Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Data driven exploratory attacks on black box classifiers in adversarial domains

Tegjyot Singh Sethi[*], Mehmed Kantardzic

Data Mining Lab, University of Louisville, Louisville, USA

ARTICLE INFO

Article history:
Received 23 March 2017
Revised 13 November 2017
Accepted 2 February 2018
Available online 8 February 2018

Communicated by Dr Lo David

Keywords:
Adversarial machine learning
Reverse engineering
Black box attacks
Classification
Data diversity
Cybersecurity

ABSTRACT

While modern day web applications aim to create impact at the civilization level, they have become vulnerable to adversarial activity, where the next cyber-attack can take any shape and can originate from anywhere. The increasing scale and sophistication of attacks, has prompted the need for a data driven solution, with machine learning forming the core of many cybersecurity systems. Machine learning was not designed with security in mind and the essential assumption of stationarity, requiring that the training and testing data follow similar distributions, is violated in an adversarial domain. In this paper, an adversary's view point of a classification based system, is presented. Based on a formal adversarial model, the *Seed-Explore-Exploit* framework is presented, for simulating the generation of data driven and reverse engineering attacks on classifiers. Experimental evaluation, on 10 real world datasets and using the Google Cloud Prediction Platform, demonstrates the innate vulnerability of classifiers and the ease with which evasion can be carried out, without any explicit information about the classifier type, the training data or the application domain. The proposed framework, algorithms and empirical evaluation, serve as a white hat analysis of the vulnerabilities, and aim to foster the development of secure machine learning frameworks.

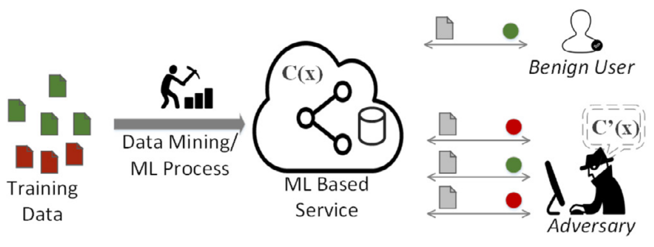
1. Introduction

The growing scale and reach of modern day web applications has increased its reliance on machine learning techniques, for providing security. Conventional security mechanisms of firewalls and rule-based black and white lists, cannot effectively thwart evolving attacks at a large scale [37]. As such, the use of data driven machine learning techniques in cybersecurity applications, has found widespread acceptance and success [15]. Whether it be for outlier detection for network intrusion analysis [51], biometric authentication using supervised classification [14], or for unsupervised clustering of fraudulent clicks [45], the use of machine learning in cybersecurity domains is ubiquitous. However, during this era of increased reliance on machine learning models, the vulnerabilities of the learning process itself have mostly been overlooked. Machine learning operates under the assumption of stationarity, i.e. the training and the testing distributions are assumed to be identically and independently distributed (IID) [53]. This assumption is often violated in an adversarial setting, as adversaries gain nothing by generating samples which are blocked by a defender's system [16]. The dynamic and contentious nature of this domain, demands a thorough analysis of the dependability and security of machine learning systems, when used in cybersecurity applications.

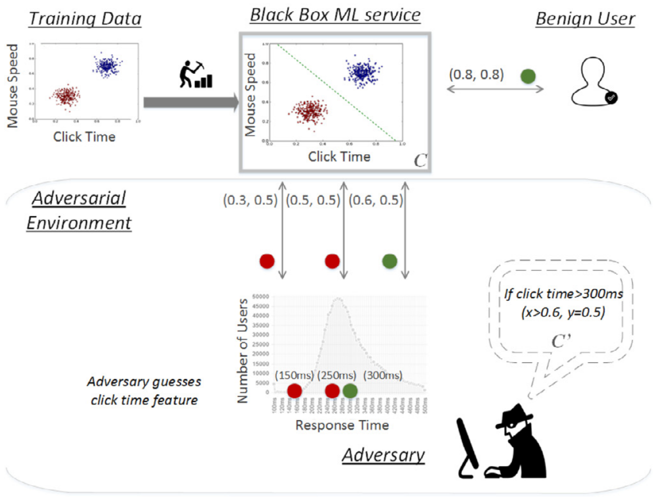
In an adversarial environment, the accuracy of classification has little significance, if an attacker can easily evade detection by intelligently perturbing the input samples [21]. Any deployed classifier is susceptible to probing based attacks, where an adversary uses the same channel as the client users, to gain information about the system, and then subsequently uses that information to evade detection [1,6]. This is seen in Fig. 1 (a), where the defender starts by learning from the training data and then deploys the classifier C, to provide services to client users. Once deployed, the model C is vulnerable to adversaries, who try to learn the behavior of the defender's classifier by submitting probes as input samples, masquerading as client users. In doing so, the defender's classifier is seen only as a black box, capable of providing tacit *Accept/Reject* feedback on the submitted samples. An adversary, backed by the knowledge and understanding of machine learning, can use this feedback to reverse engineer the model C (as C'). It can then avoid detection on future attack samples, by accordingly perturbing the input samples. It was shown recently that, deep neural networks are vulnerable to adversarial perturbations [31]. A similar phenomenon was shown to affect a wide variety of classifiers in [30], where it was demonstrated that adversarial samples are

[*] Corresponding author.
E-mail addresses: t0seth01@louisville.edu, tegjyotsingh.sethi@louisville.edu (T.S. Sethi), mehmedkantardzic@louisville.edu (M. Kantardzic).

https://doi.org/10.1016/j.neucom.2018.02.007
0925-2312/© 2018 Elsevier B.V. All rights reserved.



(a) An adversary making probes to the black box model C, can learn it as C', using active learning.



(b) Example task of attacking behavioral CAPTCHA. Black box model C, based on Mouse Speed and Click Time features, is used to detect benign users from bots. Adversary can reverse engineer C as C', by guessing click time feature and making probes based on the human response time chart, using the same input channels as regular users.

Fig. 1. Classifiers in adversarial environment, (a) shows the general adversarial nature of the problem and (b) shows an example considering a behavioral CAPTCHA system.

transferable across different classifier families. Cloud based machine learning services (such as Amazon AWS Machine Learning[1] and Google Cloud Platform[2]), which provide APIs for accessing predictive analytics as a service, are also vulnerable to similar black box attacks [42].

An example of the aforementioned adversarial environment is illustrated in Fig. 1(b), where a behavioral mouse dynamics based CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart) system is considered. Popular examples of these systems are Google's reCAPTCHA[3] and the classifier based system developed in [14]. These systems use mouse movement data to distinguish humans from bots and provide a convenient way to do so, relying on a simple point and click feedback, instead of requiring the user to infer garbled text snippets [14]. The illustrative 2D model of Fig. 1 (b), shows a linear classifier trained on the two features of - Mouse movement speed and Click time. An adversary, aiming to evade detection by this classifier, starts by guessing the click time as a key feature (intuitive in this setting), and then proceeds to makes probes to the black box model C, to learn its behavior. Probes are made by going through the spectrum of average reaction times for humans[4], guided by the *Accept*(green)/*Reject*(red) feedback from the CAPTCHA server. The information learned by reconnaissance on the black box system, can then be used to modify the attack payload so as to subsequently evade detection. While, this example was simplistic, its purpose is to illustrate the adversarial environment in which classifiers operate. Practical deployed classifiers tend to be more complex, non linear and multidimensional. However, the same reasoning and approach can be used to evade complex systems. An example of this is the good words and bad words attacks on spam detection systems [10,27]. By launching two spam emails each differing in only one word *'Sale'*, it can be ascertained that this word is important to the classification, if the email containing that word is flagged as spam. Knowing this information, the adversary can modify the word to be *'Sa1e'*, which looks visually the same but avoids detection. These evasion attacks are non-intrusive in nature and difficult to eliminate by traditional encryption/security techniques, because they use the same access channels as regular input samples and they see the same black box view of the system. From the classification perspective, these attacks occur at test time and are aimed at increasing the false negative rate of the model, i.e. increase the number of *Malicious* (positive) samples classified as *Legitimate*(negative) by C [4,5].

In addition to increasing the false negative rate of the defender's model, an intelligent data driven adversary would be interested in creating attacks of high diversity or variability. Such attacks are difficult to stop using simple blacklisting techniques and will require a more laborious task of retraining the classifier, using newly collected and curated data. We are interested in the analysis of such adversaries, as they are sophisticated, data driven, and intend to leave the machine learning model unusable over the long run.

Data driven attacks on deployed classification systems, presents a symmetric flip side to the task of learning from data. Instead of learning from labeled data to generate a model, the task of an attacker is to learn about the model, to generate evasive data samples [1]. With this motivation, we propose the Seed-Explore-Exploit(SEE) framework in this paper, to analyze the attack generation process as a learning problem, from a purely data driven perspective and without incorporating any domain specific knowledge. Research work on detecting concept drift in data streams[38,39], motivated the need for a formal analysis of the vulnerabilities of machine learning, with an initial evaluation proposed in our work in [40]. In this paper, we extend the earlier version with: i) incorporating and evaluating effects of *Diversity* of attacks on the defender's strategy, ii) introducing adversarial metrics of attack quality and the effects of varying the parameters of attack algorithms, iii) extensive detailed experimentation of the framework using a variety of defender models and the Google Cloud Prediction Service, and iv) experimentation simulating effects of diversity on blacklisting based countermeasures. To the best of our knowledge, this is the first work which presents a comprehensive adversarial model for indiscriminate exploratory black box attacks on classifier models. This work goes beyond [40], by providing a more in depth and thorough representation of adversarial activity, which can be reused for vulnerability analysis of future developed countermeasures against adversarial machine learning. The main contributions of this paper are:

- A domain independent data driven framework is presented, to simulate attacks using an Exploration-Exploitation strategy. This generic framework and the algorithms presented, can be used to analyze simple probing attacks to more sophisticated reverse engineering attacks.

[1] https://aws.amazon.com/machine-learning/.
[2] cloud.google.com/machine-learning.
[3] www.google.com/recaptcha.
[4] www.humanbenchmark.com/tests/reactiontime.

Table 1
Categorization of attacks against machine learning systems.

Influence	*Causative*- Attacks influence training data, to mislead learning
	Exploratory- Attacks affect test time data, to evade detection
Specificity	*Targeted*- Attack affect only particular instances
	Indiscriminate- Attacks irrespective of instances
Security violation	*Integrity*- Results in increased false negatives
	Availability- Denial of service attacks, due to increased errors

- Formal adversarial model and adversary's metrics are proposed, as a background for sound scientific development and testing for secure learning frameworks.
- Empirical analysis on 10 real world datasets, demonstrates that feature space information is sufficient to launch attacks against classifiers, irrespective of the type of classifier and the application domain. Additional experimentation on Google's Cloud Prediction API, demonstrates vulnerability of remote black box prediction services.
- The analysis of diversity and its effect on blacklisting based countermeasures, demonstrates that such security measures (as proposed in [21]) are ineffective when faced with reverse engineering based attacks of high diversity.

The rest of the paper is organized as follows: Section 2, presents background and related work on the security of machine learning. Section 3, presents the formal model of an adversary and the proposed Seed-Explore-Exploit(SEE) framework, for attack generation. Based on the framework, the Anchor Points attack and the Reverse Engineering attack algorithms are presented in Section 3.2 and 3.3, respectively. Experimental evaluation and detailed analysis is presented in Section 4. Additional discussion about diversity of attacks and its importance is presented in Section 5. Conclusion and avenues for further research are presented in Section 6.

2. Related work on the security of machine learning

One of the earliest works on machine learning security was presented in [4], where a taxonomy of attacks was defined, based on the principles of information security. The taxonomy categorized attacks on machine learning systems along the three axis of: Specificity, Influence and the type of Security Violation, as shown in Table 1. Based on the influence and the portion of the data mining process that these attacks affect, they are classified as being either Causative or Exploratory [7]. Causative attacks affect the training data while Exploratory attacks affect the model at test time. Specificity of the attacks, refers to whether they affect a set of targeted samples, in order to avoid detection by perturbing them, or if the attacks are aimed towards indiscriminately affecting the system, with no specific pre-selected instances. Based on the type of security violation, the attacks can aim to violate integrity, by gaining unsanctioned access to the system, or can be used to launch a denial of service availability attack.

Causative attacks aim to mislead training, by poisoning the training set, so that future attacks are easily evaded [6,24,47]. Causative attacks, although severe in effect, can be prevented by careful curation of the training data [23] and by keeping the training data secure- using database security measures, authentication and encryption. Exploratory attacks are more commonplace, less moderated and can be launched remotely without raising suspicion. These attacks affect the test time data, and are aimed at reducing the system's predictive performance [7]. This is done by crafting the attack samples, to evade detection by the defender's model [5,26]. Once a model is trained and deployed in a cybersecurity application, it is vulnerable to exploratory attacks. These attacks are non intrusive and are aimed at gaining information about the system, which is then exploited to craft evasive samples, to circumvent the system's security. Since, these attacks use the same channel as the client users, they are harder to detect and prevent.

Targeted-Exploratory attacks aim to modify a specific set of malicious input samples, minimally, to disguise them as legitimate. Indiscriminate attacks are more general in their goals, as they aim to produce any sample which will result in the defender's model to have a misclassification. Most work on exploratory attacks are concentrated on the targeted case, considering it as a constrained form of indiscriminate attacks, with the goal of starting with a malicious sample and making minimal modifications to it, to avoid detection [5,6,27,32,50]. This idea was formalized in [26], where the Minimal Adversarial Cost (MAC) metric, of a genre of classifiers, was introduced to denote the ease with which classifiers of a particular type can be evaded. The hardness of evasion was given in terms of the number of probes needed to obtain a low cost evasive sample. A classifier was considered easy to evade if making a few optimal modifications to a set of samples resulted in a high accuracy of evasion. Work in [29] shows that linear and convex inducing classifier are all vulnerable to probing based attacks, and [28] presents efficient probing strategies to carry out these attacks.

Particular strategies developed for performing exploratory attacks vary based on the amount of information available to the adversary, with a broad classification presented in [3] as: (a) Evasion attacks and (b) Reverse Engineering attacks. Evasion attacks are used when limited information about the system is available, such as a few legitimate samples only. These legitimate samples are exploited by masking techniques such as- mimicking [41] and spoofing [2], which masquerade malicious content within the legitimate samples. The mimicry attack was presented in [41], where the Mimicus tool[5] was developed, to implement evasion attacks on pdf documents, by hiding malicious code within benign documents. The good words attacks on spam emails uses a similar technique [10,27]. A spam email is inserted with benign looking words, to evade detection. Similarly, spoofing attacks are common in biometrics[2] and for phishing websites[20], where visual similarity can be achieved with totally different content. A general purpose, domain independent technique for evasion was presented in [50]. Here, using genetic programming, variants of a set of malicious samples were generated as per a monotonically increasing fitness function, which denoted success of evasion. This is an attractive technique due to its generic approach, but limited probing budgets and lack of a graded fitness function, are some of its practical limitations. In the presence of a large probing budget, or specific information about the defender's classifier model, the gradient descent evasion attack of [5], can be used. This attacks relies on knowing the exact classifier function used by the defender, or the ability to reverse engineer it using a sufficient number of probes. Once information about the classifier is known, the attack uses a gradient descent strategy to find an optimal low cost evasion sample for the classifier. Search strategies were developed for a wide range of classifiers with differentiable decision functions, including neural networks, non-linear Support vector machines, one class classifiers and for classifiers operating in discrete feature spaces [5]. An

[5] www.github.com/srndic/mimicus/blob/master/mimicus/attacks/mimicry.py.

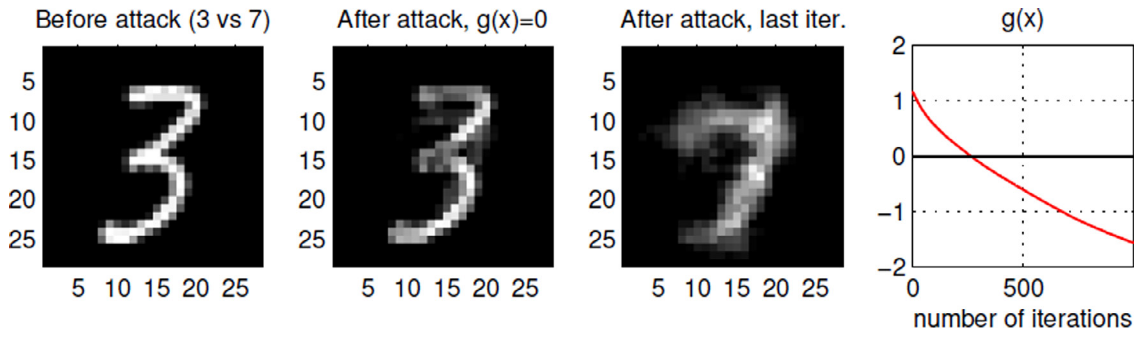


Fig. 2. Gradient descent evasion attack over 500 iterations. Left- Initial image of digit 3, Center- Image which first gets classified as 7, Right- Image after 500 iteration.[5].

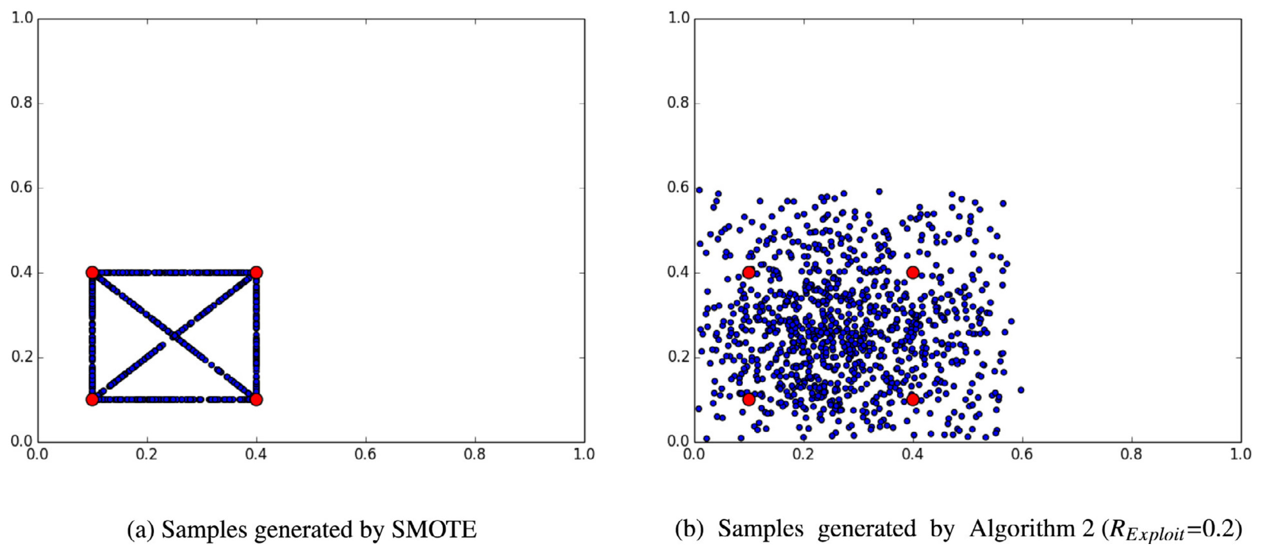


Fig. 3. Synthetic sample generation by SMOTE and the proposed AP-Exploitation algorithm. Samples in red represent the exploration samples and samples in blue represent the generated synthetic samples, based on the seed samples. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

illustration of the gradient descent attacks for masquerading a sample is shown in Fig. 2, where the image 3 is modified to be classified as 7 over 500 iterations of gradient descent.

Reverse engineering the defender's model provides avenues for sophisticated exploratory attacks, as it exposes features important to the classifier, to beFig. 3 used for mimicry attacks or large scale indiscriminate attacks. Perfect reverse engineering is not needed, as an adversary is interested only in identifying the portion of the data space which is classified as *Legitimate*. Reverse engineering was first employed in [26], where a sign witness test was used to see if a particular feature had a positive or negative impact on the decision. Reverse engineering of a decision tree classifier, as a symmetric model for defender and adversary, was presented in [49]. [50] used genetic programming as a general purpose reverse engineering tool, under the assumption of known training data distribution and feature construction algorithm. The genetic programming output, because of its intuitive tree structure, was then used to guide evasion attacks on intrusion detection systems. The idea of reverse engineering was linked to that of active learning via query generation in [3], where the robustness of SVM to reverse engineering is tested using active learning techniques of random sampling, uncertainty sampling and selective sampling.

In the above mentioned works of targeted-exploratory attacks, it is assumed that if an evasion is expensive (far from the original malicious sample), the adversary will give up. The above techniques are not designed for a determined adversary, who is willing to launch indiscriminate attacks. An adversary who wants to launch an indiscriminate attack will not bother with the near optimal evasion problem [29]. These type of attacks have been largely ignored, with the only mention we found was in [52], where it is termed - the free range attack, as an adversary is free to move about in the data space. In such attacks, the adversary will first analyze the vulnerabilities of the model, looking for prediction blind spots, before attempting an attack. Analyzing performance of models under such attack scenarios is essential to understanding its vulnerabilities in a more general real world situation, where all types of attacks are possible. Also, while most recent methodologies develop attacks as an experimental tool to test their safety mechanisms, there is very few works [5,32,44,49], which have attempted to study the attack generation process itself. Our proposed work analyzes *Indiscriminate-Exploratory-Integrity violating* attacks, under a data driven framework, with diverse adversarial goals and while considering only a black box model for the defender's classifier. We analyze the attacks from an adversary's point of view, considering the adversarial samples generation process, so as to understand the vulnerabilities of classifiers and to motivate the development of secure machine learning architectures.

3. Proposed methodology for simulating data driven attacks on classifiers

Data driven exploratory attacks on classifiers, affect the test time data seen by a deployed classifier. An adversary intending to evade classification, will begin by learning information about the system, over time, and then will launch an attack campaign to meet its goals. The adversary can only interact with the system as a black box model, receiving tacit *Accept/Reject* feedback for the submitted samples. However, an adversary can only make a limited number of probes, before it gets detected or runs out of resources. Additionally, we assume a minimal shared knowledge environment between the adversary and the defender [36]. Only the feature space information is shared between the two parties, as both operate on the same data space. All other information - such as the defender's classifier type, the model parameters and the training data, is kept hidden by the defender. Both the adversary and the defender are assumed to be capable machine learning experts, who are equipped with the tools and understanding of using a data driven approach to best suit their goals. Based on this intuitive understanding, the formal model of an adversary based on it is knowledge, goals and resources [6], is presented below:

- *Knowledge*- The adversary is aware of the number, type and range of features, used by the classification model. This could be approximated from publications, publicly available case studies in related applications, or by educated guessing [36]. For example, in case of spam classification, the feature domain could be the dictionary of English words, which is publicly available and well known. This represents a symmetric learning problem with both parties operating on the same feature space. No other information about the defender's model is known by the adversary.
- *Goals*- The adversary intends to cause false negatives for the defender's classifier, on the submitted attack samples. Additionally, the adversary also wants the attacks to be robust, such that it can avoid being detected and stopped by simple blacklisting techniques [8]. From a data driven perspective, the attacker aims to avoid detection by generating an attack set with high diversity and variability. While, repeating a single confirmed attack point, over and over, leads to ensured false negatives, such attacks are easily stopped by blacklisting that single point. We consider serious attackers only, who aim to force the retraining of the defender's classification system.
- *Resources*- The adversary has access to the system only as a client user. It can submit probes and receive binary feedback on it, upto a limited probing budget, without being detected. The adversary does not have control over the training data or the model trained by the defender.

The presented model of the adversary represents a general setting, where an attacker can take the form of an end user and then attack the system over time. Modern day web applications, which aim to reach as many users as possible, all operate under this environment and are susceptible to data driven attacks. Based on the adversary's model, the attack can be formalized here. A classifier C, trained on a set of training data D_{Train}, is responsible for classifying incoming samples into *Legitimate* or *Malicious* classes. An adversary aims to generate an attack campaign of samples D'_{Attack}, such that $C(D'_{Attack})$ has a high false negative rate. The adversary has at its disposal, a budget $B_{Explore}$ of probing data $D'_{Explore}$, which it can use to learn C and understand it as $C'(D'_{Explore})$. The number of attack samples (N_{Attack}) should be much larger than $B_{Explore}$, to justify expenditure on the adversary's part. In our analysis, the positive class is taken as the *Malicious* class samples, while the negative classification class is the *Legitimate* samples. This specified notation will be used through the rest of the paper.

The Seed-Explore-Exploit (SEE) framework is presented in Section 3.1, which provides an overview of the attack paradigm. Two specific attack strategies developed under the SEE framework, the Anchor Points attacks (AP) and the Reverse Engineering attacks (RE), are presented in Section 3.2 and Section 3.3, respectively.

3.1. The Seed-Explore-Exploit (SEE) framework

The SEE framework employs a data driven approach for generating adversarial samples. The idea of Exploration-Exploitation is common in search based optimization techniques, where the goal is to learn the data space and then emphasize only on the promising directions [12]. An adversary can also utilize a similar strategy, to best utilize the exploration budget ($B_{Explore}$), such that the resulting attack samples (D'_{Attack}) have high accuracy and high diversity. The specific steps of the framework are explained below:

- *Seed*- An attack starts with a seed phase, where it acquires a legitimate sample (and a malicious sample), to form the seed set D'_{Seed}. This seed sample can be acquired by random sampling in the feature space, by guessing a few feature values, or from an external data source of a comparable application [30]. For the case of a spam classification task, picking an email from one's own personal inbox would be a functional legitimate seed sample.
- *Explore*- Exploration is a reconnaissance task, starting with D'_{Seed}, where the goal is to obtain maximum diverse information, to understand the coverage and extent of the space of legitimately classified samples. The adversary is trying to understand the space of the features, where it will go undetected by the defender. As such, it tries to explore regions of space, where the data is classified as class *Legitimate* by the defender's model. In this phase, the adversary submits probes and receives feedback from the defender's black box. The defender can be probed upto a budget $B_{Explore}$, without being thwarted or detected. To avoid detection, it is natural that the adversary needs to spread out the attacks over time and data space, in which case the $B_{Explore}$ is the time/resources available to the adversary. The exploration phase results in a set of labeled samples $D'_{Explore}$, and the goal of the adversary is to best choose this set based on it is strategy.
- *Exploit*- The information gathered in the exploration phase is used here to generate a set of attack samples D'_{Attack}. The efficacy of the attack is based on the accuracy and the diversity of these samples.

The SEE framework provides a generic way of defining attacks on classifiers. Specific instantiations of the three phases can be developed, to suit one needs and simulation goals.

3.2. The Anchor Points Attack (AP)

The Anchor Points attack is suited for adversaries with a limited probing budget $B_{Explore}$, who have a goal of generating evasive samples for immediate benefits. An example of this would be - zero day exploits, where an adversary wants to exploit a new found vulnerability, before it is fixed [8]. These attacks start by obtaining a set of samples classified as Legitimate by C, called the Anchor Points, which serve as ground truth for generating further attack samples. From a data driven perspective, this attack strategy is defined under the SEE framework as given below.

- *Seed*- The attack begins with a single legitimate sample as the Seed (D'_{Seed}).
- *Explore*- After the initial seed has been obtained (provided or randomly sampled), the exploration phase proceeds to generate the set of Anchor Points, which will enable the understanding of the space of samples classified as *Legitimate*. The exploration phase is described in Algorithm 1, and is a radius based incremental neighborhood search technique, around the seed samples, guided by the feedback from the black box model C. Diversity of search is maintained by

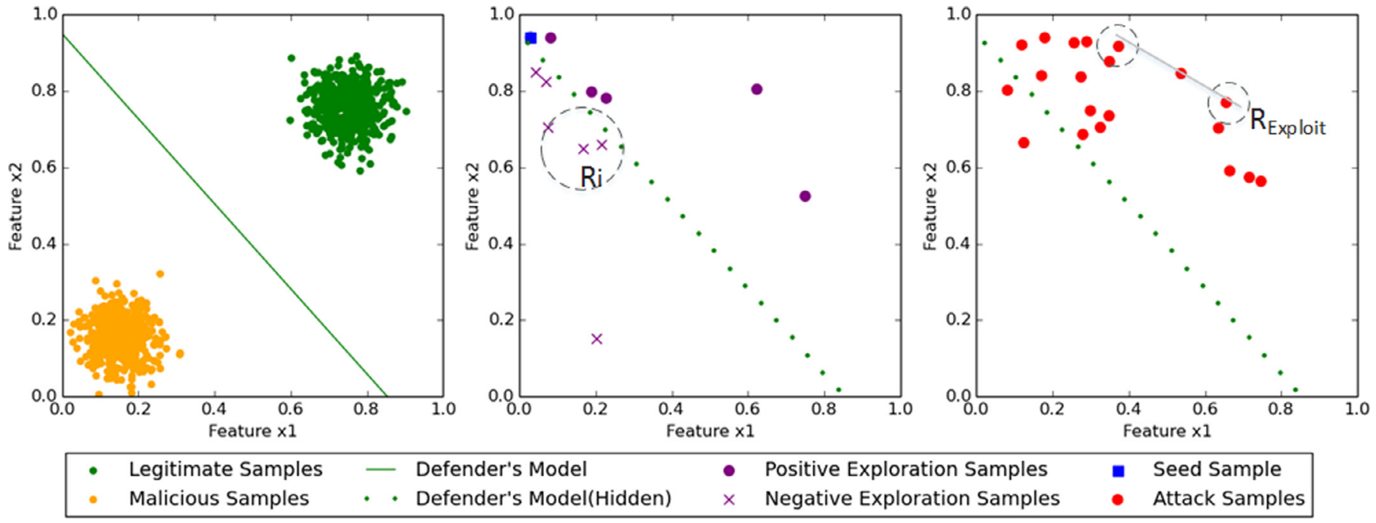


Fig. 4. Illustration of AP attacks on 2D synthetic data.*(Left - Right)*: The defender's model from it is training data. The Exploration phase depicting the seed(blue) and the anchor points samples(purple). The Exploitation attack phase samples (red) generated based on the anchor points. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Algorithm 1: AP- Exploration phase

```
Input  : Seed Data $D'_{Seed}$, Defender black box $C$.
         Parameters: Exploration budget $B_{Explore}$,
         Exploration neighborhood- $[R_{min}, R_{max}]$
Output: Exploration data set $D'_{Explore}$
1  $D'_{Explore} \leftarrow D'_{Seed}$
2  count_ legitimate=0
3  for $i = 1 .. B_{Explore}$ do
4      $x_i \leftarrow$ Select random sample from $D'_{Explore}$
5      $R_i = (R_{max} - R_{min}) * (count\_legitimate/i) + R_{min}$
6      $\hat{x_i} \leftarrow Perturb(x_i\ ,\ R_i)$
7      if $C.predict(\hat{x_i})$ is Legitimate then
8          $D'_{Explore} \cup \hat{x_i}$
9          count_legitimate ++
10 Procedure Perturb(sample, $R_{Neigh}$)
11     return sample+=random(mean=0, std=$R_{Neigh}$)
```

Algorithm 2: AP- Exploitation phase

```
Input  : Exploration data set $D'_{Explore}$, Number of attacks
         $N_{Attack}$, Radius of Exploitation $R_{Exploit}$
Output: Attacks set $D'_{Attack}$
1  $D'_{Attack} \leftarrow []$
2  for $i = 1 .. N_{Attack}$ do
3      $x_A, x_B \leftarrow$ Select random samples from $D'_{Explore}$
4      $\hat{x_A}, \hat{x_B} \leftarrow Perturb(x_A, R_{Exploit}),$
5      $Perturb(x_B, R_{Exploit})$
6      $\lambda = random(0, 1)$
7      $attack\_sample_i \leftarrow \hat{x_A} * \lambda + (1 - \lambda) * \hat{x_B}$
8      $D'_{Attack} \cup attack\_sample_i$
9  Procedure Perturb(sample, $R_{Exploit}$)
10     return sample+=random(mean=0, std=$R_{Exploit}$)
```

dynamically adjusting the search radius (R_i), based on the amount of ground truth obtained so far (Line 5). This ensures that radius of exploration increases in cases where the number of legitimate samples obtained is high, and vice versa, thereby balancing diversity of samples with their accuracy. Samples are explored by perturbing an already explored legitimate sample (Seed sample in case of first iteration), within the exploration radius (Line 7). The final exploration dataset of Anchor Points - $D'_{Explore}$, is comprised of all explored samples x_i, for which $C(x_i)$ indicated the *Legitimate* class label. The exploration phase is illustrated on a synthetic 2D dataset in Fig. 4, where the neighborhood radius R_i indicates the exploration neighborhood of a sample. Algorithm 1 has a time complexity of $O(B_{Explore})$, since perturbing and probing C, can be assumed to take constant time, as per the application it is deployed on.

- *Exploit*- The anchor points obtained as $D'_{Explore}$, forms the basis for launching the dedicated attack campaign on the classifier C. The exploitation phase (Algorithm 2) combines two techniques to ensure high accuracy and diversity of attack samples: *a) Simple perturbation*- The anchor point samples are perturbed, similar to the exploration phase, using a radius of exploitation- $R_{Exploit}$ (Line 4) and, *b) Convex combination*- The perturbed samples are combined using convex combination of samples, two at a time (Line 7). This is inspired by the Synthetic Minority Oversampling Technique (SMOTE), which is a popular oversampling technique for imbalanced datasets [11]. SMOTE was developed to balance datasets, by oversampling minority class samples, for better learning. However, Algorithm 2 extends this idea to the task of generating attack samples. While SMOTE over exploits and limits the sample generation closely to the area of known samples (shown as exploration samples in red), Algorithm 2 balances exploitation with spread of the generated attack samples. The attack set D'_{Attack}, shown in red in Fig. 4, is the final attack on the classifier C. Algorithm 2 has a time complexity of $O(N_{Attack})$, since all operations take constant time per attack sample.

The performance of the AP attack is largely dependent on the probes collected in the initial seed and exploration phase. As such, maintaining diversity is key, as larger coverage ensures more flexibility in attack generation. By the nature of these attacks, they can be thwarted by blacklists capable of approximate matching [34]. Nevertheless, they are suited for ad hoc swift blitzkriegs, before the defender has time to respond (Fig. 3).

3.3. The Reverse Engineering Attack (RE)

In case of sophisticated attackers, with a large $B_{Explore}$, direct reverse engineering of the classification boundary is more advantageous. It provides a better understanding of the classification landscape, which can then be used to launch large scale evasion or availability attacks [21]. Reverse engineering could also be an end goal in itself, as it provides information about feature importance to the classification task [26]. A reverse engineering attack, if done effectively, can avoid detection and make retraining harder on the part of the defender. However, unlike the AP attacks, these attacks are affected by the type of model used by the black box C, the dimensionality of the data and the number of probes available. Nevertheless, the goal of an adversary is not to exactly fit the decision surface, but to infer it sufficiently, so as to be able to generate attacks of high accuracy and diversity. As such, a linear approximation to the defender's model and a partial reverse engineering attempt should be sufficient for the purposes of launching a reduced accuracy attack. This reduction in accuracy can be compensated for by launching a massive attack campaign, exploiting the information provided by the reverse engineered model C'.

Effective reverse engineering relies on the availability of informative samples. As such, it is necessary to use the probing budget $B_{Explore}$ effectively. Random sampling can lead to wasted probes, with no additional information added, making it ineffective for the purposes of this attack. The query synthesis technique of [46], generates samples close to the classification boundary and spreads the samples along the boundary, to provide a better learning opportunity. The approach of [46] was developed for the purpose of active labeling of unlabeled data. We modify the approach to be used for reverse engineering as part of the SEE framework, where the attacker learns a surrogate classifier C', based on probing the defender's black box C. The SEE implementation of the RE attack is given below:

- *Seed*- The seed set consists of one legitimate and one malicious class sample.
- *Explore*- The exploration phase (Algorithm 3) uses the Gram-Schmidt process [46] to generate orthonormal samples, near the midpoint of any two randomly selected seed points of opposite classes (Line 8). This has the effect of generating points close to the separating decision boundary of the two classes, and also of spreading the samples along this boundary's surface, as depicted in the exploration phase of Fig. 5. The magnitude of the orthonormal vector is set based on λ_i, which is selected as a random value in [0,λ_{max}], to impart diversity to the obtained set of samples (Line 10–11). At the end of the exploration phase, the resulting set of labeled samples ($D'_{Explore}$), is used to train a linear classifier of choice, to form the surrogate reverse engineered model C' (Line 19). Fig. 5 shows the reverse engineered model (red), as learned from the original black box classifier C (green). Algorithm 3 has a complexity of $O(B_{Explore})$, as each of the transformation steps and the probing of C, can be assumed to take constant time, based on the application.
- *Exploit*- The surrogate model C', can be used to generate attacks with high accuracy and diversity. Ideally, a set of random points can be generated and verified against the reverse engineered model C', before adding them to the attack set D'_{Attack}. However, a practical and efficient way would be to use the exploration set samples $D'_{Explore}$ of Algorithm 3, as a seed set to generate a set of anchor points as in Algorithm 1, with the exception that we probe C' instead of the original model C. Since C' is a locally trained model, probing it does not impact $B_{Explore}$. Thus allowing an adversary to make a large number of probes, at theoretically zero cost. The anchor points obtained can then be used to generate the attack samples using Algorithm 2. A larger attack radius $R_{Exploit}$ can be used with this attack strategy, as additional validation is available via the model C'.

Algorithm 3: RE Exploration - Using Gram–Schmidt process.

```
Input : Seed Data $D'_{Seed}$, Defender black box model $C$.
        Parameters: Exploration budget $B_{Explore}$, Magnitude of
        dispersion $\lambda_{max}$
Output: Exploration data Set $D'_{Explore}$, Surrogate classifier $C'$
1  $D'_{Explore\_L}$= Legitimate samples of $D'_{Seed}$
2  $D'_{Explore\_M}$= Malicious samples of $D'_{Seed}$
3  for $i = 1 .. B_{Explore}$ do
4      $x_L \leftarrow$ Select random samples from $D'_{Explore\_L}$
5      $x_L \leftarrow$ Select random samples from $D'_{Explore\_M}$
6      $x_0 = x_L - x_M$
7      Generate random vector $x_R$
8      $x_R = x_R - \frac{<x_R,x_0>}{<x_0,x_0>} * x_0$
9      $\lambda_i = random(0, \lambda_{max})$
10     $x_R = \frac{\lambda_i}{norm(x_R)} * x_R$
11     $x_S = x_R + (x_L + x_M)/2$
12     if $C.predict(x_S)$ is Legitimate then
13         $D'_{Explore\_L} \cup x_S$
14     else
15         $D'_{Explore\_M} \cup x_S$
16 $D'_{Explore} = D'_{Explore\_L} \cup D'_{Explore\_M}$
17 Train $C'$ using $D'_{Explore}$
```

The efficacy of the reverse engineering approach is tied to the type of model being used by C, the dimensionality of the data space, and the number of reverse engineering probes possible. Nevertheless, the goal of the attacker is not to exactly fit the decision boundary, but to generate highly accurate and diverse attacks. In accordance with this, a linear approximation of a nonlinear boundary and a partial reverse engineering attempt would be sufficient. A linear approximation is not a major limitation due to the following reasons: (i) We believe that most systems would rely on linear models as a first stab at a predictive problem, and would only move to non-linear models in cases where the linear model is found to be inadequate[6], and (ii) A linear approximation of a nonlinear boundary should still provide sufficient accuracy (enough to launch a massive attack to compensate for reduced accuracy) [30]. Based on these intuitive motivations, the linear approximation is suggested as an effective attack strategy which leverages reverse engineering.

The RE attack strategy is suited for a patient adversary, who spends time/effort to probe the system and learn it, so as to have an effective attack with high diversity. Such attacks, are often hard to detect and stop by simple blacklisting techniques. However, the success of this attack relies on the goodness of the reverse engineered model, and could be affected by the nature of learning employed by the black box.

4. Experimental evaluation

This section presents experimental evaluation of the AP and the RE approaches, on classifiers trained with 7 real world datasets.

[6] http://docs.aws.amazon.com/machine-learning/latest/dg/learning-algorithm.html.

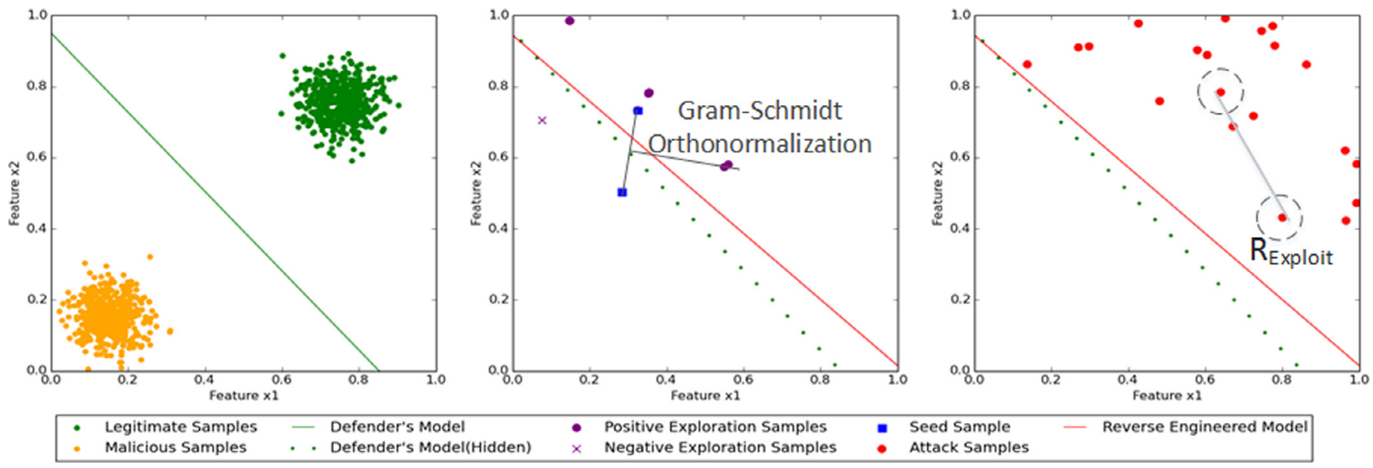


Fig. 5. Illustration of RE attacks on 2D synthetic data.*(Left - Right)*: The defender's model based on training data. The Exploration phase depicting reverse engineering(red) using the Gram–Schmidt orthonormalization process. The Exploitation attack phase samples generated after validation from the surrogate classifier (red samples). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Additionally, evaluation on 3 datasets from the cybersecurity domain is presented, to demonstrate the vulnerabilities of machine learning systems in adversarial milieus, to exploratory attacks. The experiments are presented from an adversary's point of view, who wishes to have effective attacks with high accuracy and diversity. Section 4.1 presents the metrics and the experimental protocol used, to encourage reproduce-ability of results. Experimental results and discussions are presented in Section 4.2.

4.1. Experimental methods and setup

4.1.1. Adversary metrics for attack quality

An adversary aiming to create maximum impact, needs to make the set of attack samples D'_{Attack} - *Accurate* and *Diverse*. Accuracy ensures that the attack samples will cause an increase in the false negative rate of the defender's model *C*. While, diversity ensures that the attack set has enough variability, so that they can go unnoticed for a long time. These intuitive ideas are quantified using 4 proposed quality metrics, to measure adversary effectiveness. The Effective Attack Rate (EAR) measures the accuracy of attacks, and the 3 metrics: Deviation of attacks (σ_{EA}), K-Nearest Neighbor Distance (KNN-dist) and the Minimum Spanning Tree distance (MST-dist), collectively represent the diversity of attacks. The metrics are based on the following definition of effective attacks (EA):

$$EA \quad = \quad \left\{x : \quad C(x) = Legitimate \ \wedge \ x \in D'_{Attacks}\right\} \tag{1}$$

Based on Eq. (1), an attack sample is effective if it is classified as *Legitimate* by the black box *C*. The adversarial quality metrics over the set EA are defined below:

a) *Effective Attack Rate (EAR)*: This is the accuracy of attacks, measured as the ratio of attack samples which successfully evade the defenders classifier, given by Eq. (2). A value of 1 denotes perfect evasion.

$$EAR \quad = \quad \frac{|EA|}{\left|D'_{Attacks}\right|} \tag{2}$$

b) *Deviation of effective attacks (σ_{EA})*: This is a measure of diversity, which computes the spread of data around its mean, given by Eq. (3).

$$\sigma_{EA} \quad = \quad \sqrt{\frac{1}{|EA-1|}\sum_{x_i \epsilon EA}(x_i - \mu_{EA})^2} \tag{3}$$

where, μ_{EA} indicates the Euclidean mean of samples in the effective attack set EA. A large value of σ_{EA} indicates that the data has high data space coverage.

c) *K-Nearest Neighbor distance of effective attacks ($KNN - dist_{EA}$)*: This measure of diversity, computes local density information of the data samples (motivated by [18]). It is computed by finding the average distance of the K-nearest neighbors of a sample, for all samples and then averaging this value, as given by Eq. (4).

$$KNN - dist_{EA} \quad = \quad \frac{\sum_{x \epsilon EA}\sum_{i=1}^{K} \quad dist(x, NN_i(x))}{K. \quad |EA|} \tag{4}$$

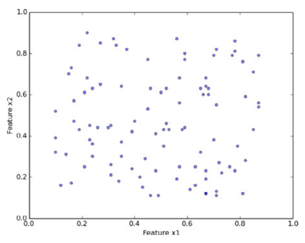
Where, the *dist(.)* function computes Euclidean distance between two vectors, and $NN_i(x)$ gives the i^{th} nearest neighbor of a sample *x*. A higher value of KNN-dist, indicates that data samples are relatively far from each other and that every sample is in a locally sparse region of space, indicating higher spread. A value of *K*=5 is chosen for experimentation.

d) *Minimum Spanning Tree distance of effective attacks ($MST - dist_{EA}$)*: This is also a measure of diversity, which is computed by finding the length of the minimum spanning tree over the set of EA samples, as per Eq. (5) [22]. This is a measure which promotes ectropy or collocation of points, in an attempt to obtain a more global uniform and diverse spread of samples. This is especially useful in recognizing multiple locally dense clusters which are far from each other.

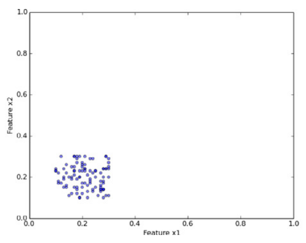
$$MST - dist_{EA} \quad = \quad \frac{length(MST(EA))}{|EA| - 1} \tag{5}$$

The MST measure computes cluster separation only once, as opposed to pairwise distance metrics which calculate distance between one point and every other point. Thus the MST provides a better sense of global diversity, by allowing sub groups of data to have less diversity. A high value of MST-distance will indicate high diversity.

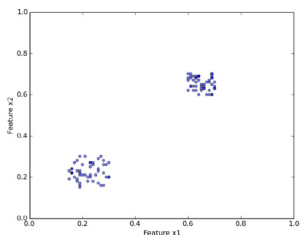
The three diversity metrics of σ, *KNN-dist* and *MST-dist*, are affected by different data distributions and together they provide a holistic representation of the variability of the attack data. Standard deviation captures the overall spread of the data and is severely affected by outliers. A larger spread of data results in higher deviation, as seen in Fig. 6 (a) where the deviation σ=0.228 is higher than in (b), where the deviation is σ=0.058. Although, deviation is effective in capturing the global spread of data, it fails at capturing the local data characteristics. As seen in Fig. 6 a) and



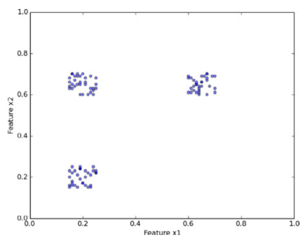
(a) σ=0.228; KNN-dist=0.071; MST-dist=0.056



(b) σ=0.058; KNN-dist=0.018; MST-dist=0.015



(c) σ=0.218; KNN-dist=0.017; MST-dist=0.037



(d) σ=0.226; KNN-dist=0.017; MST-dist=0.063

Fig. 6. Values of σ, *KNN-dist* and *MST-dist* for different 2D synthetic data distributions over 100 test points.

Table 2
Description of datasets used for experimentation of SEE framework.

Dataset	#Instances	#Dimensions
Digits08	1500	16
Credit	1000	61
Cancer	699	10
Qsar	1055	41
Sonar	208	60
Theorem	3060	51
Diabetes	768	8
Spambase	4600	57
KDD99	494021	41
CAPTCHA	1885	26

(c), which have very close deviation values ($\Delta\sigma$=0.002), but totally different distribution of data. These differences are caught by the $KNN-dist$ metric, which is higher for scattered data (Fig. 6 a), *KNN-dist*=0.071), as compared to closely packed data (Fig. 6 c), *KNN-dist*=0.017). However, the *KNN-dist* metric does not account for disjoint clusters spread out in space, as its a local measure and is myopic in scope. This distinction is caught effectively by the *MST-dist* metric, which shows a significant difference in the diversity values for Fig. 6 (c) and (d) ($\Delta MST-dist$=0.026), even though the *KNN-dist* metric shows no difference between the two. The MST metric is suitable for attacks such as the Anchor Points attacks, where the attacks are concentrated around a few ground truth points, but the ground truth points themselves are spread out in space. The three metrics together represent the variability of the samples, in high dimensional spaces, where a visual examination of the data is not possible. The subscript *EA* is omitted in the representation of the diversity metrics, through the rest of the paper, with the implicit understanding that these metrics are computed over the effective attacks set only.

4.1.2. Description of datasets used

Experimental evaluation is performed on 10 real world datasets, the details of which are presented in Table 2. The first 7 datasets were chosen from the UCI machine learning [25] repository and are popularly used for classification tasks, in literature. These datasets do not traditionally embody any security risks, but were chosen to evaluate the vulnerability of classifiers in the different data domains and distributions. The Spambase[7][25], KDD99[8][25] and the CAPTCHA [14] datasets are binary classification tasks, which represent 3 different cybersecurity domains that use machine learning as a core technique. The Spambase dataset, contains data about spam emails (such as fraud schemes, ads, etc) and legitimate personal and work emails. The KDD99 dataset is a network intrusion detection dataset, to classify normal connections from different classes of attack connections. The CAPTCHA dataset was developed in [14], for the task of blocking bots from human users, based on their mouse movement patterns, while solving a visual image based behavioral CAPTCHA puzzle.

All datasets were pre-processed by first reducing them to a binary class problem. The Digits dataset was reduced to have samples of the digit 0 and 8 only, KDD99 was reduced to represent only two classes - attacks and normal. The dataset was then converted to contain only numerical values by transforming categorical and nominal features to binary variables. The resulting number of features is shown in Table 2. The data was then normalized to the range of [0,1]. Instances were shuffled to remove any bias due to inherent concept drift. In all datasets, the class label 1 is taken to be the *Malicious* class and 0 is taken as the *Legitimate* class, as convention.

4.1.3. Experimental protocol and setup

All experiments begin with a seed sample, which is obtained by random sampling in the feature space. The Anchor Points(AP) attack requires only one legitimate seed sample while the Reverse Engineering(RE) attack requires one legitimate and one malicious sample. The seed phase concludes when the minimum required seed samples are obtained. The exploration probing budget $B_{Explore}$ is taken as 1000 samples and the number of attack samples required N_{Attack} is taken as 2000. For the AP attack, the neighborhood radius [R_{min}, R_{max}] is set at [0.1,0.5] and the exploitation radius is set at $R_{Exploit}$=0.1. The choice of [R_{min}, R_{max}] in the range of [0.1,0.5] covers a significant portion of the [0,1] normalized space. The dynamic radius adjusting technique of AP attacks, chooses an appropriate R_i value based on adjusting in this range. As such, we found that changing R_{max} and R_{min} had little impact, as the algorithm itself does a self adjustment in this range. By choosing a sufficiently large range of 0.1–0.5, we ensure that the algorithm has a wide scope to choose from. In our experimentation, we observed that the radius converges to <0.25, by the end of the exploration phase, making [0.1, 0.5] a sufficiently large starting range to explore within.

In case of the RE attack, a larger exploitation radius is taken as $R_{Exploit}$=0.5, due to additional validation from the surrogate learned classifier C'. Effects of varying this radius values are also presented in the analysis. The magnitude of dispersion λ_{max} is taken as 0.25, and it was found that changing this had little impact on the final results. The adversary's reverse engineered model is taken as a linear kernel SVM with a high regularization constant (c=10). This ensures that the model is robust and does not overfit to the explored samples, which are limited and inadequate to generalize over the entire space. All experimentation was performed using Python 2.7[9] and the scikit-learn machine learning library [33]. The results presented are averaged over 30 runs for every experiment.

[7] https://archive.ics.uci.edu/ml/datasets/Spambase.

[8] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[9] www.python.org.

Table 3
Results of Seed and Exploration phases, with linear defender model. Defender's accuracy indicates % of accuracy as perceived by the defender from the training data. Accuracy of RE model is given by the % accuracy, by testing the training data on the reverse engineered model.

Dataset	Defender's Initial Accuracy (%)	Random probes to find seed	Explored Anchor Points/$B_{Explore}$	Accuracy of RE model C' (%)
Digits08	98	4.6 ± 2.63	0.63 ± 0.01	92
Credit	79	3.13 ± 1.89	0.71 ± 0.01	71
Cancer	97	42.91 ± 29.36	0.99 ± 0.01	95
Qsar	87	49.5 ± 28.81	0.99 ± 0.01	42
Sonar	88	24.03 ± 18.92	0.98 ± 0.01	61
Theorem	72	4.07 ± 2.52	0.67 ± 0.02	57
Diabetes	78	2.93 ± 1.23	0.50 ± 0.02	71
Spambase	91	20.64 ± 12.93	0.50 ± 0.02	59
KDD99	99	6.07 ± 4.23	0.91 ± 0.01	55
CAPTCHA	100	7.27 ± 5.35	0.92 ± 0.01	91

4.2. Experimental results and analysis

Experimental analysis is presented here, by considering different models for the defender's black box, and measuring its impact on the adversary's effectiveness. Section 4.2.1 presents the results of a symmetric case, where both the adversary and the defender have similar model types (linear in this case). Results of a non symmetric setting are presented in Section 4.2.2, where we consider 4 different model types for the defender, while the adversary, agnostic of these changes, still employs a linear model. Experiments on a truly remote black box model is presented in Section 4.2.3, where we present experiments performed on Google's Cloud Prediction API. Effects of parameters on the adversary's performance is presented in Section 4.2.4.

4.2.1. Experiments with linear defender model

Experiments in this section consider a linear model for the defender's classifier C. A linear kernel SVM (regularization parameter, c=1, default provided by scikit-learn) is considered. This information is not available to the adversary, who is capable of accessing this model only via probing upto a budget $B_{Explore}$=1000.

The results of the *Seed* and *Exploration* phase are presented in Table 3. The initial accuracy of the defender, as perceived by cross-validation on its training dataset before deployment, is seen in Column 2 of Table 3. This measure is taken to understand the defender's viewpoint of the problem, where the defender being a capable data scientist, tries to maximize the predictive performance of the learned model. While other performance metrics of f-measure, precision and recall, could be employed, accuracy provides a sufficient symbolic indication of the defender's confidence in the learned model, over the presented datasets. A high accuracy ($>70\%$) is seen across all the datasets. The seed phase uses random sampling in the feature space to find seed samples. No more than 50 samples, on average, were needed for finding seeds to start the attack process. The number of Anchor Points obtained is seen to be $>50\%$ of $B_{Explore}$, indicating the ability to launch an AP attack on all 10 high dimensional domains. For the RE attack, the reverse engineering accuracy of model C' is computed by evaluating it on the original dataset, as an adhoc metric of C''s understanding of the original data space and the extent of reverse engineering.

After the exploration phase, 2000 attack samples are generated in the exploitation phase. The Effective Attack Rate (EAR) and diversity metrics are presented in Table 4 for both the AP and the RE attacks. It is seen that an EAR of 97.7% in the case of AP and $>91.2\%$ for the RE attacks, is obtained on average. This is seen even though the defender's model is perceived to have a high accuracy as per Table 3. Accuracy of classifiers is of little significance if the model can be easily evaded. The high effective attack rate for all 10 cases, highlight the vulnerability of classification models and the misleading nature of accuracy, in an adversarial environment, irrespective of the data application domain. The high EAR of the RE attacks, indicate that partial reverse engineering and the linear approximation of the defender's model surface is sufficient to launch an effective attack against it. This can be seen for the KDD99 dataset, which has a reverse engineering accuracy of 55% while its EAR for the RE attack was 93%. This is because, generating a high accuracy on the training dataset is not the goal of the RE approach. It is more concerned with generating a large number of diverse attack samples which would be classified as legitimate. This is possible even with partial reverse engineering. While a high RE accuracy indicates a high EAR (consider Cancer dataset), it is not a required condition for the RE attack, making it of practical use in high dimensional spaces.

The diversity of the RE attacks is higher than the AP attacks, on all three metrics, indicating - a larger spread of attacks, lower collocation of points and a uniform distribution in the attack space. This high diversity is obtained for RE, while still maintaining a reasonable high attack rate. The AP attacks, produces lower diversity but has high attack accuracy than the RE attacks. This is because the number of explored anchor points was $>50\%$ (Table 3), allowing a large scale AP attack to be feasible. The AP attack is therefore an attractive quick attack strategy in high dimensional spaces, irrespective of the attack domain, application type and the model used. The effectiveness of the RE attack depends on the ability of the surrogate model C' to represent the space of *Legitimate*ly classified samples by C. The reverse engineering task is dependent on the availability of enough probing budget and the complexity of the boundary represented by C. This is the cause for the higher variability in the EAR values for RE attacks in Table 4, as opposed to the AP attacks, where attacks are more tightly packed with the obtained anchor points, leading to lower variability.

The RE approach's EAR is low for the Credit, the Theorem and the Spambase datasets. In case of the Credit and Theorem dataset, the defender's accuracy is low, indicating a nonlinear separation/inseparability of samples. The RE accuracy approaches close to the defender's accuracy, but since the original model C has low accuracy, the reverse engineered model can only be so good. For the Spambase dataset, the majority of the features follow a heavy tailed distribution as shown for Feature #5 in Fig. 7. In such distributions, random sampling in the range [0,1] on each features is not the best choice. Integrating domain information which is commonly known, as in the case of text datasets having heavy tails, can be beneficial. However, following a domain agnostic approach here, a 71% attack rate is still achieved, indicating the viability of such attacks.

4.2.2. Experiments with non-linear defender model

The SEE framework considers a black box model for C. As such, it is developed as a generic data driven attack strategy, irrespective of the defender's model type, training data or the model parameters. To demonstrate the efficacy of these attacks under a variety of defender environments, experiments with different non linear

Table 4
Results of accuracy and diversity of AP and RE attacks, with linear defender model.

Dataset	Method	EAR	σ	KNN-dist	MST-dist
Digits08	AP	0.96 ± 0.01	0.23 ± 0.002	0.48 ± 0.01	0.41 ± 0.01
	RE	0.93 ± 0.06	0.273 ± 0.009	0.76 ± 0.01	0.65 ± 0.04
Credit	AP	0.98 ± 0.01	0.218 ± 0.001	1.19 ± 0.02	1.01 ± 0.02
	RE	0.80 ± 0.15	0.265 ± 0.001	2.22 ± 0.02	1.72 ± 0.31
Cancer	AP	0.99 ± 0.01	0.215 ± 0.001	0.38 ± 0.01	0.33 ± 0.01
	RE	0.99 ± 0.01	0.263 ± 0.001	0.5 ± 0.01	0.45 ± 0.01
Qsar	AP	1	0.216 ± 0.001	1.1 ± 0.01	0.94 ± 0.01
	RE	0.99+0.01	0.264 ± 0.001	1.71 ± 0.01	1.64 ± 0.01
Sonar	AP	0.99 ± 0.01	0.215 ± 0.001	1.37 ± 0.01	1.16 ± 0.01
	RE	0.98 ± 0.01	0.265 ± 0.001	2.22 ± 0.01	2.1 ± 0.015
Theorem	AP	0.97 ± 0.01	0.219 ± 0.002	1.05 ± 0.02	0.89 ± 0.02
	RE	0.87 ± 0.08	0.267 ± 0.002	1.96 ± 0.02	1.64 ± 0.15
Diabetes	AP	0.98 ± 0.01	0.217 ± 0.003	0.27 ± 0.01	0.23 ± 0.01
	RE	0.95 ± 0.04	0.262 ± 0.001	0.36 ± 0.01	0.31 ± 0.01
Spambase	AP	0.93 ± 0.01	0.233 ± 0.003	0.96 ± 0.02	0.79 ± 0.02
	RE	0.71 ± 0.2	0.273 ± 0.004	2.04 ± 0.06	1.39 ± 0.4
KDD99	AP	0.99 ± 0.01	0.215 ± 0.001	1.06 ± 0.01	0.91 ± 0.01
	RE	0.93 ± 0.04	0.263 ± 0.001	1.71 ± 0.01	1.53 ± 0.06
CAPTCHA	AP	0.99 ± 0.01	0.215 ± 0.001	0.80 ± 0.01	0.68 ± 0.01
	RE	0.97 ± 0.02	0.264 ± 0.001	1.22 ± 0.01	1.12 ± 0.03

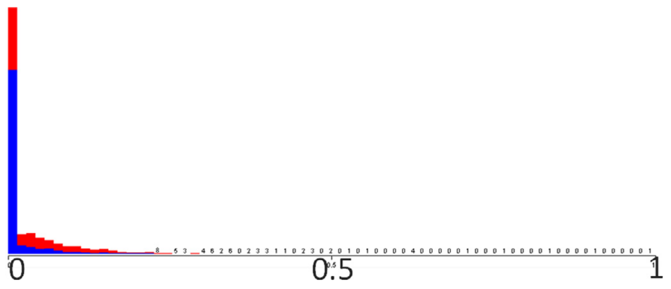


Fig. 7. Distribution of Feature #5 for Spambase dataset, showing a heavy tail. (Red - Malicious, Blue - Legitimate). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 5
Effective Attack Rate (EAR) of AP and RE attacks, with non linear defender's model (Low EAR values are italicized).

Dataset	*kNN*		*SVM-RBF*		*DT*		*RF*	
	AP	*RE*	*AP*	*RE*	*AP*	*RE*	*AP*	*RE*
Digits08	0.89	0.96	0.97	0.89	0.87	0.63	0.85	*0.48*
Credit	0.96	0.78	0.94	0.53	0.79	*0.42*	0.79	*0.33*
Cancer	0.99	0.99	0.99	0.99	0.97	0.89	0.99	0.98
Qsar	1	0.99	0.99	0.99	0.96	0.76	0.99	0.99
Sonar	0.99	0.98	1	1	0.97	0.62	0.99	0.95
Theorem	0.97	0.813	0.95	0.5	0.95	0.79	0.62	0.78
Diabetes	0.99	0.935	0.99	0.9	0.83	0.63	0.88	0.61
Spambase	0.93	0.99	*0.48*	0.84	*0.08*	*0.11*	0.99	0.98
KDD99	0.99	0.93	1	0.99	0.89	0.54	0.92	*0.27*
Captcha	0.99	0.92	0.99	0.92	0.97	0.83	0.93	0.89

black box models for C are presented here. Particularly, the following defender models were evaluated: K-Nearest Neighbors classifier with k=3 (kNN) [13], SVM with an radial basis function kernel with gamma of 0.1 (SVM-RBF) [48], C4.5 Decision Tree (DT) [35], and a Random Forest of 50 models (RF) [9], as shown in Table 5. The attacker's model is kept the same as before and the experiments are repeated for each of the defender's model. Average values of EAR over 30 runs are reported in Table 5.

The AP approach is minimally affected by the choice of defender's model, with Table 5 showing a high EAR for all defender models. The drop in case of Spambase, is attributed to the heavy tailed distributions as explained in Fig. 7. In case of the decision trees, the model trained for Spambase, focuses only on a few key features to perform the classification. Random probing attacks, space out the attack samples across dimensions, without considering their feature importance to classification. This leads to skipping over the key features in the attack generation, making the attacks less effective. However, this could be compensated by performing partial reverse engineering and using a smaller exploitation radius.

The RE results are significantly dependent on the defender's choice of model. In case of nonlinear data separation, as in the Credit and the Theorem datasets, the linear approximation is a bad choice and this is reflected in the low attack rate. In all other cases, the low attack rate is attributed to the over simplification of the understanding of the models, which in case of the decision tree and random forest tend to be complicated in high dimensional spaces. However, in a majority of the cases it is seen that a 50% attack rate is still possible with the same linear SVM model used by the adversary. This makes the SEE framework generally applicable to attack classification systems, without explicit assumptions about model types, application domain or the parameters of classification. The efficacy of these approaches, highlights the vulnerability of classifiers to purely data driven attacks, requiring only feature space information. The use of non linear models for the task of better reverse engineering and attack, is an interesting direction for further research.

4.2.3. Experiments with Google Cloud Prediction service

To demonstrate the applicability of the RE and the AP techniques on real world remote black box classifiers, we performed experiments using the Google Cloud Prediction API[10]. This API provides machine learning-as-a-service, by allowing users to upload datasets to train models, and then use the trained model to perform prediction on new incoming samples. Google's Prediction API, provides a black box prediction system, as they have not disclosed the model type or the technique used for learning, to the best of our knowledge. As such, this provides for an ideal test of the *SEE*'s attack models, where the defender is remote, accessed from a client and has no information about the defender's models [30]. We use the API's Python client library to access the cloud service,

[10] https://cloud.google.com/prediction/.

Table 6
Results of AP and RE attacks using Google Cloud Prediction API as the defender's black box.

Training accuracy	Spambase		KDD99		CAPTCHA	
	93%		99%		100%	
Attack			***Metrics***			
	AP	*RE*	*AP*	*RE*	*AP*	*RE*
EAR	1	1	1	1	0.99	0.97
σ	0.216	0.264	0.218	0.265	0.218	0.265
KNN-dist	1.324	2.148	1.105	1.714	0.813	1.228
MST-dist	1.127	2.078	0.944	1.645	0.695	1.131
Accuracy of RE model C'	48.1%		97.2%		100%	

and the results on the three cybersecurity datasets are shown in Table 6.

The results of experimentation using the Prediction API, on the three cybersecurity datasets of KDD99, CAPTCHA and Spambase, is shown in Table 6. These datasets are chosen as they represent complex and diverse real world applications, which have direct analogies to cybersecurity applications using machine learning models. The results of the experiment demonstrate that the AP and the RE attacks are effective in attacking the defender's classifier, by generating a high EAR over all datasets. The diversity of the RE approach is seen to be higher for the RE attacks on all three metrics of σ, $KNN-dist$ and $MST-dist$, indicating the variability of attacks achieved using the RE approach, in a real world setting. Furthermore, the RE accuracy in case of the Spambase dataset (48.1%) highlights that, linear approximation and partial reverse engineering are sufficient to launch an effective RE attack (EAR=1). These experiments use the same exploration budget ($B_{Explore}$=1000) as the previous sections, to generate attacks of high accuracy and high diversity. In a truly blind-folded setting, where we have no prior information about the defender's classifier, a budget of 1000 ($\approx$ \$0.5)[11] samples indicates the relative ease with which classifiers can be evaded and the need for a more comprehensive defense strategy, beyond a static machine learning model.

[11] https://cloud.google.com/prediction/pricing.

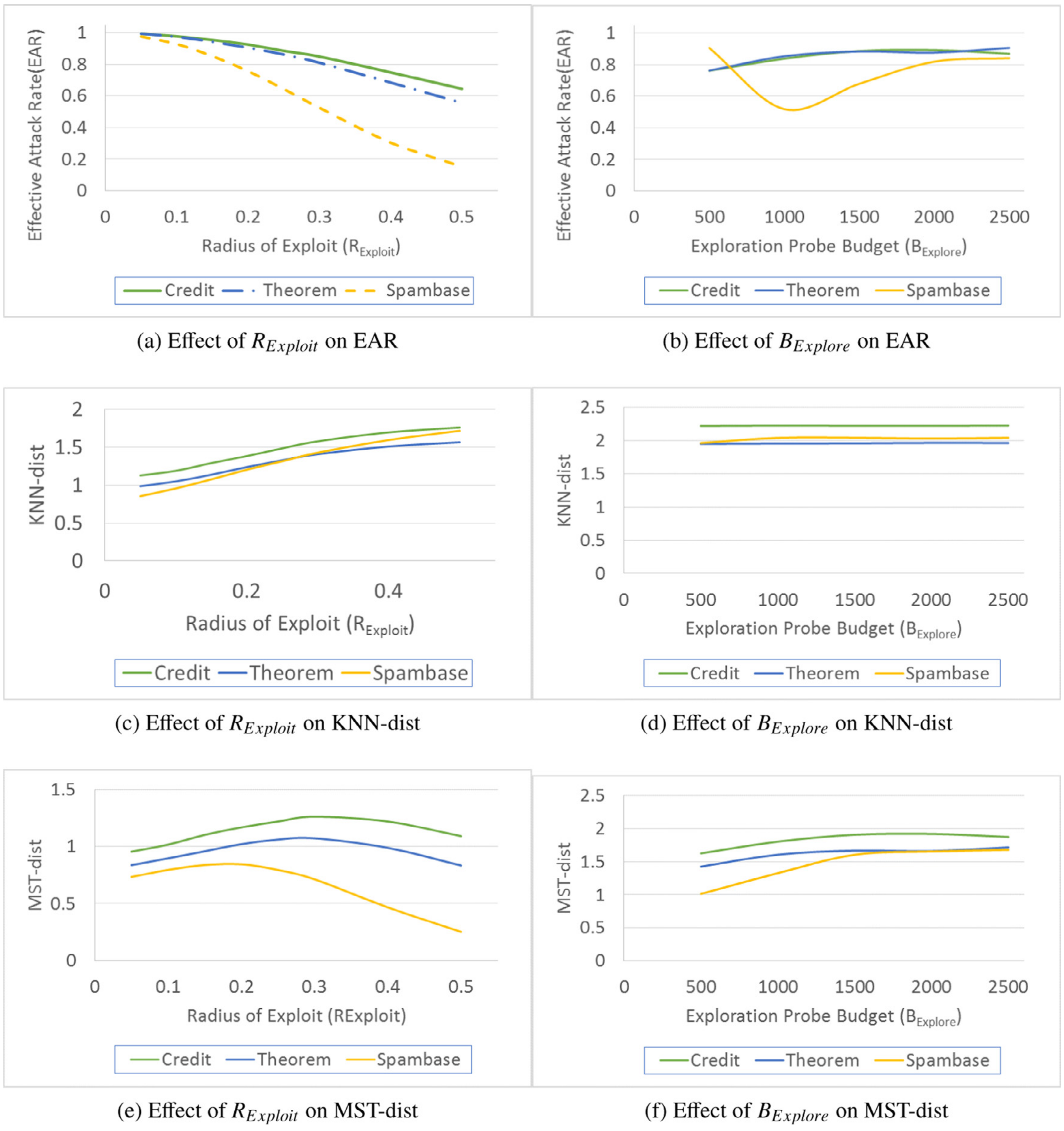


Fig. 8. Effect of changing $R_{Exploit}$, for the AP attacks (*Left*), and $B_{Explore}$, for the RE attacks (*Right*), on the Effective Attack Rate (EAR) and Diversity (KNN-dist, MST-dist).

4.2.4. *Effects of varying* $B_{Explore}$ *and* $R_{Exploit}$

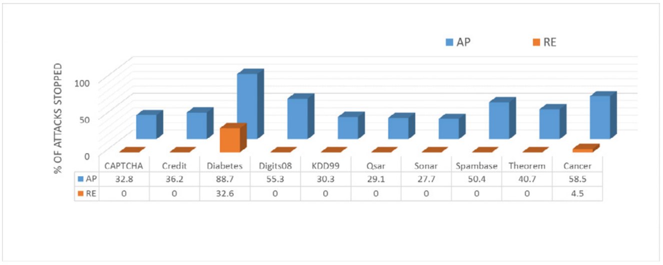
In evaluating the AP and RE approaches, the $R_{Exploit}$ was kept fixed at 0.1 for AP and 0.5 for RE. This was intuitively motivated, as confidence in attacks would reduce as distance from anchor points increases, as they are the only ground truth information available to the attackers in the AP strategy. Effect of increasing $R_{Exploit}$, on the accuracy and diversity of AP attacks, is shown in Fig. 8. The Credit, Theorem and the Spambase datasets were chosen for these evaluation, as they have low EAR for the RE approach (Table 4) and could therefore benefit from parameter tuning. Effect of increasing $R_{Exploit}$ to increase diversity of AP attacks, and increasing $B_{Explore}$ to increase EAR of RE attacks, as viable alternatives to improve performance over these three datasets is analyzed and presented.

The effective attack rate (EAR) reduces with an increase in the exploitation radius, as seen in Fig. 8(a), because attack samples move away from the anchor points. There is an associated increase in the diversity using both KNN-dist and MST-dist measures, as shown in (c) and (e). Comparison of diversity and EAR at $R_{Exploit}$=0.5 for the RE and AP approach shows, that for increasing diversity it is much better to switch to the RE approach instead of increasing $R_{Exploit}$ arbitrarily, as the effectiveness of attacks starts dropping rapidly with increased radius. The drop in MST-dist in Fig. 8 (e) is due to the reduction of the size of the Effective Attack set (EA).

As increasing diversity for AP approach leads to a drop in EAR, we investigate if we can increase the EAR of the RE approach while maintaining its high diversity. Increasing the exploration budget increases the EAR, due to availability of labeled training data for the reverse engineered model, leading to better learning of the data space. The increase in EAR ultimately plateaus, as per the Probably Approximate Learning(PAC) principles [17], indicating that it is not necessary to arbitrarily keep increasing this budget. The knee point is seen in Fig. 8(b) (around 1500 for all datasets). After the knee point, the EAR of all three datasets is 85%, and adding more probing budget has little impact on the EAR or the diversity(Fig. 8 d, f). It is necessary to have sufficient probing budget to reach this knee point, to allow effective reverse engineering in complex data spaces. This extra effort provides long term benefits as it leads to increased diversity of attacks. Below the knee point, the performance of the reverse engineered model is fairly unpredictable (as seen for the case of the Spambase datasets). However, after the knee point,the model is effective in providing high EAR. RE is suitable for patient adversaries who want to apply data science in breaking the system. In case of a low budget the AP approach is more suitable, but with the RE strategy the assumption is that the adversary wants to spend time to learn the system before attempting an attack

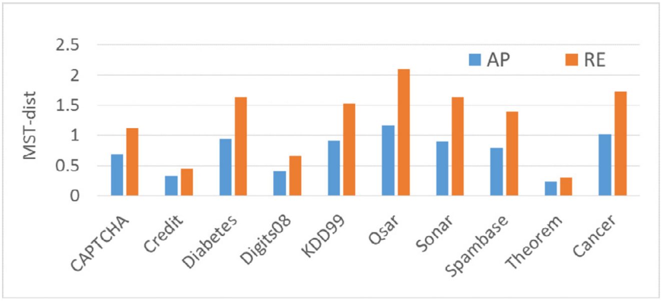
5. Why diversity is an important consideration in designing attacks?

Throughout the design and evaluation of the SEE framework, diversity of attacks has been considered as an important goal for the adversary. This was intuitively motivated, as diversity ensures that the attacks have enough variability, so as to make its detection and prevention difficult. In this section, we quantify the effects of diversity on the ability to thwart defenses, especially those based on blacklisting of samples. *Blacklists* are ubiquitous in security applications, as an approach to flag and block known malicious samples [21]. Modern blacklists are implemented using approximate matching techniques, such as Locality Sensitive Hashing, which can detect perturbations to existing flagged samples [34]. The goal of an attacker is to avoid detection by these blacklists, as they can make a large number of attack samples unusable with a quick filtering step. With high diversity, it is unlikely that blacklisting a few samples will cause the attack campaign to stop. In case of a diverse attack, the defender will have to resort to choosing between maintaining a huge blacklist of samples, or to remodel the machine learning system, both of which are expensive tasks and require time.



	CAPTCHA	Credit	Diabetes	Digits08	KDD99	Qsar	Sonar	Spambase	Theorem	Cancer
AP	32.8	36.2	88.7	55.3	30.3	29.1	27.7	50.4	40.7	58.5
RE	0	0	32.6	0	0	0	0	0	0	4.5

(a) Percentage of attacks stopped by blacklist



(b) MST-dist of AP and RE attacks

Fig. 9. Relationship between diversity and ability to thwart attacks by blacklisting samples.

To empirically evaluate the effect of diversity on blacklisting, a synthetic blacklisting experiment is presented, which simulates the effect of approximate matching filters. The blacklist is maintained as a list *BL*, of previously seen attack samples, with an associated approximation factor: ϵ. An attack is detected if a new sample falls within ϵ distance of any sample in the blacklist *BL*. The entire blacklisting process is simulated as follows: (i) the attackers use the SEE framework to generate $N_{Attacks}$ attack samples which are submitted to the defender model *C*, (ii) the defender is assumed to gain information over time about these $N_{Attacks}$ samples and then proceeds to blacklist them by storing them in *BL*, (iii) The attacker, still unaware of the blacklisting, continues to use its existing explored information (AP or RE model) to generate additionally more $N_{Attacks\_New}$ attack samples. The effectiveness of the blacklisting process is computed as the number of effective attacks in $N_{Attacks\_New}$, which are detected by *BL*. The *percentage of attacks stopped*, indicates effectiveness of blacklists and consequently the effect of diversity. A small rate would indicate that blacklisting is not effective in stopping such attacks.

Results of the blacklisting experiment, with an approximation factor of $\epsilon = 0.1$, on the UCI datasets is shown in Fig. 9. In order to balance effects of approximation across datasets, the approximation factor is multiplied by $\sqrt{d}$, where d is the number of dimensions of the dataset. The ϵ is chosen so as to balance the efficacy of the blacklists to its false positive rate. Increasing the approximation factor leads to an increase in the number of false positives, which causes legitimate samples to be misclassified as attacks. Limiting false positives is essential, as a blacklist which causes too many false alarms would be impractical. In all experiments, the exploration budget is kept fixed at 1000, the exploitation samples N_{Attack}=2000 and an additional 2000 samples(N_{Attack_New}) are generated to test the blacklisting effects.

From Fig. 9(a), it is seen that the AP approach has a higher percentage of stopped attacks than the RE approach, across all datasets. This is a result of the higher diversity of the RE attacks as seen from the MST-dist metric in Fig. 9 (b). The high diversity of the RE attack causes blacklisting to be totally ineffective (0 attacks stopped) for 8 out of the 10 datasets. In these cases, increasing the ϵ, as a countermeasure, to stop attacks is not a viable option, due to additional false alarms caused. A higher diversity in RE would force the reevaluation of the security system, leading to redesign, feature engineering and collection of additional labeled samples. All these are time taking and expensive efforts, making the RE approach effective as an attack strategy. As such, if an attacker is sophisticated and has enough probing budget $B_{Explore}$, it can launch a diverse attack campaign, which is harder to stop by adhoc security measures. The AP attack provides for high accuracy and precise attacks, as was seen in Table 4. However, a significant portion of this attack campaign (45%, on average) is stopped by using a blacklist capable of approximate matching. This experiment aims to highlight the effect of diversity and does not claim to be a concrete defense mechanism. Nevertheless, it intends to be a motivation for further analysis into the efficacy of incorporating such techniques for designing secure machine learning frameworks. Blacklists capable of heuristic matching, could empower classifiers [21] with the ability to recognize perturbed attack samples and as such continue to be effective against anchor points attacks. In case of reverse engineering attacks, a blacklisting based approach was seen to be ineffective. Taking preemptive counter-measures before deploying the classifier, to successfully detect and mislead attacks, could be a promising direction for dealing with these attacks [19].

6. Conclusion and future work

In this paper, an adversary's view of machine learning based cybersecurity systems is presented. The proposed Seed-Explore-Exploit framework, provides a data driven approach to simulate probing based exploratory attacks on classifiers, at test time. Experimental evaluation on 10 real world datasets shows that, even models having high perceived accuracy ($>90\%$), can be effectively circumvented with a high evasion rate ($>95\%$). These attacks assumed a black box model for the defender's classifier and were carried out agnostic of the type of classifier, its parameters and the training data used. Evaluation results considering 4 different non-linear classifier's and considering the Google Cloud Platform based prediction service, for the defender's black box, demonstrates the innate vulnerability of machine learning in adversarial environments, and the misleading nature of accuracy in providing a false sense of security. Also, the ability to reverse engineer the defender's classifier, and subsequently launch diverse attacks was demonstrated. Attacks with high diversity were shown to be more potent, as they can avoid detection and thwarting, by countermeasures employing blacklists and approximate matching.

The purpose of this work is to make the model designers aware of the nature of attacks that invade classification systems, from a purely data driven perspective. Wearing the white hat, we draw attention to the vulnerabilities introduced by using classifiers in cybersecurity systems. As Sun Tzu says in The Art of War- 'To know your Enemy, you must become your Enemy'[43]. We hope that this work serves as background and motivation for the development and testing of novel machine learning based security frameworks and metrics. Use of moving target defense strategies [19] and dynamic adversarial drift handling techniques [21,38], are promising directions warranting further research. Future work will concentrate on developing and analyzing preemptive strategies, in the training phase of the classifiers, to facilitate reliable attack detection and relearning for effective recovery. Also, extension of the framework to be used with multi-class and multi-label classification problems, is a direction for future work.

References

[1] M. Abramson, Toward adversarial online learning and the science of deceptive machines, in: Proceedings of 2015 AAAI Fall Symposium Series, 2015.
[2] Z. Akhtar, B. Biggio, G. Fumera, G.L. Marcialis, Robustness of multi-modal biometric systems under realistic spoof attacks against all traits, in: Proceedings of BIOMS 2011, IEEE, 2011, pp. 1–6.
[3] I.M. Alabdulmohsin, X. Gao, X. Zhang, Adding robustness to support vector machines against adversarial reverse engineering, in: Proceedings of the 23rd ACM CIKM, ACM, 2014, pp. 231–240.
[4] M. Barreno, B. Nelson, R. Sears, A.D. Joseph, J.D. Tygar, Can machine learning be secure? in: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, ACM, 2006, pp. 16–25.
[5] B. Biggio, I. Corona, D. Maiorca, B. Nelson, N. Šrndić, P. Laskov, G. Giacinto, F. Roli, Evasion attacks against machine learning at test time, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2013, pp. 387–402.
[6] B. Biggio, G. Fumera, F. Roli, Pattern recognition systems under attack: Design issues and research challenges, Int. J. Pattern Recogn. Artif. Intell. 28 (07) (2014) 1460002.
[7] B. Biggio, G. Fumera, F. Roli, Security evaluation of pattern classifiers under attack, IEEE Trans. Knowl. Data Eng. 26 (4) (2014) 984–996.
[8] L. Bilge, T. Dumitras, Before we knew it: an empirical study of zero-day attacks in the real world, in: Proceedings of the 2012 ACM conference on Computer and communications security, ACM, 2012, pp. 833–844.
[9] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001) 5–32.
[10] P.P. Chan, C. Yang, D.S. Yeung, W.W. Ng, Spam filtering for short messages in adversarial environment, Neurocomputing 155 (2015) 167–176.
[11] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[12] J. Chen, B. Xin, Z. Peng, L. Dou, J. Zhang, Optimal contraction theorem for exploration–exploitation tradeoff in search and optimization, IEEE Trans. Syst. Man Cybern.-Part A: Syst. Hum. 39 (3) (2009) 680–691.
[13] T.M. Cover, P.E. Hart, Nearest neighbor pattern classification, IEEE Trans. Inf. Theory 13 (1) (1967) 21–27.
[14] D.F. D'Souza, Avatar captcha: telling computers and humans apart via face classificationand mouse dynamics, Electronic Theses and Dissertations-1715, University of Louisville (2014).
[15] B.D.W. Group, et al., Big data analytics for security intelligence, Cloud Security Alliance (2013).
[16] P.H.C. Guerra, et al., Exploring the spam arms race to characterize spam evolution, in: Proceedings of the 7th Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS), 2010.
[17] D. Haussler, Probably Approximately Correct Learning, University of California,Computer Research Laboratory, Santa Cruz, 1990.
[18] J. He, J.G. Carbonell, Nearest-neighbor-based active learning for rare category detection, in: Proceedings of Advances in Neural Information Processing systems, 2007, pp. 633–640.
[19] J.B. Hong, D.S. Kim, Assessing the effectiveness of moving target defenses using security models, IEEE Trans. Depend. Sec. Comput. 13 (2) (2016) 163–177.
[20] J.H. Huh, H. Kim, Phishing detection with popular search engines: simple and effective, in: Foundations and Practice of Security, Springer, 2011, pp. 194–207.
[21] A. Kantchelian, S. Afroz, L. Huang, A.C. Islam, B. Miller, M.C. Tschantz, R. Greenstadt, A.D. Joseph, J. Tygar, Approaches to adversarial drift, in: Proceedings of the 2013 ACM workshop on Artificial intelligence and Security, ACM, 2013, pp. 99–110.
[22] B. Lacevic, E. Amaldi, Ectropy of diversity measures for populations in euclidean space, Inf. Sci. 181 (11) (2011) 2316–2339.
[23] H. Li, P.P. Chan, An improved reject on negative impact defense, in: Proceedings of International Conference on Machine Learning and Cybernetics, Springer, 2014, pp. 452–459.
[24] Y. Li, D.S. Yeung, A causative attack against semi-supervised learning, in: Machine Learning and Cybernetics, Springer, 2014, pp. 196–203.
[25] M. Lichman, UCI machine learning repository, 2013.
[26] D. Lowd, C. Meek, Adversarial learning, in: Proceedings of the 11th ACM SIGKDD, ACM, 2005, pp. 641–647.
[27] D. Lowd, C. Meek, Good word attacks on statistical spam filters., in: Proceedings of CEAS, 2005.
[28] B. Nelson, et al., Query strategies for evading convex-inducing classifiers, J. Mach. Learn. Res. 13 (1) (2012) 1293–1332.
[29] B. Nelson, et al., Near-optimal evasion of convex-inducing classifiers, arXiv preprint arXiv:1003.2751 (2010).
[30] N. Papernot, P. McDaniel, I. Goodfellow, Transferability in machine learning: from phenomena to black-box attacks using adversarial samples, arXiv preprint arXiv:1605.07277 (2016).
[31] N. Papernot, P. McDaniel, S. Jha, M. Fredrikson, Z.B. Celik, A. Swami, The limitations of deep learning in adversarial settings, in: Proceedings of IEEE European Symposium on Security and Privacy (EuroS&P), IEEE, 2016a, pp. 372–387.
[32] S. Pastrana Portillo, Attacks against intrusion detection networks: evasion, reverse engineering and optimal countermeasures, (Doctoral Thesis), Universidad Carlos III de Madrid(2014).
[33] F. Pedregosa, et al., Scikit-learn: Machine learning in Python 12 (2011) 2825–2830.
[34] P. Prakash, M. Kumar, R.R. Kompella, M. Gupta, Phishnet: predictive blacklisting to detect phishing attacks, in: Proceedings IEEE INFOCOM, 2010, IEEE, 2010, pp. 1–5.

[35] J.R. Quinlan, C4. 5: Programming for machine learning, Morgan Kauffmann (1993).
[36] N. Rndic, P. Laskov, Practical evasion of a learning-based classifier: a case study, in: Proceedings of IEEE Symposium on Security and Privacy (SP), 2014, IEEE, 2014, pp. 197–211.
[37] A. Saha, S. Sanyal, Application layer intrusion detection with combination of explicit-rule-based and machine learning algorithms and deployment in cyber-defence program, arXiv preprint arXiv:1411.3089 (2014).
[38] T.S. Sethi, M. Kantardzic, E. Arabmakki, Monitoring classification blindspots to detect drifts from unlabeled data, in: Proceedings of 17th IEEE International Conference on Information Reuse and Integration (IRI), IEEE, 2016.
[39] T.S. Sethi, M. Kantardzic, H. Hu, A grid density based framework for classifying streaming data in the presence of concept drift, J. Intell. Inf. Syst. 46 (1) (2016) 179–211.
[40] T.S. Sethi, M. Kantardzic, J.W. Ryu, Security theater: on the vulnerability of classifiers to exploratory attacks, in: Proceedings of 12th Pacific Asia Workshop on Intelligence and Security Informatics, (Under Consideration), Springer, 2017.
[41] C. Smutz, A. Stavrou, When a tree falls: using diversity in ensemble classifiers to identify evasion in malware detectors, in: Proceedings of NDSS Symposium, 2016.
[42] F. Tramèr, F. Zhang, A. Juels, M.K. Reiter, T. Ristenpart, Stealing machine learning models via prediction APIS, arXiv preprint arXiv:1609.02943 (2016).
[43] S. Tzu, The art of war edited by Samuel B. Griffith, 1963.
[44] D. Wagner, P. Soto, Mimicry attacks on host-based intrusion detection systems, in: Proceedings of the 9th ACM Conference on Computer and Communications Security, ACM, 2002, pp. 255–264.
[45] C. Walgampaya, M. Kantardzic, Cracking the smart clickbot, in: Proceedings of 13th IEEE International Symposium on Web Systems Evolution (WSE),, IEEE, 2011, pp. 125–134.
[46] L. Wang, X. Hu, B. Yuan, J.g. Lu, Active learning via query synthesis and nearest neighbour search, Neurocomputing 147 (2015) 426–434.
[47] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, F. Roli, Support vector machines under adversarial label contamination, Neurocomputing 160 (2015) 53–62.
[48] W.P.Z. Xiaoyan, Model selection of SVM with RBF kernel and its application, Comput. Eng. Appl. 24 (2003) 021.
[49] L. Xu, Z. Zhan, S. Xu, K. Ye, An evasion and counter-evasion study in malicious websites detection, in: Proceedings of IEEE Conference on Communications and Network Security (CNS), 2014, IEEE, 2014, pp. 265–273.
[50] W. Xu, Y. Qi, D. Evans, Automatically evading classifiers, in: Proceedings of the Network and Distributed Systems Symposium, 2016.
[51] M. Zamani, M. Movahedi, Machine learning techniques for intrusion detection, arXiv preprint arXiv:1312.2177 (2013).
[52] Y. Zhou, M. Kantarcioglu, B. Thuraisingham, B. Xi, Adversarial support vector machine learning, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2012, pp. 1059–1067.
[53] I. Žliobaitė, Learning under concept drift: an overview, arXiv preprint arXiv:1010.4784 (2010).

Tegjyot Singh Sethi received his Ph.D. from the Department of Computer Engineering and Computer Science, University of Louisville, USA. His research interests lie in the area of data mining, adversarial machine learning, change detection, learning with limited labeling, and data stream mining. He is a program committee member for INNS Big Data and ACIIDS, and has published several international conference and journal papers.

Mehmed Kantardzic is a Full Professor of the department of Computer Engineering and Computer Science (CECS) at the University of Louisville. Currently, he is the Director of the Data Mining Lab as well as the Director of CECS Graduate Studies at the CECS Department. His research focuses on data mining and knowledge discovery, machine learning, soft computing, click fraud detection and prevention, concept drift in streaming data, and distributed intelligent systems. He is the author of six books including the textbook: Data Mining: Concepts, Models, Methods, and Algorithms (John Wiley, second edition, 2011). He has served on the Editorial Boards for several international journals, and he is currently Associate Editor for WIREs Data Mining and Knowledge Discovery Journal.