Detecting Credit Card Fraud using Periodic Features

Alejandro Correa Bahnsen, Djamila Aouada, Aleksandar Stojanovic and Björn Ottersten
Interdisciplinary Centre for Security, Reliability and Trust
University of Luxembourg, Luxembourg
Email: al.bahnsen@gmail.com, djamila.aouada@uni.lu, aleksandar.stojanovic@rwth-aachen.de, bjorn.ottersten@uni.lu

***Abstract*—When constructing a credit card fraud detection model, it is very important to extract the right features from transactional data. This is usually done by aggregating the transactions in order to observe the spending behavioral patterns of the customers. In this paper we propose to create a new set of features based on analyzing the periodic behavior of the time of a transaction using the von Mises distribution. Using a real credit card fraud dataset provided by a large European card processing company, we compare state-of-the-art credit card fraud detection models, and evaluate how the different sets of features have an impact on the results. By including the proposed periodic features into the methods, the results show an average increase in savings of 13%. The aforementioned card processing company is currently incorporating the methodology proposed in this paper into their fraud detection system.**

***Keywords*—Fraud detection; von Mises distribution; Cost-sensitive learning**

I. Introduction

Credit card fraud has been a growing problem worldwide. During 2012 the total level of fraud reached 1.33 billion Euros in the Single Euro Payments Area, which represents an increase of 14.8% compared with 2011 [1]. Moreover, payments across non traditional channels (mobile, internet, ...) accounted for 60% of the fraud, whereas it was 46% in 2008. This opens new challenges as new fraud patterns emerge, and current fraud detection systems are less successful in preventing these frauds. Furthermore, fraudsters constantly change their strategies to avoid being detected, something that makes traditional fraud detection tools, such as expert rules, inadequate [2].

The use of machine learning in fraud detection has been an interesting topic in recent years. Different detection systems that are based on machine learning techniques have been successfully used for this problem, in particular: neural networks [3], Bayesian learning [4], artificial immune systems [5], hybrid models [6], support vector machines [7], peer group analysis [8], online learning [9] and social network analysis [2].

When constructing a credit card fraud detection model, it is very important to use those features that allow accurate classification. Typical models only use raw transactional features, such as time, amount, place of the transaction. However, these approaches do not take into account the spending behavior of the customer, which is expected to help discover fraud patterns [5]. A standard way to include these behavioral spending patterns is proposed in [10], where Whitrow et al. proposed a transaction aggregation strategy in order to take into account a customer spending behavior. The computation of the aggregated features consists in grouping the transactions made during the last given number of hours, first by card or account number, then by transaction type, merchant group, country or other, followed by calculating the number of transactions or the total amount spent on those transactions.

In this paper, we propose a new set of features based on analyzing the time of a transaction. The logic behind it is that a customer is expected to make transactions at similar hours. We, hence, propose a new method for creating features based on the periodic behavior of a transaction time, using the von Mises distribution [11]. In particular, these new time features should estimate if the time of a new transaction is within the confidence interval of the previous transaction time.

Furthermore, using a real credit card fraud dataset provided by a large European card processing company, we compare the different sets of features (raw, aggregated and periodic), using two kinds of classification algorithms; cost-insensitive [12] and example-dependent cost-sensitive [13]. The results show an average increase in the savings of 13% by using the proposed periodic features. Additionally, the outcome of this paper is being currently used to implement a state-of-the-art fraud detection system, that will help to combat fraud once the implementation stage is finished.

The remainder of the paper is organized as follows. In Section 2, we discuss current approaches to create the features used in fraud detection models. Then, in Section 3, we present our proposed methodology to create periodic features. Afterwards, the experimental setup and the results are given in Sections 4 and 5. Finally, conclusions and discussions of the paper are presented in Section 6.

II. Transaction aggregation strategies

When constructing a credit card fraud detection algorithm, the initial set of features (raw features) include information regarding individual transactions. It is observed throughout the literature, that regardless of the study, the set of raw features is quite similar. This is because the data collected during a credit card transaction must comply with international financial reporting standards. In TABLE I, the typical credit card fraud detection raw features are summarized.

Several studies use only the raw features in carrying their analysis [3], [4]. However, as noted in [14], a single transaction information is not sufficient to detect a fraudulent transaction, since using only the raw features leaves behind important information such as the consumer spending behavior, which is usually used by commercial fraud detection systems [10].

To deal with this, in [5], a new set of features were proposed such that the information of the last transaction made with the same credit card is also used to make a prediction.

TABLE I. SUMMARY OF TYPICAL RAW CREDIT CARD FRAUD DETECTION FEATURES

Attribute name	Description
Transaction ID	Transaction identification number
Time	Date and time of the transaction
Account number	Identification number of the customer
Card number	Identification of the credit card
Transaction type	ie. Internet, ATM, POS, ...
Entry mode	ie. Chip and pin, magnetic stripe, ...
Amount	Amount of the transaction in Euros
Merchant code	Identification of the merchant type
Merchant group	Merchant group identification
Country	Country of trx
Country 2	Country of residence
Type of card	ie. Visa debit, Mastercard, American Express...
Gender	Gender of the card holder
Age	Card holder age
Bank	Issuer bank of the card

The objective, is to be able to detect very dissimilar continuous transactions within the purchases of a customer. The new set of features include: time since the last transaction, previous amount of the transaction, previous country of the transaction. Nevertheless, these features do not take into account consumer behavior other than the last transaction made by a client, this leads to having an incomplete profile of customers.

A more compressive way to take into account a customer spending behavior is to derive some features using a transaction aggregation strategy. This methodology was initially proposed in [10]. The derivation of the aggregation features consists in grouping the transactions made during the last given number of hours, first by card or account number, then by transaction type, merchant group, country or other, followed by calculating the number of transactions or the total amount spent on those transactions. This methodology has been used by a number of studies [7]–[9], [15]–[18].

When aggregating a customer transactions, there is an important question on how much to accumulate, in the sense that the marginal value of new information may diminish as time passes. [10] discuss that aggregating 101 transactions is not likely to be more informative than aggregating 100 transactions. Indeed, when time passes, information lose their value, in the sense that a customer spending patterns are not expected to remain constant over the years. In particular, Whitrow et al. define a fixed time frame to be 24, 60 or 168 hours.

Let $\mathcal{S}$ be a set of N transactions, i.e., $N = |\mathcal{S}|$, where each transaction is represented by the feature vector $\mathbf{x}_i = [x_i^1, x_i^2, ..., x_i^k]$, where k is the number of features, and labelled using the class label $y_i \in \{0, 1\}$. Then, the process of aggregating features consists in selecting those transactions that were made in the previous t_p hours, for each transaction i in the dataset $\mathcal{S}$,

$$\mathcal{S}_{agg} \equiv TRX_{agg}(\mathcal{S}, i, t_p) = \left\{ x_l^{amt} \middle| \left(x_l^{id} = x_i^{id}\right) \wedge \left(hours(x_i^{time}, x_l^{time}) < t_p\right) \right\}_{l=1}^{N}, \quad (1)$$

where TRX_{agg} is a function that creates a subset of S associated with a transaction i with respect to the time frame t_p, $N = |\mathcal{S}|$, $|\cdot|$ being the cardinality of a set, x_i^{time} is the time of transaction i, x_i^{amt} is the amount of transaction i, x_i^{id} the customer identification number of transaction i, and

TABLE II. EXAMPLE CALCULATION OF AGGREGATED FEATURES. WHERE, x_i^{a1} IS THE NUMBER OF TRANSACTIONS IN THE LAST 24 HOURS AND x_i^{a2} IS THE SUM OF THE TRANSACTIONS AMOUNTS IN THE SAME TIME PERIOD.

Raw features						Agg. features	
TrxId	CardId	Time	Type	Country	Amt.	x_i^{a1}	x_i^{a2}
1	1	01/01 18:20	POS	Lux	250	0	0
2	1	01/01 20:35	POS	Lux	400	1	250
3	1	01/01 22:30	ATM	Lux	250	2	650
4	1	02/01 00:50	POS	Ger	50	3	900
5	1	02/01 19:18	POS	Ger	100	3	700
6	1	02/01 23:45	POS	Ger	150	2	150
7	1	03/01 06:00	POS	Lux	10	3	400

$hours(t_1, t_2)$ is a function that calculates the number of hours between the times t_1 and t_2. Afterwards the feature number of transactions and amount of transactions in the last t_p hours are calculated as:

$$x_i^{a1} = |\mathcal{S}_{agg}|, \quad (2)$$

and

$$x_i^{a2} = \sum_{x^{amt} \in \mathcal{S}_{agg}} x^{amt}, \quad (3)$$

respectively.

To further clarify how the aggregated features are calculated we show an example. Consider a set of transactions made by a client between the first and third of January of 2015, as shown in TABLE II. Then we estimate the aggregated features (x_i^{a1} and x_i^{a2}) by setting $t_p = 24$ hours. Moreover, the total number of aggregated features can grow quite quickly, as t_p can have several values, and the combination of combination criteria can be quite large as well. In [17], we used a total of 280 aggregated features. In particular we set the different values of t_p to: 1, 3, 6, 12, 18, 24, 72 and 168 hours. Then calculate the aggregated features using (1) with the following grouping criteria: country, type of transaction, entry mode, merchant code and merchant group.

III. PROPOSED PERIODIC FEATURES

When using the aggregated features, there is still some information that is not completely captured by those features. In particular we are interested in analyzing the time of the transaction. The logic behind this, is that a customer is expected to make transactions at similar hours. The issue when dealing with the time of the transaction, specifically, when analyzing a feature such as the mean of transactions time, is that it is easy to make the mistake of using the arithmetic mean. Indeed, the arithmetic mean is not a correct way to average time because, as shown in Fig. 1, it does not take into account the periodic behavior of the time feature. For example, the arithmetic mean of transaction time of four transactions made at 2:00, 3:00, 22:00 and 23:00 is 12:30, which is counter intuitive since no transaction was made close to that time.

We propose to overcome this limitation by modeling the time of the transaction as a periodic variable, in particular using the von Mises distribution [11]. The von Mises distribution, also known as the periodic normal distribution, is a distribution of a wrapped normal distributed variable across a circle. The von Mises probability distribution of a set of examples

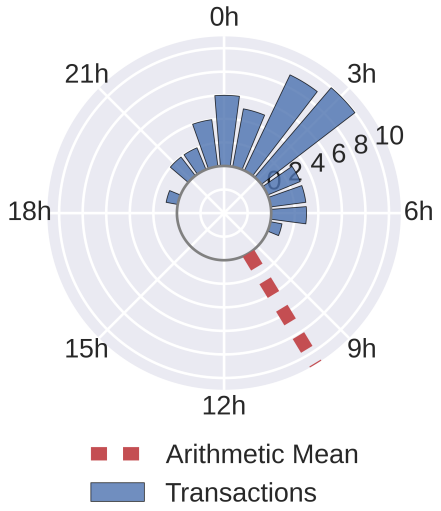

Fig. 1. Analysis of the time of a transaction using a 24 hour clock. The arithmetic mean of the transactions time (dashed line) do not accurately represents the actual times distribution.

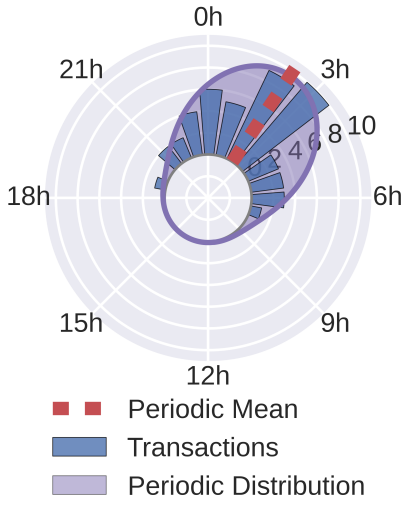

Fig. 2. Fitted von Mises distribution including the periodic mean (dashed line) and the probability distribution (purple area).

$D = \{t_1, t_2, \cdots, t_N\}$ for a given angle t_x is given by

$$f\left(t_x|\mu_{vM}, \sigma_{vM}\right) = \frac{e^{\frac{1}{\sigma_{vM}}\cos(t_x - \mu_{vM})}}{2\pi I_0\left(\frac{1}{\sigma_{vM}}\right)} \tag{4}$$

where $I_0(\kappa)$ is the modified Bessel function of order 0, and μ_{vM} and σ_{vM} are the periodic mean and periodic standard deviation, respectively. In Appendix A we present the calculation of μ_{vM} and σ_{vM}.

In particular, we are interested in calculating a confidence interval (CI) for the time of a transaction. For doing that, initially we select a set of transactions made by the same client in the last t_p hours ($\mathcal{S}_{per}$) as:

$$\mathcal{S}_{per} \equiv TRX_{vM}(\mathcal{S}, i, t_p) = \left\{ x_l^{time} \middle| \left(x_l^{id} = x_i^{id}\right) \wedge \left(hours(x_i^{time}, x_l^{time}) < t_p\right) \right\}_{l=1}^{N}. \tag{5}$$

Afterwards, the probability distribution function of the time of the set of transactions is calculated as:

$$x_i^{time} \sim vonmises\left(\mu_{vM}(\mathcal{S}_{per}), \frac{1}{\sigma_{vM}(\mathcal{S}_{per})}\right). \tag{6}$$

In Fig. 2, the von Mises distribution calculation for the earlier example is shown. It is observed that the arithmetic mean is different from the periodic mean, the latter being a more realistic representation of the actual transactional times. Then, using the estimated distribution, a new set of features can be extracted, i.e., a binary feature (x_i^{p1}) if a new transaction time is within the confidence interval range with probability α. An example is presented in Fig. 3. Furthermore, other features can be calculated, as the confidence interval range can be calculated for several values of α, and also the time period can have an arbitrary size.

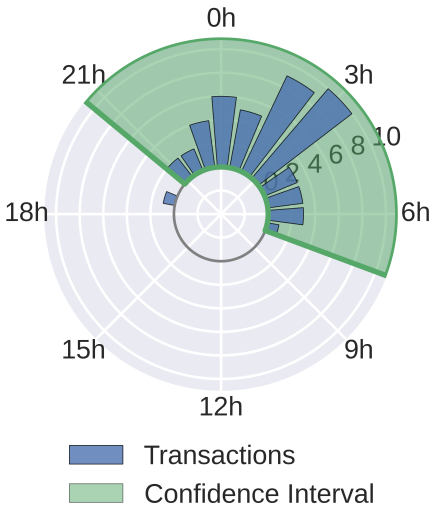

Fig. 3. Expected time of a transaction (green area). Using the confidence interval, a transaction can be flag normal or suspicious, depending whether or not the time of the transaction is within the confidence interval.

Additionally, following the same example presented in TABLE II, we calculate a feature x_i^{p1}, as a binary feature that takes the value of one if the current time of the transaction is within the confidence interval of the time of the previous transactions with a confidence of $\alpha = 0.9$. The example is shown in TABLE III, where the arithmetic and periodic means differ, as for the last transaction. Moreover, the new feature helps to get a better understanding of when a customer is expected to make transactions.

Finally, when calculating the periodic features, it is important to use longer time frames t_p, since if the distribution is calculated using only a couple of transactions it may not be as relevant of a customer behavior patterns, compared against using a full year of transactions. Evidently, if t_c is less than 24 hours, any transaction made afterwards will not be expected

TABLE III. EXAMPLE CALCULATION OF PERIODIC FEATURES. WHERE x_i^{p1} IS A BINARY FEATURE THAT INFORMS WHENEVER A TRANSACTION IS BEING MADE WITHIN THE CONFIDENCE INTERVAL OF THE TIME OF THE TRANSACTIONS.

Raw features		Arithmetic	Periodic features		
Id	**Time**	**mean**	**mean**	**Confidence interval**	x_i^{p1}
1	01/01/15 18:20	—	—	—	—
2	01/01/15 20:35	—	—	—	—
3	01/01/15 22:30	19:27	19:27	15:45 - 23:10	True
4	02/01/15 00:50	20:28	20:28	17:54 - 23:03	False
5	02/01/15 19:18	16:34	22:34	18:51 - 00:17	True
6	02/01/15 23:45	16:19	21:07	15:21 - 02:52	True
7	03/01/15 06:00	18:33	22:33	17:19 - 01:46	False

to be within the distribution of previous transactional times. To avoid this, we recommend using at least the previous 7 days of transactional information, therefore, having a better understanding of its behavioral patterns.

IV. EXPERIMENTAL SETUP

In this section, first the dataset used for the experiments is described. Second, the evaluation measure used to compare the algorithms is shown. Lastly, the algorithms used in this paper are briefly explained.

A. Database

For this paper we used a dataset provided by a large European card processing company. The dataset consists of fraudulent and legitimate transactions made with credit and debit cards between January 2012 and June 2013. The total dataset contains 120,000,000 individual transactions, each one with 27 attributes, including a fraud label indicating whenever a transaction is identified as fraud. This label was created internally in the card processing company, and can be regarded as highly accurate. In the dataset only 40,000 transactions were labeled as fraud, leading to a fraud ratio of 0.025%.

Furthermore, using the methodologies for feature extraction described in Section II, we estimate a total of 293 features. Also, for the experiments, a smaller subset of transactions with a higher fraud ratio, corresponding to transactions made with magnetic stripe, is selected. This dataset contains 236,735 transactions and a fraud ratio of 1.50%. In this dataset, the total financial losses due to fraud are 895,154 Euros. This dataset was selected because it is the one where most frauds occur.

The total dataset is divided into 3 subsets: training, validation and testing. Each one containing 50%, 25% and 25% of the transactions respectively. TABLE IV summarizes the different datasets.

B. Evaluation measure

When evaluating a credit card fraud detection model, typically a standard binary classification measure, such as misclassification error, receiver operating characteristic (ROC), Kolmogorov-Smirnov (KS) or F_1Score statistics, is used [9], [19], [20]. However, these measures may not be the most appropriate evaluation criteria when evaluating fraud detection models, because they tacitly assume that misclassification errors carry the same cost, similarly with the correct classified transactions. This assumption does not hold in practice, when wrongly predicting a fraudulent transaction as legitimate

TABLE IV. SUMMARY OF THE DATASETS

Set	**Transactions**	**%Frauds**	**Cost**
Total	236,735	1.50	895,154
Training	94,599	1.51	358,078
Validation	70,910	1.53	274,910
Testing	71,226	1.45	262,167

TABLE V. CREDIT CARD FRAUD COST MATRIX [17]

	Actual Positive $y_i = 1$	Actual Negative $y_i = 0$
Predicted Positive $c_i = 1$	$C_{TP_i} = C_a$	$C_{FP_i} = C_a$
Predicted Negative $c_i = 0$	$C_{FN_i} = Amt_i$	$C_{TN_i} = 0$

carries a significantly different financial cost than the inverse case.

In order to take into account the different costs of fraud detection during the evaluation of an algorithm, in [17], we proposed a cost matrix [13] that takes into account the actual example-dependent financial costs. In TABLE V, the cost matrix is presented, where the prediction of the algorithm c_i is a function of the k features of transaction i, $\mathbf{x}_i = [x_i^1, x_i^2, ..., x_i^k]$, y_i is the true class of the transaction i, and the cost associated with two types of correct classification, namely, true positives C_{TP_i}, and true negatives C_{TN_i}; and the two types of misclassification errors, namely, false positives C_{FP_i}, and false negatives C_{FN_i}, are presented. Moreover, our cost matrix defines the cost of a false negative to be the amount of the transaction Amt_i, and the costs of false positive and true positive to be the administrative cost C_a related to analyzing the transaction and contacting the card holder.

Afterwards, using the example-dependent cost matrix, a cost measure is calculated taking into account the actual costs of each transaction i. Let $\mathcal{S}$ be a set of N transactions, i.e., $N = |\mathcal{S}|$, where each transaction is represented by the augmented feature vector $\mathbf{x}_i^* = [\mathbf{x}_i, C_{TP_i}, C_{FP_i}, C_{FN_i}, C_{TN_i}]$, and labelled using the class label $y_i \in \{0, 1\}$. A classifier f which generates the predicted label c_i for each transaction i, is trained using the set $\mathcal{S}$. Then the cost of using f on $\mathcal{S}$ is calculated by

$$Cost(f(\mathcal{S})) = \sum_{i=1}^{N} y_i(1 - c_i)Amt_i + c_iC_a. \tag{7}$$

Lastly, in order to have a measure that is easy to interpret, we used the financial savings as we defined in [21]. In particular, the savings measure we proposed a savings measure that compare the cost of an algorithm versus the cost of using no algorithm at all. In the case of credit card fraud the cost of using no algorithm is equal to the sum of the amounts of the fraudulent transactions $\sum_{i=1}^{N} y_i Amt_i$. Then, the savings are calculated as:

$$Savings(f(\mathcal{S})) = \frac{\sum_{i=1}^{N} y_i c_i Amt_i - c_i C_a}{\sum_{i=1}^{N} y_i Amt_i}. \tag{8}$$

In other words, the sum of the amounts of the corrected predicted fraudulent transactions minus the administrative cost incurred in detecting them, divided by the sum of the amounts of the fraudulent transactions.

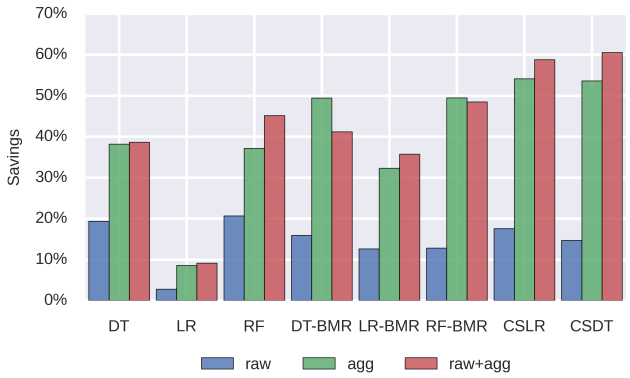

Fig. 4. Comparison of the different algorithms, trained with only the raw features (raw), only the aggregated features (agg) and both ($raw + agg$). In average, by using both the raw and the aggregated features the savings are doubled.

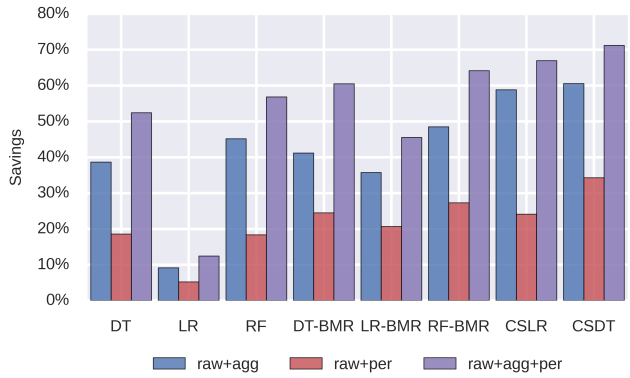

Fig. 5. Comparison of the proposed periodic (per) set of features. It is observed, that when the new set of features are combined with the aggregated features, an additional increase of savings of 16.4% is made.

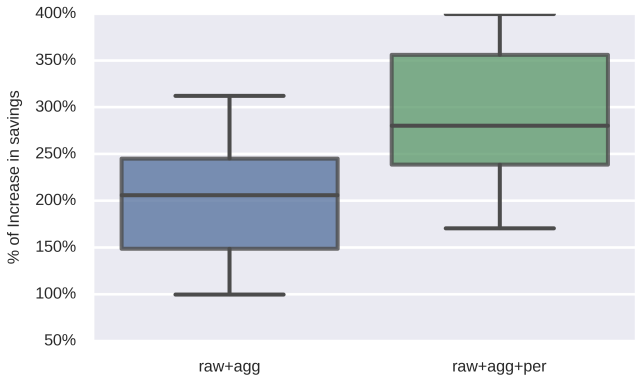

Fig. 6. Comparison of the average increase in savings when introducing each set of features compared with the results of using only the raw set of features.

C. Algorithms

For the experiments we used three cost-insensitive classification algorithms: decision tree (DT), logistic regression (LR) and a random forest (RF), using the implementation of Scikit-learn [22]. Furthermore, in previous works, we have shown the need to use algorithms that take into account the different costs associated with fraud detection [18]. In particular we also used three cost-sensitive algorithms, namely, Bayes minimum risk (BMR) [17], [18], cost-sensitive logistic regression ($CSLR$) [23] and cost-sensitive decision tree ($CSDT$) [21].

The BMR is a decision model based on quantifying tradeoffs between various decisions using probabilities and the costs that accompany such decisions. In the case of credit card fraud detection, a transaction is classified as fraud if $C_a \leq Amt_i \cdot \hat{p}_i$, and as legitimate if false. Where $\hat{p}_i$ is the estimated probability of a transaction being fraud given $\mathbf{x}_i^*$. The other two cost-sensitive methods, $CSLR$ and $CSDT$, are based on introducing the example-dependent costs into a logistic regression and a decision tree algorithm, by changing the objective function of the models to one that is cost-sensitive. For a further discussion see [23] and [21], respectively.

The implementation of the cost-sensitive algorithms is done using the *CostCla*[1] library. Moreover, each algorithm was trained, using the different sets of features: raw features (raw) as shown in TABLE I, aggregated features (agg) using equations (2) and (3), and the periodic features (per) as described in Section III.

V. Results

First we evaluate the savings of the different algorithms using only the raw features (raw), only the aggregated features (agg) and both ($raw + agg$). The results are shown in Fig. 4. Note that all the algorithms generate savings, i.e., no algorithm performs worse than using no algorithm at all. The $CSDT$ algorithm is the one that performs best, in particular when using both the raw and aggregated features. When analyzing the results using the different set of features, the aggregated features perform better than using only the raw features in all the cases. This confirms the intuition of the need of using the customer behavior patterns in order to identify fraudulent transactions. On average, by using both the raw and the aggregated features the savings are doubled.

Then, we evaluate the results of the periodic set of features. In Fig. 5, the results are shown. The new set of periodic features increase the savings by an additional 13%. The algorithm with the highest savings is the $CSDT$, closely followed by $CSLR$. Similarly to using the extended aggregated features, the periodic features do not perform well when used only with raw features. It is when combined the set of aggregated features that an increase in savings is found.

Finally, in Fig. 6, we compare the average increase in savings when introducing each set of features compared with the results of using only the set of raw features. First, the aggregated features give an average increase in savings of 201%. As previously shown, in order to improve the results, these new features need to be combined with the aggregated features in order to increase savings. Lastly, when combining the previous features with the periodic features, the results increase by 287% compared with using raw features only.

[1] https://github.com/albahnsen/CostSensitiveClassification

VI. Conclusion and discussion

In this paper we have shown the importance of using features that analyze the consumer behavior of individual card holders when constructing a credit card fraud detection model. We show that by preprocessing the data in order to include the recent consumer behavior, the performance increases by more than 200% compared to using only the raw transaction information.

Moreover, we extended the current approaches to analyze the consumer behavior by proposing a new method to analyze the periodic behavior of the time of a transaction using the von Mises distribution. The new proposed set of features increases the performance by 287%.

However, because this study was done using a dataset from a financial institution, we were not able to deeply discuss the specific features created, and the individual impact of each feature. Nevertheless, our framework is ample enough to be recreated with any kind of transactional data. Furthermore, when implementing this framework on a production fraud detection system, questions regarding response and calculation time of the different features should be addressed. In particular, since there is no limit on the number of features that can be calculated, a system may take too long to make a decision based on the time spent recalculating the features with each new transaction.

Acknowledgment

Funding for this research was provided by the Fonds National de la Recherche, Luxembourg, grant number AFR-PhD-5942749.

References

[1] European Central Bank, "Third report on card fraud," European Central Bank, Tech. Rep., 2014.

[2] 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.

[3] R. Brause, T. Langsdorf, and M. Hepp, "Neural data mining for credit card fraud detection," *Proceedings 11th International Conference on Tools with Artificial Intelligence*, pp. 103–106, 1999.

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

[5] S. Bachmayer, "Artificial Immune Systems," *Artificial Immune Systems*, vol. 5132, pp. 119–131, 2008.

[6] M. Krivko, "A hybrid model for plastic card fraud detection systems," *Expert Systems with Applications*, vol. 37, no. 8, pp. 6070–6076, Aug. 2010.

[7] 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, Feb. 2011.

[8] D. J. Weston, D. J. Hand, N. M. Adams, C. Whitrow, and P. Juszczak, "Plastic card fraud detection using peer group analysis," *Advances in Data Analysis and Classification*, vol. 2, no. 1, pp. 45–62, Mar. 2008.

[9] A. D. Pozzolo, O. Caelen, Y.-A. Le Borgne, S. Waterschoot, and G. Bontempi, "Learned lessons in credit card fraud detection from a practitioner perspective," *Expert Systems with Applications*, vol. 41, no. 10, pp. 4915–4928, Aug. 2014.

[10] C. Whitrow, D. J. Hand, P. Juszczak, D. J. 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, Jul. 2008.

[11] N. I. Fisher, *Statistical Analysis of Circular Data*, 1996, vol. 9.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*, ser. Information science and statistics. Springer, 2006, vol. 4, no. 4.

[13] C. Elkan, "The Foundations of Cost-Sensitive Learning," in *Seventeenth International Joint Conference on Artificial Intelligence*, 2001, pp. 973–978.

[14] R. Bolton and D. J. Hand, "Unsupervised profiling methods for fraud detection," in *Credit Scoring and Credit Control VII*, 2001.

[15] D. Tasoulis and N. Adams, "Mining information from plastic card transaction streams," in *Proceedings in 18th International Conference on Computational Statistics*, 2008.

[16] 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.

[17] A. Correa Bahnsen, A. Stojanovic, D. Aouada, and B. Ottersten, "Cost Sensitive Credit Card Fraud Detection Using Bayes Minimum Risk," in *2013 12th International Conference on Machine Learning and Applications*. Miami, USA: IEEE, Dec. 2013, pp. 333–338.

[18] ——, "Improving Credit Card Fraud Detection with Calibrated Probabilities," in *Proceedings of the fourteenth SIAM International Conference on Data Mining*, Philadelphia, USA, 2014, pp. 677 – 685.

[19] R. J. Bolton, D. J. Hand, F. Provost, and L. Breiman, "Statistical Fraud Detection: A Review," *Statistical Science*, vol. 17, no. 3, pp. 235–255, 2002.

[20] D. J. Hand, C. Whitrow, N. M. Adams, P. Juszczak, and D. J. Weston, "Performance criteria for plastic card fraud detection tools," *Journal of the Operational Research Society*, vol. 59, no. 7, pp. 956–962, May 2007.

[21] A. Correa Bahnsen, D. Aouada, and B. Ottersten, "Example-Dependent Cost-Sensitive Decision Trees," *Expert Systems with Applications*, vol. 42, no. 19, pp. 6609–6619, 2015.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[23] A. Correa Bahnsen, D. Aouada, and B. Ottersten, "Example-Dependent Cost-Sensitive Logistic Regression for Credit Scoring," in *2014 13th International Conference on Machine Learning and Applications*. Detroit, USA: IEEE, 2014, pp. 263–269.

Appendix

The von Mises distribution, also known as the periodic normal distribution, is a distribution of a wrapped normal distributed variable across a circle [11]. The von Mises distribution of a set of examples $D = \{t_1, t_2, \cdots, t_N\}$ is defined as $D \sim vonmises\left(\mu_{vM}, 1/\sigma_{vM}\right)$, where μ_{vM} and σ_{vM} are the periodic mean and periodic standard deviation, respectively, and are calculated as follows [12]

$$\mu_{vM}(D) = 2\tan^{-1}\left(\frac{\phi}{\left(\sqrt{\psi^2+\phi^2}+\phi\right)}\right), \tag{9}$$

and

$$\sigma_{vM}(D) = \sqrt{ln\left(\frac{1}{\left(\frac{1}{N}\phi\right)^2+\left(\frac{1}{N}\psi\right)^2}\right)}, \tag{10}$$

where $\phi = \sum_{t_j \in D} \sin(t_j)$ and $\psi = \sum_{t_j \in D} \cos(t_j)$.