


A Novel Data-Driven Attack Method on Machine Learning Models


Emre Sadıkođlu

(Yalova University, Yalova, Turkey)

 <https://orcid.org/0000-0002-7341-4621>, emre.sadikoglu@yalova.edu.tr


İrfan Kösosoy

(Kocaeli University, Kocaeli, Turkey)

 <https://orcid.org/0000-0001-5219-5397>, irfan.kosesoy@kocaeli.edu.tr

Murat Gök

(Yalova University, Yalova, Turkey)

 <https://orcid.org/0000-0003-2261-9288>, murat.gok@yalova.edu.tr

Abstract: With the increasing popularity and usage of artificial intelligence systems, it has become crucial to address their vulnerability to cyber-attacks. In this study, we propose a novel gradient descent-based method to generate fake data that can be accepted as positive by a targeted machine learning model. Our method is designed to generate a large number of positive samples with a minimal number of probes to the model, making it difficult to detect by security systems. Additionally, we develop an alternative model to the attacked model using a reverse engineering approach, trained on a dataset composed of the samples generated by our method. We evaluate the success of our proposed method and the alternative model through a series of experiments. We conducted experiments on six distinct datasets, each of which was trained using three separate machine-learning algorithms. This resulted in a total of eighteen unique models that were evaluated and compared in our analysis. In the evaluation of results, the most commonly used metrics in the literature, including effective attack rate (EAR), accuracy, precision, recall, and F1 score, were employed. Focusing particularly on EAR-oriented assessments, our method demonstrates its effectiveness with a notably high EAR of 97% in the combination of the kNN method and the Cancer dataset. According to the results of our experiments, the proposed method demonstrates high effectiveness as a data-driven attack method.

Keywords: Cyber security, Data driven attack, Adversarial data, Artificial intelligence

Categories: H.3.1, H.3.2, H.3.3, H.3.7, H.5.1

DOI: 10.3897/jucs.108445

1 Introduction

The increasing popularity of machine learning models in various domains has led to their widespread adoption in critical decision-making applications. However, the security of these models has become a growing concern in recent years. With the rise of cyber threats and adversarial attacks, the vulnerabilities of machine learning models have come into focus. These vulnerabilities can result in serious consequences, such as model manipulation, data breaches, and other malicious activities. There are several types of vulnerabilities that can exist in machine learning models, some of which are given in Table 1 by a short explanation.

Adversarial attacks are one of the most common types of attacks on machine learning models. These attacks work by intentionally adding perturbations to the input data to cause the model to make incorrect predictions [Xue et al., 2020]. Adversarial attacks can be categorized into two main types: targeted and non-targeted attacks. Targeted attacks involve an attacker trying to force the model to output a specific result. Non-targeted attacks, on the other hand, are attacks where the attacker is not trying to force a specific output but rather to cause the model to make any incorrect prediction [Dong et al., 2018].

Data poisoning attacks are another type of attack on machine learning models. In this attack, the attacker alters the training data set to manipulate the model's behavior. For example, an attacker could add false information to the training data to cause the model to predict a specific outcome [Chen et al., 2021].

Model inversion attacks involve an attacker trying to reverse-engineer (RE) the model to obtain sensitive information about the training data or the model's parameters. This type of attack can be used to extract sensitive information such as medical records or credit card numbers [Kobayashi et al., 2014].

Model stealing attacks are similar to model inversion attacks in that they involve an attacker trying to obtain the model's parameters. However, in this attack, the attacker does not try to reverse-engineer the model. Instead, they try to copy the model's parameters by querying the model with carefully crafted queries [Juuti et al., 2019].

Privacy attacks are attacks where an attacker can obtain sensitive information from the model [Papernot et al., 2018]. For example, an attacker could use a machine learning model trained on medical data to obtain sensitive information about patients [Wu et al., 2020].

Model substitution attack is a type of poisoning attack on machine learning systems that involves a malicious actor attempting to substitute a target machine learning model with a substitute model that is controlled by the attacker. In this attack, the attacker aims to manipulate the behavior of the target model by training a substitute model on a modified dataset that may contain malicious data or a backdoor. The goal of the attack is to replace the target model with the substitute model in a way that is undetected by the system's users or maintainers [Chen et al., 2017]. For example, imagine an online banking application that uses a machine learning model to predict which transactions are fraudulent. An attacker may attempt to substitute the bank's machine learning model with a malicious substitute model by first gathering data on legitimate transactions, and then training a substitute model on that data. However, the attacker also includes some fraudulent transactions in the training dataset, effectively "poisoning" the model. Once the substitute model is trained, the attacker deploys it in the bank's online application, substituting it for the original model. The attacker may then carry out fraudulent transactions that are specifically designed to evade detection by the substitute model. As a result, the bank's fraud detection system is effectively circumvented, and the attacker is able to carry out fraudulent transactions undetected.

In the scope of this study, we sought to answer the research questions provided below:

RQ1: Can we generate a multitude of distinct instances based on a single instance classified as positive by a machine learning model?

RQ2: While the obtained data is being sent to the machine learning model for evaluation, it may be blocked due to a high number of probes. How can we obtain the maximum number of data that the model will classify as positive with the minimum number of probes?

RQ3: Is it possible to train an alternative model to the existing machine learning model by using the derived data?

We can enumerate the contributions presented in the study as follows:

Attack Type	Explanation
Adversarial attacks	These are attacks where an attacker intentionally alters the input data to fool the machine learning model into making incorrect predictions.
Data poisoning	This is where an attacker adds malicious data to the training dataset to alter the model's behavior during the training phase.
Model inversion	In this type of attack, an attacker tries to reverse-engineer the model to obtain sensitive information about the training data or the model's parameters.
Model stealing	An attacker tries to copy the model's parameters by querying the model with carefully crafted queries.
Privacy attacks	Where an attacker can obtain sensitive information from the model, such as the training data or the model's parameters.
Model substitution attack	Malicious actor attempting to substitute a target machine learning model with a substitute model that is controlled by the attacker

Table 1: Some types of potential attacks can be launched against machine learning algorithms

-Leveraging a single positive example, we were able to generate a multitude of distinct positive examples rapidly (corresponding to RQ1).

-To address the possibility of probe blocking in the system, the number of generated positive samples per unit was maximized (corresponding to RQ2). At this stage, the best result in the study (cancer dataset - KNN model, see Table 4) was achieved by generating 97 out of 100 examples as positive.

-An alternative machine learning model was trained by using the derived dataset (corresponding to RQ3, see Table 7 for results). With this model, the generated examples can be tested before being sent to the actual model, thereby increasing the effectiveness of the attack.

The rest of this paper is organized as follows. In Section 3.1, proposed attack method is explained. In Section 3.2, learning methods are described briefly. In Section 4, the experimental setting and results are presented. And Section 5 concludes the paper.

2 Related Works

Below is a summary of several literature studies that have been conducted to investigate the security vulnerabilities discussed earlier.

[Sethi and Kantardzic, 2018] highlights the vulnerability of modern web applications to adversarial activity and the need for a data-driven solution to counter cyber-attacks. The paper presents an adversary's viewpoint of a classification-based system and introduces

the Seed-Explore-Exploit framework for simulating the generation of data-driven and reverse engineering attacks on classifiers.

[Biggio et al., 2014] discusses the vulnerability of pattern recognition systems in adversarial settings, particularly in security-sensitive applications like spam and malware detection. The authors analyze previous works and give examples of how such systems can be evaded by exploiting vulnerabilities in untrained components or learning algorithms. They propose the need for both reactive and proactive security paradigms to improve the security of pattern recognition systems.

[Fredrikson et al., 2015] discusses how machine learning algorithms are used in privacy-sensitive applications such as medical diagnoses, facial recognition, and lifestyle predictions. The authors introduce a new class of model inversion attacks that exploit confidence values to learn sensitive information about individuals. They experimentally show that the attacks can estimate whether a respondent in a lifestyle survey admitted to cheating on their significant other and how to recover recognizable images of people's faces given only their name and access to the ML model. The authors also investigate countermeasures to these attacks, including privacy-aware decision tree training algorithms and revealing only rounded confidence values.

[Shafahi et al., 2018] discusses the concept of data poisoning. The authors propose a type of data poisoning attack that uses clean labels, which means that attackers don't need to have control over the labeling of training data. They show that a single poisoned image can control classifier behavior in transfer learning and that reliable poisoning can be achieved using multiple (around 50) poisoned training instances.

[Hidano et al., 2018] highlights privacy concerns related to online services based on machine learning (ML) and the threat of model inversion attacks. The proposed general model inversion (GMI) framework captures scenarios where knowledge of the non-sensitive attributes is not provided. The paper proposes a new type of model inversion attack that can be carried out without the knowledge of the non-sensitive attributes by using the paradigm of data poisoning to inject malicious data into the training set. The paper provides a concrete algorithm of the model inversion attack on prediction systems based on linear regression models and evaluates the performance of the new attack through experiments with actual data sets.

[Feng et al., 2019] proposes a training time attack on deep learning models by generating bounded perturbations on the training data to manipulate the behavior of the corresponding classifier during test time. The authors use an auto-encoder-like network to generate adversarial perturbations on the training data and optimize them to cause the lowest performance for a victim classifier.

[Zang et al., 2021] explores the fragility of deep neural networks for graph-structured data and the severe security threats posed by small adversarial perturbations. The authors propose an algorithm called GUA to identify "anchor nodes," which are bad actors that compromise a trained graph neural network by flipping the connections to any targeted victim.

[Wu et al., 2022] addresses the threat of model extraction attacks against graph neural network (GNN) models, where a well-trained model can be stolen by an attacker pretending as a client. The authors formalize the threat modeling and classify the adversarial threats into seven categories based on different background knowledge of the attacker.

3 Methodology

This chapter provides a thorough description of the proposed attack method. Additionally, it presents the datasets that were utilized in the experimental studies. The learning algorithms used to classify the datasets are also briefly described.

3.1 Proposed Attack Method

The proposed method (Fake Data Generation - FDG) shows the vulnerability of binary classifiers. The main objective of the FDG method is to find a matrix used for mapping the seed data to a new sample that must be predicted as a positive class by the attacked classifier. The seed data is randomly selected from positive samples seen in Equation 1. The matrix used for mapping is initialized randomly, ranged from 0 to 1, and then updated by the gradient descent method. If the seed data is not accessible, it can be generated randomly. Then, the random seed is checked if it is positive or not by sending it to the classifier. Detecting the seed data is the most important part of the algorithm. After the seed data is determined, it can be used for deriving new samples. The process steps of the method are visualized in Figure 1. Additionally, Figure 1 illustrates how an alternative model is developed through reverse engineering the targeted model.

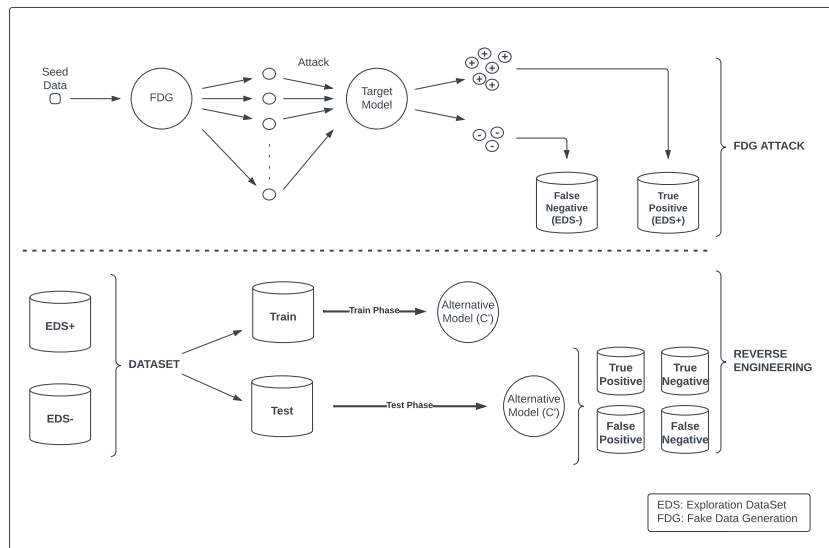


Figure 1: Illustration of the proposed attack method

The reverse engineering technique is employed as a means of mitigating the risk of security systems blocking attacks. This involves first subjecting the exploration sample to evaluation by the classifier C' , and only proceeding to submit it to the original classifier C if the predicted value is positive. The objective is to enhance the effectiveness of attacks by increasing the number of successful attempts made through this process.

$$seed = \{random(x) \mid C(x) = 1\} \quad (1)$$

Let's say the seed data is a vector X_s and the initialized mapping matrix is W_{old} with the size of $1 \times n$ and $n \times n$, respectively. And let's say the derived sample is a vector X_{new} and the updated mapping matrix is W_{new} . Given that the dot product of W_{old} and X_s is equal to the dot product of X_{new} and W_{new} as can be seen in Equation 2.

$$X_{new} \odot W_{new} = X_s \odot W_{old} \quad (2)$$

In Equation 2, all parameters are known except the X_{new} (W_{new} is updated version of the matrix W_{old} using gradient descent method. See Equation 8). X_{new} is obtained by using Equation 3.

$$\begin{aligned} X_{new} \odot W_{new} &= X_s \odot W_{old} \\ X_{new} &= (X_s \odot W_{old}) \oslash W_{new} \end{aligned} \quad (3)$$

Equation 3 is run inside an iteration (see pseudo code). In each iteration, the mapping matrices W_{old} and W_{new} are updated by using GD, but the seed data are not changed while deriving new samples.

GD is an iterative optimization method that finds the optimal values to minimize a cost function. A logarithmic function given in Equation 4 is used as cost function.

$$J(W) = \frac{1}{m} \sum_{i=1}^m cost(h(W, x^{(i)}), y^{(i)}) \quad (4)$$

$$Sigmoid : h(x) = \frac{1}{1 + e^{-W^T x}} \quad (5)$$

$$cost(h(W, x), y) = \begin{cases} -\log(h(W, x)), & \text{if } y = 1 \\ -\log(1 - h(W, x)), & \text{if } y = 0 \end{cases} \quad (6)$$

$$J(W) = - \left[\sum_{i=1}^m y^{(i)} \log h(W, x^{(i)}) + (1 - y^{(i)}) \log(1 - h(W, x^{(i)})) \right] \quad (7)$$

In Equation 4, the cost function takes two variables; the first one is the return value of the sigmoid function and the second one is the actual class value of the sample x^i . The sigmoid function also takes two variables; the first one is W which is optimized by the GD and updated in each iteration, and the second variable is the seed data x^i . The formula of the sigmoid function is given in Equation 5. Sigmoid function gives a value in the range of 0 to 1 which is used for predicting the class of the sample X . The cost function uses the actual class value y and the predicted value which is given by the sigmoid function to measure the prediction error. The formula of the cost function is given in Equation 6. The total cost value for m samples is given in Equation 7. The mapping matrix W is updated according to Equation 8. The new mapping matrix is

found by subtracting the derivative of $J(W)$ with regard to W_{old} . α is the predetermined learning rate parameter. The expanded version of the derivative of $J(W)$ is given in the equation 9.

$$W_{new} = W_{old} - \alpha \frac{\partial}{\partial W_{old}} J(W) \quad (8)$$

$$\frac{\partial}{\partial W_{old}} J(W) = \frac{1}{m} \sum_{i=1}^m (h(W, x^{(i)}) - y^{(i)}) x^{(i)} \quad (9)$$

3.2 Learning Methods

In this study, we have used three learning methods: k-Nearest Neighbour (kNN), Classification and Regression Trees (CART), and Support Vector Machine (SVM).

The kNN classification algorithm is a lazy learning method with a desirable computational speed along with acceptable classification accuracy. In the kNN algorithm, the distance of the data point, whose class is to be determined, to other points is calculated. Then, the k nearest neighbor points to this point are computed. Finally, the classes of k neighbors are voted and the data point is assigned to the class which has the most relevant points.

CART is one of the machine learning methods that can perform classification and regression analyses. This method divides data into two parts node and sub-node using Gini Index. The first node is the root. After selecting the root, the remaining features are divided into two nodes and this process is repeated until the last feature [Loh, 2014].

SVM is a statistical method that discriminates linearly separable patterns using an optimal hyperplane. The optimal hyperplane is a hyperplane that maximizes the margin of the hyperplane to the nearest point of each pattern to classify the given patterns correctly. [Hearst et al., 1998].

Algorithm 1 Generate attack data with GD

Input:

- seed,
- iteration N ,
- learning rate for GD

Output:

- attack data

```

attack data = {}
while iteration do
  exp-data = GradientDescent(seed);
  for  $i = 1, \dots, \text{size}(\text{exp-data})$  do
    if classifier.predict(exp-data[i])==1 then
      attack data  $\leftarrow$  exp-data[i];

```

4 Experimental Results

In this section, it will be argued whether the exploration data which is generated by the proposed method is close to the original data set or not. Experiments are performed on three different classifiers and six different data sets. A more detailed account of datasets is given in Section 4.1. To evaluate the results, quality metrics which are commonly used in the literature are used. The definition of the quality metrics is given in Section 4.2. The findings of the experiments are discussed in Section 4.3.

4.1 Datasets

The proposed method can only be applied to numerical datasets. Therefore, all the datasets used in the experimental study are composed of numeric values with six different datasets: Diabetes [Smith et al., 1988], QSAR [Grisoni et al., 2016], Credit [Häußler, 1979], Cancer [Wolberg and Mangasarian, 1990], Spambase [Hopkins et al., 1999], Sonar [Gorman and Sejnowski, 1988]. The number of the features and sample size of the datasets are given in Table 2. The data sets used in this study were obtained from the University of California Irvine (UCI) Machine Learning Repository. The dataset, consisting of data from positive and negative classes, is balanced so that the numbers of positive and negative data are equal.

4.2 Quality Metrics

In the results section, two different experiments were conducted. The first experiment is to observe the success of the proposed method which is exploration data generated by, and the second is the success of the classifier which is trained by reverse engineering approach. The original data sets are composed of positive and negative labeled samples

whereas the exploration data are only composed of positive labeled samples. Thus, there are no false positive and true negative predictions for the experiments that are applied to the exploration data. Therefore, the success of the proposed method cannot be evaluated by metrics such as precision, recall, and f1-score. These metrics are used to compare the reverse engineering approach with an attacked classifier. Formulas and a brief explanation of these metrics are given below. All these metrics can be derived from the confusion matrix that provides information about the number of the actual value of the instances predicted true or false. True positive (TP) and true negative (TN) refer to the correctly classified positive and negative samples respectively, and false positive and false negative shows the missed positive and negative samples.

Accuracy is the ratio of TP and TN to all positive and negative samples.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (10)$$

Precision is the ratio of TP out of Positives. Precision also gives a measure of relevant data points.

$$Precision = \frac{TP}{TP + FP} \quad (11)$$

Recall gives TP performance of our model. In other words, Recall is the coverage of positive samples.

$$Recall = \frac{TP}{TP + FN} \quad (12)$$

F-measure is the harmonic mean of Precision and Recall measures. F-measure provides a better assessment of the model performance in terms of the combined Precision and Recall measures.

$$F - measure = 2 \times \frac{precision \times recall}{precision + recall} \quad (13)$$

Kappa measure indicates whether the classifier's classification performance is by chance.

$$\kappa = \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e} \quad (14)$$

In Equation 14, p_o is the observed probability and p_e is the expected probability.

To measure the success of the proposed method, effective attack rate (EAR) is used. EAR is a metric that enables us to understand how many of the derived samples are

Dataset	# of features	# of positive samples	# of negative samples	# of total samples
Diabetes	9	268	500	768
QSAR	42	284	771	1055
Credit	21	300	700	1000
Cancer	11	212	357	569
Spambase	58	1812	2788	4600
Sonar	61	97	111	208

Table 2: The data sets used in the experiments

labeled as positive by the attacked classifier. EAR is the ratio of each exploratory data sample that manipulates the classifier to the total number of discovery data samples. The formula of the EAR is given by Equation 15.

$$EAR = \frac{|\{C(x) = 1 \ \& \ x \in x_exp\}|}{|x_exp|} \quad (15)$$

4.3 Results

We conducted two types of experiments: The efficiency of the attacks and the success of the classifiers (C') that are trained with the exploration data are measured and evaluated.

In the first experiment, before running the proposed method, positive labeled seed data are determined. Then, the seed data are given to our method to generate the exploration dataset. In this way, for each dataset that is mentioned in Table 2, an exploration dataset is generated. All the samples of the exploration data are reckoned as positive (That is why there are no true negative and false positive classified samples in the test results for the first experiment.).

kNN, CART, and SVM classifiers are performed on the datasets. As a result of the dataset and learning combinations, a total of 18 models are trained. Samples of the exploration data are given to the proposed model as inputs and the model returns a prediction value (positive or negative). Each exploration dataset is composed of 1000 samples. The goal of the algorithm is to get as many samples as possible into the original classifier as positive. The success of the algorithm is measured by EAR values. The EAR metric shows how many of the attacks are considered positive by the classifier. A large number of probes detected as negative by the classifier may cause the attack to be blocked by the security systems. Thus, a high EAR value is important in terms of the applicability of the algorithm.

The EAR results are shared in Table 3 and visualized in Figure 2 for better understanding. For each experiment, the algorithm is performed 10 times with 1000 samples. The given results are the mean of the 10 distinct experiments. Standard deviations of the results are given in Table 4. As can be seen from the standard deviations, the algorithm gives consistent results for each experiment. In general, when all datasets are considered, the best results are observed with kNN and SVM. CART is better than SVM for spambase and kNN for Sonar. It lagged behind SVM and kNN for all other datasets. When classifiers are compared, it can be said that CART is the most resistant classifier against attacks for our method. Considering the whole dataset and classifier combinations, the best result is acquired by the kNN model with the Cancer dataset. This dataset and classifier combination achieved 0.97 success, in other words, 970 out of 1000 samples are labeled positive by the classifier. It is also observed that the method generally gave better results in the Cancer dataset, and the worst results in the Diabetes dataset.

	Diabetes	QSAR	Credit	Cancer	Sonar	Spambase
kNN	0.74	0.85	0.93	0.97	0.81	0.84
CART	0.66	0.79	0.75	0.85	0.83	0.76
SVM	0.71	0.87	0.76	0.96	0.92	0.66

Table 3: EAR values of the classifiers

	Diabetes	QSAR	Credit	Cancer	Sonar	Spambase
kNN	0.006	0.001	0.006	0.006	0.003	0.007
CART	0.008	0.002	0.005	0.005	0.005	0.008
SVM	0.003	0.006	0.006	0.008	0.008	0.007

Table 4: Standard Deviations of EAR

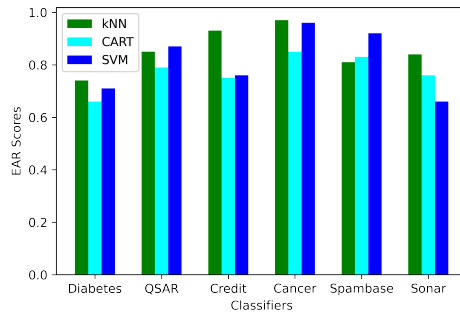


Figure 2: EAR Scores

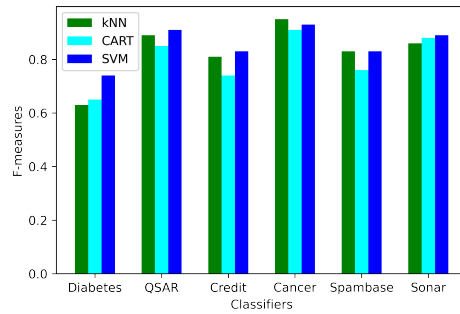


Figure 3: F-measures for RE

In the second experiment, the reverse engineering approach is used to decrease the risk of blocking attacks by security systems. To do this, the exploration sample will first be checked by the classifier C' . If the predicted value is positive, then, this sample will be sent to the original classifier C . It is aimed to increase the number of effective attacks made in this way.

The datasets shared in Table 2 are used to train the original models that are mentioned as C in this study. 10-fold cross-validation (CV) is applied to evaluate the performances of the models C . In a 10-fold CV, the dataset is randomly divided into 10 clusters with equal numbers of positive and negative samples. Nine sets are used for training and one set for testing. This process is performed by 10 iterations with a shift of the training and test clusters. Finally, the average values of the performance metrics belong to all iterations that are computed. Thus, the final performance of the model is obtained. The hyper-parameter values considered during the experiments are reported in Table 5.

	Hyper-parameters
kNN	k=3
SVM	C=10, tol=1.0E-5

Table 5: Hyper-parameters of the learning algorithms

The models' qualities are measured in terms of accuracy, precision, recall, F-measure, and Kappa metrics. The data which are used to train the C' models are generated with the proposed method. Each training dataset consists of 2000 samples, 1000 of which are labeled as negative and 1000 are positive. 10-fold CV is not used in the training of the C' models.

To measure the quality of the C' an external test set, which is composed of 500 negative and 500 positive samples, is used. The same test set is also used to show the classification results for C model. The classification results for the models C are given in Table 7.

Results for both classifiers are given in Table 6. It is not aimed to compare the success of the classifiers. Because it is expected from C' to classify both negative and positive samples correctly. All the metrics given in Table 6 have great importance. But in practice, there will not be any negative labeled samples sent to the attacked classifier. Therefore, it is not important whether the C classifier classifies the negative samples correctly. We evaluate the C classifier according to how well it classifies positive samples.

According to the performance results given in Table 6, it is revealed that the SVM classifier outperforms the other classifier methods on Diabetes, QSAR, Credit, and Spambase datasets from the point of all metrics. This means that the models trained with the reverse engineering approach have higher accuracy on predicting negative samples. But when the precision results are compared with the original models which are given in Table 5, it can be said that the RE classifiers have high predicting rates on positive samples too. The F-measurement results are also presented in Figure 3 as a bar graph.

The external dataset which is synthetically derived from the original dataset by the proposed gradient descent-based method is also used to test the original models (C). These results are also given in Table 6. Precision is the most important metric to evaluate the success of the proposed method. Because, as mentioned before, only positive samples are sent to the attacked model in practice. Thus, only the number of true positive and false positive predictions affects the success of the attack. Looking at the precision results,

		C'				
Dataset	Classifier	Accuracy	Precision	Recall	F-measure	Kappa
Diabetes	kNN	0.80	0.74	0.94	0.83	0.61
	CART	0.84	0.82	0.88	0.85	0.68
	SVM	0.77	0.68	0.99	0.81	0.54
QSAR	kNN	0.76	0.75	0.78	0.76	0.52
	CART	0.77	0.76	0.79	0.77	0.53
	SVM	0.84	0.78	0.93	0.85	0.67
Credit	kNN	0.80	0.79	0.81	0.80	0.59
	CART	0.78	0.73	0.88	0.80	0.56
	SVM	0.87	0.90	0.83	0.86	0.73
Cancer	kNN	0.83	0.75	0.99	0.85	0.66
	CART	0.82	0.74	0.99	0.85	0.64
	SVM	0.86	0.79	0.99	0.88	0.73
Sonar	kNN	0.79	0.71	0.88	0.82	0.57
	CART	0.76	0.67	0.91	0.80	0.55
	SVM	0.77	0.69	0.96	0.81	0.54
Spambase	kNN	0.67	0.61	0.97	0.75	0.35
	CART	0.72	0.88	0.52	0.66	0.45
	SVM	0.82	0.91	0.72	0.80	0.64
		C				
Diabetes	kNN	0.55	0.86	0.12	0.22	0.11
	CART	0.56	0.94	0.13	0.22	0.12
	SVM	0.55	0.91	0.12	0.22	0.11
QSAR	kNN	0.59	0.55	0.97	0.71	0.19
	CART	0.59	0.57	0.68	0.62	0.18
	SVM	0.61	0.57	0.92	0.70	0.22
Credit	kNN	0.52	0.51	0.98	0.67	0.04
	CART	0.57	0.67	0.26	0.38	0.14
	SVM	0.59	0.55	0.97	0.70	0.18
Cancer	kNN	0.65	0.85	0.37	0.52	0.31
	CART	0.58	0.69	0.31	0.42	0.17
	SVM	0.68	0.97	0.38	0.54	0.37
Sonar	kNN	0.53	0.25	0.23	0.18	0.29
	CART	0.53	0.41	0.23	0.26	0.49
	SVM	0.52	0.43	0.23	0.23	0.39
Spambase	kNN	0.75	0.91	0.55	0.67	0.50
	CART	0.75	0.69	0.89	0.78	0.50
	SVM	0.82	0.91	0.72	0.70	0.64

Table 6: $C'(X)$ and $C(X)$ classification results with external test set

four of the 18 experiments are below 50%, seven of them are in the range of 50-75%, and the rest are 85% and above. The best result was obtained with the SVM method and Cancer dataset combination with a rate of 97%.

Dataset	Classifier	Accuracy	Precision	Recall	F-Measure	Kappa
Diabetes	kNN	0.75	0.59	0.62	0.61	0.42
	CART	0.70	0.59	0.58	0.58	0.36
	SVM	0.82	0.79	0.56	0.65	0.53
QSAR	kNN	0.85	0.89	0.88	0.88	0.68
	CART	0.85	0.86	0.92	0.89	0.66
	SVM	0.85	0.87	0.89	0.89	0.67
Credit	kNN	0.67	0.72	0.77	0.74	0.28
	CART	0.60	0.64	0.66	0.65	0.19
	SVM	0.71	0.76	0.80	0.78	0.37
Cancer	kNN	0.96	0.94	0.95	0.94	0.91
	CART	0.93	0.93	0.87	0.90	0.85
	SVM	0.96	0.92	0.94	0.93	0.90
Sonar	kNN	0.88	0.95	0.79	0.86	0.77
	CART	0.71	0.74	0.71	0.73	0.42
	SVM	0.75	0.74	0.71	0.72	0.51
Spambase	kNN	0.81	0.76	0.74	0.75	0.59
	CART	0.91	0.87	0.89	0.88	0.82
	SVM	0.91	0.91	0.87	0.89	0.81

Table 7: Results of C classifiers which is trained with original datasets

5 Conclusion

The paper introduces a novel data-driven method for attacking machine-learning classifiers. The proposed technique generates large quantities of data at a fast rate, which is its primary advantage. We conducted two experiments to evaluate the approach. The first experiment assessed the efficacy of the attacks, while the second measured the success of the resulting RE models, which were trained using the exploration data. The results indicated that the method was highly effective in attacking binary classification algorithms.

It is worth emphasizing that this paper primarily concentrates on one specific type of attack, setting the stage for future research to explore comprehensive measures for safeguarding data and privacy against not only this attack but also its variations and similar threats. By addressing these concerns, we aim to pave the way for more resilient and secure machine learning systems.

To sum up, our study contributes a novel perspective to the realm of machine learning security, highlighting the importance of understanding and defending against data-driven attacks. The demonstrated effectiveness of our approach underscores the urgency for further research in this area to fortify the defenses of machine learning classifiers in an ever-evolving landscape of threats and challenges.

References

- [Biggio et al., 2014] Biggio, B., Fumera, G., and Roli, F. (2014). Pattern recognition systems under attack: Design issues and research challenges. *International Journal of Pattern Recognition and Artificial Intelligence*, 28(07):1460002.
- [Chen et al., 2021] Chen, J., Zhang, X., Zhang, R., Wang, C., and Liu, L. (2021). De-Pois: An Attack-Agnostic Defense against Data Poisoning Attacks. *IEEE Transactions on Information Forensics and Security*, 16:3412–3425.
- [Chen et al., 2017] Chen, P. Y., Zhang, H., Sharma, Y., Yi, J., and Hsieh, C. J. (2017). ZOO: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. *AISeC 2017 - Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, co-located with CCS 2017*, pages 15–26.
- [Dong et al., 2018] Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., and Li, J. (2018). Boosting Adversarial Attacks with Momentum-Mi-Fgsm. *Proceedings of the IEEE conference on computer vision and pattern recognition, (CVPR) 2018*, pages 9185–9193.
- [Feng et al., 2019] Feng, J., Cai, Q. Z., and Zhou, Z. H. (2019). Learning to confuse: Generating training time adversarial data with auto-encoder. *Advances in Neural Information Processing Systems*, 32(NeurIPS):1–11.
- [Fredrikson et al., 2015] Fredrikson, M., Jha, S., and Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. *Proceedings of the ACM Conference on Computer and Communications Security*, 2015-October:1322–1333.
- [Gorman and Sejnowski, 1988] Gorman, R. P. and Sejnowski, T. J. (1988). Analysis of hidden units in a layered network trained to classify sonar targets. *Neural networks*, 1(1):75–89.
- [Grisoni et al., 2016] Grisoni, F., Consonni, V., Vighi, M., Villa, S., and Todeschini, R. (2016). Investigating the mechanisms of bioconcentration through qsar classification trees. *Environment international*, 88:198–205.
- [Häußler, 1979] Häußler, W. (1979). Empirische ergebnisse zu diskriminationsverfahren bei kredit Scoringsystemen. *Zeitschrift für Operations Research*, 23(8):B191–B210.
- [Hearst et al., 1998] Hearst, M. A., Dumais, S. T., Osman, E., Platt, J., and B, S. (1998). Support vector machines. *Intelligent Systems and their Applications, IEEE*, 10(6):22.
- [Hidano et al., 2018] Hidano, S., Murakami, T., Katsumata, S., Kiyomoto, S., and Hanaoka, G. (2018). Model inversion attacks for prediction systems: Without knowledge of non-sensitive attributes. *Proceedings - 2017 15th Annual Conference on Privacy, Security and Trust, PST 2017*, pages 115–124.
- [Hopkins et al., 1999] Hopkins, M., Reeber, E., Forman, G., and Suermondt, J. (1999). Spambase data set. *Hewlett-Packard Labs*, 1(7).
- [Juuti et al., 2019] Juuti, M., Szyller, S., Marchal, S., and Asokan, N. (2019). PRADA: Protecting against DNN model stealing attacks. *Proceedings - 4th IEEE European Symposium on Security and Privacy, EURO S and P 2019*, pages 512–527.
- [Kobayashi et al., 2014] Kobayashi, S., Sawada, K., Hara, T., Kushima, H., and Kimura, K. (2014). Heat-to-heat variation in creep rupture ductility of ASME GR.91 steels in the long-term -investigation into recovery of microstructure and void formation-. *Advances in Materials Technology for Fossil Power Plants - Proceedings from the 7th International Conference*, pages 637–647.
- [Loh, 2014] Loh, W.-Y. (2014). Fifty years of classification and regression trees. *International Statistical Review*, 82(3):329–348.
- [Papernot et al., 2018] Papernot, N., McDaniel, P., Sinha, A., and Wellman, M. P. (2018). SoK: Security and Privacy in Machine Learning. *Proceedings - 3rd IEEE European Symposium on Security and Privacy, EURO S and P 2018*, pages 399–414.

- [Sethi and Kantardzic, 2018] Sethi, T. S. and Kantardzic, M. (2018). Data driven exploratory attacks on black box classifiers in adversarial domains. *Neurocomputing*, 289:129–143.
- [Shafahi et al., 2018] Shafahi, A., Ronny Huang, W., Najibi, M., Suciu, O., Studer, C., Dumitras, T., and Goldstein, T. (2018). Poison frogs! Targeted clean-label poisoning attacks on neural networks. *Advances in Neural Information Processing Systems*, 2018-December(NeurIPS):6103–6113.
- [Smith et al., 1988] Smith, J. W., Everhart, J. E., Dickson, W., Knowler, W. C., and Johannes, R. S. (1988). Using the adap learning algorithm to forecast the onset of diabetes mellitus. In *Proceedings of the annual symposium on computer application in medical care*, page 261. American Medical Informatics Association.
- [Wolberg and Mangasarian, 1990] Wolberg, W. H. and Mangasarian, O. L. (1990). Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the national academy of sciences*, 87(23):9193–9196.
- [Wu et al., 2022] Wu, B., Yang, X., Pan, S., and Yuan, X. (2022). *Model Extraction Attacks on Graph Neural Networks: Taxonomy and Realisation*, volume 1. Association for Computing Machinery.
- [Wu et al., 2020] Wu, M., Zhang, X., Ding, J., Nguyen, H., Yu, R., Pan, M., and Wong, S. T. (2020). Evaluation of Inference Attack Models for Deep Learning on Medical Data.
- [Xue et al., 2020] Xue, Y., Xie, M., and Roshan, U. (2020). Defending against substitute model black box adversarial attacks with the 01 loss. pages 1–6.
- [Zang et al., 2021] Zang, X., Xie, Y., Chen, J., and Yuan, B. (2021). Graph Universal Adversarial Attacks: A Few Bad Actors Ruin Graph Learning Models. *IJCAI International Joint Conference on Artificial Intelligence*, pages 3328–3334.