Fraud Detection in Big Data using Supervised and Semi-supervised Learning Techniques

Germán E. Melo-Acosta
I&S Research Group
Universidad de Antioquia
Calle 67 No. 53 - 108, 050010, Medellín, Colombia
Email: geduardo.melo@udea.edu.co

Freddy Duitama-Muñoz, Julián D. Arias-Londoño
Dpt. of Systems Engineering and Computers Science
Universidad de Antioquia
Calle 67 No. 53 - 108, 050010, Medellín, Colombia
Email: {john.duitama, julian.ariasl}@udea.edu.co

***Abstract*—This paper addresses the credit card fraud detection problem in the context of Big Data, based on machine learning techniques. In the fraud detection task, typically the available datasets for ML training present some peculiarities, such as the unavoidable condition of a strong class imbalance, the existence of unlabeled transactions, and the large number of records that must be processed. The present paper aims to propose a methodology for automatic detection of fraudulent transactions, that tackle all these problems. The methodology is based on a Balanced Random Forest, that can be used in supervised and semi-supervised scenarios through a co-training approach. Two different schemes for the co-training approach are tested, in order to overcome the class imbalance problem. Moreover, a Spark platform and Hadoop file system support our solution, in order to enable the scalability of the proposed solution. The proposed approach achieves an absolute improvement of around 24% in terms of geometric mean in comparison to a standard random forest learning strategy.**

***Keywords*—Fraud Detection, Machine Learning, Random Forest, Semi-supervised Learning, Spark**

I. Introduction

Nowadays, most of the organizations, companies and government agencies are conducting a large portions of their activities using information systems, for example, accounting, human resources, customer relationship management, marketing, and sales [1]. Following this perspective, organizations have adopted practices associated to electronic commerce to increase their productivity or efficiency in trading products or services [2]. Simultaneously, the accessibility, omnipresence and high performance of the personal computing are motivating individuals to do their banking, shopping, and entertainment electronically. This increasing trend has lured criminals to move their efforts into this new territory making the electronic fraud a serious problem in several contexts. Based on the review presented in [2], Figure 1 shows a taxonomy of the areas where the electronic fraud can be found.

According to PwC consulting, in the Global Economic Crime Survey 2016 [3] approximately one in three (36%) surveyed companies reported have been victim of some kind of cybercrime, which means an increase in about 12% according to the 2014 survey [4]. Also from the 2016 survey, the cybercrime was found to be the second of the most commonly

978-1-5386-1060-2/17/$31.00

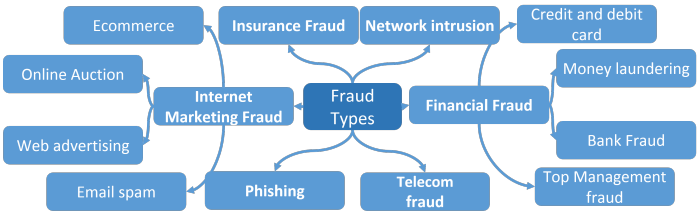

Fig. 1. Fraud taxonomy based on the results of [2]

types of economic crime analyzed in the report; it was only behind of the asset misappropriation.

Technically speaking, two approaches are used to combat these crimes, the fraud prevention systems (FPS) and fraud detection systems (FDS). The FPS are the pro-active hardware and software mechanisms to avoid the fraud occurrence, on the other hand, the FDS are used when the fraudsters have overtaken the FPS. The main objective of the FDS is identify the fraud as soon as possible in order to take the necessary actions to revert it [5].

In particular, credit card fraud detection is one of the most explored kind of fraud [6] and is defined as the activity of obtaining financial advantage or causing economic loss to a credit card holder without his explicit knowledge. In other words, all activity performed by a person who is not the cardholder. There are many ways in which fraudsters obtain the information to perpetrate the fraud [7]: a) Association with commercial establishments which share the information of their customers in unauthorized manner, b) Using phishing trough email, cloned bank web pages or fake shopping websites. c) Using software tools to generate valid credit card numbers, and/or d) By physical robbery and personal investigation.

In the last few years, Machine Learning (ML) techniques have emerged as alternative for the development of automatic FDS. ML makes possible to take into account the wide range of ways in which fraud can be performed. From a machine learning point of view, the fraud detection problem corresponds to a supervised classification task aiming to determine whether a new transaction is legitimate or fraudulent one. The training phase of supervised ML algorithms is intended to adjust model that generalize the class distribution; however, the generalized model to be built has three challenges that make the process a

specially tough task. The first one is the unavoidable imbalance in the class distribution on the datasets. On the real world, the number of fraud transactions are (fortunately) much less than legal transactions, but the skewness on the distribution provokes a bias in the model prediction [8]. Secondly, there are an important number of unlabeled samples consequence of reject policies of a particular commerce, or due to communication fails into the banking system. Therefore, the transactions rejected by these cases cannot be associated to fraud. Lastly, since the large number of transactions and the amount of the information that must be processed every day, it is necessary to use training techniques with proved scalability. Bearing all this in mind, the learning methods for designing automatic FDS should be able to take advantage of the unlabeled samples integrating supervised and semi-supervised methods and also be able to work in a Big Data context.

During the last few years, several techniques have been used for credit card fraud detection. As in many applications, Artificial Neural Networks (ANN) has been used extensively in this context; FDS based on Multilayer perceptrons (MLP) were proposed in [9]–[11], Self-Organizing Maps were presented on [12], and a proposal for using both types of ANN was reported in [13]. Other techniques that can be found in the state of the art include: Decision Trees (DT) [14], Support Vector Machine (SVM) [15], [16], Logistic Regression (LR) [17] and Bayesian learning [18]. Other authors have focused on the comparison of several of the former techniques; for instance in [19], the authors used a data aggregation technique with various classification algorithms such as SVM, LR, Bayesian learning, DT and a committee of DT called Random Forest (RF). In [20] there was performed a comparison among SVM, RL and RF. For its part, in [21], the authors designed a real time methodology for a FDS and compared RF, LR, and ANN. In all the mentioned comparative works, the authors found RF as the best technique, moreover it is worth to highlight that the results were obtained with different databases composed of common attributes of the transaction like date and time, location, amount, demographics, and derived attributes which are computed from common attributes in order to gather information about frequency of transactions or behavior patterns of the buyer.

In order to overcome the class imbalance problem, FDS approaches based on ML have followed standard methodologies which can be categorized into two types: data level and algorithmic level approaches [22]. At the data level, the works were focused on the use of preprocessing stages, mainly down sampling the majority class, even working with less than 10% of the original database [20]. A slightly different approach presented in [6], proposed a meta-classifier technique where various classifiers were trained with small subsets of the original database. On the other hand, the algorithmic level approach was adopted through the use of cost-sensitive learning methods. For instance, the authors in [5] used cost-sensitive DT modifying the Gini index during the learning of the tree. Similarly, [15] adopted a improved Imbalance Class Weighted Support Vector Machine (ICW-SVM) to balance the SVM learning process. Also, in [10] a methodology based on nonlinear discriminant analysis neural models was tested, aiming to deal with the imbalance class issue, and in [11] the authors constructed a fraud density map using training data to correct the bias of the ANN model. Although all these works got some improvements in comparison with the original imbalance data sets, the main drawback is that most of them use relatively small datasets (less than 100000 samples), which is far from the number of transactions that a real system has to deal with in a very short period of time. But more importantly, most of the works did not propose their approach to be implemented and validated in scalable and distributed environments, which prevent their use in real transaction processing systems. Additionally, the inclusion of unlabeled data has not been studied deeply in the context of fraud detection. Except for some works based on unsupervised learning [10], [12]; To the best of our knowledge, no works have proposed the use of labeled and unlabeled samples in the design of machine learning based FDS.

The present paper aims to propose a FDS of credit card transactions addressing the three pointed out challenges. Taking into account the RF is one of the best models found in the state of the art for this task, the proposed FDS is based on a Balanced Random Forest (BRF) [23], which allows to compensate the class imbalance problem and have achieved better results than other cost-sensitive learning methods [23]. Furthermore, the unlabeled samples are integrated using the co-training approach presented in [24] on the BRF predictors; this approach have proven be superior to many semi-supervised learning solutions when there are not two sufficiently independent and redundant set of attributes [23] and have the advantage to implement semi-supervised learning without modify the training algorithm of the base model. Moreover, our approach implements two different co-training schemes in order to overcome the class imbalance problem. Lastly but no less important, the whole methodology is implemented in Apache Spark [25] with the SQL and DataFrame Libraries [26], guaranteeing the scalability of the proposed solution. The rest of the paper is organized as follows: section II presents a brief description of the features definition and the classification models. Section III describes the database used and the experimental setup. Then, in section IV the results of the experimental validation are presented and discussed. Lastly, section V presents some conclusions derived from the results.

II. Methodology

A. Characterization

The set of features consists of 138 variables extracted from the transactions and related with the buyer, commerce, card features, amount and device used to make the transaction, among others. The information used at this stage corresponds to the features reported in some of the works reviewed [12], [13], [16], [17], [19]–[21], and also some expert knowledge from the partner company that provided the data.

All the calculation and the procedures to get the 138 variables are based on Apache Spark used the DataFrame API. The selection of this API instead of the Core of Apache Spark is due the implementation of the catalyst optimizer on the API mentioned [26], this optimizer is a tool developed to generate efficient code based on SQL-like operations.

B. Detection methods

The RF model is an ensemble of DT grouped in order to reduce the risk of over-fitting [27]. Like DT, RF handles

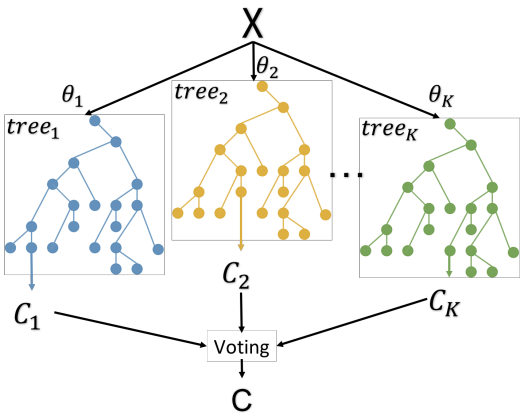

Fig. 2. A general architecture of a random forest classifier.

categorical features, extends to the multi-class classification setting, does not require feature scaling, and is able to capture non-linearities and feature interactions. The basic algorithm of RF trains a set of DT separately from multiple bootstrap sampling of the training set, and injecting additional randomness into the selection of the features set analyzed in every decision node; in this way each DT is a bit different from each other in the ensemble. Combining the predictions from each tree reduces their variance improving the performance on test data [27].

Apart from the above randomizations, DT training is done in the same way as for individual DT. The final decision about the class to which sample belongs is taken by consensus among the different trees. Two different combinations can be carried out, a hard decision based on majority vote and a soft decision based on the class conditional probability estimated as the joint probability that every single tree of the ensemble produces the same output. A scheme of RF used as classifier is shown in the Fig. 2.

Formally, assuming a set of training data $\chi = (\mathbf{x}_m, y_m), m = 1, \ldots, M$, where $\mathbf{x}_m$ is an input observation and y_m is a target output. A predictor with low bias and high variance $f(\mathbf{x}, \chi)$ can be created using the training set χ. Now, using randomly sampling from the set χ, a collection of predictors $f(\mathbf{x}, \chi, \theta_k)$ can be created, with $f(\mathbf{x}, \chi, \theta_k)$ being the k-th predictor, and θ_k the random vector selecting the data samples for the respectively predictor. If θ_k is generated applying bootstrap sampling and since the several θ_k, $k = 1, \cdots K$, are independent and identically distributed (*i.i.d.*), it can be shown that combining *i.i.d.* randomized predictors into a committee, it leaves the bias approximately unchanged while reduces the variance by a factor equal to the mean value of the correlation among the ensemble's predictors [27]. Thus, if correlation and bias of *i.i.d.* randomized predictors are kept low, a big reduction in test set error can be obtained [27].

Due to the independence of each DT in the RF, the algorithm can be implemented in a distributed environment in order to train several DT in parallel. In this sense, the authors in [28] propose the implementation of RF based on Map-Reduce, which is framework for processing parallelizable problems across large datasets using a large number of computers (nodes), collectively referred to as a cluster [29]. Apache Spark [25] extends the Map-Reduce model to efficiently support more types of computations, including interactive queries and stream processing based on a structure called Resilient Distributed Dataset (RDD), thanks to the open-source community, in Apache Spark the RF based on the work of [28] was implemented and included in the Machine Library (MLlib). Specifically, the Apache Spark RF implementation has the following parameters:

1) Number of trees
2) Maximum depth of the trees
3) Impurity measure (Entropy or Gini index)
4) Number of bins used when discretizing continuous features.

As it was mentioned previously, the basic RF cannot deal with the imbalance class issue. With the aim of adapting the RF for tackling this problem, in [23] a Balance RF algorithm was proposed, which consists on the training of each DT of the ensemble using a perfectly balanced dataset. The training set of every DT is composed by the total number of samples from the minority class, and a equally-sized subset from the majority class. In other words, down-sampling the majority class to the number of minority class samples. At the same time the RF should be built with a number of trees equal to the rate between number of samples from minority class vs majority class. Defining the training dataset like $\chi = \chi_\alpha \cup \chi_\omega$, thus, the union between the subsets of the majority class χ_α and minority class χ_ω, the number of trees in a BRF is given by

$$K^{'} = \left\lceil \frac{M_\alpha}{M_\omega} \right\rceil \tag{1}$$

where M_α and M_ω are the sizes of the subsets χ_α and χ_ω respectively, and $\lceil \cdot \rceil$ indicates the least succeeding integer (ceiling function). Currently, the Apache Spark MLlib does not include implementation of neither BRF or other approach to deal with skewed classes distribution. In this work, a modification of the current Spark RF API was performed, in order to include the BRF algorithm [1].

On the other hand, the inclusion of the unlabeled samples is based on the democratic co-learning algorithm [24], in which a set of learners with different inductive biases are trained separately using the originally labeled examples. After this training phase, the learners are used to make prediction over the unlabeled samples by using the soft-voting decision scheme, and if a majority of the classifiers agree on the labeling of some unlabeled examples, and the probability of the predicted label is greater than a inclusion probability threshold γ, the newly-labeled samples are added to a new training set χ_c composed by these newly-labeled samples and the original training set. Finally, all learners are trained in a second phase using χ_c as training set.

Considering the class imbalance approach, two strategies for the inclusion of unlabeled samples into the χ_c set are proposed. The first one consists in the inclusion of a sample if

[1] As soon as the results of the research be published, such implementation will be free available in GitHub.

the predicted label probability for such sample is greater than γ. The second strategy includes the unlabeled sample whether the predicted label belongs to the minority class and the first condition is also satisfied. The proposed scheme is shown in Figure 3.

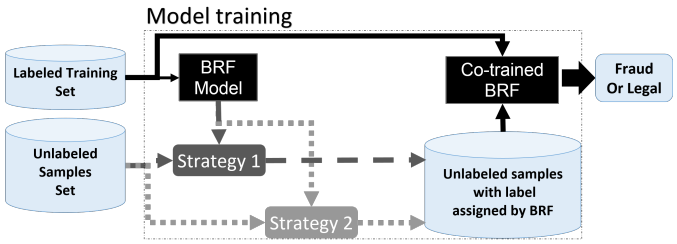

Fig. 3. Schema of the co-training approach to include unlabeled samples according to the prediction probability.

Finally, taking advantage of the meta-classification approach, the final model evaluated consists in the union of 2 best models obtained with the algorithms described above. The predictions of the 2 models are aggregated using soft-voting with the probability of each classifier.

III. Database and Experimental Set up

The dataset used in this work was provided by a Colombian payment gateway company, as part of a joint research project. The original database provided by the partner company is a relational database. This database has a structure made to support the business model of company, therefore a pre-processing of the information was required. After the characterization stage, the dataset was exported from the relational engine o Apache Hive[2], a data warehouse based on map-reduce.

The dataset on Hive have the distribution of the three classes: fraud, legal and unlabeled, shown in Table I.

TABLE I. Distribution of classes of the database used

Class	N° of samples	%	% (Fraud And Legal)
Legal	6488341	80.288	99.927
Fraud	4726	0.058	0.073
Unlabeled	1538104	19.033	NA

All the experiments were carried out using a cross-validation strategy with 5 folds. In order to get a baseline model to compare the BRF and the co-training approaches, the first model trained and validated is the Apache Spark RF implementation. Also, this first model is trained to evaluate the influence of the four parameters of Apache Spark RF implementation. A grid search of the four parameters was performed with the following values: depth=$[10, 20, 30]$), bins=$[32, 128]$, impurity=$[entropy, gini]$, trees=$[2, 10, 50, 100]$. The total number of experiments was 240. The values of the grid for BRF are the same excluding the number of trees, due to the fact that in this model the trees are pre-calculated with the eq.(1). In the co-training model the parameters evaluated were $\gamma = [0.75, 0.85, 0.95]$ and the condition of whether both classes are added to the new training set or only the fraudulent one.

[2]https://hive.apache.org/

The different approaches tested were evaluated in terms of sensitivity, specificity and accuracy. However, since the accuracy is not able to distinguish between the relative weight of the different classes, which, in the context of imbalanced problems may lead to erroneous conclusions; for this reason, an additional set of evaluation measures was included. Lets TP be the number of fraudulent samples correctly classified (True Positives), TN be the number of legal samples correctly classified (True Negatives), FN be the number of fraudulent samples wrongly classified (False Negatives), and FP be the number of legal samples wrongly classified (False Positives). The sensitivity, specificity and accuracy are defined as

$$S = \frac{TP}{TP + FN}; \quad E = \frac{TN}{TN + FP} \tag{2}$$

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

Sensitivity (eq. (2) left) and specificity (eq. (2) right) measure the accuracy on the positive (fraud) and negative (legal) cases. A trade-off between these true positives and true negatives is typically sought. Therefore, the set of metrics able to quantify the performance in imbalance scenarios includes the G-mean, which can be defined as

$$G\text{-}mean = (S \cdot E)^{1/2} \tag{4}$$

and the weighted-Accuracy expressed as

$$wtdAcc = \beta S + (1 - \beta)E \tag{5}$$

Both $G\text{-}mean$ and $wtdAcc$ provide summary performance indicators of the tradeoff between S and E, taking into account the relative weight of each class in the test sample. The β used is 0.7 to indicate higher weights for accuracy on the fraud cases. Well known Area Under the ROC Curve (AUC) was also included. AUC is often considered a better measure of overall performance [30], and is independent of specific classification cutoff values.

IV. Results

Figure 4 shows the results in terms of S, E and $G\text{-}mean$ obtained using the default Spark RF model, and according to the grid search used for the different parameters listed in section III. From Figure 4 is possible to observe that after a number of trees equal to 50, the performance did not change significantly. Also the depth of the trees has an important effect in the performances unlike the number of bins and the impurity measurement, whose influence was not determinant. The results for all the validation folds are included in the plots, which allows to see the stability of the results through different runs of the system. Another important aspect to emphasizes is the difference between the S and E measures; the results show that the standard RF is biased to the legal class and detect it almost in a perfect manner with a trade off in sensibility. Such a trade off is more easily to observe in the values of the $G\text{-}mean$.

In contrast, the results of the BRF shown in Figure 5 prove that the balanced strategy reduce significantly the imbalance

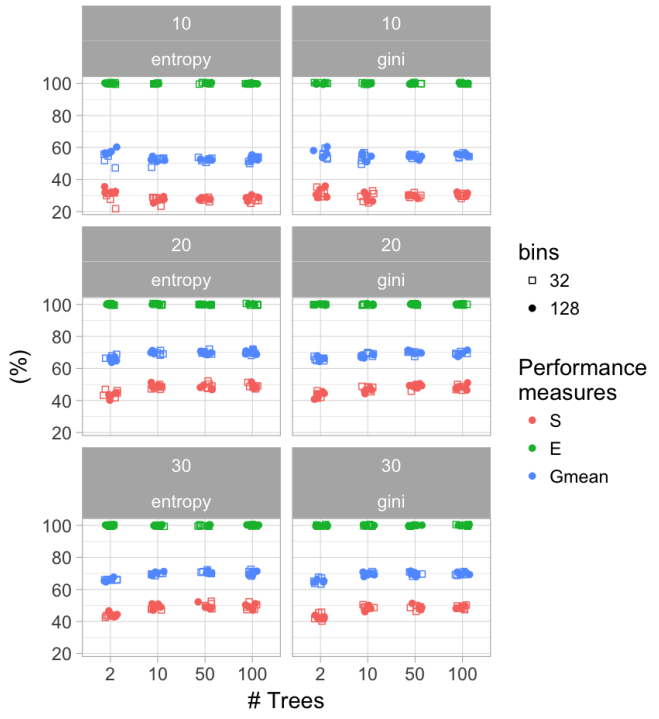

Fig. 4. Performance of the standard Spark RF model for the different parameters evaluated. At the top of every plot is the impurity measure used during the information gain calculation and the maximum depth of the trees. Results for each validation fold are included.

impact on model training. The BRF increases the $G\text{-}mean$ in more than 23% in absolute terms, and provides similar values for S, E and $G\text{-}mean$, showing a less biased performance. On the other hand, the influence of trees's depth is not as strong as in the normal RF, which is very important since it reduces the number of experiments required during validation. Also the impact due to the number of bins and the impurity measure remains negligible.

The results of the co-train approaches are shown in Figure 6, which presents the values of S, E and $G\text{-}mean$ on the test dataset for the five validation folds. None of the two strategies based on the co-training methodology, improve the prediction of the positive class, the values of S obtained by co-trained BRF are a little bit lower than normal BRF. Nevertheless, the co-trained BRF improves the values of E meaning that the accuracy on the negative class prediction increased. The control parameter γ showed a very important effect on the performance of the co-training methodology, as it increases, the values of S and $G\text{-}mean$, in average, increase too in both strategies evaluated.

The last strategy evaluated was the combination of the Standard BRF and the best Co-trained BRF model, which is called here Meta-classifier. Table II summarizes the best results for all the strategies, including the Meta-Classifier. The tuples (trees, depth, impurity, bins) of the grid search which achieved the performances shown in Table II were, for Spark RF and BRF (50,30,entropy,32) and (1373,20, Gini, 32), respectively. For the co-training approaches the result was obtained using a model with $\gamma = 0.95$ in both strategies mentioned. According with these results, all the strategies used

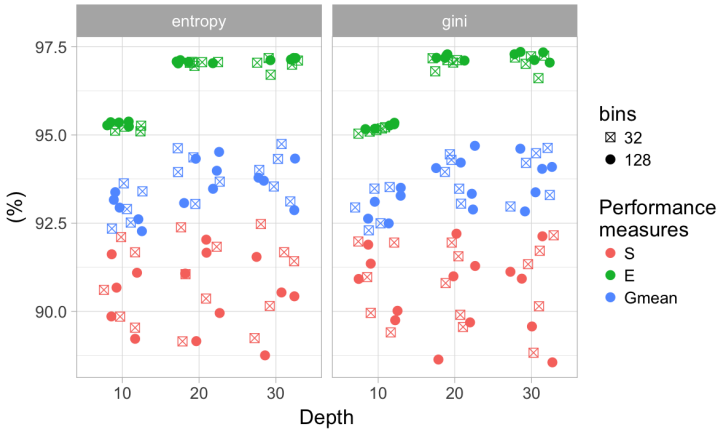

Fig. 5. Performance obtained by BRF according to S, E and $G\text{-}mean$ for the different set of parameters in the grid search

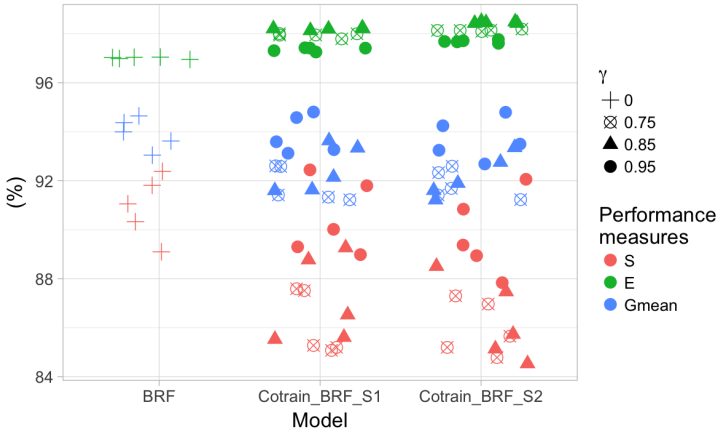

Fig. 6. Performance obtained by BRF using the second con-training strategy according to S, E and $G\text{-}mean$ for the different set of parameters in the grid search

to tackle the imbalance class problem, achieved prediction improvements in S, $G\text{-}mean$, $AUCROC$ and $wtdAcc$ above of 80%, 30%, 25% and 40% respectively, with a reduction in specificity less than 3%. The best strategy for fraud detection was the proposed combination of BRF and co-trained BRF using the first strategy yielding to a 94.63% in $G\text{-}mean$ with a very low confidence interval. From table II is also possible to confirm that Acc is not an appropriate measure in this context, since the class imbalance problem. Both the $G\text{-}mean$ and $wtdAcc$ showed the improvement in the performance of the balanced strategies in comparison to the standard Spark RF model.

V. Conclusions

A fraud detection system for credit card transactions was presented. The system was designed to tackle the three challenges related with fraud detection data sets, namely a strong class imbalance, the inclusion of labeled and unlabeled samples, and the ability to process a large number of transactions. The result showed that the proposed successfully overcome all the challenges. A BRF based on the Spark RF model was implemented, in order to compensate the class imbalance of

TABLE II. BEST RESULTS OBTAINED FOR ALL THE STRATEGIES EVALUATED

Model	Acc (%)	$S(\%)$	$E(\%)$	$G\text{-}mean(\%)$	$AUC\ ROC$	$wtdAcc$
Spark RF	99.96 ± 0.00 *	50.02 ± 1.50	100.00 ± 0.00	70.72 ± 1.06	0.75 ± 0.01	65.01 ± 1.05
BRF	97.02 ± 0.02	90.95 ± 1.28	97.03 ± 0.04	93.94 ± 0.64	0.94 ± 0.10	92.77 ± 0.91
BRF co-trained with Strategy 1	97.37 ± 0.04	90.53 ± 1.54	97.37 ± 0.04	93.89 ± 0.78	0.94 ± 0.01	92.58 ± 1.09
BRF co-trained with Strategy 2	97.70 ± 0.04	89.81± 1.65	97.70 ± 0.04	93.67 ± 0.85	0.94 ± 0.01	92.17 ± 1.17
Meta-Classifier	97.13 ± 0.02	91.67 ± 1.11	97.14 ± 0.02	94.36 ± 0.57	0.94 ± 0.01	93.31 ± 0.78

*mean ± standard deviation

the dataset and the unlabeled samples were used trough a co-training approach using the BRF model. Moreover, a proposed strategy based on a meta-classification approach that combines BRF and Co-Trained BRFs achieved the best performance. All the different strategies evaluated were implemented on Apache Spark guaranteeing the scalability of the proposed approach.

ACKNOWLEDGMENT

This research was carried out under grant 020C-2016 funded by Ruta N corporation, Medellín, Colombia.

REFERENCES

[1] M. Behdad, L. Barone, M. Bennamoun, and T. French, "Nature-inspired techniques in the context of fraud detection," *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 42, no. 6, pp. 1273–1290, 2012.

[2] A. Abdallah, M. A. Maarof, and A. Zainal, "Fraud detection system: A survey," *Journal of Network and Computer Applications*, vol. 68, pp. 90–113, 2016.

[3] PwC, "Global Economic Crime Survey," Tech. Rep., 2016. [Online]. Available: https://www.pwc.com/gx/en/economic-crime-survey/pdf/GlobalEconomicCrimeSurvey2016.pdf

[4] PricewaterhouseCoopers, "PwC Global Economic Crime 2014 Survey: Cybercrime and Electronic Fraud," 2014. [Online]. Available: http://www.pwc.com/gx/en/economic-crime-survey/cybercrime.jhtml

[5] Y. Sahin and E. Duman, "Detecting Credit Card Fraud by Decision Trees and Support Vector Machines," *International Association of Engineers*, vol. I, 2011.

[6] P. K. Chan, W. Fan, A. Prodromidis, and S. J. Stolfo, "Distributed data mining in credit card fraud detection," *IEEE Intelligent Systems*, vol. 14, pp. 67–74, 1999.

[7] R. Patidar and L. Sharma, "Credit Card Fraud Detection Using Neural Network," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 1, no. May, pp. 13–14, 2011.

[8] N. Chawla and K. Bowyer, "SMOTE: synthetic minority over-sampling technique," *arXiv preprint arXiv: ...*, vol. 16, pp. 321–357, 2011.

[9] E. Aleskerov, B. Freisleben, and B. Rao, "Cardwatch : A Neural Network Based Database stern for Credit Card Fraud Detection," *Proceedings of the IEEE/IAFE*, pp. 220–226, 1997.

[10] J. R. Dorronsoro, F. Ginel, C. Sánchez, and C. Santa Cruz, "Neural fraud detection in credit card operations," *IEEE Transactions on Neural Networks*, vol. 8, no. 4, pp. 827–834, 1997.

[11] M.-J. Kim and T.-S. Kim, "A neural classifier with fraud density map for effective credit card fraud detection," *Intelligent Data Engineering and Automated LearningIDEAL 2002*, pp. 21–30.

[12] D. Olszewski, "Fraud detection using self-organizing map visualizing the user profiles," *Knowledge-Based Systems*, vol. 70, pp. 324–334, 2014.

[13] F. Ogwueleka, "Data mining application in credit-card Fraud detection system," *Journal of Engineering Science and Technology*, vol. 6, no. 3, pp. 311–322, 2011.

[14] Y. Sahin, S. Bulkan, and E. Duman, "A cost-sensitive decision tree approach for fraud detection," *Expert Systems with Applications*, vol. 40, no. 15, pp. 5916–5923, 2013.

[15] Qibei Lu; Chunhua Ju, "Research on Credit Card Fraud Detection Model Based on Class Weighted Support Vector Machine," *Journal of Convergence Information Technology*, vol. 6, no. 1, pp. 62–68, 2011.

[16] V. Dheepa and R. Dhanapal, "Behavior Based Credit Card Fraud Detection Using Support Vector Machines," *ICTACT Journal on Soft Computing*, vol. 6956, no. July, pp. 391–397, 2012.

[17] S. Jha, M. Guillen, and J. Christopher Westland, "Employing transaction aggregation strategy to detect credit card fraud," *Expert Systems with Applications*, vol. 39, no. 16, pp. 12 650–12 657, 2012.

[18] S. Panigrahi, A. Kundu, S. Sural, and a. K. Majumdar, "Credit card fraud detection: A fusion approach using Dempster-Shafer theory and Bayesian learning," *Information Fusion*, vol. 10, no. 4, pp. 354–363, 2009.

[19] C. Whitrow, D. J. Hand, P. Juszczak, D. Weston, and N. M. Adams, "Transaction aggregation as a strategy for credit card fraud detection," *Data Mining and Knowledge Discovery*, vol. 18, no. 1, pp. 30–55, 2009.

[20] S. Bhattacharyya, S. Jha, K. Tharakunnel, and J. C. Westland, "Data mining for credit card fraud: A comparative study," *Decision Support Systems*, vol. 50, no. 3, pp. 602–613, 2011.

[21] V. Van Vlasselaer, C. Bravo, O. Caelen, T. Eliassi-Rad, L. Akoglu, M. Snoeck, and B. Baesens, "APATE: A Novel Approach for Automated Credit Card Transaction Fraud Detection using Network-Based Extensions," *Decision Support Systems*, vol. 75, pp. 38–48, 2015.

[22] V. López, A. Fernández, J. G. Moreno-Torres, and F. Herrera, "Analysis of preprocessing vs. cost-sensitive learning for imbalanced classification. Open problems on intrinsic data characteristics," *Expert Systems with Applications*, vol. 39, no. 7, pp. 6585–6608, 2012.

[23] C. Chen, A. Liaw, and L. Breiman, "Using random forest to learn imbalanced data," *University of California, Berkeley*, no. 1999, pp. 1–12, 2004.

[24] Y. Zhou and S. Goldman, "Democratic Co-learning," *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, no. Ictai, pp. 594–602, 2004.

[25] M. Zaharia, M. Chowdhury, T. Das, and A. Dave, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," *NSDI'12 Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, pp. 2–2, 2012.

[26] M. Armbrust, A. Ghodsi, M. Zaharia, R. S. Xin, C. Lian, Y. Huai, D. Liu, J. K. Bradley, X. Meng, T. Kaftan, and M. J. Franklin, "Spark SQL: Relational Data Processing in Spark," *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data - SIGMOD '15*, pp. 1383–1394, 2015.

[27] L. Breiman, "Random Forests," *Machine learning*, pp. 5–32, 2001.

[28] B. Panda, J. S. Herbach, S. Basu, and R. J. Bayardo, "PLANET : Massively Parallel Learning of Tree Ensembles with MapReduce," in *Proceedings of the VLDB Endowment*, 2009, pp. 1426–1437.

[29] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[30] C. X. Ling, J. Huang, and H. Zhang, "Auc: a statistically consistent and more discriminating measure than accuracy," in *IJCAI*, vol. 3, 2003, pp. 519–524.