ranganathan, Michael Gribskov, Kenta Nakai, and Christian Schönbach, editors, *Encyclopedia of Bioinformatics and Computational Biology - Volume 1*, pages 546–560. Elsevier, 2019. doi : 10.1016/b978-0-12-809633-8.20351-8. URL https://doi.org/10.1016/b978-0-12-809633-8.20351-8.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, NY, 2006. Ch. 1.2.4 The Gaussian distribution.

Avrim Blum and Moritz Hardt. The ladder : A reliable leaderboard for machine learning competitions. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1006–1014. JMLR.org, 2015. URL http://proceedings.mlr.press/v37/blum15.html.

Ludovico Boratto, Gianni Fenu, and Mirko Marras. Interplay between upsampling and regularization for provider fairness in recommender systems. *User Model. User Adapt. Interact.*, 31(3) :421–455, 2021. doi : 10.1007/s11257-021-09294-8. URL https://doi.org/10.1007/s11257-021-09294-8.

Alexei Botchkarev. Performance metrics (error measures) in machine learning regression, forecasting and prognostics : Properties and typology. *CoRR*, abs/1809.03006, 2018. URL http://arxiv.org/abs/1809.03006.

Kevin J Boudreau and Karim R Lakhani. Using the crowd as an innovation partner. *Harvard business review*, 91(4) :60–9, 2013.

Antoine Boutet, Mathieu Cunche, Sébastien Gambs, Benjamin Nguyen, and Antoine Laurent. DARC : Data Anonymization and Re-identification Challenge. In *RESSI 2020 - Rendez-vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information*, Nouan-le-Fuzelier, France, December 2020. URL https://inria.hal.science/hal-02512677.

Xavier Bouthillier and Gaël Varoquaux. Survey of machine-learning experimental methods at NeurIPS2019 and ICLR2020. Research report, Inria Saclay Ile de France, January 2020. URL https://hal.science/hal-02447823.

Xavier Bouthillier, Pierre Delaunay, Mirko Bronzi, Assya Trofimov, Brennan Nichyporuk, Justin Szeto, Nazanin Mohammadi Sepahvand, Edward Raff, Kanika Madan, Vikram Voleti, Samira Ebrahimi Kahou, Vincent Michalski, Tal Arbel, Chris Pal, Gaël Varoquaux, and Pascal Vincent. Accounting for variance in machine learning benchmarks. In Alex Smola, Alex Dimakis, and Ion Stoica, editors, *Proceedings of Machine Learning and Systems 2021, MLSys 2021, virtual, April 5-9, 2021*. mlsys.org, 2021. URL https://proceedings.mlsys.org/paper/2021/hash/cfecdb276f634854f3ef915e2e980c31-Abstract.html.

Paul C Boutros, Adam D Ewing, Kyle Ellrott, Thea C Norman, Kristen K Dang, Yin Hu, Michael R Kellen, Christine Suver, J Christopher Bare, Lincoln D Stein, et al. Global optimization of somatic variant identification in cancer genomes with a global community challenge. *Nature genetics*, 46(4) :318–319, 2014.

Pavel Brazdil and Carlos Soares. A comparison of ranking methods for classification algorithm selection. In Ramón López de Mántaras and Enric Plaza, editors, *11th European Conference on Machine Learning ECML 2000, Barcelona, Catalonia, Spain*, volume 1810 of *Lecture Notes in Computer Science*, pages 63–74. Springer, 2000. doi : 10.1007/3-540-45164-1_8. URL https://doi.org/10.1007/3-540-45164-1_8.

Pavel Brazdil, Jan N. van Rijn, Carlos Soares, and Joaquin Vanschoren. Metalearning : Applications to automated machine learning and data mining. 2022.

T. Caliński and J Harabasz. A dendrite method for cluster analysis. *Communications in Statistics*, 3(1) :1–27, 1974. doi : 10.1080/03610927408827101. URL https://www.tandfonline.com/doi/abs/10.1080/03610927408827101.

Gürol Canbek, Seref Sagiroglu, Tugba Taskaya Temizel, and Nazife Baykal. Binary classification performance measures/metrics : A comprehensive visualized roadmap to gain new insights. In *2017 International Conference on Computer Science and Engineering (UBMK)*, pages 821–826, 2017. doi : 10.1109/UBMK.2017.8093539.

Harald Carlens. State of competitive machine learning in 2022. *ML Contests*, 2023.

Rich Caruana and Alexandru Niculescu-Mizil. Data mining in metric space : an empirical analysis of supervised learning performance criteria. In Won Kim, Ron Kohavi, Johannes Gehrke, and William DuMouchel, editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA, August 22-25, 2004*, pages 69–78. ACM, 2004. doi : 10.1145/1014052.1014063. URL https://doi.org/10.1145/1014052.1014063.

Diogo V. Carvalho, Eduardo M. Pereira, and Jaime S. Cardoso. Machine learning interpretability : A survey on methods and metrics. *MDPI Electronics*, 2019. URL https://www.mdpi.com/2079-9292/8/8/832/pdf.

Vítor Cerqueira, Luís Torgo, and Igor Mozetic. Evaluating time series forecasting models : an empirical study on performance estimation methods. *Mach. Learn.*, 109(11) :1997–2028, 2020. doi : 10.1007/s10994-020-05910-7. URL https://doi.org/10.1007/s10994-020-05910-7.

David L Chandler. A doctor in the palm of your hand : how the qualcomm tricorder x-prize could help to revolutionize medical diagnosis. *IEEE pulse*, 5 (2) :50–54, 2014.

Irene Y. Chen, Fredrik D. Johansson, and David A. Sontag. Why is my classifier discriminatory? In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31 : Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 3543–3554, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/1f1baa5b8edac74eb4eaa329f14a0361-Abstract.html.

Tianqi Chen and Carlos Guestrin. Xgboost : A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

Francois Chollet et al. Keras, 2015. URL https://github.com/fchollet/keras.

Alexandra Chouldechova and Aaron Roth. The frontiers of fairness in machine learning. *CoRR*, abs/1810.08810, 2018. URL http://arxiv.org/abs/1810.08810.

Christopher Cieri and Mark Liberman. Issues in corpus creation and distribution : The evolution of the linguistic data consortium. In *Proceedings of the Second International Conference on Language Resources and Evaluation, LREC 2000, 31 May - June 2, 2000, Athens, Greece*. European Language Resources Association, 2000. URL http://www.lrec-conf.org/proceedings/lrec2000/html/summary/209.htm.

Paul R. Cohen. *Empirical methods for artificial intelligence*. MIT Press, 1995. ISBN 978-0-262-03225-4.

Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307) : 756–760, 2010.

Wayne W. Daniel. Spearman rank correlation coefficient. In *Applied Nonparametric Statistics (2nd ed.)*, pages 358 – 365. Boston : PWS-Kent, 1990. ISBN 978-0-534-91976-4.

Rhiju Das, Bin Qian, Srivatsan Raman, Robert Vernon, James Thompson, Philip Bradley, Sagar Khare, Michael D Tyka, Divya Bhat, Dylan Chivian, et al. Structure prediction for casp7 targets using extensive all-atom refinement with rosetta@ home. *Proteins : Structure, Function, and Bioinformatics*, 69 (S8) :118–128, 2007.

Etienne David, Simon Madec, Pouria Sadeghi-Tehran, Helge Aasen, B. Zheng, Shouyang Liu, Norbert Kirchgeßner, Goro Ishikawa, Koichi Nagasawa, Minhajul A. Badhon, Curtis Pozniak, Benoit de Solan, Andreas Hund, Scott C. Chapman, Frédéric Baret, Ian Stavness, and Wei Guo. Global wheat head detection (GWHD) dataset : a large and diverse dataset of high resolution RGB labelled images to develop and benchmark wheat head detection methods. *CoRR*, abs/2005.02162, 2020. URL https://arxiv.org/abs/2005.02162.

D. L. Davies and D. W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2) :224–227, 1979. ISSN 0162-8828. doi : 10.1109/TPAMI.1979.4766909.

Joost de Winter. Can chatgpt pass high school exams on english language comprehension ? *Preprint*, 2023.

Janez Demăr. On the Appropriateness of Statistical Tests in Machine Learning . 2008. URL http://www.site.uottawa.ca/ICML08WS/papers/J_Demsar.pdf.

Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. 1998. URL https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.1005.1391&rep=rep1&type=pdf.

B. Donnot. Grid2op- A testbed platform to model sequential decision making in power systems. . https://GitHub.com/rte-france/grid2op, 2020.

David Donoho. 50 years of data science. *Journal of Computational and Graphical Statistics*, 26(4) :745–766, 2017. doi : 10.1080/10618600.2017.1384734. URL https://doi.org/10.1080/10618600.2017.1384734.

Matthias Dorfer, Anton R. Fuxjäger, Kristián Kozák, Patrick M. Blies, and Marcel Wasserer. Power grid congestion management via topology optimization with alphazero. *CoRR*, abs/2211.05612, 2022. doi : 10.48550/arXiv.2211.05612. URL https://doi.org/10.48550/arXiv.2211.05612.

Chris Drummond. Machine learning as an experimental science (revisited). *Machine Learning*, 2006. URL http://www.aaai.org/Papers/Workshops/2006/WS-06-06/WS06-06-002.pdf.

Bradley Efron and Robert Tibshirani. *An Introduction to the Bootstrap*. Springer, 1993a. ISBN 978-1-4899-4541-9. doi : 10.1007/978-1-4899-4541-9. URL https://doi.org/10.1007/978-1-4899-4541-9.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Number 57 in Monographs on Statistics and Applied Probability. Chapman & Hall/CRC, Boca Raton, Florida, USA, 1993b.

Adrian El Baz, Ihsan Ullah, Edesio Alcobaça, André C. P. L. F. Carvalho, Hong Chen, Fabio Ferreira, Henry Gouk, Chaoyu Guan, Isabelle Guyon, Timothy Hospedales, Shell Hu, Mike Huisman, Frank Hutter, Zhengying Liu, Felix Mohr, Ekrem Öztürk, Jan N van Rijn, Haozhe Sun, Xin Wang, and Wenwu Zhu. Lessons learned from the NeurIPS 2021 MetaDL challenge : Backbone fine-tuning without episodic meta-learning dominates for few-shot learning image classification. In *NeurIPS 2021 Competition and Demonstration Track*, On-line, United States, December 2021. URL https://hal.science/hal-03688638.

Pär G Engström, Tamara Steijger, Botond Sipos, Gregory R Grant, André Kahles, Gunnar Rätsch, Nick Goldman, Tim J Hubbard, Jennifer Harrow, Roderic Guigó, et al. Systematic evaluation of spliced alignment programs for rna-seq data. *Nature methods*, 10(12) :1185–1191, 2013.

Kim H. Esbensen and Paul Geladi. Principles of proper validation : use and abuse of re-sampling for validation. *Journal of CHEMOMETRICS*, 2010. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cem.1310.

Hugo Jair Escalante, Isabelle Guyon, Sergio Escalera, Júlio C. S. Jacques Júnior, Meysam Madadi, Xavier Baró, Stéphane Ayache, Evelyne Viegas, Yagmur Güçlütürk, Umut Güçlü, Marcel A. J. van Gerven, and Rob van Lier. Design of an explainable machine learning challenge for video interviews. In *2017 International Joint Conference on Neural Networks, IJCNN 2017, Anchorage, AK, USA, May 14-19, 2017*, pages 3688–3695. IEEE, 2017. doi : 10.1109/IJCNN.2017.7966320. URL https://doi.org/10.1109/IJCNN.2017.7966320.

Hugo Jair Escalante, Heysem Kaya, Albert Ali Salah, Sergio Escalera, Yagmur Güçlütürk, Umut Güçlü, Xavier Baró, Isabelle Guyon, Júlio C. S. Jacques Júnior, Meysam Madadi, Stéphane Ayache, Evelyne Viegas, Furkan Gürpinar, Achmadnoer Sukma Wicaksana, Cynthia C. S. Liem, Marcel A. J. van Gerven, and Rob van Lier. Explaining first impressions : Modeling, recognizing, and explaining apparent personality from videos. *CoRR*, abs/1802.00745, 2018. URL http://arxiv.org/abs/1802.00745.

M.J. Evans and J.S.Rosenthal. *Probability and Statistics - The Science of Uncertainty*. W.H.Freeman and Company, New York, 2004.

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, and Frank Hutter. Auto-sklearn : Efficient and robust automated machine learning. In Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors, *Automated Machine Learning - Methods, Systems, Challenges*, The Springer Series on Challenges in Machine Learning, pages 113–134. Springer, 2019. doi : 10.1007/978-3-030-05318-5_6. URL https://doi.org/10.1007/978-3-030-05318-5_6.

Peter C. Fishburn and Steven J. Brams. Paradoxes of preferential voting. *Mathematics Magazine*, 56(4) :207–214, 1983. doi : 10.2307/2689808.

R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annual Eugenics*, 7(Part II) :179–188, 1936. The competition protocol was designed by Isabelle Guyon. This challenge was generated using ChaLab for Codalab v1.5.

George Forman and Martin Scholz. Apples-to-apples in cross-validation studies : pitfalls in classifier performance measurement. *SIGKDD Explor.*, 12(1) : 49–57, 2010. doi : 10.1145/1882471.1882479. URL https://www.kdd.org/exploration_files/v12-1-p49-forman-sigkdd.pdf.

RTE France. Futurs énergétiques 2050 principaux résultats. October 2021. URL https://assets.rte-france.com/prod/public/2021-10/Futurs-Energetiques-2050-principaux-resultats_0.pdf.

Simon Frieder, Luca Pinchetti, Ryan-Rhys Griffiths, Tommaso Salvatori, Thomas Lukasiewicz, Philipp Christian Petersen, Alexis Chevalier, and Julius Berner. Mathematical capabilities of chatgpt. *CoRR*, abs/2301.13867, 2023. doi : 10.48550/arXiv.2301.13867. URL https://doi.org/10.48550/arXiv.2301.13867.

Jerome H Friedman. Greedy function approximation : a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.

Wulf Gaertner. *A Primer in Social Choice Theory*. Oxford University Press, 2006.

William V. Gehrlein. Condorcet's paradox and the condorcet efficiency of voting rules. 1997.

Allan Gibbard. Manipulation of voting schemes : A general result. *Econometrica*, 1973.

Fabrizio Gilardi, Meysam Alizadeh, and Maël Kubli. Chatgpt outperforms crowd-workers for text-annotation tasks. *Preprint*, 2023.

Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations : An overview of interpretability of machine learning. In Francesco Bonchi, Foster J. Provost, Tina Eliassi-Rad, Wei Wang, Ciro Cattuto, and Rayid Ghani, editors, *5th IEEE International Conference on Data Science and Advanced Analytics, DSAA 2018, Turin, Italy, October 1-3, 2018*, pages 80–89. IEEE, 2018. ISBN 978-1-5386-5090-5. doi : 10.1109/DSAA.2018.00018. URL https://doi.org/10.1109/DSAA.2018.00018.

Anthony Goldbloom and Ben Hamner. Kaggle. 2010. URL https://www.kaggle.com/.

Benjamin M Good and Andrew I Su. Crowdsourcing for bioinformatics. *Bioinformatics*, 29(16) :1925–1933, 2013.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

Joseph K Goodman, Cynthia E Cryder, and Amar Cheema. Data collection in a flat world : The strengths and weaknesses of mechanical turk samples. *Journal of Behavioral Decision Making*, 26(3) :213–224, 2013.

Yehoram Gordon, Alexander Litvak, Carsten Schütt, and Elisabeth Werner. On the minimum of several random variables. *Proceedings of the American Mathematical Society*, 134(12) :3665–3675, 2006.

Margherita Grandini, Enrico Bagli, and Giorgio Visani. Metrics for multi-class classification : an overview. *CoRR*, abs/2008.05756, 2020. URL https://arxiv.org/abs/2008.05756.

Angela K Green, Katherine E Reeder-Hayes, Robert W Corty, Ethan Basch, Mathew I Milowsky, Stacie B Dusetzina, Antonia V Bennett, and William A Wood. The project data sphere initiative : accelerating cancer research by sharing data. *The oncologist*, 20(5) :464, 2015.

Alibaba Group. Tianchi. 2014. URL https://tianchi.aliyun.com/.

Odd Erik Gundersen and Sigbjørn Kjensmo. State of the art : Reproducibility in artificial intelligence. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1644–1651. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17248.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 2017. URL http://proceedings.mlr.press/v70/guo17a.html.

Guy Gur-Ari, Ethan Dyer, Ambrose Slone, Jascha Sohl-Dickstein, Noah Fiedel, Jaehoon Lee, Daniel Freeman, Aitor Lewkowycz, Anders Andreassen, Gaurav Mishra, Vedant Misra, Vinay Ramasesh, Noah Constant, Rosanne Liu, and et al. Beyond the imitation game : Quantifying and extrapolating the capabilities of language models. *CoRR*, abs/2206.04615, 2022. doi : 10.48550/arXiv.2206.04615. URL https://doi.org/10.48550/arXiv.2206.04615.

William H. Guss, Cayden Codel, Katja Hofmann, Brandon Houghton, Noburu Kuno, Stephanie Milani, Sharada P. Mohanty, Diego Perez Liebana, Ruslan Salakhutdinov, Nicholay Topin, Manuela Veloso, and Phillip Wang. The minerl competition on sample efficient reinforcement learning using human priors. *CoRR*, abs/1904.10079, 2019. URL http://arxiv.org/abs/1904.10079.

Isabelle Guyon, John Makhoul, Richard Schwartz, and Vladimir Vapnik. What size test set gives good error rate estimates? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1) :52–64, 1998a.

Isabelle Guyon, John Makhoul, Richard M. Schwartz, and Vladimir Vapnik. What size test set gives good error rate estimates? *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(1) :52–64, 1998b. doi : 10.1109/34.655649. URL https://doi.org/10.1109/34.655649.

Isabelle Guyon, Steve Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004. URL https://proceedings.neurips.cc/paper/2004/file/5e751896e527c862bf67251a474b3819-Paper.pdf.

Isabelle Guyon, Amir Reza Saffari Azar Alamdari, Gideon Dror, and Joachim M. Buhmann. Performance prediction challenge. 2006a.

Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and Lofti A. Zadeh. Feature extraction, foundations and applications. 2006b.

Isabelle Guyon, Gavin C. Cawley, Gideon Dror, and Vincent Lemaire. Results of the active learning challenge. In Isabelle Guyon, Gavin C. Cawley, Gideon Dror, Vincent Lemaire, and Alexander R. Statnikov, editors, *Active Learning and Experimental Design workshop, In conjunction with AISTATS 2010, Sardinia, Italy, May 16, 2010*, volume 16 of *JMLR Proceedings*, pages 19–45. JMLR.org, 2011. URL http://proceedings.mlr.press/v16/guyon11a/guyon11a.pdf.

Isabelle Guyon, Lisheng Sun-Hosoya, Marc Boullé, Hugo Jair Escalante, Sergio Escalera, Zhengying Liu, Damir Jajetic, Bisakha Ray, Mehreen Saeed, Michèle Sebag, Alexander R. Statnikov, Wei-Wei Tu, and Evelyne Viegas. Analysis of

the automl challenge series 2015-2018. In Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren, editors, *Automated Machine Learning - Methods, Systems, Challenges*, The Springer Series on Challenges in Machine Learning, pages 177–219. Springer, 2019a. doi : 10.1007/978-3-030-05318-5_10. URL https://doi.org/10.1007/978-3-030-05318-5_10.

Isabelle Guyon, Lisheng Sun-Hosoya, Marc Boullé, Hugo Jair Escalante, Sergio Escalera, Zhengying Liu, Damir Jajetic, Bisakha Ray, Mehreen Saeed, Michèle Sebag, et al. Analysis of the automl challenge series. *Automated Machine Learning*, page 177, 2019b.

Donna Harman. Overview of the first trec conference. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '93, page 36–47, New York, NY, USA, 1993. Association for Computing Machinery. ISBN 0897916050. doi : 10.1145/160688.160692. URL https://doi.org/10.1145/160688.160692.

Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning : Data Mining, Inference, and Prediction, 2nd Edition*. Springer Series in Statistics. Springer, 2009. ISBN 9780387848570. doi : 10.1007/978-0-387-84858-7. URL https://doi.org/10.1007/978-0-387-84858-7.

Willem J. Heiser and Antonio D'Ambrosio. Clustering and prediction of rankings within a kemeny distance framework. In Berthold Lausen, Dirk Van den Poel, and Alfred Ultsch, editors, *Algorithms from and for Nature and Life - Classification and Data Analysis*, pages 19–31. Springer, 2013. doi : 10.1007/978-3-319-00035-0_2. URL https://doi.org/10.1007/978-3-319-00035-0_2.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 3207–3214. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16669.

José Hernández-Orallo, Peter A. Flach, and César Ferri. A unified view of performance metrics : translating threshold choice into expected classification loss. *J. Mach. Learn. Res.*, 13 :2813–2869, 2012. doi : 10.5555/2503308.2503332. URL https://dl.acm.org/doi/10.5555/2503308.2503332.

M Herrera. Galileo, bernoulli, leibniz and newton around the brachistochrone problem. *Rev Mexicana Fis*, 40(3) :459–475, 1994.

Roger A Hoskins, Susanna Repo, Daniel Barsky, Gaia Andreoletti, John Moult, and Steven E. Brenner. Reports from cagi : The critical assessment of genome interpretation. *Human Mutation*, 38(9) :1039–1041, 2017. doi : https://doi.org/10.1002/humu.23290. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/humu.23290.

Jeff Howe. The rise of crowdsourcing. *Wired magazine*, 14(6) :1–4, 2006.

Fan Huang, Haewoon Kwak, and Jisun An. Is chatgpt better than human annotators? potential and limitations of chatgpt in explaining implicit hate speech. *CoRR*, abs/2302.07736, 2023. doi : 10.48550/arXiv.2302.07736. URL https://doi.org/10.48550/arXiv.2302.07736.

Frank Hutter. A Proposal for a New Competition Design Emphasizing Scientific Insights. *CiML Workshop, NeurIPS*, December 2019.

Rob J Hyndman. Forecasting competitions, 2023. URL https://robjhyndman.com/hyndsight/forecasting-competitions/.

Rob J. Hyndman and George Athanasopoulos, editors. *Forecasting : principles and practice*. OTexts, 2021.

Bertrand Iooss, Vincent Chabridon, and Vincent Thouvenot. Variance-based importance measures for machine learning model interpretability. In *Actes du Congrès*, Oct 2022. URL https://hal.archives-ouvertes.fr/hal-03741384.

Richard Isdahl and Odd Erik Gundersen. Out-of-the-box reproducibility : A survey of machine learning platforms. In *15th International Conference on eScience, eScience 2019, San Diego, CA, USA, September 24-27, 2019*, pages 86–95. IEEE, 2019. doi : 10.1109/eScience.2019.00017. URL https://doi.org/10.1109/eScience.2019.00017.

Nathalie Japkowicz and Mohak Shah, editors. *Evaluating Learning Algorithms : A Classification Perspective*. Cambridge University Press, 2011. ISBN 9780521196000.

Scott M. Jordan, Yash Chandak, Daniel Cohen, Mengxue Zhang, and Philip S. Thomas. Evaluating the performance of reinforcement learning algorithms. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4962–4973. PMLR, 2020. URL http://proceedings.mlr.press/v119/jordan20a.html.

James Jordon, Daniel Jarrett, Evgeny Saveliev, Jinsung Yoon, Paul W. G. Elbers, Patrick Thoral, Ari Ercole, Cheng Zhang, Danielle Belgrave, and Mihaela van der Schaar. Hide-and-seek privacy challenge : Synthetic data generation vs. patient re-identification. In Hugo Jair Escalante and Katja Hofmann, editors, *NeurIPS 2020 Competition and Demonstration Track, 6-12 December 2020, Virtual Event / Vancouver, BC, Canada*, volume 133 of *Proceedings of Machine Learning Research*, pages 206–215. PMLR, 2020. URL http://proceedings.mlr.press/v133/jordon21a.html.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Applying and improving alphafold at casp14. *Proteins : Structure, Function, and Bioinformatics*, 89(12) :1711–1721, 2021a.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873) :583–589, 2021b.

Wendy Kan, Phil Culliton, and Douglas Sterling. Dog image generation competition on kaggle. *Competitions in Machine Learning (CiML) workshop at Neural Information Processing Systems (NIPS) 2019*, 2019.

Balázs Kégl, Alexandre Boucaud, Mehdi Cherti, Akin Osman Kazakci, Alexandre Gramfort, Guillaume M Lemaitre, Joris Van den Bossche, Djalel Benbouzid, and Camille Marini. The RAMP framework : from reproducibility to transparency in the design and optimization of scientific workflows. International Conference On Machine Learning, July 2018. URL https://hal.archives-ouvertes.fr/hal-02072341. Poster.

M. Kendall and J.D. Gibbons. *Rank Correlation Methods.* 5th Edition, Edward Arnold, London., 1990a.

M. G. Kendall and J. D. Gibbons. Rank correlation methods. *New York, NY : Oxford University Press*, 1990b.

M. G. Kendall and B. Babington Smith. The Problem of m Rankings. *The Annals of Mathematical Statistics*, 10(3) :275 – 287, 1939. doi : 10.1214/aoms/1177732186. URL https://doi.org/10.1214/aoms/1177732186.

Thanh Gia Hieu Khuong and Benedictus Kent Rachmat. Auto-survey challenge : Advancing the frontiers of automated literature review. In *Junior Conference on Data Science and Engineering 2023*, Orsay, France, Sep 2023. URL https://inria.hal.science/hal-04206578. ⁂hal-04206578⁂.

Lukasz Kidzinski, Sharada Prasanna Mohanty, Carmichael F. Ong, Zhewei Huang, Shuchang Zhou, Anton Pechenko, Adam Stelmaszczyk, Piotr Jarosik, Mikhail Pavlov, Sergey Kolesnikov, Sergey M. Plis, Zhibo Chen, Zhizheng Zhang, Jiale Chen, Jun Shi, Zhuobin Zheng, Chun Yuan, Zhihui Lin, Henryk Michalewski, Piotr Milos, Blazej Osinski, Andrew Melnik, Malte Schilling, Helge J. Ritter, Sean F. Carroll, Jennifer L. Hicks, Sergey Levine, Marcel Salathé, and Scott L. Delp. Learning to run challenge solutions : Adapting reinforcement learning methods for neuromusculoskeletal environments. *CoRR*, abs/1804.00361, 2018. URL http://arxiv.org/abs/1804.00361.

R.D. King, C. Feng, and A. Sutherland. Statlog : Comparison of classification algorithm on large real-world problems. *Applied Artificial Intelligence*, 9(3) : 289–333, 1995. doi : 10.1080/08839519508945477. URL https://doi.org/10.1080/08839519508945477.

Diederik P. Kingma and Max Welling. An introduction to variational autoencoders. *CoRR*, abs/1906.02691, 2019. URL http://arxiv.org/abs/1906.02691.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, pages 1137–1145. Morgan Kaufmann, 1995a. URL http://web.cs.iastate.edu/~jtian/cs573/Papers/Kohavi-IJCAI-95.pdf.

Ron Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, IJCAI 95, Montréal Québec, Canada, August 20-25 1995, 2 Volumes*, pages 1137–1145. Morgan Kaufmann, 1995b. URL http://ijcai.org/Proceedings/95-2/Papers/016.pdf.

Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2) :273–324, 1997. doi : 10.1016/S0004-3702(97)00043-X. URL https://doi.org/10.1016/S0004-3702(97)00043-X.

Ron Kohavi, Carla E Brodley, Brian Frasca, Llew Mason, and Zijian Zheng. Kdd-cup 2000 organizers' report : Peeling the onion. *Acm Sigkdd Explorations Newsletter*, 2(2) :86–93, 2000.

Ron Kohavi et al. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai*, volume 14, pages 1137–1145. Montreal, Canada, 1995.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Andriy Kryshtafovych, Torsten Schwede, Maya Topf, Krzysztof Fidelis, and John Moult. Critical assessment of methods of protein structure prediction (casp)—round xiv. *Proteins : Structure, Function, and Bioinformatics*, 89(12) : 1607–1617, 2021.

Martin Krzywinski and Naomi Altman. Error bars. *Nature Methods*, 10(10) : 921–922, Oct 2013. ISSN 1548-7105. doi : 10.1038/nmeth.2659. URL https://doi.org/10.1038/nmeth.2659.

S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22 :79–86, 1951.

Ravi Kumar and Sergei Vassilvitskii. Generalized distances between rankings. In *International Conference on World Wide Web, WWW 2010*. ACM, 2010. doi : 10.1145/1772690.1772749. URL https://doi.org/10.1145/1772690.1772749.

Martina Kutmon, Anders Riutta, Nuno Nunes, Kristina Hanspers, Egon L Willighagen, Anwesha Bohler, Jonathan Mélius, Andra Waagmeester, Sravanthi R Sinha, Ryan Miller, et al. Wikipathways : capturing the full diversity of pathway knowledge. *Nucleic acids research*, 44(D1) :D488–D494, 2016.

Doudou LaLoudouana and Mambobo Bonouliqui Tarare. Data set selection. *Neural Information Processing Systems (NIPS)*, 2002.

Charline Le Lan, Marc G. Bellemare, and Pablo Samuel Castro. Metrics and continuity in reinforcement learning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 8261–8269. AAAI Press, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17005.

John Langford. The cross validation problem. In Peter Auer and Ron Meir, editors, *Learning Theory, 18th Annual Conference on Learning Theory, COLT 2005, Bertinoro, Italy, June 27-30, 2005, Proceedings*, volume 3559 of *Lecture Notes in Computer Science*, pages 687–688. Springer, 2005a. doi : 10.1007/11503415_47. URL https://doi.org/10.1007/11503415_47.

John Langford. Tutorial on practical prediction theory for classification. *J. Mach. Learn. Res.*, 6 :273–306, 2005b. URL https://www.jmlr.org/papers/volume6/langford05a/langford05a.pdf.

Pat Langley. Machine learning as an experimental science. *Machine Learning*, 3 :5–8, 1988. doi : 10.1007/BF00115008. URL https://doi.org/10.1007/BF00115008.

Pat Langley and Dennis Kibler. The experimental study of machine learning, 1991.

Stefan M Larson, Christopher D Snow, Michael Shirts, and Vijay S Pande. Folding@ home and genome@ home : Using distributed computing to tackle previously intractable problems in computational biology. *arXiv preprint arXiv :0901.0866*, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *IEEE*, 1998.

Jeehyung Lee, Wipapat Kladwang, Minjae Lee, Daniel Cantu, Martin Azizyan, Hanjoo Kim, Alex Limpaecher, Snehal Gaikwad, Sungroh Yoon, Adrien Treuille, et al. Rna design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences*, 111(6) :2122–2127, 2014.

G. W. Leibniz. Leibniz' participation oí his solution and oí those oí j. bernoulli and oí marquis de i'hospital, to the problem published by j. bernoulli, and at the same time, the solutions to his second problem. *Acta Eruditorum Lipsi*, page 201, 1697.

Xingxuan Li, Yutong Li, Linlin Liu, Lidong Bing, and Shafiq R. Joty. Is GPT-3 a psychopath ? evaluating large language models from a psychological perspective. *CoRR*, abs/2212.10529, 2022. doi : 10.48550/arXiv.2212.10529. URL https://doi.org/10.48550/arXiv.2212.10529.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel J. Orr, Lucia Zheng, Mert Yüksekgönül, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri S. Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. Holistic evaluation of language models. *CoRR*, abs/2211.09110, 2022. doi : 10.48550/arXiv.2211.09110. URL https://doi.org/10.48550/arXiv.2211.09110.

Mark Liberman. Fred Jelinek. *Computational Linguistics*, 36(4) :595–599, 12 2010. ISSN 0891-2017. doi : 10.1162/coli_a_00032. URL https://doi.org/10.1162/coli_a_00032.

Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 140 :1–55, 1932.

Cheng-Yuan Liou, Wei-Chen Cheng, Jiun-Wei Liou, and Daw-Ran Liou. Autoencoder for words. *Neurocomputing*, 139 :84–96, 2014.

Sijia Liu, Yanshan Wang, and Hongfang Liu. Selected articles from the biocreative/ohnlp challenge 2018, 2019.

Xuechen Liu, Xin Wang, Md. Sahidullah, Jose Patino, Héctor Delgado, Tomi Kinnunen, Massimiliano Todisco, Junichi Yamagishi, Nicholas W. D. Evans, Andreas Nautsch, and Kong Aik Lee. Asvspoof 2021 : Towards spoofed and deepfake speech detection in the wild. *IEEE ACM Trans. Audio Speech Lang. Process.*, 31 :2507–2522, 2023. doi : 10.1109/TASLP.2023.3285283. URL https://doi.org/10.1109/TASLP.2023.3285283.

Zhengying Liu. *Automated Deep Learning : Principles and Practice*. Theses, Université Paris-Saclay, November 2021. URL https://tel.archives-ouvertes.fr/tel-03464519.

Zhengying Liu, Adrien Pavao, Zhen Xu, Sergio Escalera, Fabio Ferreira, Isabelle Guyon, Sirui Hong, Frank Hutter, Rongrong Ji, Júlio C. S. Jacques Júnior, Ge Li, Marius Lindauer, Zhipeng Luo, Meysam Madadi, Thomas Nierhoff, Kangning Niu, Chunguang Pan, Danny Stoll, Sébastien Treguer, Jin Wang, Peng Wang, Chenglin Wu, Youcheng Xiong, Arber Zela, and Yang Zhang. Winning solutions and post-challenge analyses of the chalearn autodl challenge 2019. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(9) :3108–3125, 2021a. doi : 10.1109/TPAMI.2021.3075372. URL https://doi.org/10.1109/TPAMI.2021.3075372.

Zhengying Liu, Adrien Pavao, Zhen Xu, Sergio Escalera, Fabio Ferreira, Isabelle Guyon, Sirui Hong, Frank Hutter, Rongrong Ji, Julio C. S. Jacques Junior, Ge Li, Marius Lindauer, Zhipeng Luo, Meysam Madadi, Thomas Nierhoff, Kangning Niu, Chunguang Pan, Danny Stoll, Sebastien Treguer, Jin Wang, Peng Wang, Chenglin Wu, Youcheng Xiong, Arbër Zela, and Yang Zhang. Winning solutions and post-challenge analyses of the chalearn autodl challenge 2019. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43 (9) :3108–3125, 2021b. doi : 10.1109/TPAMI.2021.3075372.

Kazuaki Maeda and Stephanie M. Strassel. Annotation tools for large-scale corpus development : Using AGTK at the linguistic data consortium. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, May 26-28, 2004, Lisbon, Portugal*. European Language Resources Association, 2004. URL http://www.lrec-conf.org/proceedings/lrec2004/summaries/761.htm.

Spyros Makridakis and Michèle Hibon. The m3-competition : results, conclusions and implications. *International Journal of Forecasting*, 16(4) :451–476, Oct 2000. doi : 10.1016/S0169-2070(00)00057-1.

Spyros Makridakis, A Andersen, R Carbone, R Fildes, Michèle Hibon, R Lewandowski, J Newton, E Parzen, and R Winkler. The accuracy of extrapolation (time series) methods : Results of a forecasting competition. *Journal of Forecasting*, 1(2) :111–153, Apr 1982. doi : 10.1002/for.3980010202.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The m4 competition : Results, findings, conclusion and way forward. *International Journal of Forecasting*, 34(4) :802–808, 2018.

Adam A Margolin, Erhan Bilal, Erich Huang, Thea C Norman, Lars Ottestad, Brigham H Mecham, Ben Sauerwine, Michael R Kellen, Lara M Mangravite, Matthew D Furia, et al. Systematic analysis of challenge-driven improvements in molecular prognostic models for breast cancer. *Science translational medicine*, 5(181) :181re1–181re1, 2013.

Marianthi Markatou, Hong Tian, Shameek Biswas, and George Hripcsak. Analysis of variance of cross-validation estimators of the generalization error. *J. Mach. Learn. Res.*, 6 :1127–1168, 2005. URL http://www.jmlr.org/papers/volume6/markatou05a/markatou05a.pdf.

A. Marot, N. Megel, V. Renault, and M. Jothy. ChroniX2Grid - The Extensive PowerGrid Time-serie Generator. https://github.com/BDonnot/ChroniX2Grid, 2020.

Antoine Marot, Benjamin Donnot, Camilo Romero, Luca Veyrin-Forrer, Marvin Lerousseau, Balthazar Donon, and Isabelle Guyon. Learning to run a power network challenge for training topology controllers. *CoRR*, abs/1912.04211, 2019. URL http://arxiv.org/abs/1912.04211.

Antoine Marot, Benjamin Donnot, Gabriel Dulac-Arnold, Adrian Kelly, Aïdan O'Sullivan, Jan Viebahn, Mariette Awad, Isabelle Guyon, Patrick Panciatici, and Camilo Romero. Learning to run a power network challenge : a retrospective analysis. *CoRR*, abs/2103.03104, 2021. URL http://proceedings.mlr.press/v133/marot21a/marot21a.pdf.

A. Martin and M. Przybocki. The nist speaker recognition evaluations : 1996-2001. In *A Speaker Odyssey, A Speaker Recognition Workshop*, 2001. URL https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=151520.

Nestor Maslej, Loredana Fattorini, Erik Brynjolfsson, John Etchemendy, Katrina Ligett, Terah Lyons, James Manyika, Helen Ngo, Juan Carlos Niebles, Vanessa Parli, Yoav Shoham, Russell Wald, Jack Clark, and Raymond Perrault.

The ai index 2023 annual report. Technical report, Institute for Human-Centered AI, Stanford University, Stanford, CA, April 2023. The AI Index 2023 Annual Report by Stanford University is licensed under Attribution-NoDerivatives 4.0 International.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2) :153–157, June 1947. doi : 10.1007/bf02295996. URL https://doi.org/10.1007/bf02295996.

Adriënne M. Mendrik and Stephen R. Aylward. A framework for challenge design : Insight and deployment challenges to address medical image analysis problems, 2019.

Dirk Merkel. Docker : lightweight linux containers for consistent development and deployment. *Linux journal*, 2014(239) :2, 2014.

D. Michie, D.J. Spiegelhalter, and C.C. Taylor. *Machine Learning, Neural and Statistical Classification*. 1994.

Divyabh Mishra. Crowdanalytix. 2012. URL https://www.crowdanalytix.com/.

Mustafa Misir and Michèle Sebag. Alors : An algorithm recommender system. *Artif. Intell.*, 244 :291–314, 2017. doi : 10.1016/j.artint.2016.12.001. URL https://doi.org/10.1016/j.artint.2016.12.001.

Sharada Mohanty, Shivam Khandelwal, and Marcel Salathé. Aicrowd. 2016. URL https://github.com/AIcrowd/AIcrowd.

Sharada P. Mohanty, Jyotish Poonganam, Adrien Gaidon, Andrey Kolobov, Blake Wulfe, Dipam Chakraborty, Grazvydas Semetulskis, João Schapke, Jonas Kubilius, Jurgis Pasukonis, Linas Klimas, Matthew J. Hausknecht, Patrick MacAlpine, Quang Nhat Tran, Thomas Tumiel, Xiaocheng Tang, Xinwei Chen, Christopher Hesse, Jacob Hilton, William Hebgen Guss, Sahika Genc, John Schulman, and Karl Cobbe. Measuring sample efficiency and generalization in reinforcement learning benchmarks : Neurips 2020 procgen benchmark. *CoRR*, abs/2103.15332, 2021. URL https://arxiv.org/abs/2103.15332.

Annette M. Molinaro, Richard Simon, and Ruth M. Pfeiffer. Prediction error estimation : a comparison of resampling methods. *Bioinform.*, 21(15) :3301–3307, 2005. doi : 10.1093/bioinformatics/bti499. URL https://pubmed.ncbi.nlm.nih.gov/15905277/.

Jonathan M Mortensen, Evan P Minty, Michael Januszyk, Timothy E Sweeney, Alan L Rector, Natalya F Noy, and Mark A Musen. Using the wisdom of the

crowds to find critical errors in biomedical ontologies : a study of snomed ct. *Journal of the American Medical Informatics Association*, 22(3) :640–648, 2015.

Takao Murakami, Hiromi Arai, Koki Hamada, Takuma Hatano, Makoto Iguchi, Hiroaki Kikuchi, Atsushi Kuromasa, Hiroshi Nakagawa, Yuichi Nakamura, Kenshiro Nishiyama, Ryo Nojima, Hidenobu Oguri, Chiemi Watanabe, Akira Yamada, Takayasu Yamaguchi, and Yuji Yamaoka. Designing a location trace anonymization contest. *Proc. Priv. Enhancing Technol.*, 2023(1) :225–243, 2023. doi : 10.56553/popets-2023-0014. URL https://doi.org/10.56553/popets-2023-0014.

Tom Murph. The first level of super mario bros. is easy with lexicographic orderings and time travel... after that it gets a little tricky. 2013.

Kevin P. Murphy. *Machine Learning : A Probabilistic Perspective*. MIT Press, Cambridge, MA, 2012. Ch. 6.4.2 Unbiased estimators.

Claude Nadeau and Yoshua Bengio. Inference for the generalization error. *Mach. Learn.*, 52(3) :239–281, 2003. doi : 10.1023/A:1024068626366. URL https://papers.nips.cc/paper/1661-inference-for-the-generalization-error.pdf.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2901–2907. AAAI Press, 2015. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9667.

R.B. Nelsen. Kendall tau metric. In *Encyclopedia of Mathematics*. EMS Press, 2001.

Benedict Neo. 12 data science and ai competitions to advance your skills in 2021. *Towards Data Science*, 2021. URL https://towardsdatascience.com/12-data-science-ai-competitions-to-advance-your-skills-in-2021-32e3fcb95d8c.

Manh Hung Nguyen, Lisheng Sun, Nathan Grinsztajn, and Isabelle Guyon. Meta-learning from Learning Curves Challenge : Lessons learned from the First Round and Design of the Second Round. working paper or preprint, August 2022. URL https://hal.science/hal-03725313.

Alex Nichol, Vicki Pfau, Christopher Hesse, Oleg Klimov, and John Schulman. Gotta learn fast : A new benchmark for generalization in RL. *CoRR*, abs/1804.03720, 2018. URL http://arxiv.org/abs/1804.03720.

Raquel Norel, John Jeremy Rice, and Gustavo Stolovitzky. The self-assessment trap : can we all be better than average ? *Molecular systems biology*, 7(1) :537, 2011.

OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. doi : 10.48550/arXiv.2303.08774. URL https://doi.org/10.48550/arXiv.2303.08774.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty ? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.

Julio-Omar Palacio-Niño and Fernando Berzal. Evaluation metrics for unsupervised learning algorithms. *CoRR*, abs/1905.05667, 2019. URL http://arxiv.org/abs/1905.05667.

Ismail Parsa. Kdd-cup-97 : A knowledge discovery and data mining tools competition talk. Newport beach, CA, USA, 1997. URL http://www-aig.jpl.nasa.gov/public/kdd97/.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch : An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

A. Pavao, D. Kalainathan, L. Sun-Hosoya, K. Bennett, and I. Guyon. Design and Analysis of Experiments : A Challenge Approach in Teaching. *CiML Workshop, NeurIPS*, December 2019. URL http://ciml.chalearn.org/ciml2019/accepted/Pavao.pdf?attredirects=0&d=1.

Adrien Pavao. ranky. https://github.com/didayolo/ranky, 2020.

Adrien Pavao, Michael Vaccaro, and Isabelle Guyon. Judging competitions and benchmarks : a candidate election approach. *European Symposium on Artificial Neural Networks (ESANN) Proceedings*, 2021a.

Adrien Pavao, Isabelle Guyon, Anne-Catherine Letournel, Xavier Baró, Hugo Escalante, Sergio Escalera, Tyler Thomas, and Zhen Xu. CodaLab Competitions : An open source platform to organize scientific challenges. Technical report, Université Paris-Saclay, FRA., April 2022a. URL https://hal.inria.fr/hal-03629462.

Adrien Pavao, Isabelle Guyon, and Zhengying Liu. Filtering participants improves generalization in competitions and benchmarks. In *(ESANN) 2022 - European Symposium on Artificial Neural Networks*, Bruges, Belgium, 2022b. URL https://www.esann.org/sites/default/files/proceedings/2022/ES2022-72.pdf.

Adrien Pavao, Isabelle Guyon, Anne-Catherine Letournel, Dinh-Tuan Tran, Xavier Baro, Hugo Jair Escalante, Sergio Escalera, Tyler Thomas, and Zhen Xu. Codalab competitions : An open source platform to organize scientific challenges. *Journal of Machine Learning Research*, 24(198) :1–6, 2023a. URL http://jmlr.org/papers/v24/21-1436.html.

Adrien Pavao, Evelyne Viegas, Isabelle Guyon, et al. *AI Competitions and Benchmarks : The Science Behind the Contests*. 2023b. Submitted and under review at Data-driven Machine Learning Research (DMLR).

Adrien Pavao et al. Airplane numerical twin : A time series regression competition. *International Conference on Machine Learning and Applications (ICMLA)*, 2021b.

Joseph Pedersen, Rafael Muñoz-Gómez, Jiangnan Huang, Haozhe Sun, Wei-Wei Tu, and Isabelle Guyon. Ltu attacker for membership inference, 2022.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn : Machine learning in python. *the Journal of machine Learning research*, 12 :2825–2830, 2011.

Pécresse, Valérie and Conseil régional d'Ile-de-France. StratÉgie Énergie-climat de la rÉgion Île-de-france. 2018. URL https://www.iledefrance.fr/sites/default/files/medias/rapports/RAPCR%202018-016RAP.pdf.

Edward Raff. A step toward quantifying independently reproducible machine learning research. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32 : Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5486–5496, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/c429429bf1f2af051f2021dc92a8ebea-Abstract.html.

Janarthanan Rajendran, Alexander Irpan, and Eric Jang. Meta-learning requires meta-augmentation. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5705–5715. Curran Associates,

Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/3e5190eeb51ebe6c5bbc54ee8950c548-Paper.pdf.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad : 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016. URL http://arxiv.org/abs/1606.05250.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know : Unanswerable questions for squad. *CoRR*, abs/1806.03822, 2018. URL http://arxiv.org/abs/1806.03822.

Sebastian Raschka. Model evaluation, model selection, and algorithm selection in machine learning. *CoRR*, abs/1811.12808, 2018. URL http://arxiv.org/abs/1811.12808.

Martin Rees. A longitude prize for the twenty-first century. *Nature News*, 509 (7501) :401, 2014.

Nils Reimers and Iryna Gurevych. Reporting score distributions makes a difference : Performance study of LSTM-networks for sequence tagging. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 338–348, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi : 10.18653/v1/D17-1035. URL https://aclanthology.org/D17-1035.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?" : Explaining the predictions of any classifier. *CoRR*, abs/1602.04938, 2016. URL http://arxiv.org/abs/1602.04938.

John R. Rice. The algorithm selection problem. volume 15 of *Advances in Computers*. 1976. doi : https://doi.org/10.1016/S0065-2458(08)60520-3. URL https://www.sciencedirect.com/science/article/pii/S0065245808605203.

Nicholas Roberts, Samuel Guo, Cong Xu, Ameet Talwalkar, David Lander, Lvfang Tao, Linhang Cai, Shuaicheng Niu, Jianyu Heng, Hongyang Qin, Minwen Deng, Johannes Hog, Alexander Pfefferle, Sushil Ammanaghatta Shivakumar, Arjun Krishnakumar, Yubo Wang, Rhea Sukthanker, Frank Hutter, Euxhen Hasanaj, Tien-Dung Le, Mikhail Khodak, Yuriy Nevmyvaka, Kashif Rasul, Frederic Sala, Anderson Schneider, Junhong Shen, and Evan Sparks. Automl decathlon : Diverse tasks, modern methods, and efficiency at scale. In Marco Ciccone, Gustavo Stolovitzky, and Jacob Albrecht, editors, *Proceedings of the NeurIPS 2022 Competitions Track*, volume 220 of *Proceedings of Machine Learning Research*, pages 151–170. PMLR, 28 Nov–09 Dec 2022. URL https://proceedings.mlr.press/v220/roberts22a.html.

Nicholas Roberts, Spencer Schoenberg, Tzu-Heng Huang, Dyah Adila, Changho Shin, Jeffrey Li, Sonia Cromp, Cong Xu, Samuel Guo, Adrien Pavao, Ameet Talwalkar, and Frederic Sala. Toward data-centric automl. In *Competition, Poster*. AutoML Conference 2023, 2023.

William B. Roberts, D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, and Michael Young. Machine learning : The high interest credit card of technical debt. 2014. URL https://api.semanticscholar.org/CorpusID:15225610.

Rebecca Roelofs, Sara Fridovich-Keil, Moritz Hardt, John Miller, Ludwig Schmidt, Vaishaal Shankar, and Benjamin Recht. A meta-analysis of overfitting in machine learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019a. URL https://proceedings.neurips.cc/paper/2019/file/ee39e503b6bedf0c98c388b7e8589aca-Paper.pdf.

Rebecca Roelofs, Vaishaal Shankar, Benjamin Recht, Sara Fridovich-Keil, Moritz Hardt, John Miller, and Ludwig Schmidt. A meta-analysis of overfitting in machine learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32 : Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9175–9185, 2019b. URL https://proceedings.neurips.cc/paper/2019/hash/ee39e503b6bedf0c98c388b7e8589aca-Abstract.html.

Jörg Rothe. *Economics and Computation : An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division.* Springer, 2015. ISBN 9783662479049.

David Rousseau and Andrey Ustyuzhanin. Machine learning scientific competitions and datasets. 2020. URL https://arxiv.org/abs/2012.08520.

Peter J. Rousseeuw. Silhouettes : A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20 :53–65, 1987. ISSN 0377-0427. doi : 10.1016/0377-0427(87)90125-7.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat. Mach. Intell.*, 1 (5) :206–215, 2019. doi : 10.1038/s42256-019-0048-x. URL https://doi.org/10.1038/s42256-019-0048-x.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3) :211–252, 2015. doi : 10.1007/s11263-015-0816-y. URL https://doi.org/10.1007/s11263-015-0816-y.

S., C. Greenberg, E. Singer, D. Reynolds, L. Mason, and J. Hernandez-Cordero. The 2019 nist speaker recognition evaluation cts challenge. In *The Speaker and Language Recognition Workshop : Odyssey 2020*, 2020. URL https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=929506.

S Saeb, L Lonini, A Jayaraman, DC Mohr, and KP Kording. Voodoo machine learning for clinical predictions. biorxiv, 059774, 2016.

Julio Saez-Rodriguez, James C Costello, Stephen H Friend, Michael R Kellen, Lara Mangravite, Pablo Meyer, Thea Norman, and Gustavo Stolovitzky. Crowdsourcing biomedical research : leveraging communities as innovation engines. *Nature Reviews Genetics*, 17(8) :470–486, 2016.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts : On the importance of regularization for worst-case generalization. *arXiv preprint arXiv :1911.08731*, 2019.

I.H. Sarker. Deep learning : A comprehensive overview on techniques, taxonomy, applications and research directions. *SN COMPUT. SCI.*, 2 :420, 2021. doi : 10.1007/s42979-021-00815-1. URL https://doi.org/10.1007/s42979-021-00815-1.

Mark Allen Satterthwaite. Strategy-proofness and Arrow's conditions : Existence and correspondence theorems for voting procedures and social welfare functions. *Journal of Economic Theory*, 10(2) :187–217, April 1975. URL https://ideas.repec.org/a/eee/jetheo/v10y1975i2p187-217.html.

Julia A. Schnabel, Christos Davatzikos, Gabor Fichtinger, Alejandro F. Frangi, and Carlos Alberola-López. Special issue on miccai 2018. *Medical Image Analysis*, 58 :101560, 2019. ISSN 1361-8415. doi : https://doi.org/10.1016/j.media.2019.101560. URL https://www.sciencedirect.com/science/article/pii/S1361841519301021.

Nihar B. Shah and Martin J. Wainwright. Simple, robust and optimal ranking from pairwise comparisons. *J. Mach. Learn. Res.*, 18 :199 :1–199 :38, 2017. URL http://jmlr.org/papers/v18/16-206.html.

Zheyan Shen, Jiashuo Liu, Yue He, Xingxuan Zhang, Renzhe Xu, Han Yu, and Peng Cui. Towards out-of-distribution generalization : A survey, 2021.

Sidney Siegel and Jr. Castellan, N. John. *Nonparametric Statistics for the Behavioral Sciences (2nd ed.)*. New York : McGraw-Hill, 1988. ISBN 978-0-07-057357-4.

John H Smith. Aggregation of preferences with variable electorate. *Econometrica*, 41 :1027–1041, 1973.

Dava Sobel. *Longitude : The true story of a lone genius who solved the greatest scientific problem of his time*. Macmillan, 2005.

Ray Solomonoff. Algorithmic probability – its discovery – its properties and application to strong ai. *Randomness Through Computation : Some Answers, More Questions*, pages 149–157, 2011.

Gustavo Stolovitzky, DON Monroe, and Andrea Califano. Dialogue on reverse-engineering assessment and methods : the dream of high-throughput pathway inference. *Annals of the New York Academy of Sciences*, 1115(1) :1–22, 2007.

Rainer Storn and Kenneth V. Price. Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.*, 11(4) :341–359, 1997. doi : 10.1023/A:1008202821328. URL https://doi.org/10.1023/A:1008202821328.

Devin P. Sullivan, Casper F. Winsnes, Lovisa Åkesson, Martin Hjelmare, Mikaela Wiking, Rutger Schutten, Linzi Campbell, Hjalti Leifsson, Scott Rhodes, Andie Nordgren, Kevin Smith, Bernard Revaz, Bergur Finnbogason, Attila Szantner, and Emma Lundberg. Deep learning is combined with massive-scale citizen science to improve large-scale image classification. *Nature Biotechnology*, 36(9) :820–828, Oct 2018. ISSN 1546-1696. doi : 10.1038/nbt.4225. URL https://doi.org/10.1038/nbt.4225.

Lisheng Sun-Hosoya, Isabelle Guyon, and Michèle Sebag. Activmetal : Algorithm recommendation with active meta learning. In Georg Krempl, Vincent Lemaire, Daniel Kottke, Adrian Calma, Andreas Holzinger, Robi Polikar, and Bernhard Sick, editors, *Workshop on Interactive Adaptive Learning @ Proceedings of the European Conference on Machine Learning Principles and Practice of Knowledge Discovery in Databases (ECML-PKDD), Dublin, Ireland, September 10th, 2018*, volume 2192 of *CEUR Workshop Proceedings*, pages 48–59. CEUR-WS.org, 2018. URL http://ceur-ws.org/Vol-2192/ialatecml_paper3.pdf.

Lisheng Sun-Hosoya, Isabelle Guyon, and Michèle Sebag. Activmetal : Algorithm recommendation with active meta learning. In *IAL 2018 workshop, ECML PKDD*, 2018.

James Surowiecki. *The wisdom of crowds*. Anchor, 2005.

Ines Thiele, Neil Swainston, Ronan MT Fleming, Andreas Hoppe, Swagatika Sahoo, Maike K Aurich, Hulda Haraldsdottir, Monica L Mo, Ottar Rolfsson, Miranda D Stobbe, et al. A community-driven global reconstruction of human metabolism. *Nature biotechnology*, 31(5) :419–425, 2013.

Sebastian Thrun, Michael Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia M. Oakley, Mark Palatucci, Vaughan R. Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary R. Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara V. Nefian, and Pamela Mahoney. Stanley : The robot that won the DARPA grand challenge. *J. Field Robotics*, 23(9) :661–692, 2006. doi : 10.1002/rob.20147. URL https://doi.org/10.1002/rob.20147.

T. Nicolaus Tideman. Independence of clones as a criterion for voting rules. *Social Choice and Welfare*, 4(3) :185–206, 1987.

David Trafimow and Michael Marks. Editorial. basic and applied social psychology. 2015.

Adrien Treuille and Rhiju Das. Scientific rigor through videogames. *Trends in biochemical sciences*, 39(11) :507–509, 2014.

E. Triantaphyllou. *Multi-Criteria Decision Making : A Comparative Study*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000. ISBN 0-7923-6607-7.

E. Triantaphyllou and S.H. Mann. An examination of the effectiveness of multidimensional decision-making methods : A decision-making paradox. *International Journal of Decision Support Systems*, 5(3) :303–312, 1989. doi : 10.1016/0167-9236(89)90037-7.

Ioannis Tsamardinos and Constantin F. Aliferis. Towards principled feature selection : Relevancy, filters and wrappers. In Christopher M. Bishop and Brendan J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, AISTATS 2003, Key West, Florida, USA, January 3-6, 2003*. Society for Artificial Intelligence and Statistics, 2003. URL http://research.microsoft.com/en-us/um/cambridge/events/aistats2003/proceedings/133.pdf.

Ioannis Tsamardinos, Elissavet Greasidou, and Giorgos Borboudakis. Bootstrapping the out-of-sample predictions for efficient and accurate cross-validation. *Mach. Learn.*, 107(12) :1895–1922, 2018. doi : 10.1007/s10994-018-5714-4. URL https://arxiv.org/pdf/1708.07180.pdf.

Hung-Yu Tseng, Yi-Wen Chen, Yi-Hsuan Tsai, Sifei Liu, Yen-Yu Lin, and Ming-Hsuan Yang. Regularizing meta-learning via gradient dropout. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

Renbo Tu, Nicholas Roberts, Mikhail Khodak, Junhong Shen, Frederic Sala, and Ameet Talwalkar. Nas-bench-360 : Benchmarking neural architecture search on diverse tasks, 2023.

Ryan Turner, David Eriksson, Michael McCourt, Juha Kiili, Eero Laaksonen, Zhen Xu, and Isabelle Guyon. Bayesian optimization is superior to random search for machine learning hyperparameter tuning : Analysis of the black-box optimization challenge 2020. In Hugo Jair Escalante and Katja Hofmann, editors, *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*, volume 133 of *Proceedings of Machine Learning Research*, pages 3–26. PMLR, 06–12 Dec 2021. URL https://proceedings.mlr.press/v133/turner21a.html.

Kadir Uludag and Jiao Tong. Testing creativity of chatgpt in psychology : interview with chatgpt. *Preprint*, 2023.

Rohit Vashisht, Anupam Kumar Mondal, Akanksha Jain, Anup Shah, Priti Vishnoi, Priyanka Priyadarshini, Kausik Bhattacharyya, Harsha Rohira, Ashwini G Bhat, Anurag Passi, et al. Crowd sourcing a new paradigm for interactome driven drug target identification in mycobacterium tuberculosis. *PloS one*, 7(7) :e39808, 2012.

Mariya I. Vasileva. The dark side of machine learning algorithms : How and why they can leverage bias, and what can be done to pursue algorithmic fairness. In Rajesh Gupta, Yan Liu, Jiliang Tang, and B. Aditya Prakash, editors, *KDD '20 : The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 3586–3587. ACM, 2020. doi : 10.1145/3394486.3411068. URL https://doi.org/10.1145/3394486.3411068.

Cédric Villani. *The Wasserstein distances*, pages 93–111. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-71050-9. doi : 10.1007/978-3-540-71050-9_6. URL https://doi.org/10.1007/978-3-540-71050-9_6.

Ulrike von Luxburg, Robert C. Williamson, and Isabelle Guyon. Clustering : Science or art? In Isabelle Guyon, Gideon Dror, Vincent Lemaire, Gra-

ham W. Taylor, and Daniel L. Silver, editors, *Unsupervised and Transfer Learning - Workshop held at ICML 2011, Bellevue, Washington, USA, July 2, 2011*, volume 27 of *JMLR Proceedings*, pages 65–80. JMLR.org, 2012. URL http://proceedings.mlr.press/v27/luxburg12a.html.

Jérôme Waldispühl, Attila Szantner, Rob Knight, Sébastien Caisse, and Randy Pitchford. Leveling up citizen science. *Nature Biotechnology*, 38(10) :1124–1126, 2020.

A.S. Weigend and N.A. Gershenfeld. Results of the time series prediction competition at the santa fe institute. In *IEEE International Conference on Neural Networks*, pages 1786–1793 vol.3, 1993. doi : 10.1109/ICNN.1993.298828.

Neil A. Weiss. *A Course in Probability*. Addison–Wesley, 2005.

D. R. Woodall. Properties of preferential election rules. *Voting Matters*, pages 8–15, December 1994a.

Douglas Woodall. Properties of preferential election rules. *Voting Matters*, pages 8–15, December 1994b.

Zhen Xu, Wei-Wei Tu, and Isabelle Guyon. Automl meets time series regression design and analysis of the autoseries challenge. *CoRR*, abs/2107.13186, 2021.

Zhen Xu, Sergio Escalera, Adrien Pavão, Magali Richard, Wei-Wei Tu, Quanming Yao, Huan Zhao, and Isabelle Guyon. Codabench : Flexible, easy-to-use, and reproducible meta-benchmark platform. *Patterns*, 3(7) : 100543, 2022. ISSN 2666-3899. doi : https://doi.org/10.1016/j.patter.2022.100543. URL https://www.sciencedirect.com/science/article/pii/S2666389922001465.

Deshraj Yadav, Rishabh Jain, Harsh Agrawal, Prithvijit Chattopadhyay, Taranjeet Singh, Akash Jain, Shivkaran Singh, Stefan Lee, and Dhruv Batra. Evalai : Towards better evaluation systems for AI agents. *CoRR*, abs/1902.03570, 2019. URL http://arxiv.org/abs/1902.03570.

Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin P. Bennett. Privacy preserving synthetic health data. *European Symposium on Artificial Neural Networks (ESANN)*, 2019a.

Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin P. Bennett. Privacy preserving synthetic health data. In *27th European Symposium on Artificial Neural Networks, ESANN 2019, Bruges, Belgium, April 24-26, 2019*, 2019b. URL http://www.elen.ucl.ac.be/Proceedings/esann/esannpdf/es2019-29.pdf.

Andrew Yale, Saloni Dash, Ritik Dutta, Isabelle Guyon, Adrien Pavao, and Kristin P. Bennett. Generation and evaluation of privacy preserving synthetic health data. *Neurocomputing*, 416 :244–255, 2020. doi : 10.1016/j.neucom.2019.12.136. URL https://doi.org/10.1016/j.neucom.2019.12.136.

Junichi Yamagishi, Xin Wang, Massimiliano Todisco, Md. Sahidullah, Jose Patino, Andreas Nautsch, Xuechen Liu, Kong Aik Lee, Tomi Kinnunen, Nicholas W. D. Evans, and Héctor Delgado. Asvspoof 2021 : accelerating progress in spoofed and deepfake speech detection. *CoRR*, abs/2109.00537, 2021. URL https://arxiv.org/abs/2109.00537.

Jaesik Yoon, Taesup Kim, Ousmane Dia, Sungwoong Kim, Yoshua Bengio, and Sungjin Ahn. Bayesian model-agnostic meta-learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/e1021d43911ca2c1845910d84f40aeae-Paper.pdf.

H. P. Young. Social choice scoring functions. *SIAM Journal on Applied Mathematics*, 28 :824–838, 1975a.

H. P. Young. Social choice scoring functions. *SIAM Journal on Applied Mathematics Vol. 28, No. 4*, pages 824 – 838, 1975b.

Yongli Zhang and Yuhong Yang. Cross-validation for selecting a model selection procedure. 2015a. URL http://users.stat.umn.edu/~yangx374/papers/ACV_v30.pdf.

Yongli Zhang and Yuhong Yang. Cross-validation for selecting a model selection procedure. *Journal of Econometrics*, 187(1) :95–112, 2015b. URL https://EconPapers.repec.org/RePEc:eee:econom:v:187:y:2015:i:1:p:95-112.

Julian G. Zilly, Jacopo Tani, Breandan Considine, Bhairav Mehta, Andrea F. Daniele, Manfred Diaz, Gianmarco Bernasconi, Claudio Ruch, Jan Hakenberg, Florian Golemo, A. Kirsten Bowser, Matthew R. Walter, Ruslan Hristov, Sunil Mallya, Emilio Frazzoli, Andrea Censi, and Liam Paull. The AI driving olympics at neurips 2018. *CoRR*, abs/1903.02503, 2019. URL http://arxiv.org/abs/1903.02503.

James Zou and Londa Schiebinger. Ai can be sexist and racist—it's time to make it fair, 2018.

Łukasz Kidziński et al. Ai for prosthetics challenge, 2018. URL https://www.crowdai.org/challenges/neurips-2018-ai-for-prosthetics-challenge.