Detecting Fraudulent Advertisements on a Large E-Commerce Platform

Tim Zimmermann[1], Timo Djürken[1], Arne Mayer[1], Michael Janke[1],
Martin Boissier[2], Christian Schwarz[2], Rainer Schlosser[2], Matthias Uflacker[2]
[1,2]Hasso Plattner Institute, University of Potsdam, Germany
{firstname.lastname}@{[1]student.hpi.de, [2]hpi.de}

ABSTRACT

E-commerce platforms face the challenge of efficiently and accurately detecting fraudulent activity every day. Manually checking every advertisement for fraud does not scale and is financially unviable. By using automated learning algorithms, we can drastically reduce the number of advertisements that need to be checked by humans. In this paper, we present the results of a joint project with a large e-commerce company selling used goods. Using our partner's advertisement data, we implemented several classification approaches to automatically recognize fraudulent activity. With the help of the proposed fraud detection, customer service agents only need to check about 8% of all advertisements manually for fraud. Simultaneously, we detect more than 93% of all fraudulent advertisements.

Keywords

Fraud Detection; Random Forests; XGBoost; Logistic Regression; Boosted Trees; E-Commerce

1. DETECTING ONLINE FRAUD

The success of e-commerce platforms strongly depends on the trust that customers have in it. If a platform is home to fraudulent offerings, customers are less likely to interact with that platform. Hence, minimizing the number of fraudulent advertisements is crucial for every e-commerce company.

In 2012, online retailers lost an estimated revenue of USD 3.5 billion due to fraud [2]. This figure does not even include the expenses that retailers face to fight crime on their platforms. Technology already helps to reduce the number of people required by automatically suggesting a classification for a given offering. However, many companies still have to employ a large number of dedicated staff to make the final decision whether an offering is fraudulent or not.

Fraudulent activity cannot always be easily identified. It is crucial to take the dependencies between various characteristics of the offering into account. While these dependencies can be simple in some cases, they are often complex and not at all obvious to humans. Approaches using hand-crafted rule sets pose several problems. On the one hand, it is hard for humans to recognize fraud patterns and maintain them over time as fraudsters adapt their behavior. To create such rules, highly skilled experts are needed which are both rare and expensive. On the other hand, fraudsters and companies are in constant competition to come up with ever-improving strategies to carry out fraud and to fight it, respectively. Therefore, complex models comprised of rules that humans once developed are likely to prove ineffective once fraudsters steadily evolve their strategy. Fraud detection needs to be both adaptable and verifiable.

In general, there is always a trade-off between correctly identifying fraudulent advertisements on the one hand and incorrectly classifying actually legitimate advertisements as fraudulent on the other. For every e-commerce company it is expensive to have fraud on the platform. Consequentially, we want to identify as many of the fraudulent advertisements as possible. We can accept if we incorrectly classify some of the legitimate advertisements as fraudulent, because the automatic classification is solely used as a filter for the customer support agent. These agents then check those advertisements manually. Therefore, if a fraudulent advertisement is not classified as such it will not be checked by an agent unless a user files a complaint. On the other hand, if an advertisement is classified as fraudulent but is actually legitimate, the customer agent can overrule the classification of the algorithm. Another challenge of hand-crafted rules for classification is to determine how likely a given offering is fraudulent based on the rule set. However, having that functionality can reduce the workload of the agents reviewing the classification as the probability of a prediction being correct above a certain threshold might already be sufficient. In that case, the advertisement's classification could be determined without any human looking at it.

In this paper, we compare three approaches to classify fraud: logistic regression and decision tree-based models in the form of Random Forests and XGBoost. We compare their effectiveness to recognize fraud with two different feature sets: One feature set is based on the characteristics and values of the offerings (see Section 4). The other one is based on the average fraud probability for each of these values.

1.1 Columnar In-Memory Databases

For our implementation, we decided for an architecture using the statistical language *R* and a columnar in-memory database (in our case *SAP HANA*). The decision for R was rather straightforward as R is open source and the de-facto standard for fast prototyping of machine learning algorithms.

The decision for a columnar in-memory database was made for several reasons. Databases with analytical capabilities allow to us to select a new data set for training on the fly without long running MapReduce or ETL (extract, transform, and load) jobs. At the same time, we can exploit the fact that we always have access to the most recent data to train our models on [4]. This is particularly interesting for identifying the steadily changing behaviour by fraudsters. Whenever a major shift in fraudulent behaviour is observed, models can be retrained within minutes.

During this project, we have used the columnar database to select data on the fly and used a stand-alone R installation for iterative work on our models. In case our proposed approach shall be put into production, there are two ways – both offering high performance – to proceed. First, R and HANA can be linked using the system's shared memory as described in [3]. This way, data can be directly shared between the two processes without any serialization or network transmission, while still having all capabilities of R at hand. Second, SAP HANA provides the so-called *predictive analysis library* (PAL) offering a large variety of machine learning algorithms that are directly executed on the actual data inside of the database engine.

2. PRELIMINARIES

For our research, we worked with data from a production system of our project partner. It includes a subset of advertisements along with some meta information (see Figure 1).

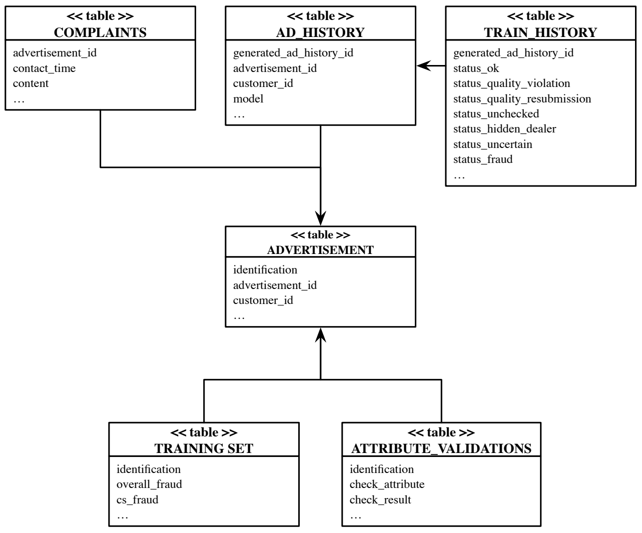


Figure 1: Data schema of our project partner's data set.

The data set contains a total of ~1.4 million advertisements. For some of these advertisements there is more than one version, which means that the advertisement has changed over the observation period. We treat these versions as separate advertisements and can therefore work with about 1.9 million advertisements. For around 1.5 million ($\approx$ 80%) of those advertisements we know the correct classification, i.e., whether a given advertisement is fraudulent. Apart from the current states of advertisements, the data also includes so-called *attribute validations* as well as historical information. Attribute validations are checks that are performed whenever an advertisement is created or changed, e.g., comparing the price to the average price for this offered item. The history table contains different versions of the advertisements.

3. STRATEGIES FOR MODELING FRAUD

In order to successfully reduce the number of advertisements that have to be checked manually, there are two major aspects we need to consider. First, the algorithm should produce as few false negatives as possible. That is, it should not misclassify as legitimate advertisements that are in fact fraudulent. At the same time, the pool of possibly fraudulent advertisements should be as small as possible. Second, the algorithm needs to keep up with the latest strategies of fraudsters. Fraudsters react to defensive mechanisms employed by companies and improve the techniques they use to fool and exploit both users and companies.

3.1 Dynamic Adaption

Traditional approaches are based on domain experts. They use their knowledge and experience to create a set of rules. A rule consists of a number of checks that an advertisement is tested against. The result is a classification as fraudulent or legitimate, depending on whether the rule matched or not. Results of multiple rules are aggregated to make a final decision on the classification of the advertisement.

This approach has two major drawbacks. First, it takes a lot of time to develop a sophisticated set of rules that are tuned to the individual use case. Second, these rules constantly need to be updated by these experts, which in turn again takes time. Besides having to pay the experts, this time span can already cause very significant damage and result in big losses to the company.

In contrast, machine learning offers techniques with which the criteria to classify advertisements can be generated semi- or fully automatically, while at the same time adapting to the ever-changing deceptions of fraudsters. We compare three different techniques to classify advertisements: logistic regression, Random Forest, and XGBoost. The process of engineering the criteria is described in Section 4.

3.2 Logistic Regression

Logistic regression is a form of supervised learning. A function is fitted to a labeled data set. In our case the advertisements are labeled with *fraud (1)* or *no fraud (0)*. The function is then used to predict whether new advertisements are fraudulent or not. The result is always a value between 0 and 1 - the higher it is, the more likely that advertisement is fraudulent. Due to the nature of regression it is not able to work with categorical (non-numerical) features. One work-around is to transform categorical into binary features, so-called *one-hot encoding.* Assume the *brand* attribute only had three distinct values: *Nike*, *Reebok* and *Adidas.* Instead of having a single feature *brand* we would have three features, specifically `is_nike`, `is_reebok` and `is_adidas`. The value of each of these features is either 0 or 1 and it can thus be used by logistic regression. The main disadvantage of this approach is that it adds a lot of features to the model and therefore increases the computation time required to generate the model. At the same time, many classification algorithms are limited in the number of features they can efficiently handle.

3.3 Random Forest

Random Forests are a collection of decision trees that are trained by repeatedly taking random samples from the training set. As it also relies on labeled data it is a supervised learning technique as well. In contrast to logistic regression, Random Forests can natively handle categorical features with an arbitrary number of distinct values. However, the R-implementation we used is only able to handle features with a maximum of 53 distinct values. While this is enough for some of the features we used, it does not solve the problem in general. The `brand` attribute, for example, has more than 53 different brands in it. Often times it is not necessary to know the exact brand - it might be sufficient to know the segment, e.g., *sports shoes* or *running shoes*. The segment can be used as a "reduced" feature instead.

We built a forest of 4,000 trees. This forest was assembled by 20 parallel workers building forests of 200 trees each, which were combined to a single model in the end.

3.4 XGBoost

XGBoost – for e**x**treme **g**radient-**boost**ed trees – is a recent algorithm that has attracted many researchers over the past year [1]. Similar to Random Forests, XGBoost creates (boosted) decision trees and belongs to the group of supervised learning techniques. One of the key advantages of XGBoost – besides the pure learning results – is its performance and scalability, allowing for very fast training and thus faster parameter tuning.

3.5 Visualizing the Decision Process

One of our main objectives was an easy-to-understand and digestible visualization of the automated fraud detection. Understanding why a – from a customer agent perspective – black-box algorithm classified a particular advertisement as legitimate or fraudulent is essential to build up trust between the system and the customer agent. Additionally, it helps the users to quickly decide which characteristics of the advertisement to look at.

3.5.1 Logistic Regression

In the training stage, logistic regression builds a model which consists of an intercept value and feature coefficients that are used to predict the fraud probability. We built two visualizations in order to help the customer agent better understand the learned model. The first model is supposed to give the customer agent an overview of the features that the model uses to predict. Hereby, we simply show the highest/lowest coefficients of the model (please note that normalization is required for this step). This very simple approach helps to make the results interpretable as the customer agent gains trust in the model as it (hopefully) detects fraud characteristics that conform with the agent's experience.

The second model is rather simple and optimized to help the agent decide whether a particular advertisement is legitimate or fraudulent (for an example see Figure 2). For all historical legitimate and fraudulent advertisements, we calculate average explanatory variables that are put into the regression model. Now, for the advertisement that is currently examined, we compare against the advertisement's explanatory variables and show the ratio (i.e., the 'fraud influence'). We do not aim to give the customer agent a full understanding of the model's decision as even simple models are usually to complex to re-enact mentally in a reasonable time frame. Instead, we want to give hints which features should be examined and allow to check for plausibility.

3.5.2 Decision Tree-Based Models

While it is possible to visualize decision trees, it is difficult to condense the process into a few simple but meaningful figures. On the one hand, providing a figure for each tree of a larger forest is unpractical for the number of trees in our models. On the other hand, using a random sample of trees can be non-representative and might confuse the user.

Hence, we decided to solely show the relevance results of the logistic regression to the customer agents.

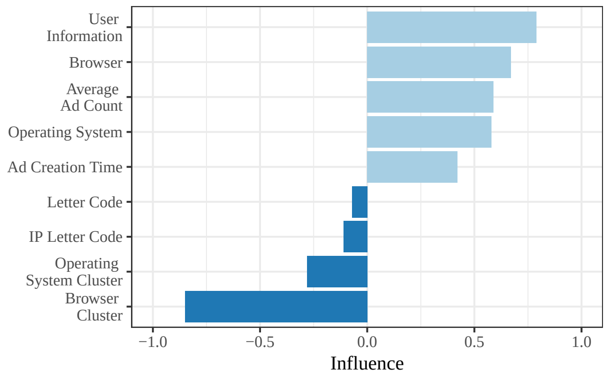


Figure 2: Visualization of the relevant features and their influence on fraud probability for an exemplary advertisement.

4. FEATURE ENGINEERING

To detect fraud, we worked with two very different feature sets. The first feature set relies on the attributes and features provided by our project partner as well as additional features we manually created. We call this feature set *value-based feature set*. The second feature set is based on the average fraud probability for a given value and we therefore call it *average feature set*.

4.1 Advertisement Characteristics

The value-based feature set consists of multiple features with varying complexity. This includes basic attributes like the geographical origin of the user's IP address but also more complex combinations such as the *IP Usage Counter*. A fraudster might have numerous advertisements online at the same time to increase the chances of success. For this case we calculate the number of advertisements that were created with the same IP address and try to find IP addresses with a large number of advertisements.

IP addresses can also be analyzed with regards to their usage and connections to fraud in the past. However, it is not easy to identify users based on the IP since for most customers the IP changes every day. Nevertheless, the IP can in many cases be connected to an internet service provider simply by using its sub ranges. Users typically only log in with IP addresses of a small number of different providers, which in turn means that the number of different providers can be used as a feature. The current fraud detection system generates attribute validations (see Section 2). We have included the most useful of these attribute validations in our feature set as well. In total we are using 24 (partially hand-crafted) features in our value-based feature set.

4.2 Using Historical Fraud Information

This feature set makes use of the average fraud probability for all values in the data set. As a first step, for every value in (almost) every column, we calculate the share of fraudulent advertisements with that value compared to all advertisements with that value.

Specifically, for each of these columns we calculate the conditional probability of fraud $P(fraud \mid v)$ for every possible value v. We then use these probabilities as features. Table 1 illustrates this process.

(a)

ID	BRAND	BROWSER	FRAUD
1	Nike	Chrome	1
2	Nike	Chrome	1
3	Reebok	Firefox	1
4	Adidas	Firefox	0
5	Nike	Firefox	0

(b)

ID	BRAND	BROWSER	FRAUD
1	0.67	1	1
2	0.67	1	1
3	1	0.33	1
4	0	0.33	0
5	0.67	0.33	0

Table 1: Initial data (a) and generated features (b).

Let us assume in the learning data, the brand *Nike* appears a total of three times. Two of these advertisements are fraudulent. Therefore, the fraud probability for *Nike* calculates as:

$$P(fraud \mid brand = Nike) =$$

$$\frac{P(fraud \cap brand = Nike)}{P(brand = Nike)} = \frac{0.4}{0.6} \approx 0.67$$

Thus, all values of *Nike* in the *brand* column are replaced with 0.67 in the feature set. Accordingly, Google's *Chrome* as a browser has two advertisements in the table, of which both are fraudulent, yielding a 1.0.

In case of numerical values (e.g., price, sales rank on platform) we group them into buckets and determine the share of fraudulent advertisements for the buckets instead. We do not transform columns with unique values (e.g., advertisement ids) and do not use them as features.

The main advantage of using average fraud probabilities is the fact that the features itself already contain direct information about the variable to be determined, namely fraud. Each value is a representation of the historical likelihood of fraud for that specific characteristic of the advertisement. Another advantage of this feature set is that all features used by the models are numerical. Therefore, we do not have to create more features to deal with categorical values, which would make the model both more complex as well as computationally more expensive.

One disadvantage of the feature set is that for values which have not been seen before there is no average fraud value. One way to address this is to initialize that value with the average fraud probability for all advertisements. Alternatively, these advertisements could be flagged as suspicious so that a customer agent has to decide how to continue. Another disadvantage is the added calculation effort during testing. Either, we have to calculate the averaged values before we can test an advertisement or we have to materialize the averages. With column in-memory databases, calculating the averages on the fly is comparatively fast and thus we decided for this approach. However, if the added load of these aggregations tends to harm system performance, the averages can be materialized and updated from time to time with only negligible accuracy hits.

	Value-Based	Averages
Logistic regression	0.73	0.82
Random Forest	0.80	**0.87**
XGBoost	0.82	**0.89**

Table 2: F_3-scores by feature set and model.

Predicted \ Actual	Legitimate	Fraud
Legitimate	$280,289$	$1,180$
Fraud	$9,162$	$16,343$

Table 3: Confusion matrix for XGBoost with the *averages* feature set.

5. EVALUATION

In this section we will compare and evaluate the six combinations of models (Section 3) and features (Section 4). As mentioned in Section 2, the data set contains correct classifications for 80% of the advertisements. This part of the data set is used for training the models and for cross-validation. For the remaining 20% of unclassified advertisements we predict the classification with each of the models. The results are checked with a dedicated validation service that will judge the predicted classification. It knows the correct classification for every advertisement and will return the number of true/false positives/negatives for our prediction.

5.1 F_β-Score Measures

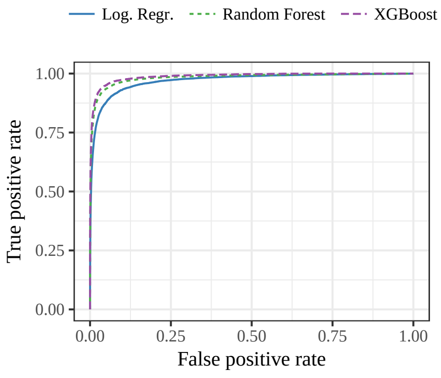
To measure the accuracy of our classifications we calculate different F_β-scores. F_β-scores offer a way to weigh recall higher than precision by choosing $\beta > 1$. As outlined in Section 1, we prioritize maximizing the share of fraudulent advertisements found (i.e., recall). We chose $\beta = 3$ for a significantly higher importance of recall while not disregarding precision altogether. Additionally, we provide the F1-scores for the interested reader for better comparability.

5.2 Results

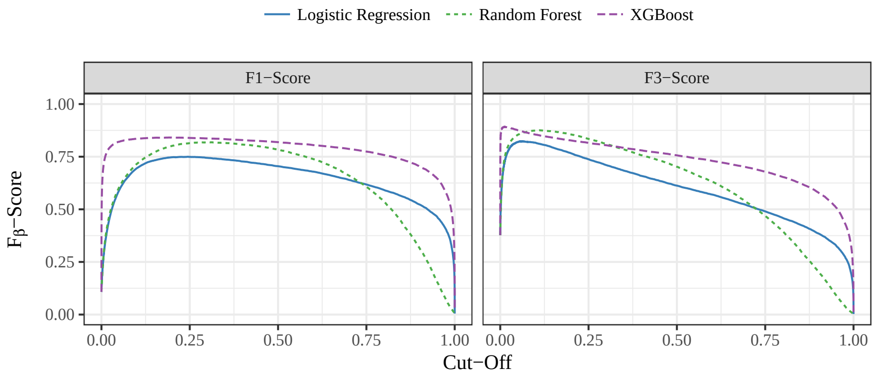
We achieved the bes F_3-score (0.892) using the *averages* feature set in combination with XGBoost (cf. Table 2). An overview of the results is shown in Figure 3 and the confusion matrix is shown in Table 3.

The recall was 0.93, meaning 93% of all fraudulent advertisements have been correctly classified as such. Over 64% of advertisements that customer support agents deal with are actually fraudulent. Our project partner adapts that ratio based on the absolute number of advertisements that have to be checked and the agents' workload. Our approach allows to flexibly change the number of advertisements that have to be checked by adjusting the cutoff value in the model.

The cutoff value is the threshold that all three models use to decide whether a particular advertisement is classified as fraudulent or not. If the predicted value is below the cutoff threshold the advertisement will be classified as legitimate,



(a) True and false positive rates.



(b) F1 and F3-scores.

Figure 3: Comparison of true/false positives rates and F1/F3-scores using the averages feature set.

whereas it will be classified as fraudulent if the value is larger than the cutoff value. Therefore, a higher value means that fewer advertisements are classified as fraudulent, which in turn reduces the workload of the agents. The cutoff value to maximize the F_β-score can be calculated by iteratively narrowing down its range (similarly to gradient descent).

An advantage of decision forests over logistic regression is that they are able to model dependencies between multiple features. As a simple example, consider the price of a tennis racket by a large tennis brand and the exact racket model. Fraudulent actors often try to lower the prices in their advertisements to lure people into buying. Which price is to be considered low strongly depends on the model. Logistic regression is not able to differentiate between a simple entry-level tennis racket and a professional top-class tennis racket, both offered for USD 50. Decision forests, on the other hand, are able to recognize the dependency between them and can decide that this might be a low price for a top-class racket, but not for an entry-level one. We believe this to be the major reason why decision forests performs better for both feature sets.

One of the reasons that the *averages* feature set performs better is that the numerical features are not necessarily monotonic in their influence on fraud. Logistic regression is not directly capable to model these ranges appropriately without further feature engineering. In contrast, Random Forest is (to some extend) able to recognize particularly interesting ranges of numerical features, but there is no guarantee that it recognizes all of these ranges for every feature [5]. When using averages instead, the features all become monotonic with regards to their importance on fraud: a higher value always means a higher probability for fraud.

Looking at recent data mining challenges, the superiority of XGBoost over Random Forest was somewhat expectable. Especially XGBoost's out-of-the box performance allows for fast parameter tuning.

6. CONCLUSION

In this paper, we presented the results of various approaches we implemented in order to detect fraud on a real-world data set of a large e-commerce company. The most promising approach to detect fraud was XGBoost in combination with the *averages feature set* yielding an F_3-score of 0.89. We identified over 93% of all fraudulent advertisements and incorrectly classified 3% of all non-fraudulent advertisements as fraudulent. Besides the Random Forest model, we used logistic regression to improve the understanding and interpretability of the fraud detection for the customer agents.

The proposed models can be easily adjusted to control the number of items that are forwarded to the customer agents. Having that kind of control enables our partner to adjust the agents' workload, e.g., during peaks when customer agents could be overloaded with too items to process.

For our implementation, we used a columnar in-memory database (i.e., SAP HANA) in conjunction with R. Using these technologies, we can iterate on our models fast while being highly flexible at the same time. For large real-world setups, this setup has the advantage of not requiring to add a new IT system to the landscape in order to train our models. The analytical capabilities of column stores allow us selectively train on a subset of the data without expensive ETL pre-processing and to work on the most recent data, which allows fast reactions to changing fraudulent behaviours.

7. REFERENCES

[1] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proc. 22nd ACM SIGKDD*, pages 785–794, 2016.

[2] CyberSource. 2013 online fraud report - online payment fraud trends, merchant practices, and benchmarks. Technical Report 14th Annual Edition, 2013.

[3] P. Große et al. Bridging two worlds with RICE integrating R into the SAP in-memory computing engine. *PVLDB*, 4(12):1307–1317, 2011.

[4] H. Plattner. The impact of columnar in-memory databases on enterprise systems. *PVLDB*, 7(13):1722–1729, 2014.

[5] J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, Mar. 1986.