A Machine Learning-based System for Financial Fraud Detection

*

João Paulo A. Andrade[1], Leonardo S. Paulucio[1], Thiago M. Paixão[1,2], Rodrigo F. Berriel[1], Teresa Cristina Janes Carneiro[1], Raphael V. Carneiro[1], Alberto F. De Souza[1], Claudine Badue[1], Thiago Oliveira-Santos[1]

[1]Universidade Federal do Espírito Santo (UFES), Brazil

[2]Instituto Federal do Espírito Santo (IFES), Brazil

Email: p.paulo.andrade@gmail.com

Abstract. *Companies created for money-laundering or as a means for tax-evasion are harmful to the country's economy and society. This problem is usually tackled by governmental agencies by having officials to pore over companies' financial data and to single out those that exhibit fraudulent behavior. Such work tends to be slow-paced and tedious. This paper proposes a machine learning-based system capable of classifying whether a company is likely to be involved in fraud or not. Based on financial and tax data from various companies, four different classifiers – k-Nearest Neighbors, Random Forest, Support Vector Machine (SVM), and a Neural Network – were trained and then used to indicate fraud. The best-performing model achieved a macro-averaged F1-score of 92.98% with the Random Forest.*

1. Introduction

According to the Oxford Dictionary [Simpson 2006], fraud is defined as "wrongful or criminal deception intended to result in financial or personal gain". There are several types of frauds such as in telecommunications, insurance frauds, money laundering, health care, tax evasion, credit cards, etc. All of them cause huge financial losses every year.

In recent decades, the number of scams involving fraud has grown considerably. A survey on fraud and economic crime conducted by PricewaterhouseCoopers (PwC) [Lavion et al. 2018] in 2018, with 7,200 companies from 123 different countries, showed that 49% of the companies were victims of some kind of fraud between 2016 and 2018, which is an increase when compared to 36% of the last survey in 2016. Despite this growth, the number of frauds represents a small percentage of all transactions, resulting in many more legitimate cases than fraudulent ones [Maes et al. 2002].

Fraud detection has always been an important task for governments and private enterprises. This task is traditionally accomplished through costly internal audits [Ngai et al. 2011] that require the analysis of lots of data and law details, which makes the auditor's work slow and error-prone. Therefore, there is a large demand for automatic

*This study was financed in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001; Conselho Nacional de Desenvolvimento Científico e Tecnológico – Brasil (CNPq) – grants 311120/2016-4 and 311504/2017-5; and Fundação de Amparo à Pesquisa do Espírito Santo - Brazil (FAPES) – grant 84304057.

solutions to increase the efficiency of the detection process, and, in this context, machine learning algorithms play a central role.

Several works investigate the use of machine learning for general fraud detection [Mittal and Tyagi 2019, SADGALI et al. 2019, Yao et al. 2018, Awoyemi et al. 2017, Thennakoon et al. 2019]. As noticeable in the literature, the most common techniques to solve this problem are: Neural Networks [Yao et al. 2018, Maes et al. 2002], Bayesian Networks [Maes et al. 2002], Logistic Regression [Nadim et al. 2019], Support Vector Machine [Pai et al. 2011, SADGALI et al. 2019], Random Forest [Liu et al. 2015, Kumar et al. 2019], Naive Bayes [Awoyemi et al. 2017] and k-Nearest Neighbors [Thennakoon et al. 2019, Awoyemi et al. 2017, SADGALI et al. 2019].

In addition to the aforementioned techniques, Deep Neural Networks have gained attention in the fraud detection problem [Fu et al. 2016, Abakarim et al. 2018, Najadat et al. 2020, Paula et al. 2016] due to the large success in other applications. In this context, it can be highlighted the use of (unsupervised) autoencoders for the detection of suspicious operations of exporting companies [Paula et al. 2016] and for real-time detection of fraudulent credit card transactions [Abakarim et al. 2018]. In the supervised paradigm, Convolutional Neural Networks (CNNs) have been used, for example, to detect frauds in credit card transactions [Fu et al. 2016] and in telecommunications [Chouiekh and Haj 2018]. Despite the promising results, the use of deep networks requires large amounts of manually labeled data. This is an issue in the context of the current application (tax evasion) since the acquisition of labeled data is time-consuming and demands the expertise of auditors [Wu et al. 2019].

To enable automatic detection of tax evasion in companies, this work proposes a classification-based system that processes the past transactions of a company and predicts whether it is fraudulent or regular. The system aims at making the job of auditors more efficient when looking for fraudulent companies. This work was evaluated on real-world transaction records provided by the Treasury Office of the state of Espírito Santo (SEFAZ-ES), Brazil, *i.e.*, a database comprising more than 2,000 companies and more than 300,000 transactions. In the proposed system, training and inference (prediction) rely on a pre-processing procedure specifically designed for the provided database. Since inference is conducted at the transaction level, a fusion function is leveraged to integrate transaction-level information and to decide on the regularity status at company level.

Different classifiers, that are commonly used in approaches in the literature, were investigated: Neural Networks, Support Vector Machine, Random Forest, and k-Nearest Neighbors. To provide a fair experimental assessment, we used a k-fold cross-validation protocol. Results show that Random Forest yielded the best performance with a macro-averaged F1-score of 92.98%. Considering a semi-automatic approach where the system is used as a preliminary filter for the auditors, the results show that the manual workload can be reduced in approximately 81% with a loss of nearly 15% of the fraudulent cases.

It is worth mentioning that tax evasion databases are scarce in the literature, which makes it difficult to evaluate the generalization of the different fraud detection systems. To benefit the research community, the SEFAZ-ES database will be released[1] (preserving companies anonymity) as an additional contribution of this work, together with the trained

[1]Upon acceptance of this paper.

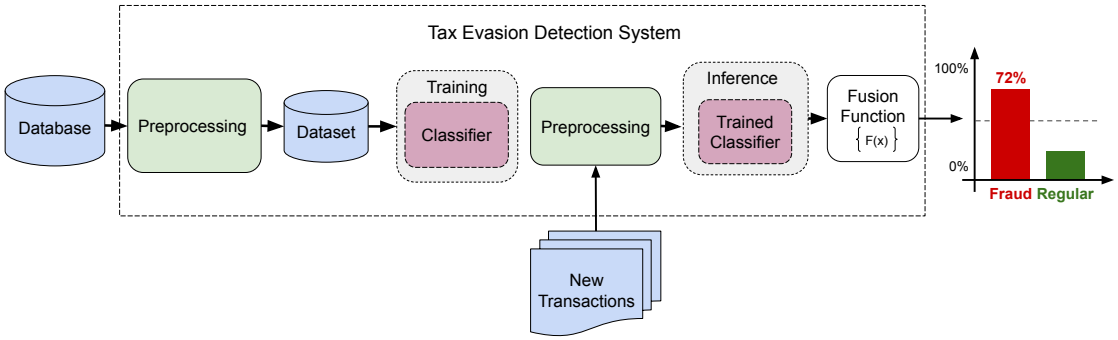

Figure 1. Overview of the proposed system. The training stage includes the preprocessing of the original database (producing a dataset with numerical features) and the training itself. At run-time, the same preprocessing is applied to new transactions and then submitted to the classifier, which assigns a confidence score indicating the fraud probability. The predictions associated with a company are combined by the fusion function, determining the regularity status of the respective company (regular or fraud).

models and the source code of our system.

The rest of this paper is organized as follows. The next section presents the proposed tax-evasion detection system. Section 3 describes the experimental methodology, and the obtained results are presented and discussed in Section 4. Finally, conclusion and future works are drawn in Section 5.

2. Tax Evasion Detection System

The conceptual workflow of our system is illustrated in Figure 1. To avoid terminology misunderstanding, the term *database* is used throughout the text to denote the original unmodified data provided by SEFAZ-ES, whereas *dataset* refers to the data effectively used in the *training* of a model. Both training and inference share the same data preprocessing procedure, however, while the training leverages a fixed database, the *inference* stage aims at processing new transactions of a company to decide its regularity status. Since inference is conducted at transaction level, a fusion function is responsible to define the final status of the company (fraud or regular). The aforementioned elements of the system are described in this section.

2.1. Data Description

The database has information from 248,867 companies comprehending a single table that was automatically generated by processing data from invoices. A row, also referred to as an *instance* or *entry*, from the tabular database references a company by a masked identifier (to preserve the company's anonymity) and its content summarizes the invoices issued in a particular month in a given year: fiscal indicators, quantities (*e.g.*, how many transactions were made on a referenced date), and amounts from revenue and expenditure detailed in many different categories. A single entry or as many as hundreds of entries can be associated to a given company: this number is not fixed for each company and some have far more than others. Companies are labeled as either "Regular" or "Fraud", and all

rows (or instances) associated with a company have consequently the same label. There are no duplicate entries in the dataset.

The dataset, on its turn, was obtained by processing the database (as detailed in the following Subsection), resulting in a reduced set of categorical and numerical features. The categorical features comprise codes and flags related to fiscal activity. The codes identify information from tax payment, tax incentives, business classification (autonomous, micro-entrepreneur, etc.), and from income tax returns. The different flags indicate whether a company is legally authorized to print fiscal documents and generate customer invoices, and whether it has a legal person as co-owner. The numerical data comes from state-specific tax returns and from standard and electronic invoices. These values are already tax-deducted amounts from the company's income, income from transfers, net income, expenses, and other various values from tax-related financial operations regarding income and expenses. All amounts represent a monthly summary of a company's financial activities. There are also numerous quantities that indicate, for instance, how long (in months) a company has omitted information on economical and fiscal activity, the number of blank income tax returns filled, how many co-owners a company has, the total of invoices issued on the referenced date and some additional quantities on economical and tax-related matter.

2.2. Data Preprocessing

Originally, the database contains 92 features, including the anonymized ID of the companies and the label. From these 92 features, 45 numerical and 9 categorical features (which go through one-hot encoding and become 62 in total) are selected to constitute the dataset, as well as the anonymized ID and the label. Many features of the database were left out of the dataset because they deal mainly with textual data and dates that aid human auditors in their work but are unfit, irrelevant (e.g., features having all entries with the same value), or improper for a machine learning model. The remaining features in the dataset required some adjustments, such as standardization for the numerical features and one-hot encoding for the categorical ones, resulting in 109 features on the final dataset.

Some features may have missing values, but handling them is not always straightforward. For instance, when a feature has just a few null values (*e.g.*, 5% or less of all entries from a label), it may be preferable to eliminate all exceeding entries from the other features than fully discarding it. In the proposed system, features with more than 1% of null entries were discarded. For the other features with missing values, the excess from all other features was removed to reduce the loss of data when handling missing values.

Features with textual data are either discarded or one-hot encoded. The textual features that are removed are those that provide written description of categorical features, dates, codes that are given *after* a company is labeled (introducing bias) and features that have too many categories (*i.e.*, more than 10) to be one-hot encoded. All the remaining categorical features (flags and codes) undergo the process of one-hot encoding. As mentioned at the beginning of this Subsection, there are 9 categorical features in the dataset, but after one-hot encoding them, they amount to 62 because every value from each categorical feature now becomes itself a feature. The last step in the preprocessing phase is the standardization of the numerical features. Finally, the data is ready to be used by the model for classification.

2.3. System Input and Output

An entry from a company serves as an input for the system. The vast majority of the companies in the dataset is linked to more than one entry, and, as mentioned in Section 2.1, all entries linked to the same company share the same label (*e.g.*, "Fraud"). The model is fed entry by entry and classifies them independently, which means that the system can output different labels for different entries from the same company. After that, the outputs of a company are fused and the system classifies the company accordingly. The final output of the system is the company's classification together with the confidence (in percentage) with which the system classified the company as "Fraud".

This confidence value is important because human auditors will further investigate and validate all classifications. Therefore, they can use it as an indicator of the companies they should focus on, making their job of finding frauds more efficient. In this context, our fraud detection system must present as fewer false positives (regular entities predicted as fraud) as possible, translating to more efficiency for the human workers.

2.4. Machine Learning Models

As mentioned in the introduction, four classical classification techniques were used to assess the effectiveness of the proposed system: Random Forest (RF) [Ho 1995], k-Nearest Neighbors (KNN) [Cover and Hart 1967] , Support Vector Machine (SVM) [Cortes and Vapnik 1995] and a fully-connected Neural Network (NN) [McCulloch and Pitts 1988]. A summary of each of them is presented next.

KNN: it predicts an example based on plurality vote of its k closest neighbors in the space of features. As the name implies, the example, or entry in our case, will be assigned to the most common class among its k-nearest neighbors. This method can be problematic when classes tend to overlap in the feature space of a dataset.

RF: it comprises an ensemble of Decision Trees that work by constructing and training each of them with a random subset of all available features and outputs the mode of the classes of the individual trees in the ensemble. This results in a low variance, which is favorable when training a model.

Support Vector Machine: it constructs a linear decision surface in a high dimensional feature space and maps examples of the different categories (or labels, in our case) in the linear space in a way that keeps them far apart as possible. Its predictions are made by mapping new samples into the linear space and classifying them based on where they land on the decision boundary.

Neural Network: it is inspired by the human neural system, comprises multiple units (artificial neurons) organized in layers that process the received signal and forward them to further layers. These signals can be quantified by real numbers and act as inputs to the units. A unit, on its turn, processes the input signals by combining them with weights (weighted sum) and applying an activation function (usually non-linear) to the resulting

sum. The final result (in our case, a classification) is computed in the last layer of the network.

2.5. Training and Inference

After data preprocessing, the generated dataset can be employed in the training of the machine learning classifiers. In the inference step, the classifier receives an entry containing a summary of monthly transactions of a company and predicts whether the entry is fraudulent or not (i.e., regular).

2.6. Fusion function

As previously mentioned, the classifier prediction is based on a single entry, which means that a company with multiple entries is subject to conflicting classifications. In that regard, together with the class, the classifier also outputs a confidence score (in percentage) indicating the probability of such a company entry being fraudulent. Therefore, an entry is predicted as "Regular" if its confidence score is less than a threshold (*e.g.*, empirically set to 50%). To produce a single output, the system aggregates the confidence scores of each prediction (using a simple arithmetic mean) and maps the resulting value back into a class, *i.e.*, "Fraud" if such value is greater or equal to the given threshold, or "Regular", otherwise.

3. Experimental Methodology

This section describes the dataset, the performance metrics, the conducted experiments, and the computational platform where these experiments were performed. To enable further comparison, code and data are intended to be made available upon acceptance[2].

3.1. Dataset

The database comprises data of 248,867 companies, with 13,333,020 lines (entries) and 92 columns (features). However, only 2,247 of the total number of companies from the database are labeled as "Regular" or "Fraud", so that a relatively small portion of the database is available for the supervised learning. The number of companies and entries of each class in the resulting dataset is shown in Table 1. The ground-truth labels were assigned by human auditors (experts), who manually analyzed the companies' transactions. When auditors assign a label to a company, all the entries (transactions) associated to this company receive the same label. The data preprocessing (Section 2.2) results in a dataset with 315,588 entries and 109 features in total.

Table 1. Overview of the dataset.

Label	Number of Companies	Total of Entries
Fraud	460	8,771
Regular	1,787	306,816

[2]The release of the anonymized data is undergoing formal approval.

3.2. Performance Metrics

The performance metrics include precision, recall, and F1-score (all of which are macro averaged). Since the main purpose of the system is to assist auditors by filtering the number of companies that they have to analyze to uncover frauds, the F1-score is the most relevant metric because it is the harmonic mean between precision and recall. As both take into account the number of true positives a model generates, they quantify how successful a model was in classifying fraudulent companies.

3.3. Experiments

The experiments are conducted using a k-fold cross-validation protocol ($k = 10$) where, in each run, 7 folds are reserved for training, 1 for validation, and 2 for test. The folds are stratified with respect to the companies, *i.e.*, each fold has 10% of all the regular/fraudulent companies. The data is partitioned on a rolling basis, *i.e.*, the training-validation-test partition for the first run comprises the folds $\{1, \ldots, 7\}$ (training), $\{8\}$ (validation), $\{9, 10\}$ (test), $\{2, \ldots, 8\}$, $\{9\}$, $\{10, 1\}$ for the second run, and so on, resulting in 10 runs/partitions. It is worth emphasizing that there is no overlap of companies between the training, validation and test sets. All runs are performed independently and do not use each other's results.

For a single run, the classification models are trained using different hyper-parameter configurations and the resulting models are evaluated on the validation fold. In other words, a *grid-search* is conducted to determine the best validation model (for each classifier) to be assessed in the test folds, *i.e.*, the model that maximizes the macro-averaged F1-score on the validation fold. After the 10 runs, the best models for each fold were evaluated and their corresponding performance metrics on the test folds are averaged over the runs. The range of the tested hyper-parameters for each classifier is defined as follows:

KNN: Number of neighbors: 5, 6, 7, 8, 9, 10; Leaf size passed to the algorithm: 1, 2, 3, 5; Weight function used in prediction: uniform weights (all points in each neighborhood are weighted equally), weight points by the inverse of their distance; Algorithm utilized to compute nearest neighbors: automatic (attempts to decide the most appropriate algorithm [Pedregosa et al. 2011]), balltree, kdtree, brute-force search.

RF: Maximum depth of the tree: 80, 90, 100, 110; Maximum number of features to consider when looking for the best split: 2, 3; Minimum number of samples required at a leaf node: 3, 4, 5; Minimum number of samples required to split an internal node: 8, 10, 12; Number of trees in the forest: 100, 200, 300, 1000.

SVM: Regularization parameter (C): 6, 7, 8, 9, 10, 11, 12; Kernel type: linear, radial basis function (RBF); Kernel coefficient (gamma): 1, 0.1, 0.01, 0.001, 0.0001.

NN: Number of hidden layers: 2; Hidden layer sizes: 200; Activation function for the hidden layer: hyperbolic tan (tanh), rectified linear unit (relu), logistic sigmoid; Solver for

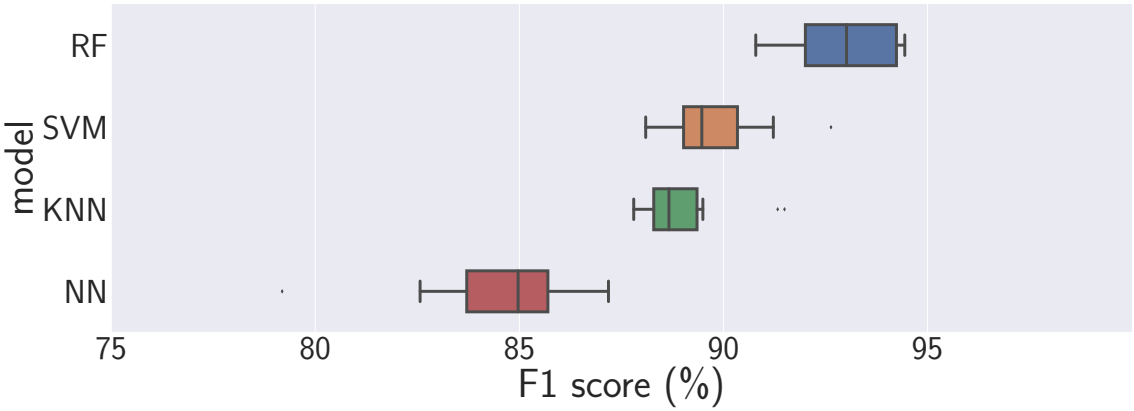

Figure 2. F1 scores from the test partitions.

weight optimization: stochastic gradient descent (sgd), adam; Initial learning rate: 0.01, 0.001, 0.0001; Learning rate schedule for weight updates: constant, adaptive.

3.4. Quantitative vs. Qualitative Analysis

The analysis of the experimental results is two-fold. First, in the quantitative analysis, the performances obtained with different classifiers are compared based on the Precision, Recall, and F1-score metrics (Section 3.2) for a fixed confidence score threshold of 50%. After that, we qualitatively analyze the trade-off between performance and manual workload by using the proposed system with the best-performed classifier. Such a discussion is grounded on the distribution of True/False Positive/Negatives for different confidence score thresholds – including the usual 50% adopted in the quantitative analysis – and assumes a simple protocol where negative predictions (*i.e.*, companies predicted as regular) are discarded and positive predictions (*i.e.*, companies predicted as fraudulent) are submitted to manual review of the auditors. This protocol tends to alleviate significantly the manual workload since the majority (regular) class is intended to be discarded.

3.5. Experimental Setup

The experiments were carried out on an Intel Core i7-8700 3.2 GHz with 16 GB of RAM, running Linux Ubuntu 16.04. The *sklearn* framework [Pedregosa et al. 2011] was adopted for all the steps involving training and evaluation of the models. The training-validation phase (for all models) took approximately 144 hours, whereas the test lasted approximately 12 hours.

4. Results and Discussion

4.1. Quantitative Analysis

Table 2 shows the quantitative results obtained with the cross-validation protocol. As it can be noted, RF achieved the highest performance for all metrics, being the only classifier to achieve F1-score superior to 90%. On the other hand, the NN yielded the lowest F1-score and the highest standard deviation for this metric. This is mostly caused by the low recall, which means that the fraudulent companies are not being properly recovered with the use of the NN.

Table 2. Performance of the best validation models obtained with the cross-validation protocol (macro-averaged metrics).

Models	Metric (%)		
	Precision	**Recall**	**F1-score**
Random Forest	$\mathbf{94.24 \pm 1.49}$	$\mathbf{92.04 \pm 2.31}$	$\mathbf{92.98 \pm 1.23}$
SVM	88.39 ± 1.43	91.92 ± 2.70	89.82 ± 1.26
KNN	88.97 ± 1.49	89.26 ± 1.48	89.10 ± 1.24
Neural Network	89.49 ± 1.65	81.26 ± 2.79	84.42 ± 2.15

Concerning the precision metric, SVM, KNN, and NN perform very similarly, being the highest variation of 1.10% (NN and SVM). In addition to the high F1-score average, RF achieved the lowest deviation for this metric, which is indicative of the robustness of the method for this application. Such a result shows that our system can be a valuable resource to reduce the manual effort of auditors in the analysis of large amounts of data.

The chart in Figure 2 shows the classifiers' performance focusing on the F1-score metric. Visually, there is some overlap between SVM and KNN boxplots. For a more accurate evaluation, the paired Student's t-test was used to assess the statistical performance equivalence between every two classifiers. A threshold of 5% was adopted for the p-value, which means that two classifiers whose the F1-score yields a p-value greater or equal 5% are considered statistically equivalent in performance. In fact, the p-value for SVM $\times$ KNN is approximately 31%, which implies the performance equivalence of the two classifiers. For the other cases (pairs), the p-value was below 0.025% ($\ll$5%), indicating a statistically significant difference in performance.

4.2. Qualitative Analysis

A possible use of the system is to filter out regular companies (negative) so that the remaining companies predicted as fraudulent (positives) can be reassessed by the auditors. Figure 3 shows the trade-off between performance and manual workload by using the proposed system with Random Forest, the classifier which achieved the best performance on the quantitative analysis. The dashed line (gray) represents the positive ratio across different values for the confidence threshold. This line indicates the amount (in %) of companies detected as fraudulent by the system, therefore the decreasing curve indicates the decrease in manual workload. The blue area (FP) indicates the proportion of false positives, which are regular companies wrongly detected as fraudulent by the system. As it can be seen in the chart, the amount of false positives decays sharply from 0 of confidence threshold to 5%, whereas the true positives (TP) remains stable until around 50% of threshold. This implies a significant reduction of manual effort by correctly discarding regular companies without losing track of the fraudulent companies, which is a desirable behavior of the system.

From 5% on, the curve decreases smoothly and we gradually notice the appearance of false negatives (FN). Although the manual review workload is decaying, such a movement is slow and happens at the cost of losing some fraud detections. Interestin-

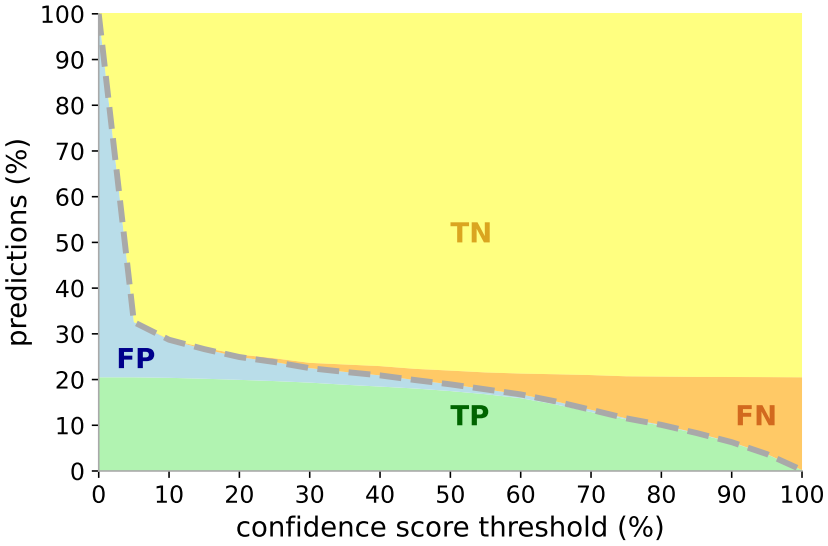

Figure 3. Qualitative effort analysis based on the distribution of True/False Positive/Negatives across confidence score threshold. The dashed line encompasses the positive/total ratio, *i.e.*, the proportion between the predicted fraudulent companies and the total of companies.

gly, the usual threshold of 50% adopted in our quantitative analysis, yields 81.05% of reduction in manual effort and the loss of 14.67% of the fraudulent detections.

5. Conclusion and Future Work

This work presented a machine-learning based system to detect tax evasion in the state of Espírito Santo (Brazil). The system works by classifying transactions performed by companies as either fraudulent or regular, and then creating a consensus about the company regularity status through a fusion function. The experimental analysis focused on investigating the effectiveness of 4 classical classifiers in the identification of fraudulent companies whose transaction records were provided by SEFAZ-ES.

Results showed that Random Forest yielded the best performance with a macro-averaged F1-score of 92.98%. SVM and KNN achieved statistically equivalent performance, whereas the lowest F1-score was achieved with the Neural Network. Random Forest also showed robustness given the lower standard deviation for all metrics, in particular for the F1-score ($<$1.25%). Additionally, it was discussed the use of the system as a preliminary filter to reduce the manual workload of the auditors. We showed that using our experimental setup (*i.e.*, confidence threshold of 50%) and the Random Forest classifier, the manual effort can be reduced in approximately 81% with a loss of nearly 15% of the fraudulent companies.

Overall, the investigation indicates the feasibility of the proposed system for automatic detection in a real-world environment, as it is the SEFAZ-ES database. This would imply scaling up the number of processed transactions, and, therefore, to increase the

public revenue by fighting corruption. Alternatively, our methodology can be used to assist auditors in the analysis of tax evasion and match the manual workload to the current capacity of the auditors' team.

Future work includes incorporating into the model the features to be introduced in the database by the SEFAZ-ES team. Since there are far more regular companies than fraudulent ones, we will also address the data imbalance problem to try to improve the system performance. In addition to the current performance metrics, it will be investigated the incorporation of financial transactions into our model. Finally, our research will address transfer learning across geographic regions to deal with the problem of availability of annotated data in some regions, as done in [Zhu et al. 2018].

Referências

Abakarim, Y., Lahby, M., and Attioui, A. (2018). An efficient real time model for credit card fraud detection based on deep learning. In *Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications*, page 30. ACM.

Awoyemi, J. O., Adetunmbi, A. O., and Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 International Conference on Computing Networking and Informatics (ICCNI)*, pages 1–9.

Chouiekh, A. and Haj, E. H. I. E. (2018). Convnets for fraud detection analysis. *Procedia Computer Science*, 127:133–138.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

Fu, K., Cheng, D., Tu, Y., and Zhang, L. (2016). Credit card fraud detection using convolutional neural networks. In *International Conference on Neural Information Processing*, pages 483–490. Springer.

Ho, T. K. (1995). Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE.

Kumar, M. S., Soundarya, V., Kavitha, S., Keerthika, E. S., and Aswini, E. (2019). Credit card fraud detection using random forest algorithm. In *2019 3rd International Conference on Computing and Communications Technologies (ICCCT)*, pages 149–153.

Lavion, D. et al. (2018). Pulling fraud out of the shadows. *PwC's*.

Liu, C., Chan, Y., Alam Kazmi, S. H., and Fu, H. (2015). Financial fraud detection model: Based on random forest. *International journal of economics and finance*, 7(7).

Maes, S., Tuyls, K., Vanschoenwinkel, B., and Manderick, B. (2002). Credit card fraud detection using bayesian and neural networks. In *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, pages 261–270.

McCulloch, W. S. and Pitts, W. (1988). *A Logical Calculus of the Ideas Immanent in Nervous Activity*, page 15–27. MIT Press, Cambridge, MA, USA.

Mittal, S. and Tyagi, S. (2019). Performance evaluation of machine learning algorithms for credit card fraud detection. In *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pages 320–324.

Nadim, A. H., Sayem, I. M., Mutsuddy, A., and Chowdhury, M. S. (2019). Analysis of machine learning techniques for credit card fraud detection. In *2019 International Conference on Machine Learning and Data Engineering (iCMLDE)*, pages 42–47.

Najadat, H., Altiti, O., Aqouleh, A. A., and Younes, M. (2020). Credit card fraud detection based on machine and deep learning. In *2020 11th International Conference on Information and Communication Systems (ICICS)*, pages 204–208.

Ngai, E., Hu, Y., Wong, Y., 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.

Pai, P.-F., Hsu, M.-F., and Wang, M.-C. (2011). A support vector machine-based model for detecting top management fraud. *Knowledge-Based Systems*, 24(2):314–321.

Paula, E. L., Ladeira, M., Carvalho, R. N., and Marzagao, T. (2016). Deep learning anomaly detection as support fraud investigation in brazilian exports and anti-money laundering. In *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 954–960. IEEE.

Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

SADGALI, I., SAEL, N., and BENABBOU, F. (2019). Fraud detection in credit card transaction using machine learning techniques. In *2019 1st International Conference on Smart Systems and Data Science (ICSSD)*, pages 1–4.

Simpson, J. (2006). *Oxford Dictionary of English*. Oxford University Press, Oxford, United Kingdom.

Thennakoon, A., Bhagyani, C., Premadasa, S., Mihiranga, S., and Kuruwitaarachchi, N. (2019). Real-time credit card fraud detection using machine learning. In *2019 9th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pages 488–493.

Wu, Y., Zheng, Q., Gao, Y., Dong, B., Wei, R., Zhang, F., and He, H. (2019). Tedm-pu: A tax evasion detection method based on positive and unlabeled learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 1681–1686. IEEE.

Yao, J., Zhang, J., and Wang, L. (2018). A financial statement fraud detection model based on hybrid data mining methods. In *2018 International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pages 57–61.

Zhu, X., Yan, Z., Ruan, J., Zheng, Q., and Dong, B. (2018). IRTED-TL: An inter-region tax evasion detection method based on transfer learning. In *17th IEEE Intl. Conf. On Trust, Security And Privacy In Computing And Communications/12th IEEE Intl. Conf. On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1224–1235.