Hacking Smart Machines with Smarter Ones, How to Extract Meaningful Data from Machine Learning Classifiers

Critical report - IASD / Anonymization, privacy

Elie Kadoche - 24/02/2020

Abstract Privacy is a crucial problematic in the machine learning field. For example, It is always tricky to share to the world interesting and valuable data without compromising personal information. Such a problem, called data publishing is a well known problem and has been the object of numerous research. Some techniques has then been developed to allow personal data publishing without compromising privacy. But a similar - quite not identical - problem exists. If a classifier is released without its training data, is it possible, based only on such a classifier, to find the data on which it has been trained?

Introduction

The 19th June 2013 an Italian team published an article answering this question: "Hacking Smart Machines with Smarter Ones: How to Extract Meaningful Data from Machine Learning Classifiers" [1]. The authors explain how they have developed a classifier allowing them to "infer unexpected but useful information from ML classifiers", from their training set for example.

Even if the subject as already been the center of some research, this paper does not seems to rely on other works since the authors say they "put forward a new type [...] to the best of our knowledge, has not been considered before". Beside, this touch of caution is very appreciated. According to Google Scholar, this paper has been cited more than 124 times and seems to have inspired several researches about the same subject: attacking machine learning models.

In the following document we propose a critical analysis of this paper. To do so, our guideline will be the following: 1) present the context and the main idea of the paper, 2) present the authors, 3) describe the general attack strategy, 4) describe the different attacks made by the authors, 5) give my critics and conclude.

1 Extracting meaningful information from a classifier

Lets consider that we are the company A and our competitor is company B. Its classifier C_b is better than ours (C_a). The assumption is made that the algorithms used are quite standard and well known. In consequence, it is easy to replicate and optimizing them. From there, the reason why C_b is better than C_a is not the choice of algorithm but the choice of training data. In this paper, the authors show that "a classifier can be hacked and that it is possible to extract from it meaningful information about its training set".

The issues here does not concern privacy leaks, but more general information about training sets, information being the reason why a classifier would be better than another. Therefore, the type of leakage they are interested in "is quite different than that considered in privacy preserving data mining". The experiences are not done on commercial products due to legal issues but are done on similar open source projects. Classification problems are considered in this paper.

A classifier can be viewed as a condensation of its training data: we want to "compress" all the knowledge we have into a compact version, the classifier, in order to use it to a specific task. So it makes sense to say that it might be possible to extract information about the training set from the classifier only.

2 6 authors working in computer security

Overall the team seems to be a strong one and each member seems to have abilities and knowledge adapted to the the paper we are going to comment. From Università di Roma La Sapienza, Dipartimento di Informatica, **Giuseppe Ateniese**, is a researcher working on cloud security and machine learning applied to security and intelligence issues. He has today more than 17022 citations and seems influential in his field. His actual position seems very suitable for the article we are commenting here. **Luigi Vincenzo Mancini** is a professor researcher cited more than 6762 times, working cyber-security and network security. So it seems perfectly related with the paper. **Angelo Spognardi** is a researcher working in computer security, network security, applied cryptography and machine learning. **Domenico Vitali** is a Phd student under the supervision of professor Luigi Vincenzo Mancini. It seems logical that they are both working on the same paper. Working at Consiglio Nazionale delle Ricerche, Istituto di Analisi dei Sistemi ed Informatica Roma, **Giovanni Felici** works in particular in optimization and data mining, perfectly adapted to this paper. From Università di Roma Tre, Dipartimento di Matematica, **Antonio Villani** seems to be a less known researcher working in computer science and computer security.

3 A general attack strategy

The classifier from which we want to infer some information, the target classifier, is noted $\mathcal{C}_x$. Such a classifier can be encoded in a set of feature vectors $\mathcal{F}_{\mathcal{C}_x}$. For example, in the case of SVM (Support Vector Machines), it consists in a list of all the support vectors. This classifier has been trained on a data set $\mathcal{D}_x$. We want to know if a specific property $\mathbb{P}$ is preserved by $\mathcal{D}_x$, i.e $\mathbb{P} \approx \mathcal{D}_x$.

To do so, the authors describe a general attack strategy. First, we need to build a special data set $\mathbf{D}$ of several $\mathcal{D}_i$ with $i \in \{0, ..., n\}$. Ideally we want that 50% of $\mathbf{D} \approx \mathbb{P}$ and the other 50% $\approx \bar{\mathbb{P}}$ (i.e. the property $\mathbb{P}$ is not preserved by $\mathcal{D}_i$). Each data set $\mathcal{D}_i$ is associated to a label $l_i \in \{\mathbb{P}, \bar{\mathbb{P}}\}$. Next, we build a meta-classifier $\mathbf{MC}$ using the algorithm 1.

Now, since our meta-classifier is trained to recognize if the data set $\mathcal{D}$ on which a classifier $\mathcal{C}$ is trained preserves a certain property $\mathbb{P}$ by looking at its feature vectors $\mathcal{F}_{\mathcal{C}}$, we just need to give to our meta-classifier the feature vectors $\mathcal{F}_{\mathcal{C}_x}$ of the target model $\mathcal{C}_x$. The prediction of the meta-classifier will tell if $\mathbb{P} \approx \mathcal{D}_x$ or not.

Such a technique is possible because the assumption is made that the classifier $\mathcal{C}_x$ is disclosed after the training phase, it means that the adversary has full access to all of its parameters, its structure and its instruction sequences. In practice, we use a brute-force approach, by creating a different meta-classifier per property. An important point to note is that the property $\mathbb{P}$ does not appear in the attributes of the training set.

Algorithm 1: $\mathbf{MC}$ (meta-classifier) training

```
1 $\mathcal{D}_\mathcal{C} = \{\emptyset\}$ // Begins with an empty data set (main)
2 foreach $\mathcal{D}_i \in \mathbf{D}$ do
3     $\mathcal{C}_i \longleftarrow$ train($\mathcal{D}_i$) // For each dataset $\mathcal{D}_i$ previously built we train a classifier $\mathcal{C}_i$ over it
4     $\mathcal{F}_{\mathcal{C}_i} \longleftarrow$ get_feature_vectors($\mathcal{C}_i$) // We extract its feature vectors
5     foreach $a \in \mathcal{F}_{\mathcal{C}_i}$ do
6         $\mathcal{D}_\mathcal{C} = \mathcal{D}_\mathcal{C} \bigcup \{a, l_i\}$ // We add each feature vector associated to its label to the main dataset
7     end
8 end
9 $\mathbf{MC} \longleftarrow$ train($\mathcal{D}_\mathcal{C}$) // We train the meta-classifier over the main data set
```

4 Several attacks

4.1 Hidden Markov Models

Definition "A Markov Model is a stochastic process that can be represented as a finite state machine in which the transition probability depends only on the current state". In a HMM, only the outputs are observed. We describe a HMM by a set of hidden states $Q = q_1, ..., q_n$, a transition probability matrix $A \in \mathbb{R}^{n\times n}$ where $a_{i,j}$ is the probability of moving from state i to state j and a emission probability matrix $B \in \mathbb{R}^{m\times n}$ where $b_{k,j}$ is the probability to produce the observable o_k from the state j.

The problem The meta-classifier is a decision tree. The authors decided to used strongly biased training data sets to show how their attack strategy performs. A learning problem consists in reconstructing the 2 matrices A and B given the states Q and one (or more) observation sequence Y. The authors use specific algorithms to train the HMM. The attack is performed on a speech recognition system using HMM: the goal is to detect if the HMM was trained only with people from the same nationality.

The attack The property studied is $\mathbb{P}$ = *the classifier was trained only with people who speak an Indian English dialect*. The authors used 11137 recordings from the VoxForge corpus. They trained the meta-classifier using the algorithm 1. One key was to encode a HMM (not described here, because not interesting for this report) into a feature vector $\mathbf{a}$. On the test set, the results are the following: an accuracy of 0.909 for Indian and 0.907 for not Indian and a meta-classifier of more than 811 nodes with 610 leaves.

Improvements Since the decision tree is quite huge, the authors used a filter on $\mathcal{D}_\mathcal{C}$ to improve their meta-classifier. To do so, they used the Kullback-Leibler (KL) divergence between the output probability distributions of the models. The results are the following: an accuracy of 0.966 for Indian and 0.986 for not Indian and a meta-classifier of 21 nodes with 11 leaves, much simpler.

4.2 Support Vector Machines

The problem The definition of SVM is not done in this document (out of scope). The authors use SVM to build a Network Traffic Classifier able to distinguish between DNS and WEB traffic. The goal is to determine if the training set was mostly constituted of Google web traffic. So here, $\mathbb{P}$ = *the classifier was trained mostly with Google web traffic*.

The attack The meta-classifier is a decision tree. The authors decided to used strongly biased training data sets to show how their attack strategy performs. To train the meta-classifier, 70 data sets were created. The procedure is identical, and the results are the following: an accuracy of 0.93 for Google and an accuracy of 0.96 for not Google. Here again, the results are good.

4.3 Differential Privacy

Finally the authors show that Differential Privacy (DP), one of the most commonly used data protection, is ineffective against their attack strategy. The goal of DP is to minimize the probability of identifying one single record of a database. DP works with 3 different approaches, all ineffective against the authors' attack strategy.

Transform $\mathcal{D}$ First, DP can transform, "obfuscate" the original database $\mathcal{D}$ into $\mathcal{D}'$. However, during the classifier's training, $\mathcal{D}'$ is used and it exactly what the adversary is after. The adversary is not interested in $\mathcal{D}$.

Adding noise to the output Another way for DP to perform is to add noise to the output. It is also ineffective since the adversary has a full access to the classifier and can simply disable the noise.

Adding noise during the training Finally, DP can perform by adding noise during training. This approach is still ineffective because the authors say that "the final classifier must anyway converge to classify correctly the training set". So the effect of the noise is not that strong and does not affect this much the model.

A example attack To demonstrate the ineffectiveness of DP against their attack, the authors implemented 2 variants of a network traffic classifier that makes use of the K-Means algorithm. The data is the same used for SVM. The results are shown in the figures 1 and 2 (no DP) and 3 and 4 (with DP).

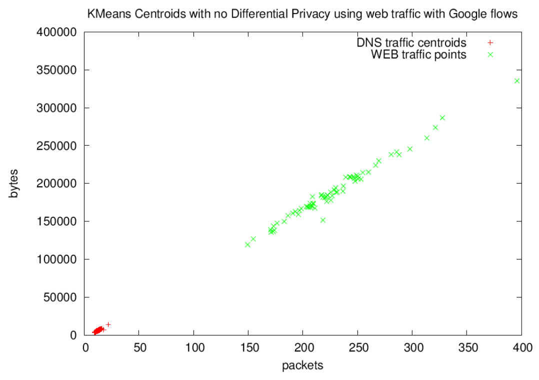

Figure 1: Training set contains Web traffic directed to Google

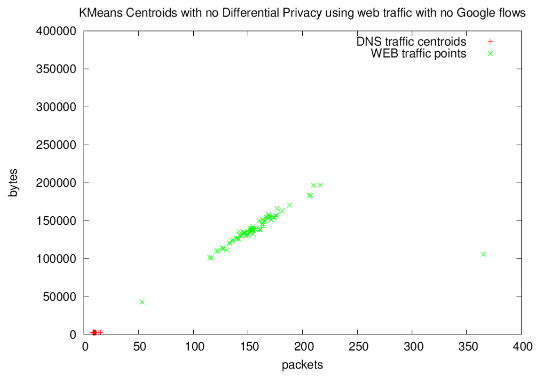

Figure 2: Training set does not contain Web traffic directed to Google

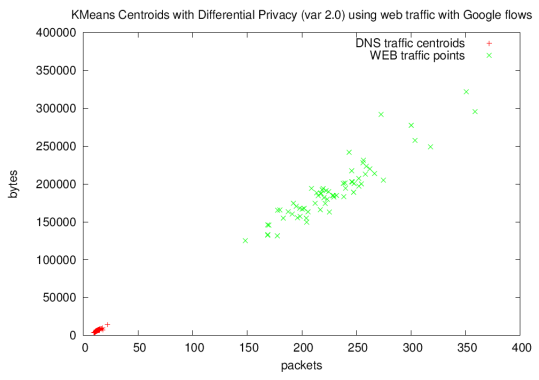

Figure 3: Training set contains Web traffic directed to Google

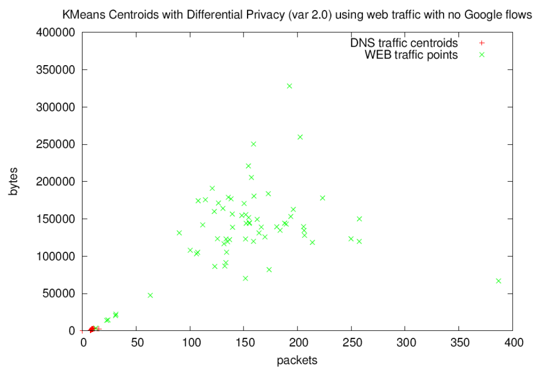

Figure 4: Training set does not contain Web traffic directed to Google

The authors run the 2 classifiers (with and without DP) on 70 training sets, obtaining 70 distinct centroids. The positions of the centroids are different in both case: the adversary can distinguish either there is a Google traffic or not.

What I understand I find the part on DP the less clear of the paper. What I more simply understand from their explications is that what their attack targets, is not what the DP protects. The role of DP is to avoid the identification on 1 single sample. The goal of their attack is to infer some statistical information about the training data. So DP can not really alter such an attack.

5 My critics

Strong assumption The assumption is made that the algorithms used are quite standard and well known, and so, the difference of performance of 2 classifiers often comes from the training set. In my opinion it is not always the case and often, the difference of performance comes from the algorithms and their optimization.

Disclosed classifier The assumption is made that the classifier is disclosed after the training phase. The authors say that this assumption is "reasonable since it is possible to extract the plain classifier also from a binary executable". In practice, I think today it is possible to release a classifier while keeping its inner parameters protected.

Difficulties in practice Since it is to the adversary to choose the properties he wants to infer from the target classifier, and since it is to the adversary to build a specific data set around this property, I think that the authors' attack strategy is difficult to use in practice. Indeed, build a data set can often be a difficult task. And in my opinion, for the meta-classifier to perform well, the data sets on which it is trained need to be as good as $\mathcal{D}_x$.

Biased training sets Since the authors decided to used strongly biased training sets to show how their technique works, We could ask ourselves if it is working as well on not biased training sets. Indeed, can the meta-classifier distinguish if a data set $\mathcal{D}$ preserves a property $\mathbb{P}$ when $\mathcal{D}$ is more balanced? The conditions in which the authors made their experiments are very far from the reality.

Typography The typography is very bad and not homogenize. Indention has to be done at the beginning of each paragraph or not done at all, but not both. The paragraphs are not used properly: a space bigger than the one being used between lines of a paragraph must be used to separate paragraphs. In the references there is a link going out of the page, etc. I find it very not serious for a scientific paper, even if the paper is good: this is the role of LaTeX to make good and clean presentation, and here the presentation is not done correctly in my opinion, which taint the entire paper.

Reinforcement learning Today many models are trained using reinforcement learning: there is no more "real" training set. And all information which can be extracted from such a model will be in a sense artificial, because created by the model itself.

Conclusion

This paper presents a novel data leakage, infer statistical information about the training set of classifier, and a general attack strategy to exploit it. I find this new data leakage very interested: it makes a individual or a company ask more questions about the releasing of a classifier. Data breaches can be everywhere and point out a new way to infer data from a model is a great discovery in my opinion.

Then, the authors present an attack strategy to exploit this classifier's weakness. I find this strategy quite smart but very brutal. This is basically brute-force and the time and computer power it requires are in my sens huge. Building multiple data sets, train multiple classifiers, train a meta-classifier over it... It is a lot and I do not think that such a technique can be used efficiently in practice.

But I am curious about performing their attack strategy on artificial neural networks (for the target classifier and for the meta-classifier). In their paper, the authors performs some experiments to demonstrate the effectiveness of their attack. Even if they get good results, I find these experiments too far away from the reality to be significant.

So overall I found this article interested and I learn a new kind of attack on a classifier. Using the feature vectors of a model to infer some statistical information about the training set is smart. Also, their attack strategy, as interesting as it is, might be difficult to use in practice.

References

[1] Giuseppe Ateniese, Giovanni Felici, Luigi V. Mancini, Angelo Spognardi, Antonio Villani, and Domenico Vitali. Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers. *CoRR*, abs/1306.4447, 2013.