# Adversarial Sampling Attacks Against Phishing Detection

Hossein Shirazi, Bruhadeshwar Bezawada, Indrakshi Ray, Charles Anderson

# Adversarial Sampling Attacks Against Phishing Detection

Hossein Shirazi[1], Bruhadeshwar Bezawada[2],
Indrakshi Ray[1], and Chuck Anderson[1]

[1] Colorado State University, Fort Collins CO 80523, USA
[2] Mahindra È cole Centrale, Hyderabad, Telangana, India

**Abstract.** Phishing websites trick users into believing that they are interacting with a legitimate website, and thereby, capture sensitive information, such as user names, passwords, credit card numbers and other personal information. Machine learning appears to be a promising technique for distinguishing between phishing websites and legitimate ones. However, machine learning approaches are susceptible to *adversarial learning* techniques, which attempt to degrade the accuracy of a trained classifier model. In this work, we investigate the robustness of machine learning based phishing detection in the face of adversarial learning techniques. We propose a simple but effective approach to simulate attacks by generating adversarial samples through direct feature manipulation. We assume that the attacker has limited knowledge of the features, the learning models, and the datasets used for training. We conducted experiments on four publicly available datasets on the Internet. Our experiments reveal that the phishing detection mechanisms are vulnerable to adversarial learning techniques. Specifically, the identification rate for phishing websites dropped to 70% by manipulating a single feature. When four features were manipulated, the identification rate dropped to zero percent. This result means that, any phishing sample, which would have been detected correctly by a classifier model, can bypass the classifier by changing at most four feature values; a simple effort for an attacker for such a big reward. We define the concept of *vulnerability level* for each dataset that measures the number of features that can be manipulated and the cost for each manipulation. Such a metric will allow us to compare between multiple defense models.

**Keywords:** Phishing, Machine Learning, Adversarial Sampling, Classifiers

## 1 Introduction

### 1.1 Motivation

Phishing, as defined in [1], is an attempt to obtain sensitive information such as user-names, passwords, and credit card details by masquerading as a trustworthy entity in an electronic communication. The first recorded mention of the term is found in the hacking tool against American Online (AOL) users in the 1995

named *AOHell* while the technique was elaborated earlier in a presentation to the International HP Users group, Interix, by Felix and Hauck in 1987 [2]. Phishing attacks have shown remarkable resilience against a multitude of defensive efforts, and attackers continue to generate sophisticated phishing websites that closely mimic legitimate websites. While there were 328,000 unique attacks reported in 2007, this number almost quadrupled by 2017 [3].

Phishing, when viewed as a social-engineering attack, cannot be solved solely by educating the end users, and hence, automatic detection techniques are essential. Several defenses were proposed against phishing attacks, such as URL blacklisting, keyword-based filtering, IP address filtering, and machine learning based techniques. Solutions like URL-blacklisting are no longer effective as attackers can bypass such techniques through simple URL manipulation or by hosting websites on popular free hosting services on the Internet. Machine learning based techniques appear to be a promising direction.

### 1.2 Problem Statement

The studies in the existing literature emphasize on feature definition or enhancing the statistical learning models to discriminate between phishing and legitimate websites. The state-of-the-art solutions for phishing detection [4–8] use engineered features based on observations made by the research experts in this domain on publicly available datasets. One crucial assumption, in existing machine learning based phishing detection approaches, is that the training data collection process is independent of the attackers' actions [9]. However, in adversarial contexts, *e.g.* phishing or spam filtering, this is far from the reality as attackers either generate noisy data samples or generate new attack samples by manipulating features of existing ones. The noisy data samples result in a classification model with low accuracy and requires a higher effort for an attacker. The manipulation of features results is a more dangerous scenario wherein an attacker can bypass an existing classifier without much effort. In this work, we explore and study the effect of adversarial sampling on phishing detection algorithms in depth, starting with some simple feature manipulation strategies, and show some surprising results that demonstrate impact on the classification accuracy with trivial feature manipulation.

### 1.3 Proposed Approach and Key Contributions

We gathered four separate publicly available datasets developed by other researchers and applied adversarial sampling techniques to evaluate the robustness of the trained model against artificially generated adversarial samples. Although we do not show any solution to address this current threat, we show the vulnerability of the current approaches, and explored the robustness of the datasets against the engineered features, and the learning models. Our key contributions are as follows:

– We modeled the threat against the current defense and detection mechanism and explained the attackers' access and knowledge, which the attackers utilize to attack any given trained classifier model.

- We define the vulnerability level of phishing instances, which quantifies the attackers' efforts, and describe an approach to manipulate phishing instances and create new samples.
- We surveyed a full range of phishing detection techniques focusing on the machine learning based approaches. We showed the weakness of some well-known machine learning approaches and emphasized on how a phisher can generate new phishing website instances to evade a trained classifier in each of these approaches.
- We built an experimental setup and conducted a wide range of experiments and analyzed how vulnerable the datasets and learning model are by testing against the adversarial samples.

The rest of this paper is organized as follows. In Section 2, we describe a wide range of defense mechanisms against phishing attacks in the literature. Also, we describe the various adversarial attacks against the machine learning classifiers in non-phishing domains. In Section 3, we model the threat from three points of view: attackers' goal, knowledge, and influence. In Section 4, we simulated adversarial sampling attack followed by the assessing vulnerability level and quantifying the cost of the attack. In Section 5, we explain the results of our experiments to prove the robustness of the classifiers and datasets against these attacks. In Section 6, we conclude the paper and discuss some future work.

## 2 Related Work

### 2.1 Machine Learning for Phishing Detection

For phishing website detection, machine learning algorithms are well suited as they can assimilate common attack patterns such as hidden fields, keywords, and page layouts, across multiple phishing data instances and create learning models that are resilient to small variations in future unknown phishing data instances. In the prior machine learning approaches, researchers engineered novel sets of features from diverse perspectives based on public datasets of phishing and legitimate websites. While these approaches have demonstrated excellent results for detecting phishing websites, they also suffer from serious disadvantages due to adversarial sampling as we show in the following discussion.

Niakanlahiji *et al.* [4] introduced PhishMon, a scalable feature-rich framework with a series of new and existing features derived from HTTP responses, SSL certificates, HTML documents, and JavaScript files. The authors reported an accuracy of 95% on their datasets.

According to a Symantec report [10], the number of URL obfuscation based phishing attacks was up by 182.6% in 2017. Some URL obfuscation techniques used by attackers are: misspelling of the targeted domain name, using the targeted domain name in other parts of the URL like the sub-domain, adding sensitive keywords like 'login', 'secure' or 'https' etc. Sahinguz *et al.* [11] proposed a real-time detection mechanism based on Natural Language Processing (NLP) of URLs. The technique used a large dataset without requiring third-party services, and focused on features derived from URL obfuscation, and achieved an accuracy of more than 95%.

Verma *et al.* [12] defined lexical, distance, and length related features for the detection pf phishing URLs. They employed the two-sample Kolmogorov-Smirnov statistical test along with other features to detect phishing websites. They conducted a series of experiments on four large proprietary datasets and reported an accuracy of 99.3% with a false positive rate of less than 0.4%.

Jiang *et al.* [5] merged information from DNS and the URL to develop a Deep Neural Network (DNN) with the help of NLP to detect phishing attacks. While other approaches need to specify features explicitly, this method extracts hidden features automatically. The approach relies on the information from DNS and thus, requires third-party services.

Attackers use Domain Generation Algorithms (DGA) to dynamically generate a large number of random domain names for adversarial purposes including phishing attacks. Pereira *et al.* [6] introduced an approach for detecting such domains. These domains are considered as legitimate for detection mechanisms and human analysis. The authors used a graph-based algorithm to extract the dictionaries that have been used by attackers to detect malicious domains.

While these proposed approaches are promising, they often do not considering the page content. Attackers have full control over the URL except for the Second Domain Level (SLD), and thus, they can create any URL to bypass the classifier. Also, the content of the website is the most critical factor to lure the end-users rather than the URL or domain name themselves. Therefore, any solution not considering the website content would not be useful in the real world.

Tian *et al.*, [13] studied five types of domain squatting over a large DNS dataset of over 224 million registered domains. They identified 657 thousand domains that potentially targeted 702 popular websites. Using visual and Optical Character Recognition (OCR) analysis, they created a highly accurate classifier and found more than one thousand new phishing instances of which 90% of them successfully evaded well-known blacklists even after one month. The authors combined two powerful techniques: domain squatting and OCR analyses on a large dataset. The advantage of this approach is in finding new instances that evaded the current classifiers. However, there is significant cost in keeping this information current.

Shirazi *et al.* [7] observed two concerns with existing machine learning approaches: a large number of training features and bias in the type of datasets used. The study focused on the features derived from the domain name usage in phishing and legitimate websites and reported an accuracy of $97 - 98\%$ on the chosen datasets. To prove the performance of the whole model, they evaluated it with unseen phishing samples from a completely different source and achieved a detection rate of 99%.

Recently, Li *et al.* [8] proposed an approach to extract the features from both URL and web page content and ran multiple machine learning techniques including GBDT, XGBoost, and LightGBM, in multiple layers, referred to as stacking approaches. The URL-based feature set includes eight features in total *e.g. using IP address*, *suspicious symbols*, *sensitive vocabulary*. The HTML based category includes features like *Alarm Window*, *Login Form*, *Length of HTML*

*Content*. With The dataset has 20 features in total. The experiment has been conducted on three datasets, of which two are large ones with 50K instances and the accuracy is more than 97% in all cases. Although this approach is similar to recent machine learning approaches and does not use third-party services, it is similar to other previous work like [7].

### 2.2 Learning in Adversarial Context

The proposed defense mechanisms in the literature widely employed machine learning techniques to counter phishing attacks. However, adversarial sampling attacks can threaten the current defense mechanisms. While there are some general analysis of the vulnerabilities of classification algorithms and the corresponding attacks [14], to the best of our knowledge, there is no other study on adversarial sampling in the context of the phishing attacks. Thus far, researchers studied and formulated these threats in a general manner or in other application contexts like image recognition. In the following, we briefly explore these efforts.

Dalvi *et al.* [9] studied the problem of adversary learning as a game between two active agents: data miner and adversary. The goal of each agent is to minimize its cost and maximize the cost to the other agent. The classifier adapts to the environment and its settings either manually or automatically in this approach. The authors assumed that both sides, including data miner and adversary, have perfect knowledge about a problem. This assumption, however, does not hold in many situations. For example, in the phishing detection system, the adversary does not know the training set or the actual classification algorithm used. In Section 3, we modeled the adversary and elaborated why the adversary cannot have perfect knowledge. The attackers may directly or indirectly target the vulnerabilities in the feature selection procedure. Although the attackers might target the trained classification system, it still is an indirect attack on the chosen features.

Xiao *et al.* [15] explored the vulnerabilities of feature selection algorithms under adversarial sampling attacks. They extended a previous framework [16] to investigate the robustness of three well-known feature selection algorithms.

There are few approaches that create more secure machine learning models. Designing a secure learning algorithm is one way to build a more robust classifier against these attacks. Demontis *et al.* [17] investigated a defense method that can improve the security of linear classifier by learning more evenly-distributed feature weights. They presented a secure SVM called Sec-SVM to defend against evasion attacks with feature manipulation. Wang [18] theoretically guaranteed robustness of k-nearest neighbors algorithm in the context of adversarial examples. They introduced a modified version of k-nearest neighbor classifier while k is equal to 1 and theoretically guaranteed its robustness in a large dataset.

Finally, there are some tools for bench-marking and standardizing performance of machine learning classifier against adversarial attacks in the literature. *Cleverhans* [19] is an open-source library that provides an implementation of adversarial sample construction techniques and adversarial training for image datasets. Given the lack of such bench-marking tools for the phishing problem, we tested our approach with our own attack strategies and implementation.

## 3 Threat Model

In this section, we model the adversarial sampling attack against machine learning based phishing detection approaches. We start with the attacker's *goal*, *knowledge*, and *influence* in general machine learning solutions, and then we explain them in the context of phishing problem. We model the adversarial sample generation for existing phishing instances based on the attacker's abilities and then evaluate the cost that the adversary has to pay for the successful execution of this attack. Finally, we define the vulnerability level for the dataset.

### 3.1 Attacker's Goal

Biggioa *et al.* explored three different goals for the attackers namely *security violation*, *attack specificity*, and *error specificity* [20]. The goal of an attacker in the *security violation* is to evade well-known security metrics including availability, privacy, and integrity. The attacker may violate the availability of the system by denial-of-service attack. In this case, if the system cannot accomplish the desired task due to the attacker's behavior, the availability of the service would be affected. The attacker needs to obtain sensitive and private information of users with approaches like reverse-engineering to violate the user's privacy.

In the phishing context, the adversary will attack the integrity of the system. The integrity is violated if the attack does not violate the regular system behavior; however, the attacker violates the accuracy of the classifier *e.g.* by luring classifier to label maliciously crafted phishing instances as legitimate to evade the classifier. The attack *specificity* depends on whether an attacker wants to mis-classify a specific set of samples (like phishing) or any given sample. The error *specificity* relates to the attacker's effort to increase a specific type of error in the system and degrade other classifier scores.

In this study, we consider that the adversary desires to attack the *specificity* of the learning model. This leads to the incorrect classification of the adversarial phishing samples as legitimate and thereby, these samples will deceive the end users. Also, with respect to error *specificity*, the adversary wants to decrease the True Positive Rate (TPR).

### 3.2 Attacker's Knowledge

An attacker may have different levels of knowledge about the machine learning model. An attacker might have detailed knowledge, *i.e.*, *white-box* or *perfect knowledge*, minimal knowledge about the model called *zero knowledge* [15, 20] and limited knowledge about the model known as the *gray-box*. If the adversary knows everything about the learning model, parameters, and the training dataset including the classifier parameters, then the attacker has *perfect knowledge*. In the *zero knowledge*, the adversary can probe the model by sending instances and observing the results. The adversary infers information about the model by choosing appropriate data samples. In the *limited knowledge*, it is assumed that adversary knows about features and their representation, and the learning

algorithm. However, the adversary does not know about the training set or the algorithm's parameters.

From the dataset point of view, the attacker may have partial or full access to the training dataset. Attacker may also have partial or full knowledge about the feature representation or feature selection algorithm and its criteria. In the worst case scenario, an attacker may know about the subset of selected features. In our study, we assumed that the adversary has *limited knowledge*. The adversary knows about the classifier model and the feature set but does not know about training set, the classifier, or classifier's training parameters.

### 3.3 Attacker Influence

Two major types of attacker influence have been defined in the literature namely *poisoning* and *evasion* attacks. The *poisoning* attack refers to the case where the adversary injects adversarial instances into the training phase. This injection leads to bypass data later on in the testing phase of the experiment or even at the practical usage of the model. For example, email providers use spam detection services to block emails that include a link to phishing websites. The email system gives the users ability to override the email's label *e.g.* re-labeling a spam email as non-spam to deal in cases of False Positive detection, The system benefits from user's labeling to improve the accuracy by updating the training set. However, in a *poisoning* attack, an attacker with an authorized email account in the system can re-label the correctly detected spam emails as non-spam to poison the training set of the classifier.

While the attacker does not have access to the training set in the *evasion* attack, it tries to intentionally and smartly manipulate features to avoid samples being labeled correctly by the classifier at the testing phase. Similar to the previous example on the spam detection system, a phisher may send an email with intentionally misspelled words to evade the classifier.

In this work, we assume that the attacker has a *limited knowledge* about the learning model, but has unlimited access to the *predict* function of the learning model. The attacker can test as many instances as needed and get the results. With this assumption, an attacker can create a large number of new samples and test them against the classifier to see if they can bypass the model. In the next section, we describe our adversarial sampling approach and outline our method for measuring the effectiveness of the samples in lowering the classifier's accuracy. Section 5 studies how this attack can be effective by showing the degradation of the classifier's score under this attack.

## 4 Adversarial Sampling for Phishing

We simulate the attacker's approach to generate new adversarial samples based on the existing phishing instances that are detected by the classifier. The adversary generates new instances based on these current phishing instances to check whether the generated instances are able to evade the classifier. We assume that the attacker has full control on the URL and web page content except for the domain name, which is unique. The attacker has *limited knowledge* about the classifier and features, as we discussed earlier.

### 4.1 Defining the Dataset

We use similar notation to that used in [15]. The whole dataset has been generated by a procedure $\mathcal{P} : \mathcal{X} \mapsto \mathcal{Y}$. In the experiment, we defined two types of instances: Legitimate ($L$) and Phishing ($P$). A learning algorithm trains from this dataset and will label the new instances. Each instance in the dataset has $d$ features that are represented as a $d$-dimensional vector and is labeled as legitimate ($L$) or phishing ($P$).

$$x_i = [x_i^1, \cdots x_i^d]^T \in \mathcal{X} \tag{1}$$

Each instance relates to the target label of $y_i \in \mathcal{Y}$ , $\mathcal{Y} \in \{0, 1\}$. We denote this set $D$ with $n$ samples as follows: $\mathcal{D} = \{x_i, y_i\}_{i=1}^n$. The set $\mathcal{T}$ is the subset with $t$ instances that the adversary can access, $\mathcal{T} \subseteq \mathcal{D}$ , $t \leq n$

### 4.2 Selecting Features for Manipulation

To specify a subset of features, we introduce the notation $\Phi = \{0, 1\}^d$, where each element denotes whether the corresponding feature has been selected (1) or not (0). The first step for creating adversarial samples is to select one or more features for manipulation. $\Phi^s$ denote the set of all possible combinations of $s$ features, $\binom{n}{s}$, that have been selected and $\pi_i^s$ denotes $i^{th}$ such choice of features. For example, $\pi_1^3 = (0, 1, 1, 1, 0)$ means that, the first combination from $\Phi^s$, chooses features 2, 3, and 4, for manipulation. We formalize this in Equation 2:

$$\pi_i^s \in \Phi^s \quad where\ i \in \binom{n}{s} \quad and \quad \sum_{i=1}^{d} \pi_i^s = s \tag{2}$$

Assigning new feature values is the next step after defining the subset of features for manipulation. We assumed that each feature value may be replaced by values that appeared in existing phishing instances. The intuition is that, if the value has been found to be assigned to that feature previously for a phishing instance, then the feature is more likely to get that value again in another phishing instance.

Let $T^i$ denote the set of all values that have appeared for the feature $i$ among the phishing instances. For example, $T^2 = \{-1, 0, 1\}$ denotes existing phishing instances have values $-1$, 0, and 1 in the second feature.

For generating new instances, first, we need to generate all possible feature combinations with different lengths and then, for each combination, we need to permute all possible feature values from $T^i$. This process is done only for phishing instances that have been predicted correctly by the classifier. Algorithm 1 explains this process for a given phishing instance. It shows how the adversarial instances will be generated based on an original input and desired features for manipulation. There are two inputs for Algorithm 1: an original phishing instance and the selected features to manipulate, and returns as output, the new adversarial instances that have been generated. In lines 2 and 3, the algorithm loops over all of the selected features for manipulation. The algorithm gets all available values for them from the array $T$ that is previously defined. The algorithm adds a series of available feature values to the list $L$. Line 5 calculates a

**Algorithm 1:** Generating the Adversarial Samples

**Result:** New Adversarial Samples
**Input :** $x$ , $selFeatures$
**Output:** genSamples

Let $L$ and $genSamples$ be new array
/* Get possible values for selected features */
**foreach** *featurePos in* $selFeatures$ **do**
    L.append ← $T[featurePos]$
**end**
/* Product possible values to generate all combination */
L_pr ← $product(L^*)$
/* Generate new instances based on new feature values */
**foreach** *new_val in L_pr* **do**
    /* Making a temporary copy of instance */
    temp ← $Copy(x)$
    **for** $k \leftarrow$ 0 *to* $Len($ *new_val* $)$ **do**
        /* Override the feature value with new value */
        temp[$x[k]$] = new_val[k]
    **end**
    /* Adding the new instance to the result array */
    genSamples.append ( temp )
**end**
return genSamples

*Product* function over array $L$ to calculate all possible values for selected features and saves them in $L\_pr$.

Now, each row in $L\_pr$ has the values for all of the selected features for manipulation. We make another loop over $L\_Pr$ to assign new values to the original phishing instance $x$. The algorithm saves them in the result array of $genSamples$.

### 4.3 Adversary Cost

Attackers have to handle two challenges for generating adversarial instances. From a machine learning point of view, the dataset includes vectors, but the attacker has to change the website in a way that it generates the desired vector similar to adversarial samples. This is not a trivial process, and it has considerable cost for the attacker. Whereas adversarial samples may have a higher chance of evading the classifier, but they may not be visually or functionally similar to the targeted websites. This increases the chance of being detected by the end-user. Thus, the adversary wants to minimize two parameters: the number of manipulated features and the assigned feature values. We consider this as a cost function for the adversary.

In the previous section, we discussed how the attacker controls the number of manipulated features, but it is not the only parameter. If the manipulated

feature values are far from the original values, it will increase the chance of evading the classifier. We study this hypothesis in Section 5. But, this will also change the visual appearance or behavioral functionality of the website from the targeted website, thereby, increasing the chance of phishing website being detected by the end-user.

In this work, we used the *Euclidean distance* between the original phishing sample and newly generated sample to estimate the cost, a higher distance indicates a larger cost. Consider $x_i$ to be a phishing instance and $x_i^{'}$ a manipulated one based on the original $x_i$ instance. Both are vectors of size $n$. The *Euclidean distance* between $x_i$ and $x_i^{'}$ will be calculated by Equation 3:

$$d(x_i, x_i^{'}) = \sqrt{\sum_{k=1}^{n} (x_i^k - x_i^{'k})^2} \tag{3}$$

If $l$ is the number of manipulated features to generate $x_i^{'}$ from $x_i$, and $d$ is *Euclidean distance* between them, the total cost $c$ will be derived from this equation: $\mathcal{C}(x_i, x_i^{'}) = (l, d)$. This tuple will be used to evaluate the total cost for generating the adversarial instances.

### 4.4 Vulnerability Level

A phishing instance that has been predicted correctly is vulnerable at the level of $l$ with the cost of $d$ if there is at least one adversarial instance that can bypass with $l$ manipulated features and distance $d$. We call this instance vulnerable if, by manipulating $l$ features and with the distance of $d$, it can bypass the classifier. The goal of the attacker here is optimizing the $l$ and $d$; a multi-objective optimization problem for the attacker. For example, if we have a phishing instance, which has been detected by the classifier, but there is the new instance generated by manipulating 3 features and a *Euclidean distance* of 2.7 bypasses the classifier, the original sample is vulnerable at the level of 3 with a cost of 2.7.

## 5 Experiments and Results

In this section, we show the effectiveness of our threat model and proposed adversarial sampling attack that degrades the accuracy and efficacy of existing learning models. First, we discuss the datasets utilized and then, we elaborate on three different experiments we have conducted and their results.

### 5.1 Used Datasets

We obtained four publicly available phishing datasets on the Internet and the details of these datasets are given below.

**Dataset** 1**: DS-1**: This set includes 1000 legitimate websites from *Alexa.com* and 1200 phishing websites from *PhishTank.com*; 2200 in total. Each instance in this dataset has eight features and all are related to the domain name of

the websites. The features used are domain length, presence of non-alphabetic character in the domain name, the ratio of hyperlinks referring to the domain name, the presence of HTTPS protocol, matching domain name with copyright logo, and matching domain name with the page title. With these features, Shirazi *et al.* [7] reported an ACC of 97-98% in the experiments, which is significantly high.

**Dataset 2: DS-2**: Rami *et al.* [21] created this dataset in 2012 and shared it with UCI machine learning repository [22]. This set includes 30 features and are categorized into five categories: *URL based*, *abnormal based*, *HTML-based*, *JavaScript based*, and *domain-based* features. This dataset includes 4898 legitimate instances from *Alexa.com* merged with 6158 phishing instances from *PhishTank.com*; more than 11000 in total making it the most extensive dataset that we have used in this study.

**Dataset 3: DS-3**: In 2014, Abdelhamid *et al.* [23] shared their dataset on UCI machine learning repository [22]. This dataset includes 651 legitimate websites and 701 phishing websites; 1352 instances in total and includes ten features combination of third-party services and HTML based features for each instance. Authors report an ACC between 90%-95% in their experiments.

**Dataset 4: DS-4**: This dataset is the most recent dataset publicly available in the literature that we could find and was published in 2018. It has been created by Tan *et al.* [24] and was published on Mendeley [3] dataset library. This set contains 5000 websites from *Alexa.com* and as well as those obtained by web crawling, labeled as legitimate, and 5000 phishing websites from *PhishTank.com* and *OpenPhish.com*. The authors collected this data from January to May 2015 and from May to June 2017. This dataset includes 48 features, a combination of URL-based, and HTML-based features. While this dataset includes the URL length and it may bias the dataset as Shirazi *et al.* [7] explained it, but we kept all features.

Table 1 summarises the number of instances, features, and the portion of legitimate vs phishing instances in each dataset. We have a dataset with a large number of instances, DS-2, and DS-4 with 11000 and 10000 respectively. We also have small dataset DS-3 with 1250 instances. With respect to the number of features, DS-1, with just seven features is a dataset with a limited number of features and in comparison, DS-4 with 48 features is a large dataset. Also, we used an unbiased dataset like DS-1 and used DS-4 as well though it may be biased concerning some of the features like URL length. Besides, the features in each dataset are selected from different points of view such as URL-based features in DS-2, DS-3, and DS-4, or domain-related features in DS-1, and HTML-Based features in DS-2 and DS-4. These variations validate our hypothesis in a stronger and more general sense. Also, it shows that adversarial sampling is a serious problem that may be happening in different situations and needs to be addressed.

[3] https://data.mendeley.com/

**Table 1.** Number of instances, features, and portion of legitimate and phishing websites in each dataset

| Dataset | Data shape (#) | | Instances (%) | |
|---|---|---|---|---|
| | Size | Features | Legitimate | Phishing |
| **DS-1** | 2210 | 7 | 44.71 | 55.29 |
| **DS-2** | 11055 | 30 | 55.69 | 44.31 |
| **DS-3** | 1250 | 9 | 43.84 | 56.16 |
| **DS-4** | 10000 | 48 | 50.0 | 50.0 |

### 5.2 Exp-1: Evaluation of Datasets

In the first experiment, we tested the performance of each dataset against a wide range of classifiers. The experiment is as follows. We labeled phishing websites in all datasets as $+1$ and legitimate websites as $-1$. We used five-fold cross-validation to avoid issues of over-fitting and to test the performance of the learning model against unknown data instance classification. We used six different classifiers namely Decision Tree Decision Tree (DT), Gradient Boosting (GB), Random Forest (RF) , K-Nearest Neighbors (KNN), and Support Vector Machine (SVM) with two different kernels: Linear (lin) and Gaussian (rbf) to make the comparison between classifiers. We repeated each experiment 10 times and reported the average and standard deviation of the results. Table 2 explains the achieved results in this experiment.

For DS-1, RF and GB both generate the highest ACC and the TPR in each classifier is almost the same. Also, DS-1 has the best average of TPR among all classifiers. This means that, despite different classifiers, the features are well-defined. RF gives the best TPR (94.25%) and ACC (95.76%) on DS-2. Interestingly, the DT does not generate a good TPR (86.77%).

The experiments on DS-3 dataset did not yield a high TPR or the ACC. Both GB and SVM with Gaussian kernel has the TPR of 87%, which are not that much good. The best ACC, for this dataset, is from GB, with 83%. The experiment on DS-3 gave very good results. Both GB and RF gave a TPR over 97% and accuracy of 97%, which are very high. Also, this dataset has the best average of ACC among different classifiers meaning this dataset performs very well with different types of classifiers. With six different classifiers, the experiments on both DS-1 and DS-4 show an average ACC of more than 94%, which is significantly high. This confirms that these datasets are well-defined and have a good set of distinguishing features.

We used a single metric of f1 to compare all classifiers and datasets together. Table 3 shows the best f1 score for each dataset with the classifier that has produced that result. It is evident from this table that both GB and RF generate the best results among all of the experiments, so we selected these two classifiers for the next experiments.

### 5.3 Generating Adversarial Samples

In each dataset, we reserved 200 phishing instances and then trained the model without the 200 reserved phishing instances. The generated adversarial samples

**Table 2.** Evaluation of model against different classifiers with two metrics.

(a) TPR

| Cls. | DS-1 | DS-2 | DS-3 | DS-4 | Avg. |
|---|---|---|---|---|---|
| **DT** | 95.25 | 86.77 | 84.97 | 96.14 | **95.25** |
| **GB** | 96.18 | 92.25 | 87.23 | 97.65 | **96.18** |
| **KNN** | 95.93 | 90.61 | 84.95 | 93.97 | **95.93** |
| **RF** | 96.25 | 94.25 | 85.84 | 97.85 | **96.25** |
| **SVM(l)** | 95 | 89.62 | 86.71 | 94.93 | **95** |
| **SVM(r)** | 93.67 | 91.88 | 87.88 | 95.69 | **93.67** |
| **Best** | **96.25** | **94.25** | **87.88** | **97.85** | |

(b) ACC

| Cls. | DS-1 | DS-2 | DS-3 | DS-4 | Avg. |
|---|---|---|---|---|---|
| **DT** | 94.8 | 92.1 | 82.51 | 95.73 | **91.29** |
| **GB** | 95.49 | 94.32 | 83.76 | 97.52 | **92.77** |
| **KNN** | 94.82 | 92.21 | 81.16 | 93.76 | **90.49** |
| **RF** | 95.35 | 95.76 | 82.89 | 97.8 | **92.95** |
| **SVM(l)** | 93.96 | 92.4 | 79.16 | 94.38 | **89.98** |
| **SVM(r)** | 93.96 | 94.14 | 82.4 | 95.2 | **91.43** |
| **Best** | **95.49** | **95.76** | **83.76** | **97.8** | |

**Table 3.** The classifier that holds best f1 on each dataset has been selected. TPR and ACC are also reported for comparison

| Metric | DS-1 | DS-2 | DS-3 | DS-4 |
|---|---|---|---|---|
| **Best Classifier** | **GB** | **RF** | **GB** | **RF** |
| **Best f1** | **95.94** | **95.17** | **85.83** | **97.8** |
| **TPR** | 96.18 | 94.25 | 87.23 | 97.85 |
| **ACC** | 95.49 | 95.76 | 83.76 | 97.8 |

need to be similar and valid to the phishing examples; otherwise, those cannot be assumed to be phishing instances. To assign new values to the features and generate new instances, we just used previously seen values in the phishing instances. With this strategy, it is guaranteed that the newly assigned value is valid and has already been seen in other phishing instances in the dataset. We discussed this process earlier in Section 4. We randomly selected combination of features, up to four different features, and changed the values of each feature with all possible feature values.

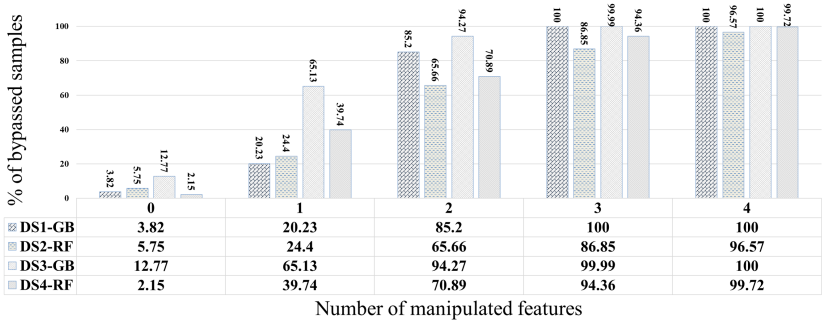
After creating each new sample, we tested our new sample against the selected classifier and checked whether it could bypass the classifier or not. If it did, we consider the original phishing instance to be a vulnerable instance. We calculate the distance between the new instance and the original one to find the closest instance that can bypass the classifier.

### 5.4 Exp-2: Robustness of Learning Model

This experiment studies the robustness of datasets and learning models against generated adversarial samples. We selected one classifier that performs best for each dataset based on the f1 score from Table 3. For the datasets DS-1 and DS-3, we selected GB and RF for DS-2 and DS-4.

In this experiment, we counted the number of reserved phishing instances that are vulnerable. This means that, there should be at least one optimized manipulated instance based on the original sample that can bypass the classifier. With small perturbation on these instances, they can bypass the classifier and elude the users to release their critical information. Based on our hypothesis, these are vulnerable instances and can be assumed as a threat to the learning

**Fig. 1.** Robustness of datasets against adversarial samples




model. We repeated each experiment ten times and reported the average of the results.

Figure 1 shows the results of Exp-2. While the x-axis shows the number of manipulated features, zero manipulated feature means that the test happened with the original phishing instances without any perturbation. The trend of results reveals that by increasing the number of perturbation, the number of evaded samples increase proportionally. We continued increasing the perturbed features for up to four different features at a time. We observed that with four features, almost all phishing instances bypass the classifier model.

For example, Figure 1 shows that less than 4% of phishing instances in DS-1 can bypass the classifier without any perturbation. With only one manipulated feature, more than 20% of phishing instances can bypass the classifier. With two manipulated features, almost all of instances can bypass the GB. The results are almost the same for other datasets. In another case, while just 12% of original phishing instances (the instances without any changes) have been misclassified in DS-3, the results significantly go up to 65% with only one perturbed feature.

This experiment shows how vulnerable the machine learning models are for the phishing problem. Small perturbation on features can bypass the classifier and degrade the accuracy significantly.

### 5.5 Exp-3: Dataset Vulnerability Level

In this experiment, we studied the cost that an adversary has to pay to bypass a classifier. From an adversary point of view, it is not inexpensive to manipulate an instance with new feature values and bypass the classifier. In Section 4.3, we assessed the cost and in Section 4.4, we defined the term *vulnerability level* for one instance. Similar to previous experiment 5.4, we reserved 200 phishing instances from each dataset and chose the classifier for each dataset based on Table 3. For datasets DS-1 and DS-3, we chose GB while we chose RF for both DS-2 and DS-4 datasets. Averaging the *vulnerability level* for each of the 200 selected instances and repeating the experiment ten times, we assessed the vulnerability level for the whole dataset.

Figure 2 presents the results of this experiment for all datasets for two parameters: the number of manipulated features and the average cost of adversarial

instances. It is evident that, by increasing the number of manipulated features, the cost also increases steadily. For example, for the dataset DS-1, the average cost, for adversarial samples, with one manipulated feature is 0.95 and with four manipulated features the cost is 3.93.

Furthermore, it is clear that the average cost for some datasets is more than that of other datasets. For example, in the DS-4, the adversary has to pay more cost especially when the number of features increases to three and four in comparison to the other datasets. This shows that this dataset is more robust against these attacks and has a lower vulnerability level. Also, it is clear that with one single feature manipulation with a small cost, it is possible to bypass a classifier. This needs to be considered when a dataset and features are designed.

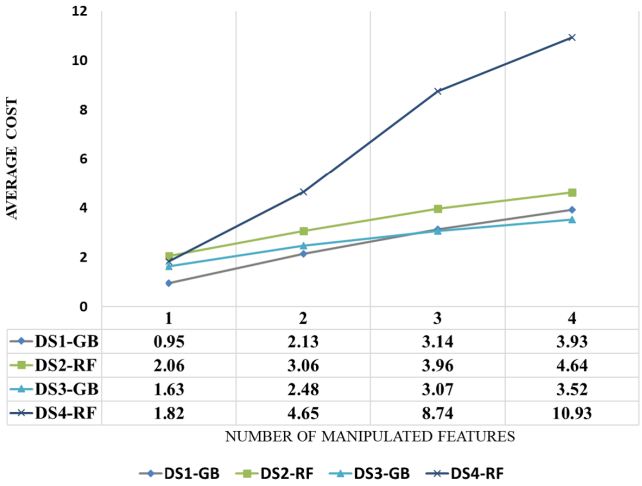
**Table 4.** Comparisons of different approaches in the literature including our proposed approach

| Author | Description | Size | ACC |
|---|---|---|---|
| **Niakanlahiji** *et al.* [4] | -Scalable feature-rich framework with a series of new and existing features<br>-Not using third-party services, Language agnostic | 22.3K | 95% |
| **Sahinguz** *et al.* [11] | -Real-time detection mechanism based on NLP of URLs, Language independent<br>-Tested on a large dataset, Not using third-party service | 73K | 97% |
| **Verma** *et al.* [12] | -Features based on lexical-, distance-, and length-related features of the URL<br>-Using four large datasets | 115K | 99.3% |
| **Jiang** *et al.* [5] | -Combined the URL and DNS information, Used a deep neural network with the help of NLP, Automatically extracts hidden features | 7M | 96% |
| **Tian** *et al.* [13] | -Studied five types of domain squatting, Using dataset of over 224 million registered domains, Using visual and OCR analysis, Found new phishing instances that evaded common blacklist | 234M | N/A |
| **Pereira** *et al.* [6] | -Detecting algorithmically generated domain, Graph-based algorithm to extract the dictionaries that are being used to generate algoritmically domains | 80K | 99% |
| **Shirazi** *et al.* [7] | -Studying limitation current approaches: large number of features and bias in the datasets , Focused on the domain name, Running at the client-side<br>-Not using third-party services | 2.2 K | 97-98% |
| **Li *et al.* [8]** | -Extract the features from both URL and HTML of the page<br>-Not using third-party services | 50K | 97% |
| **Bulakh** *et al.* [25] | -Companies can define their phishing detection mechanism and protect the customers<br>-Can be used as an complimentary service besides other detection approaches | 1.3K | 96.34% |
| **Our work** | -Evaluate the performance of existing datasets including [7, 21, 23, 24]<br>-Using multiple classifiers and comparing the results | 2-10K | 81-95% |
| **Our work** | - Proposing adversarial sampling attack against the learning model, Showing the feasibility of the attack, Prove the vulnerability of current model, Modeling the vulnerability level and cost | 2-10K | 0% |

### 5.6 Comparing the results with previous experiments

In this section, we compare our approach with some of the previous researches in this field. Table 4 compared nine different approaches in the literature. We summarized the pros and cons of each approach and show the dataset size and best accuracy results of each approach. We studied a wide range of previous efforts by focusing on machine learning techniques. Some of the techniques solely focused on the URL itself [11, 13] but others look at both URL and the content of the page [7, 25]. The use of third-party services is another difference between approaches. While using third-party services like search or DNS inquires leverage the feature set and make the feature set more reliable it also endangers

**Fig. 2.** The manipulation cost for adversarial samples based on number of manipulated features




the privacy of the users. Third-party enquiries to fetch the feature value reveal the browsing history of the end-users The previous studies have been done on variable sizes of datasets. While some of the datasets have less than 5 thousand records [7, 25], there are also datasets with millions of instances [5, 13]. Also, for approaches analyzing just the URL without the webpage content, creating massive datasets are easier. Most of the approaches achieved high accuracy of over the 95%. Both [6, 12] achieved accuracy of 99%, which is significantly high. Tian *et al.* [13] found new phishing samples that were not detected by common phishing detection mechanisms even after one month. We also added the results of this study to Table 4. We trained the classifier on the four public datasets and achieved very high accuracy. When we added the manipulated features in the testing phase, the accuracy degraded significantly and finally became zero. These experiments prove that our proposed attack is sufficient to evade existing classifiers for phishing detection.

## 6 Conclusion and Future work

In this work, we explained the limitation of machine learning techniques when adversarial samples are taken into consideration. We introduced the notion of vulnerability level for data instances and datasets based on the adversarial attacks and quantified it. We achieved high accuracy in the absence of this attack using seven different well-studied classifiers in the literature: more than 95% for all classifiers except one that had 82%. However, when we evaluated the best-performing classifier against the adversarial samples, the performance of the classifier degraded significantly. With only one feature perturbation, the TPR falls from 82-97% to 79%-45% and, increasing the number of perturbed features to four, the TPR fell to 0%, meaning that all of the phishing instances were able to bypass the classifier. We continued our experiments by considering the adver-

sary cost in the experiment. We showed that both the number of manipulated features and the total manipulation cost, which can be derived from the difference between original phishing sample and the adversarial sample, are essential. This means that from an attacker point of view, not only changing the minimum number of instances is desired, but also it is important that the adversarial sample has the minimum cost. This is an impressive result and shows the weakness of well-known defense mechanisms against phishing attacks. In future, we want to design robust machine learning models that are immune to adversarial learning attacks.

## Acknowledgements

This work is supported in part by funds from NSF Awards CNS 1650573, CNS 1822118 and funding from CableLabs, Furuno Electric Company, SecureNok, and AFRL. Research findings and opinions expressed are solely those of the authors and in no way reflect the opinions of the NSF or any other federal agencies.

## References

1. Yanping Zhang, Yang Xiao, Kaveh Ghaboosi, Jingyuan Zhang, and Hongmei Deng. A survey of cyber crimes. *Security and Communication Networks*, 5:422–437, 2012.
2. Jerry Felix and Chris Hauck. System security: a hacker's perspective. *Interex Proceedings*, 1:6–6, 1987.
3. APWG. Phishing attack trends report - 3q 2018, 2018. [Online; accessed 24-Jan-2019].
4. Amirreza Niakanlahiji, Bei-Tseng Chu, and Ehab Al-Shaer. Phishmon: A machine learning framework for detecting phishing webpages. In *Intelligence and Security Informatics*, pages 220–225, 2018.
5. Jianguo Jiang, Jiuming Chen, Kim-Kwang Raymond Choo, Chao Liu, Kunying Liu, Min Yu, and Yongjian Wang. A deep learning based online malicious url and dns detection scheme. In *Security and Privacy in Communication Systems*, pages 438–448, 2017.
6. Mayana Pereira, Shaun Coleman, Bin Yu, Martine DeCock, and Anderson Nascimento. Dictionary extraction and detection of algorithmically generated domain names in passive dns traffic. In *Research in Attacks, Intrusions, and Defenses*, pages 295–314, 2018.
7. Hossein Shirazi, Bruhadeshwar Bezawada, and Indrakshi Ray. "kn0w thy doma1n name": Unbiased phishing detection using domain name based features. In *Access Control Models and Technologies*, pages 69–75, 2018.
8. Yukun Li, Zhenguo Yang, Xu Chen, Huaping Yuan, and Wenyin Liu. A stacking model using url and html features for phishing webpage detection. *Future Generation Computer Systems*, 94:27–39, 2019.
9. Nilesh Dalvi, Pedro Domingos, Sumit Sanghai, Deepak Verma, et al. Adversarial classification. In *international conference on Knowledge discovery and data mining*, pages 99–108, 2004.
10. ISTR Internet Security Threat Report Volume 23. Technical report.
11. Ozgur Koray Sahingoz, Ebubekir Buber, Onder Demir, and Banu Diri. Machine learning based phishing detection from urls. *Expert Systems with Applications*, 117:345–357, 2019.

12. Rakesh Verma and Keith Dyer. On the character of phishing urls: Accurate and robust statistical learning classifiers. In *Data and Application Security and Privacy*, pages 111–122, 2015.
13. Ke Tian, Steve TK Jan, Hang Hu, Danfeng Yao, and Gang Wang. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Internet Measurement Conference*, pages 429–442, 2018.
14. Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and JD Tygar. Adversarial machine learning. In *ACM workshop on Security and artificial intelligence*, pages 43–58, 2011.
15. Huang Xiao, Battista Biggio, Gavin Brown, Giorgio Fumera, Claudia Eckert, and Fabio Roli. Is feature selection secure against training data poisoning? In *International Conference on Machine Learning*, pages 1689–1698, 2015.
16. Battista Biggio, Giorgio Fumera, and Fabio Roli. Security evaluation of pattern classifiers under attack. *IEEE transactions on knowledge and data engineering*, 26:984–996, 2014.
17. Ambra Demontis, Marco Melis, Battista Biggio, Davide Maiorca, Daniel Arp, Konrad Rieck, Igino Corona, Giorgio Giacinto, and Fabio Roli. Yes, machine learning can be more secure! a case study on android malware detection. *Dependable and Secure Computing*, 2017.
18. Yizhen Wang, Somesh Jha, and Kamalika Chaudhuri. Analyzing the robustness of nearest neighbors to adversarial examples. In *International Conference on Machine Learning*, pages 5120–5129, 2018.
19. Nicolas Papernot, Ian Goodfellow, Ryan Sheatsley, Reuben Feinman, and Patrick McDaniel. cleverhans v1. 0.0: an adversarial machine learning library. *arXiv preprint arXiv:1610.00768*, 10, 2016.
20. Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *arXiv preprint arXiv:1712.03141*, 2017.
21. Rami M Mohammad, Fadi Thabtah, and Lee McCluskey. An assessment of features related to phishing websites using an automated technique. In *Internet Technology And Secured Transactions*, pages 492–497, 2012.
22. Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.
23. Neda Abdelhamid, Aladdin Ayesh, and Fadi Thabtah. Phishing detection based associative classification data mining. *Expert Systems with Applications*, 41(13):5948–5959, 2014.
24. Choon Lin Tan. Phishing dataset for machine learning: Feature evaluation, 2018.
25. Vlad Bulakh and Minaxi Gupta. Countering phishing from brands' vantage point. In *International Workshop on Security And Privacy Analytics*, pages 17–24, 2016.