Identify Theft Detection on e-Banking Account Opening

Roxane Desrousseaux, Gilles Bernard and Jean-Jacques Mariage
LIASD Laboratory, Paris 8 University, France

Keywords: Machine Learning, Fraud Detection, Behavior, Pattern, Banking, Identity Theft.

Abstract: Banks are compelled by financial regulatory authorities to demonstrate whole-hearted commitment to finding ways of preventing suspicious activities. Can AI help monitor user behavior in order to detect fraudulent activity such as identity theft? In this paper, we propose a Machine Learning (ML) based fraud detection framework to capture fraudulent behavior patterns and we experiment on a real-world dataset of a major European bank. We gathered recent state-of-the-art techniques for identifying banking fraud using ML algorithms and tested them on an abnormal behavior detection use case.

1 INTRODUCTION: BACKGROUND AND MOTIVATIONS

Banking activity is changing. The customer relationship, hitherto physical and face-to-face, is becoming digitalized. Customers no longer need to go to agencies and carry out by themselves more and more online procedures. Even account opening is now often completely digital and banks need to insure that the prospect (i.e person not yet customer) is the person that he/she pretends to be and the reasons for its account opening. Banks are particularly concerned about detecting identity theft because this type fraud can lead to huge losses of money, when an identity theft has obtained a personal loan for example. This type of fraud also generates a bad image of the bank and is subject to international sanctions in the case of an account used for financial trafficking.

Banks once had a close relationship with their customers which is now often remote. The AI approach is based on Knowing Your Customer (KYC) by knowing his behavior. For banking, the KYC mainly refers to the requirement of checking a customer's ID when opening a new account. However, faking an ID is so easy that identity theft is one of the fastest-growing banking issues. There is no better way to know your future client and identify risk than by analyzing his/her behavior.

The main focus of this study is to help decide whether a prospect opening a banking account on the bank's website is the person he/she claims to be.

Analysis of interaction between bank and customer on a mass scale is new. The scientific community has put much effort to propose solutions that could benefit to the banking industry. Non-linear techniques as neural networks (NNs) seem doubtless appropriate for detecting elaborate and heterogeneous banking frauds. NNs offer means of dimensional reduction that maintain the characteristics and complexity of the data structure. By preserving the relationships, they make it possible to work on reduced formats while also giving the ability to analyze more complex problems (Zaslavsky and Strizhak, 2006).

2 RELATED WORK

The scientific literature signals as financial anomalies: transactional fraud, money laundering, impersonation or creation of false accounts, account theft, promotional abuse, user behavior, cybersecurity.

The analysis of abnormal activity requires a large amount of past data to learn what characterizes a behavior, based on time series, over several scales of time. The main idea is to model a normal behavior and find the deviant ones. Anomaly detection leads to specific problems identified in the literature: 1) defining what a normal behavior is, because types of behaviors are numerous, 2) difficulty to adapt to a normal change of behavior without causing false alarms. In order to model user's behavior on a website or application, many researches use variables such as the number of visited pages, the keystroke or click speed, the number of failed connections, the IP address and device used, the number of specific events in a cer-

Desrousseaux, R., Bernard, G. and Mariage, J.
Identify Theft Detection on e-Banking Account Opening.
DOI: 10.5220/0008648605560563
In *Proceedings of the 11th International Joint Conference on Computational Intelligence (IJCCI 2019)*, pages 556-563
ISBN: 978-989-758-384-1

tain time-lapse (Shanmugapriya and Ganapathi, 2017; Mazzawi et al., 2017).

Detection of banking anomalies gathers information on the customer, his accounts, his geographical location, time, and information related to his subscribed products. The space of banking transactions is characterized by potentially large dimensions and sparse vectors. The data brings together millions of transactions per day, millions of individual account activities, hundreds of product types. It is difficult to effectively extract relevant attributes; money laundering for example can range from a single transaction to a series of complex transactional activities spanning over several months. In terms of engineering, this represents one of the worst forms of dimensionality disorder with large scale differences and overloading of variables.

Confidentiality restrictions of banking related data have limited the progression of research. There is a real lack of access to financial data for fraud detection research. In order to deal with this problem, novel approaches in fraud detection involve the use of simulators to produce enough financial data which contains both the normal and fraudulent behavior. This technique uses the advantages and benefits of simulation, applied to financial domains to avoid the legal and privacy concerns of real datasets. The main concept behind the approach is to learn how the real customers behave and interact as described by the real data, and through simulation, recreate this behavior by simulated fictitious customers in a way that does not represent a leak of private financial records.

The main research topic in the literature concerning banking fraud is the detection of fraudulent credit card transactions (Ngai et al., 2011; Krishnapriya, 2017; Anandakrishnan et al., 2018). A multitude of techniques have been applied, ranging from simple regression to sophisticated NNs. In a study (Sundarkumar and Ravi, 2015), Logistic Regression performances on credit card churn prediction and insurance fraud detection was compared to an undetermined Decision Tree model, Support Vector Machine, Probabilistic Neural Network, Group Method of Data Handling and Multi-Layer Perceptron. More recently, studies focus on using more elaborated NNs for detecting transactional fraud, like Convolutional Neural Networks (Fu et al., 2016), Autoencoder and Generative Adversarial Network (Chen et al., 2019).

In a study, (Wang et al., 2017) use Recurrent Neural Networks (RNNs) for detecting fraudulent transactions on an E-commerce site which is similar to our experiments because they also used browsing history to characterize users behavior. They ignored all sessions that did not lead to an order, labelled the fraudulent orders using a business's department sample and modeled web sessions as sequences of clicks. They also included information like dwell time (time spent on a particular page), page loading time, browsing time, geolocation, URL... Clicks of the same session are fed into the model in the time order, and a risk score is outputted for the last click (i.e. the check-out action) for each session, indicating how suspicious the session is. Performances of RNNs (varying number of layers and of neurons) are compared to traditional methods including logistic regression, Naive Bayes, SVM and Random Forest. Results are validated using historical data and their research concluded that their framework was able to support the transaction volumes they had, while providing an accuracy never achieved by traditional methods based on aggregate features.

To our knowledge this is the first study on detecting banking identity theft with ML algorithms on a real-world dataset. The remainder of this paper is organized as follows: next section presents a description of the real dataset used for our study. We will then introduce the experimental setup with algorithms tested and assessment metrics used. Finally we will provide some results and analysis.

3 REAL DATA ANALYSIS

In this section we introduce the dataset used for detecting identity theft, the preprocessing steps done such as feature creation, and the preliminary statistical analysis that allowed us to get more insights on the fraudulent patterns.

3.1 Dataset Description

Data is provided by our industrial partner and due to confidentiality policy it can not be provided in public access. The dataset consists in navigational logs, containing information about users activity on the bank's website. The browsing information captured in these logs is spread among ≈ 50 tables (Page table, Click table, Geolocation table, Device Table, etc...). The initial volume of logs available was enormous (Click table for example represents between 50-90 millions of lines per month). Approximatively 11 month of browsing stored in a Hive database on a Hadoop cluster, representing more than 2 million connexions per month, 80 000 – 85 000 sessions for account opening.

The first preprocessing step was to gather, filter and aggregate the navigational logs. Fraud team provided files containing samples of prospect numbers and their corresponding label (account opened or refusal because of a manual identity theft detection by the expert teams).

At some point of the online boarding process, the prospect is assigned a prospect number and has to login to his/her temporary account, to upload his/her ID justification and other supporting documents. Prospects can take several web sessions to complete their onboarding process, they can decide to interrupt their inscription, come back later to terminate and submit their form.

In our study, we filtered out navigational sessions of interest by finding all sessions and activity related to the prospects included in our sample. This was done by finding trace in the navigational logs of a connexion on the temporary account. The join queries between all navigational tables are done on primary database keys such as the web browsing session id, session timestamps and other action/trigger event ids, keys we can use to link information included in the scattered data source. We created the explanatory variables with Spark, we modeled the general behavior of each prospect on the bank's website and onboarding form: number of sessions, sessions duration, navigation speed (between pages, clicks), types and average number of pages visited, device used, screen resolution, geolocation, information provided for account opening. The information given on the online form can be numerical (savings amount, property titles value) or categorical (income bracket, type of working contract...).

Finally we looked at the nature of the features created, transformed and scaled them properly following the general rules. Continuous values are scaled in the range of 0 to 1, distributed normally with mean 0 and standard deviation 1 (this is not fixed but ensuring a correct range of input can help training). For categories without ordinal relationship, we used a One-hot encoding, producing a binary vector with 0 in all dimensions except 1 for the category the data belongs to.

3.2 Preliminary Statistical Analysis

After gathering the data and constructing the features, we did a preliminary analysis of the dataset. This step is essential in every Data Science project and can provide interesting insights on the data. In our study, we plotted variable distribution and correlation matrix using Python visualization libraries and found the following facts:

- Number of pages visited per session and click speed don't evolve over time and stay approximatively the same for each user session (thus behavior doesn't change so much).
- A bigger number of pages visited does not always mean longer sessions.
- Fraudsters tend to be faster in their navigation and make shorter sessions.
- All fraudsters declare the same kind of profile revenue.

4 EXPERIMENTS

This section is divided into three subsections. In the first subsection we will describe the ML models used for identity theft detection and preprocessing stage. In the second, the assessment metrics used. The last section presents our results.

The questions we wish to address are the following:

- Which algorithms are going to work best; especially do multi-layer NNs have better results than standard classifiers?
- Will oversampling lead to better performances?
- More generally how will preprocessing affect detection results? Will the results be better with multi-layer Autoencoder (AE) than with Principal Component Analysis (PCA)?

4.1 Machine Learning Models

We present here the ML classifiers used in this study. The low proportion of abnormal observations (outliers) is an issue in anomaly detection and unbalanced learning. Anomalies are scarce and therefore little present, and standard ML algorithms are not able to learn classes of data that are not enough represented in the learning dataset. Therefore, specific algorithms have been devised for this task.

We studied three types of ML models: basic models, outlier dedicated algorithms, both from Scikit-Learn library, and sophisticated NNs built with Keras library. In the preprocessing stage, for dimension reduction, we compared PCA (from Scikit-Learn) with Multi Layer AE (built with Keras) before applying these models.

4.1.1 ML Classifiers

The following basic models were tested: k-Nearest Neighbors (k-NN), linear SVM (SVM), CART Decision Tree (DT), Random Forest (RF), Multi Layer Perceptron, in reality a Bilayer Perceptron with just one hidden layer (BLP), Adaboost, Naives Bayes (NB) and Quadratic Discriminant Analysis (QDA).

As specific outlier detection models, we have chosen three algorithms that are gaining popularity in such tasks: Local Outlier Factor (LOF), Isolation Forest (IForest), One-class SVM (OCSVM). A multitude of variants of these algorithms have been proposed.

As a more sophisticated model, we built a Multi Layer Perceptron (MLP) with three hidden layers having rectified linear unit (ReLu) as activation function, a sigmoid function for the last layer, and a binary cross entropy loss function because we are in a binary classification problem. The number of neurons of each hidden layer is set to $\approx 80\%$ of the number of input features, as this was the value that gave the best preprocessing performances.

We give below a brief description focusing on the less well known algorithms.

IForest (Liu et al., 2008). This algorithm is based on decision trees. The method takes advantage of two quantitative properties of the outliers: *(a)* they have fewest instances and *(b)* their attribute values greatly differ from those of the more frequent classes. Thus, outliers are more likely to be isolated early in the tree and hence have shorter paths to the root.

LOF (Breunig et al., 2000). The LOF algorithm focuses on analyzing local densities of the data space. It identifies regions of similar density and consider as outliers data points from lower density regions. It is based on k-NN; the density is estimated from the distances between neighbours, computing the local reachability of a point, given by the inverse average reachability distance from its k neighbours (equation 1).

$$\mathrm{lrd}(p) = 1/\frac{\sum_{o\in \mathrm{KNN}(p)} \mathrm{rdist}_{\,k}(p \leftarrow o)}{|\mathrm{KNN}(p)|} \qquad (1)$$

where KNN(p) is the set of p's nearest neighbours. The (asymmetric) reachability distance *rdist* from o to p is defined by equation 2.

$$rdist_k(p \leftarrow o) = \max\{k - \mathrm{dist}(o),\, d(p,o)\} \qquad (2)$$

The final LOF score then compares the locally relevant lrd values:

$$\mathrm{LOF}_k(p) = \frac{1}{|\mathrm{KNN}(p)|} \sum_{o\in \mathrm{KNN}(p)} \frac{\mathrm{lrd}_k(o)}{\mathrm{lrd}_k(p)} \qquad (3)$$

OCSVM (Rätsch et al., 2000). In 2000, the popular Support Vector Machines algorithm was adapted to focus on outliers in the following way. The idea is to classify all non outlier data in one group, compute the probability distribution of this group. A discriminant function compares the probability of membership of a given point, comparing it with a parameter $\rho \in [0,1]$; this parameter is estimated by solving a quadratic function. Let $x_1, x_2, \ldots, x_l$ be training examples belonging to one class X, where X is a compact subset of R^N. $\Phi : X \longrightarrow H$ is a kernel map which transforms the training examples to another space. Then, to separate the data set from the origin, one needs to solve the following quadratic programming problem:

$$\min \frac{1}{2}\|w\|^2 + \frac{1}{\nu l}\sum_{i=1}^{l} \xi_i - \rho \qquad (4)$$

subject to

$$(w \cdot \Phi(x_i)) \geq \rho - \xi_i \quad i = 1,2,\ldots,l \quad \xi_i \geq 0 \qquad (5)$$

If w and ρ solve this problem, then the decision function

$$f(x) = \mathrm{sign}((w \cdot \Phi(x)) - \rho) \qquad (6)$$

will be positive for most examples contained in the training set.

4.1.2 AE Dimensional Reduction

We provide below a description of the multi layer sparse Autoencoder NN that we used for preprocessing, against PCA.

AE. An AE is an autosupervised NN that applies backpropagation and whose objective is to learn to reproduce input vectors $\{x(1), x(2), \ldots x(m)\}$ as outputs $\{\hat{x}(1), \hat{x}(2), \ldots \hat{x}(m)\}$.

In other words, it learns an approximation of the identity function. By forcing constraints on the network, such as limiting the number of hidden units, interesting structure about the data can be discovered. Anomaly detection using dimensionality reduction is based on the assumption that data has variables correlated with each other and can be embedded into a lower dimensional subspace in which normal samples and anomalous samples appear significantly different.

Consider an AE with several hidden layers; activation of unit i in layer l is given by equation 7.

$$a_i^{(l)} = f\left(\sum_{j=1}^{n} W_{ij}^{(l-1)} a_j^{(l-1)} + b_i^{(l)}\right) \qquad (7)$$

where W and b parameters are respectively weight and bias.

Given a training set of m examples, the overall cost function to minimize is shown in equation 8.

$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\|x(i)-\hat{x}(i)\|^2\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_l+1}\left(W_{ji}^{(l)}\right)^2 \quad (8)$$

The first term in the definition of $J(W,b)$ in equation 8 is an average sum-of-squares error term. The second term is a regularization term, a weight decay term aiming to prevent overfitting. The weight decay parameter λ controls the relative importance of each of those terms. Four main extensions of AEs have been developed: Sparse (Deng et al., 2013), Denoising (Vincent et al., 2008), Contractive (Rifai et al., 2011) and Variational (Goodfellow et al., 2014).

The sparsity constraint we chose in our study forces the neurons to be inactive most of the time, but interesting structure in the data can still be discovered. To enforce the constraint $\hat{\rho}_j = \rho$, where ρ is a sparsity parameter (usually close to zero), we add an extra penalty term to our optimization objective that penalizes $\hat{\rho}_j$ significantly deviating from ρ. Many choices of the penalty term are possible but common choice is based on the concept of Kullback–Leibler divergence (see equation 9).

$$J^{Sparse}(W,b) = J(W,b) + \beta\sum_{j=1}^{s_2} KL(\rho\|\hat{\rho}_j) \quad (9)$$

where $J(W,b)$ is defined in equation 8, and β controls the weight of the penalty term.

In Keras library, the Dense function constructs a fully connected NN layer. If we enforce a representation constrained only by the size of the hidden layer, the hidden layer is learning an approximation of PCA. To add a sparsity constraint on the activity of the hidden representations, so fewer units fire at a given time, we add in Keras an activity regularizer to our Dense layer. When we add more than one hidden layer to an AE, it helps to reduce a high dimensional data to a smaller code representing important features. The network used in our study is a 4-layered sparse AE where each layer is a sparse encoding layer reducing the number of features by 25% (we observed better performances with a smooth decay of neuron number in each layer). Each hidden layer becomes a more compact representation than the last hidden layer.

4.2 Assessment Metrics

The most frequently used metrics are accuracy and error rate. Considering a basic two-class classification problem, then a representation of classification performance can be formulated by a confusion matrix (contingency table), as illustrated in table 1. In this paper, the minority (fraudulent) class is used as the positive class and the majority (non fraud) class as the negative class. With this convention, accuracy and error rate are respectively defined as in 10 and 11.

Table 1: Confusion matrix. *P*, *N* stand for predicted positive and negative classes, *p*, *n* for ground-truth positive and negative label.

		Ground truth	
		p	**n**
Program prediction	**P**	TP (True Positive)	FP (False Positive)
	N	FN (False Negative)	TN (True Negative)
Count:		P_C	N_C

$$Accuracy = \frac{TP+TN}{P_c+N_c} \quad (10)$$

$$ErrorRate = 1 - Accuracy \quad (11)$$

It can be difficult to mesure performances of a classifier on unbalanced data. In the case of highly skewed datasets, ROC curve may provide an overly optimistic view of an algorithm's performance. Under such situations there are three standard assessment metrics which are useful to unbalanced learning, defined as follows:

$$Precision = \frac{TP}{TP+FP} \quad (12)$$

$$Recall = \frac{TP}{TP+FN} \quad (13)$$

$$F1\text{-}Score = \frac{(1+\beta)^2 \cdot precision \cdot recall}{\beta^2 \cdot precision + recall} \quad (14)$$

where β is the relative importance of precision and recall, usually $\beta = 1$.

For our study, we focused on precision, recall (sensitivity), F1-score and specificity (false alarm rate).

4.3 Evaluation Results

All the hyper-parameters of the models mentioned above have been tuned using GridSearch() class. Performances described in this subsection are calculated on a k-stratified cross validation (Mosteller and Tukey, 1968).

In the preprocessing stage, we tested dimension reduction using PCA or Multi Layer AE, removal of categorical features, removal of features with low variance, and oversampling.

To compute confusion matrices and evaluate performance metrics of the classifiers, we divided our dataset (2200 non fraud / 396 frauds) into a learning dataset (70%) and a testing dataset (30%). Test set consists of 660 observations of class 0 (non fraud) et 119 observations of class 1 (fraud). Anomalies are equally distributed in both learning and test datasets. Each observation is described by more than 250 features (including categorical features with One-hot representation). Tables 2 to 5 show performances obtained by the candidate models on different preprocessings of the dataset.

Performances are worse on the PCA reduced data for every classifier and thus demonstrate it is unsuitable to our problem. MLP achieved best F1-score (0.83) on the PCA preprocessing (where the simpler BLP only performed 0.67). Table 3 shows classifier performances after having applied dimensional reduction using Multi Layer sparse AE. Performances are improved for all classifiers. Best performance was achieved by OCSVM with a F1-score of 0.90. Tree based models like AdaBoost, DT and RF performed well too. On the 119 fraudulent cases in test dataset, RF correctly classified 93 of them (with 2 false alarms) and DT detected 101 of them (with 8 false alarms). The MLP also achieved a good F1-score of 0.84 but with very little improvement compared to performances on PCA preprocessing.

We had more than 20 categorical features in our initial dataset. When embedded using the One-hot representation, this created a sparse data matrix with a lot of low variance features. We decided to test candidate models with removal of these attributes and performances picked up as shown in Table 4. OCSVM with F1-score of 0.93 performed best, followed by SVM with a F1-score of 0.91. Both of the SVM-based models were able to detect all fraudulent cases. We also observed that Naive Bayes made its best performances with this preprocessing with a F1-score of 0.88.

Table 5 shows performances after oversampling. In our case, we created synthetical copies of the fraudulent cases until we reach a balanced distribution (same number of fraud/non fraud observations). Test dataset consists of 660 non fraud cases and 660 fraud cases. As expected, classification performances picked up and all models achieved their best scores (with a minimum F1-score of 0.68). Random Forest got the best F1-score of 0.99 (7 false alerts and 5 frauds missed). SVM and OCSVM were able to detect all fraudulent cases and raised less than 30 false alarms.

Overall, IForest and LOF did not perform well, as they mislabeled the fraudulent cases as normal and caused too many false alarms. They were capable of detecting at least 75% of the frauds on the oversampled dataset but they were far from meeting Random Forest or SVM based models performances.

Table 2: Performances with PCA dimensional reduction.

Algorithm	Accuracy	Recall	Precision	F1 Score
k-NN	0.8929	0.4797	0.7252	0.5775
SVM	0.9402	0.6085	1.0000	0.7566
DT	0.6737	0.7190	0.8735	0.7888
RF	0.9479	0.6994	0.9453	0.8040
BLP	0.9183	0.5656	0.8484	0.6787
AdaBoost	0.9437	0.7525	0.8612	0.8032
NB	0.2858	0.9570	0.1710	0.2901
QDA	0.8536	0.0404	1.0000	0.0776
IForest	0.4568	0.4604	0.8195	0.5896
OCSVM	0.8447	0.8908	0.4953	0.6366
LOF	0.3880	0.3250	0.8730	0.4736
MLP	0.9538	0.7601	0.9233	0.8338

Table 3: Performances with AE dimensional reduction.

Algorithm	Accuracy	Recall	Precision	F1 Score
k-NN	0.9318	0.6464	0.8737	0.7431
SVM	0.9564	1.0000	0.7778	0.8750
DT	0.9666	0.8487	0.9266	0.8860
RF	0.9641	0.7815	0.9789	0.8692
BLP	0.9332	0.7143	0.8252	0.7297
AdaBoost	0.9537	0.7676	0.9156	0.8351
NB	0.6470	0.6723	0.2532	0.3678
QDA	0.3979	0.9412	0.1951	0.3232
IForest	0.9148	0.5378	0.8486	0.6584
OCSVM	0.9730	0.8232	1.0000	0.9030
LOF	0.9248	0.5909	0.87640	0.7058
MLP	0.7330	0.8359	0.8470	0.8414

The amount of time needed for learning and the time needed for labeling a new observation are two important criteria that we have not yet taken into account in this study. Our raw logs needed a heavy initial cleaning step, and feature creation was the most CPU consuming. Once the final dataset prepared and the large learning data matrix created we focused on reducing the dimensionality disorder by selecting the best features and by applying different reduction techniques. The preliminary analysis mentioned (subsection 3.2) revealed that several categorical features were useless. With feature selection and application of dimensional reduction we were able to train all models in acceptable time although BLP/MLP were very long to train as their number of hidden neurons was quite large.

Table 4: With low variance and category features removal.

Algorithm	Accuracy	Recall	Precision	F1 Score
k-NN	0.7253	0.8159	0.8535	0.8343
SVM	0.8509	1.0000	0.8504	0.9191
DT	0.9476	0.7601	0.8801	0.8157
RF	0.7723	0.8563	0.8726	0.8644
BLP	0.7369	0.7977	0.8805	0.8371
AdaBoost	0.7272	0.7850	0.8022	0.8298
NB	0.8027	0.9163	0.8600	0.8873
QDA	0.2114	0.0781	0.9005	0.1438
IForest	0.6675	0.7668	0.8281	0.7963
OCSVM	0.9451	1.0000	0.8722	0.9318
LOF	0.9537	0.7601	0.9233	0.8337
MLP	0.8124	0.9527	0.8455	0.8959

Table 5: Performances with oversampling.

Algorithm	Accuracy	Recall	Precision	F1 Score
k-NN	0.8097	0.9009	0.8777	0.8891
SVM	0.9713	1.0000	0.9442	0.9713
DT	0.9689	0.9682	0.9697	0.9689
RF	0.9909	0.9924	0.9894	0.9909
BLP	0.9523	0.9621	0.9435	0.9527
AdaBoost	0.9826	0.9712	0.9938	0.9824
NB	0.6750	0.7076	0.6643	0.6853
QDA	0.5477	0.9758	0.5477	0.6833
IForest	0.7222	0.8209	0.8466	0.8336
OCSVM	0.9886	1.0000	0.9778	0.9888
LOF	0.9333	0.6590	0.8729	0.7510
MLP	0.9667	0.9803	0.9543	0.9671

From a business validation point of view, we selected the best models (oversampled Random Forest and OCSVM) and tested classification to all historical onboarding accounts (only non fraud). We observed that our framework labeled some of them as fraudulent and reported them to the dedicated team. On these alerts, 25% were totally false alerts. In the remaining 75%, approximatively half of them were actually fraud cases mislabeled (often discovered later and not updated in the tracking file). The other half were accounts actually under observation because of fraud suspicion.

5 CONCLUSION

In this paper we proposed a ML-based framework for detecting identity theft on e-banking account opening. We aggregated massive amounts of data, combined several types of features, including behavioral browsing characteristics. We applied different popular preprocessing techniques such as oversampling and dimensional reduction with PCA. Most recent scientific work now supports the use of auto- or unsupervised NNs to detect anomalies and reduce dimensions while preserving information. In particular, studies have shown that AEs are able to detect much more subtle anomalies than conventional linear techniques such as PCA (Sakurada and Yairi, 2015). We benchmarked 12 classifiers with different preprocessings and concluded that using multi layer sparse AE did improve our classification performances compared to PCA. We also observed a clear improvement of performances just by removing the embedded categorical features, outperforming scores on PCA and AE preprocessing. This clearly demonstrates the importance of the training dataset used and the feature engineering process. However, applying a dimensional reduction was imperative in our case as each observation was described by a large number of features and without dimensional reduction step, classifiers did require a lot more training time. Like other related banking fraud research (Quah and Sriganesh, 2008; Ahmad et al., 2017; Ryman-Tubb and D'Avila Garcez, 2010; Kamesh et al., 2019) our work will now focus on adapting our framework for real-time streaming data and on the extraction of comprehensible rules for the business team.

Integrating data-level solutions with classifiers, resulting in robust and efficient learners, is very popular in recent years for unbalanced problems (Abdi and Hashemi, 2016). Some works propose the hybridization of sampling techniques and the use of a cost-sensitive learning or of kernel-based methods. We applied the same techniques in our study and performed expected results: tree-based models (DT, RF, Adaboost) and SVM based models achieved best scores, especially with oversampling.

In such cases where the unbalance ratio in the data is so important, the anomaly is poorly represented, lacks a clear structure and potentially strong variance is induced. Direct application of relationship based preprocessing methods such as oversampling can actually damage the classification performance and lead to over-fitting. In our case, business validation on historical data indicates that the selected RF and OCSVM models are able to fit on new observations.

Our proposed framework is an interesting decision making tool for detecting identity theft on the digital account opening. We discovered interesting fraud patterns and overall, we observed the same results as other related works: behavior of the fraudsters is statistically different from legitimate users and real users browse following a certain pattern. In contrast, fraudsters behave more uniformly, generally going directly to the purpose of their theft, browsing quickly. Unfortunately these types of deeper analysis are often restrained from publication and research studies like ours can rarely provide detailed experimental setup or disclose chosen solution.

REFERENCES

Abdi, L. and Hashemi, S. (2016). To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE Transactions on Knowledge & Data Engineering*.

Ahmad, S., Lavin, A., Purdy, S., and Agha, Z. (2017). Unsupervised real-time anomaly detection for streaming data. *Neurocomputing*, 262:134–147.

Anandakrishnan, A., Kumar, S., Statnikov, A., Faruquie, T., and Xu, D. (2018). Anomaly Detection in Finance: Editors' Introduction. In *KDD 2017 Workshop on Anomaly Detection in Finance*, pages 1–7.

Breunig, M. M., Kriegel, H.-P., Ng, R. T., and Sander, J. (2000). LOF: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104.

Chen, J., Shen, Y., and Ali, R. (2019). Credit Card Fraud Detection Using Sparse Autoencoder and Generative Adversarial Network. In *2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference, IEMCON 2018*, pages 1054–1059. IEEE.

Deng, J., Zhang, Z., Marchi, E., and Schuller, B. (2013). Sparse autoencoder-based feature transfer learning for speech emotion recognition. In *2013 Humaine Association Conference on Affective Computing and Intelligent Interaction*, pages 511–516. IEEE.

Fu, K., Cheng, D., Tu, Y., and Zhang, L. (2016). Credit card fraud detection using convolutional neural networks. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9949 LNCS, pages 483–490. Springer, Cham.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Kamesh, V., Karthick, M., Kavin, K., Velusamy, M., and Vidhya, R. (2019). Real-time fraud anomaly detection in e-banking using data mining algorithm. *South Asian Journal of Engineering and Technology*, 8(S 1):144–148.

Krishnapriya, D. (2017). Identification of Money Laundering based on Financial Action Task Force Using Transaction Flow Analysis System. *Bonfring International Journal of Industrial Engineering and Management Science*, 7(1):01–04.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422.

Mazzawi, H., Dalal, G., Rozenblatz, D., Ein-Dorx, L., Niniox, M., and Lavi, O. (2017). Anomaly detection in large databases using behavioral patterning. In *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*, pages 1140–1149.

Mosteller, F. and Tukey, J. W. (1968). Data analysis, including statistics. In Lindzey, G. and Aronson, E., editors, *Handbook of Social Psychology, Vol. 2*. Addison-Wesley.

Ngai, E. W. T., Hu, Y., Wong, Y. H., Chen, Y., and Sun, X. (2011). The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision support systems*, 50(3):559–569.

Quah, J. T. S. and Sriganesh, M. (2008). Real-time credit card fraud detection using computational intelligence. *Expert systems with applications*, 35(4):1721–1732.

Rätsch, G., Schölkopf, B., Mika, S., and Müller, K.-R. (2000). *SVM and boosting: One class*. GMD-Forschungszentrum Informationstechnik.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 833–840. Omnipress.

Ryman-Tubb, N. F. and D'Avila Garcez, A. (2010). SOAR - Sparse Oracle-based Adaptive Rule extraction: Knowledge extraction from large-scale datasets to detect credit card fraud. In *Proceedings of the International Joint Conference on Neural Networks*, pages 1–9. IEEE.

Sakurada, M. and Yairi, T. (2015). Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis - MLSDA'14*, pages 4–11, New York, New York, USA. ACM Press.

Shanmugapriya, D. and Ganapathi, P. (2017). A Wrapper-Based Classification Approach for Personal Identification through Keystroke Dynamics Using Soft Computing Techniques. In *Identity Theft: Breakthroughs in Research and Practice*, pages 267–290. IGI Global.

Sundarkumar, G. G. and Ravi, V. (2015). A novel hybrid undersampling method for mining unbalanced datasets in banking and insurance. *Engineering Applications of Artificial Intelligence*, 37:368–377.

Vincent, P., Larochelle, H., Bengio, Y., and Manzagol, P.-A. (2008). Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM.

Wang, S., Liu, C., Gao, X., Qu, H., and Xu, W. (2017). Session-Based Fraud Detection in Online E-Commerce Transactions Using Recurrent Neural Networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 241–252.

Zaslavsky, V. and Strizhak, A. (2006). Credit card fraud detection using self-organizing maps. *Information and Security*, 18:48.