Feature Engineering Strategies for Credit Card Fraud Detection

Article *in* Expert Systems with Applications · January 2016

DOI: 10.1016/j.eswa.2015.12.030

4 authors, including:

Alejandro Correa Bahnsen
University of Luxembourg
11 PUBLICATIONS **38** CITATIONS
SEE PROFILE

Djamila Aouada
University of Luxembourg
57 PUBLICATIONS **186** CITATIONS
SEE PROFILE

Björn Ottersten
University of Luxembourg
646 PUBLICATIONS **12,751** CITATIONS
SEE PROFILE

Expert Systems With Applications 51 (2016) 134–142

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Feature engineering strategies for credit card fraud detection

Alejandro Correa Bahnsen*, Djamila Aouada, Aleksandar Stojanovic, Björn Ottersten

Interdisciplinary Centre for Security, Reliability and Trust, University of Luxembourg, Luxembourg

ARTICLE INFO

Keywords:
Cost-sensitive learning
Fraud detection
Preprocessing
Von Mises distribution

ABSTRACT

Every year billions of Euros are lost worldwide due to credit card fraud. Thus, forcing financial institutions to continuously improve their fraud detection systems. In recent years, several studies have proposed the use of machine learning and data mining techniques to address this problem. However, most studies used some sort of misclassification measure to evaluate the different solutions, and do not take into account the actual financial costs associated with the fraud detection process. Moreover, when constructing a credit card fraud detection model, it is very important how to extract the right features from the 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 expand the transaction aggregation strategy, and 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. Then, 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%.

1. Introduction

The use of credit and debit cards has increased significantly in the last years, unfortunately so has fraud. Because of that, billions of Euros are lost every year. According to the European Central Bank (European Central Bank, 2014), 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. Moreover, payments across non traditional channels (mobile, internet, etc.) 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 (Van Vlasselaer et al., 2015), moreover, machine learning methods as well can be inadequate if they miss to adapt to new fraud strategies, i.e., static models that are never updated (Dal Pozzolo, Caelen, Le Borgne, Waterschoot, & Bontempi, 2014).

The use of machine learning in fraud detection has been an interesting topic in recent years. Several detection systems based on machine learning techniques have been successfully used for this problem (Bhattacharyya, Jha, Tharakunnel, & Westland, 2011). When constructing a credit card fraud detection model, there are several factors that have an important impact during the training phase: Skewness of the data , cost-sensitivity of the application, short-time response of the system, dimensionality of the search space and how to preprocess the features (Bachmayer, 2008; Bolton, Hand, Provost, & Breiman, 2002; Dal Pozzolo et al., 2014; Van Vlasselaer et al., 2015; Whitrow, Hand, Juszczak, Weston, & Adams, 2008). In this paper, we address the cost-sensitivity and the features preprocessing to achieve improved fraud detection and savings.

Credit card fraud detection is by definition a cost-sensitive problem, in the sense that the cost due to a false positive is different than the cost of a false negative. When predicting a transaction as fraudulent, when in fact it is not a fraud, there is an administrative cost that is incurred by the financial institution. On the other hand, when failing to detect a fraud, the amount of that transaction is lost (Hand, Whitrow, Adams, Juszczak, & Weston, 2007). Moreover, it is not enough to assume a constant cost difference between false positives and false negatives, as the amount of the transactions varies quite significantly; therefore, its financial impact is not constant but depends on each transaction. In Correa Bahnsen, Stojanovic, Aouada, and Ottersten (2013), we proposed a new cost-based measure to evaluate credit card fraud detection models, taking into account the different financial costs incurred by the fraud detection process.

* Corresponding author. Tel.: +57 3045462842.
E-mail addresses: al.bahnsen@gmail.com (A. Correa Bahnsen), djamila.aouada@uni.lu (D. Aouada), aleksandar.stojanovic@rwth-aachen.de (A. Stojanovic), bjorn.ottersten@uni.lu (B. Ottersten).

http://dx.doi.org/10.1016/j.eswa.2015.12.030
0957-4174/

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 (Bachmayer, 2008). A standard way to include these behavioral spending patterns is proposed in (Whitrow et al., 2008), 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 first propose a new savings measure based on comparing the financial cost of an algorithm versus using no model at all. Then, we propose an expanded version of the transaction aggregation strategy, by incorporating a combination criteria when grouping transactions, i.e., instead of aggregating only by card holder and transaction type, we combine it with country or merchant group. This allows to have a much richer feature space.

Moreover, we also propose a new method for extracting periodic features in order to estimate if the time of a new transaction is within the confidence interval of the previous transaction times. The motivation is that a customer is expected to make transactions at similar hours. The proposed methodology is based on analyzing the periodic behavior of a transaction time, using the von Mises distribution (Fisher, 1995).

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, extended aggregated and periodic), using two kind of classification algorithms; cost-insensitive (Hastie, Tibshirani, & Friedman, 2009) and example-dependent cost-sensitive (Elkan, 2001). 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 explain the background on credit card fraud detection, and specifically the measures to evaluate a fraud detection model. Then in Section 3, we discuss current approaches to create the features used in fraud detection models, moreover, we present our proposed methodology to create periodic based features. Afterwards, the experimental setup is given in Section 4. In Section 5, the results are shown. Finally, conclusions and discussions of the paper are presented in Section 6.

2. Credit card fraud detection evaluation

A credit card fraud detection algorithm consists in identifying those transactions with a high probability of being fraud, based on historical fraud patterns. 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 (Maes, Tuyls, Vanschoenwinkel, & Manderick, 2002), Bayesian learning (Maes et al., 2002), artificial immune systems (Bachmayer, 2008), association rules (Sánchez, Vila, Cerda, & Serrano, 2009), hybrid models (Krivko, 2010), support vector machines (Bhattacharyya et al., 2011), peer group analysis (Weston, Hand, Adams, Whitrow, & Juszczak, 2008), random forest (Correa Bahnsen et al., 2013; Dal Pozzolo et al., 2014), discriminant analysis (Mahmoudi & Duman, 2015) and social network analysis (Van Vlasselaer et al., 2015).

Table 1
Classification confusion matrix.

	Actual positive $y = 1$	Actual negative $y = 0$
Predicted positive $c = 1$	True positive (*TP*)	False positive (*FP*)
Predicted negative $c = 0$	False negative (*FN*)	True positive (*TN*)

Table 2
Cost matrix (Elkan, 2001).

	Actual positive $y_i = 1$	Actual negative $y_i = 0$
Predicted positive $c_i = 1$	C_{TP_i}	C_{FP_i}
Predicted negative $c_i = 0$	C_{FN_i}	C_{TN_i}

Most of these studies compare their proposed algorithms with a benchmark algorithm and then make the comparison using a standard binary classification measure, such as misclassification error, receiver operating characteristic (*ROC*), Kolmogorov–Smirnov (*KS*), $F_1 Score$ (Bolton et al., 2002; Hand et al., 2007) or AUC statistics (Dal Pozzolo et al., 2014). Most of these measures are extracted by using a confusion matrix as shown in Table 1, where the prediction of the algorithm c_i is a function of the k features of transaction i, $\mathbf{x}_i = [x_i^1, x_i^2, \ldots, x_i^k]$ and y_i is the true class of the transaction i.

From this table, several statistics are extracted. In particular:

- $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
- $Recall = \frac{TP}{TP+FN}$
- $Precision = \frac{TP}{TP+FP}$
- $F_1 Score = 2\frac{Precision \cdot Recall}{Precision+Recall}$

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 carries a significantly different financial cost than the inverse case. Furthermore, the accuracy measure also assumes that the class distribution among transactions is constant and balanced (Provost, Fawcett, & Kohavi, 1998), and typically the distributions of a fraud detection dataset are skewed, with a percentage of frauds ranging from 0.005% to 0.5% (Bachmayer, 2008; Bhattacharyya et al., 2011).

In order to take into account the different costs of fraud detection during the evaluation of an algorithm, we may use the modified cost matrix defined in (Elkan, 2001). In Table 2, the cost matrix is presented, where the cost assof 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. This is an extension of Table 1, but in this case the costs are example-dependent, in other words, specific to each transaction i.

Hand et al. (Hand et al., 2007) proposed a cost matrix, where in the case of false positive the associated cost is the administrative cost $C_{FP_i} = C_a$ related to analyzing the transaction and contacting the card holder. This cost is the same assigned to a true positive $C_{TP_i} = C_a$, because in this case, the card holder will have to be contacted. However, in the case of a false negative, in which a fraud is not detected, the cost is defined to be a hundred times larger,

Table 3
Credit card fraud cost matrix (Correa Bahnsen et al., 2013).

	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 $c_i = 0$	$C_{FN_i} = Amt_i$	$C_{TN_i} = 0$

i.e. $C_{FN_i} = 100C_a$. This same approach was also used in (Bachmayer, 2008).

Nevertheless, in practice, losses due to a specific fraud range from few to thousands of Euros, which means that assuming constant cost for false negatives is unrealistic. In order to address this limitation, in Correa Bahnsen et al. (2013), we proposed a cost matrix that takes into account the actual example-dependent financial costs. Our cost matrix defines the cost of a false negative to be the amount $C_{FN_i} = Amt_i$ of the transaction i. We argue that this cost matrix is a better representation of the actual costs, since in practice fraud detection teams are measured by either by total monetary savings or total amount saved, while it may be of interest to a financial institution to minimize false positives, the ultimately goal of the company is to maximize profits which is better addressed by the minimization of the financial costs. The costs are summarized in Table 3.

Moreover, this framework is flexible enough to include additional costs such as one that takes into account the expected intangible cost by an irritated customer due to a false positive, or on the other hand, the profit due to a satisfy customer that feels safe by being contacted by the bank.

Afterwards, using the example-dependent cost matrix, a cost measure is calculated taking into account the actual costs $[C_{TP_i}, C_{FP_i}, C_{FN_i}, C_{TN_i}]$ of each transaction i. Let $\mathcal{S}$ be a set of N transactions i, $N = |\mathcal{S}|$, 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

$$\begin{aligned} Cost(f(\mathcal{S})) &= \sum_{i=1}^{N} \Big(y_i(c_i C_{TP_i} + (1-c_i)C_{FN_i}) \\ &\quad + (1-y_i)(c_i C_{FP_i} + (1-c_i)C_{TN_i}) \Big). \\ &= \sum_{i=1}^{N} y_i(1-c_i)Amt_i + c_i C_a. \end{aligned} \tag{1}$$

However, as noted in (Whitrow et al., 2008), the total cost may not be easy to interpret. So Whitrow et al. proposed a normalized cost measure by dividing the total cost by the theoretical maximum cost, which is the cost of misclassifying every example.

$$Cost_n(f(\mathcal{S})) = \frac{\sum_{i=1}^{N} y_i(1-c_i)Amt_i + c_i C_a}{|\mathcal{S}_0|C_a + \sum_{i=1}^{N} Amt_i \cdot \mathbf{1}_1(y_i)}, \tag{2}$$

where, $\mathcal{S}_0 = \{\mathbf{x}_i^* | y_i = 0, i \in 1, \ldots, N\}$, and $\mathbf{1}_c(z)$ is an indicator function that takes the value of one if $z = c$ and zero if $z \neq c$.

We propose a similar approach in Correa Bahnsen, Aouada, and Ottersten (2015), by defining the savings of using an algorithm as the cost of the algorithm versus the cost of using no algorithm at all. To do that, we set the cost of using no algorithm as

$$Cost_l(\mathcal{S}) = \min\{Cost(f_0(\mathcal{S})), Cost(f_1(\mathcal{S}))\}, \tag{3}$$

where f_0 refers to a classifier that predicts all the examples in $\mathcal{S}$ as belonging to the class c_0, and similarly f_1 refers to a classifier that predicts all the examples in $\mathcal{S}$ as belonging to the class c_1, the cost

Table 4
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

improvement can be expressed as the cost savings as compared with $Cost_l(\mathcal{S})$.

$$Savings(f(\mathcal{S})) = \frac{Cost_l(\mathcal{S}) - Cost(f(\mathcal{S}))}{Cost_l(\mathcal{S})}. \tag{4}$$

Moreover, 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 $Cost_l(\mathcal{S}) = \sum_{i=1}^{N} y_i Amt_i$. Then, the savings are calculated as:

$$Savings(f(\mathcal{S})) = \frac{\sum_{i=1}^{N} y_i c_i Amt_i - c_i C_a}{\sum_{i=1}^{N} y_i Amt_i}. \tag{5}$$

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

For our analysis, we choose to use the savings measure instead of the normalized cost, since in the field of credit card fraud detection, a general observation is that companies do not use predictive models. Therefore, the savings measure makes more sense for this application. Indeed the savings measure may lead to negative values which is counterintuitive, however, in the industry it makes sense to compare the results of the algorithm versus not using any algorithm at all.

Lastly, it may be argued that this example-dependent strategy is focusing solely on large amount transaction and that smaller frauds would not matter. However, this framework is flexible enough to allow modifying the cost matrix to include the available amount in the credit card as the cost of a false negative. Then, small amount frauds with a high potential loss would have a higher importance, because a lot of money is available in the credit card.

3. Feature engineering for fraud detection

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 (American Institute of CPAs, 2011). In Table 4, the typical credit card fraud detection raw features are summarized.

3.1. Capturing customer spending patterns

Several studies use only the raw features in carrying their analysis (Brause, Langsdorf, & Hepp, 1999; Minegishi & Niimi, 2011; Panigrahi, Kundu, Sural, & Majumdar, 2009; Sánchez et al., 2009).

However, as noted in (Bolton & Hand, 2001), 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 (Whitrow et al., 2008).

To deal with this, in (Bachmayer, 2008), 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. 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 (Whitrow et al., 2008). 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 (Bhattacharyya et al., 2011; Correa Bahnsen et al., 2013; 2014b; Dal Pozzolo et al., 2014; Jha, Guillen, & Christopher Westland, 2012; Sahin, Bulkan, & Duman, 2013; Tasoulis & Adams, 2008; Weston et al., 2008).

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. Whitrow et al. (2008) 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 h.

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 $\mathcal{S}$,

$$\mathcal{S}_{agg} \equiv TRX_{agg}(\mathcal{S}, i, t_p) = \left\{ x_l^{amt} \middle| \left(x_l^{id} = x_i^{id}\right) \wedge \left(hours(x_i^{time}, x_l^{time}) < t_p\right) \right\}_{l=1}^{N}, \quad (6)$$

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 $hours(t_1, t_2)$ 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:

$$x_i^{a1} = |\mathcal{S}_{agg}|, \quad (7)$$

and

$$x_i^{a2} = \sum_{x^{amt} \in \mathcal{S}_{agg}} x^{amt}, \quad (8)$$

respectively.

We note that this aggregation is not enough, in the sense that the combination of different features is not being taken into account. For example, it is not only interesting to see the total transactions, but also group them following a certain criteria, such as: transactions made in the last t_p hours, in the same country and of the same transaction type. For calculating such features, first we expand (6) as follows

$$\mathcal{S}_{agg2} \equiv TRX_{agg}(\mathcal{S}, i, t_p, cond_1, cond_2) = \left\{ x_l^{amt} \middle| \left(x_l^{id} = x_i^{id}\right) \wedge \left(hours(x_i^{time}, x_l^{time}) < t_p\right) \wedge \left(x_l^{cond_1} = x_i^{cond_1}\right) \wedge \left(x_l^{cond_2} = x_i^{cond_2}\right) \right\}_{l=1}^{N}, \quad (9)$$

where, $cond_1$ and $cond_2$, could be either of the features of a transaction listed in Table 4. Then, the features are calculated as:

$$x_i^{a3} = |\mathcal{S}_{agg2}|, \quad (10)$$

and

$$x_i^{a4} = \sum_{x^{amt} \in \mathcal{S}_{agg2}} x^{amt}. \quad (11)$$

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 5. Then we estimate the aggregated features (x_i^{a1}, x_i^{a2}, x_i^{a3} and x_i^{a4}) by setting $t_p = 24$ h. The different aggregated features give us different information of the customer spending behavior. 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 Correa Bahnsen et al. (2013), 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 h. Then calculate the aggregated features using (6), and also using (9) with the following grouping criteria: country, type of transaction, entry mode, merchant code and merchant group.

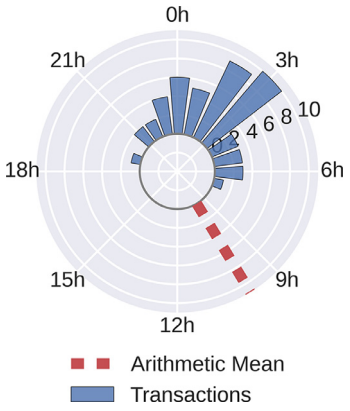

Fig. 1. Analysis of the time of a transaction using a 24 h clock. The arithmetic mean of the transactions time (dashed line) do not accurately represents the actual times distribution.

3.2. Time 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.

Table 5
Example calculation of aggregated features. Where, x_i^{a1} is the number of transactions in the last 24 h, x_i^{a2} is the sum of the transactions amounts in the same time period, x_i^{a3} is the number of transactions with the same transaction type and same country in the last 24 h and x_i^{a4} is the sum of the transactions amounts of the transactions with the same type and country in the last 24 h.

Raw features						Aggregated features			
TrxId	CardId	Time	Type	Country	Amount	x_i^{a1}	x_i^{a2}	x_i^{a3}	x_i^{a4}
1	1	01/01/15 18:20	POS	Luxembourg	250	0	0	0	0
2	1	01/01/15 20:35	POS	Luxembourg	400	1	250	1	250
3	1	01/01/15 22:30	ATM	Luxembourg	250	2	650	0	0
4	1	02/01/15 00:50	POS	Germany	50	3	900	0	0
5	1	02/01/15 19:18	POS	Germany	100	3	700	1	50
6	1	02/01/15 23:45	POS	Germany	150	2	150	2	150
7	1	03/01/15 00:00	POS	Luxembourg	10	3	400	0	0

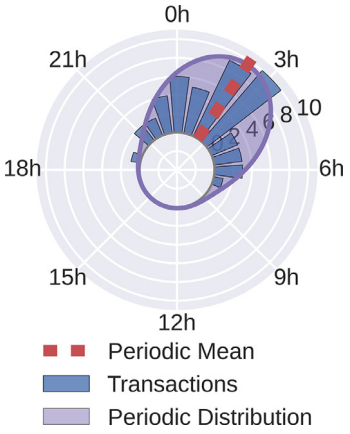

Fig. 2. Fitted von Mises distribution including the periodic mean (dashed line) and the probability distribution (purple area). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

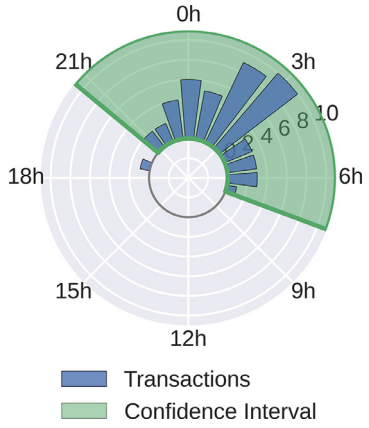

Fig. 3. Expected time of a transaction (green area). Using the confidence interval, a transaction can be flag normal or suspicious, depending whether or not the time of the transaction is within the confidence interval. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

We propose to overcome this limitation by modeling the time of the transaction as a periodic variable, in particular using the von Mises distribution (Fisher, 1995). 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 distribution of a set of examples $D = \{t_1, t_2, \cdots, t_N\}$ is defined as

$$D \sim vonmises\left(\mu_{vM}, \frac{1}{\sigma_{vM}}\right), \tag{12}$$

where 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,

$$\begin{aligned} \mathcal{S}_{per} &\equiv TRX_{vM}(\mathcal{S}, i, t_p) \\ &= \left\{ x_l^{time} \middle| \left(x_l^{id} = x_i^{id}\right) \wedge \left(hours(x_i^{time}, x_l^{time}) < t_p\right) \right\}_{l=1}^{N}. \end{aligned} \tag{13}$$

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

$$x_i^{time} \sim vonmises\left(\mu_{vM}(\mathcal{S}_{per}), \frac{1}{\sigma_{vM}(\mathcal{S}_{per})}\right). \tag{14}$$

In Fig. 2, the von Mises distribution calculation for the earlier example is shown. It is observed that the arithmetic mean is quite 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, ie., 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.

Additionally, following the same example presented in Table 5, 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 6, where the arithmetic and periodic means differ, as for the last transaction both means are significantly different. Moreover, the new feature helps to get a better understanding of when a customer is expected to make transactions.

Table 6
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

Table 7
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

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_p is less than 24 h, any transaction made afterwards will not be expected 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. Lastly, this approach can also be used to estimate features such as the expected day of the week of transactions, as some customers may only use their credit cards during the weekend nights, or during working hours.

4. Experimental setup

In this section, first the dataset used for the experiments is described. Afterwards, the partitioning of the dataset is presented. Lastly, the algorithms used to detect fraud are shown.

4.1. 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 3, we estimate a total of 293 features. Also, for the experiments, a smaller subset of transactions with a higher fraud ratio, corresponding to a specific group of transactions, 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.

4.2. Database partitioning

From the total dataset, 3 different datasets are extracted: training, validation and testing. Each one containing 50%, 25% and 25% of the transactions respectively. Table 7 summarizes the different datasets.

4.3. 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 (Pedregosa et al., 2011). We also used the Bayes minimum risk (*BMR*) model we proposed in Correa Bahnsen, Stojanovic, Aouada, and Ottersten (2014b). 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 the following condition holds true:

$$C_a P(p_f|\mathbf{x}_i^*) + C_a P(p_l|\mathbf{x}_i^*) \leq Amt_i P(p_f|\mathbf{x}_i^*), \tag{15}$$

and as legitimate if not, where $P(p_l|\mathbf{x}_i^*)$ is the estimated probability of a transaction being legitimate given $\mathbf{x}_i^*$. Similarly $P(p_f|\mathbf{x}_i^*)$ is the probability of a transaction being fraud given $\mathbf{x}_i^*$. An extensive description of the methodology can be found in Correa Bahnsen et al. (2013).

Additionally, we used the cost-sensitive logistic regression algorithm, proposed in Correa Bahnsen, Aouada, and Ottersten (2014a). This method introduces the example-dependent costs into a logistic regression, by changing the objective function of the model to one that is cost-sensitive. The new cost function is defined as:

$$J^c(\theta) = \frac{1}{N}\sum_{i=1}^{N}\Big(y_i(h_\theta(X_i)C_{TP_i} + (1-h_\theta(X_i))C_{FN_i}) + (1-y_i)(h_\theta(X_i)C_{FP_i} + (1-h_\theta(X_i))C_{TN_i})\Big), \tag{16}$$

where $h_\theta(X_i) = g(\sum_{j=1}^{k}\theta^j x_i^j)$ refers to the hypothesis of i given the parameters θ, and $g(\cdot)$ is the logistic sigmoid function, defined as $g(z) = 1/(1+e^{-z})$. To find the coefficients of the regression θ, the cost function is minimized by using binary genetic algorithms.

Lastly, we used a cost-sensitive decision tree algorithm, proposed in (Correa Bahnsen et al., 2015). In this method a new splitting criteria is used during the tree construction. In particular instead of using a traditional splitting criteria such as Gini, entropy or misclassification, the *Cost* as defined in (2), of each tree node is calculated, and the gain of using each split evaluated as the decrease in total *Savings* of the algorithm.

Finally, because these algorithms suffer when the label distribution is skewed towards one of the classes (Hastie et al., 2009), it is common to perform sampling procedures in order to have a more balanced class distribution (Hulse & Khoshgoftaar, 2007). However, in previous studies we investigated the impact of sampling on the fraud detection models, and indeed neither under-sampling or over-sampling improved the results measured by financial savings, as the distribution of the example-dependent costs is modified by the sampling procedures (Correa Bahnsen et al., 2013; 2014b).

5. Results

We estimated the different algorithms presented in Section 4.3, in particular: decision tree (*DT*), logistic regression (*LR*), random forest (*RF*), with and without the Bayes minimum risk threshold (*BMR*), and the cost-sensitive algorithms, cost-sensitive logistic regression (*CSLR*) and cost-sensitive decision tree (*CSDT*). Moreover, each algorithm was trained, using the different sets of features: raw features (*raw*) as shown in Table 4, aggregated features (*agg*1) using Eqs. (7) and (8), expanded aggregated features (*agg*2) using Eqs. (10) and (11), lastly the periodic features (*per*) as described in Section 3.2.

In Fig. 4, we present the results measured by savings, see (5), of the different algorithms using only the raw features (*raw*), only the aggregated features (*agg*1) and both (*raw* + *agg*1). First, 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.

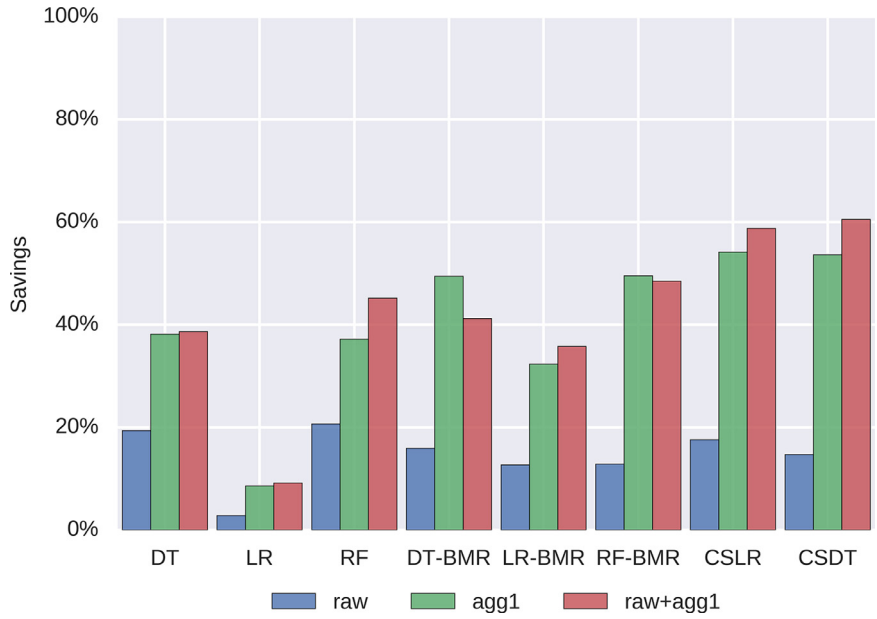

Fig. 4. Comparison of the different algorithms, trained with only the raw features (*raw*), only the aggregated features (*agg*1) and both ($raw + agg1$). In average, by using both the raw and the aggregated features the savings are doubled.

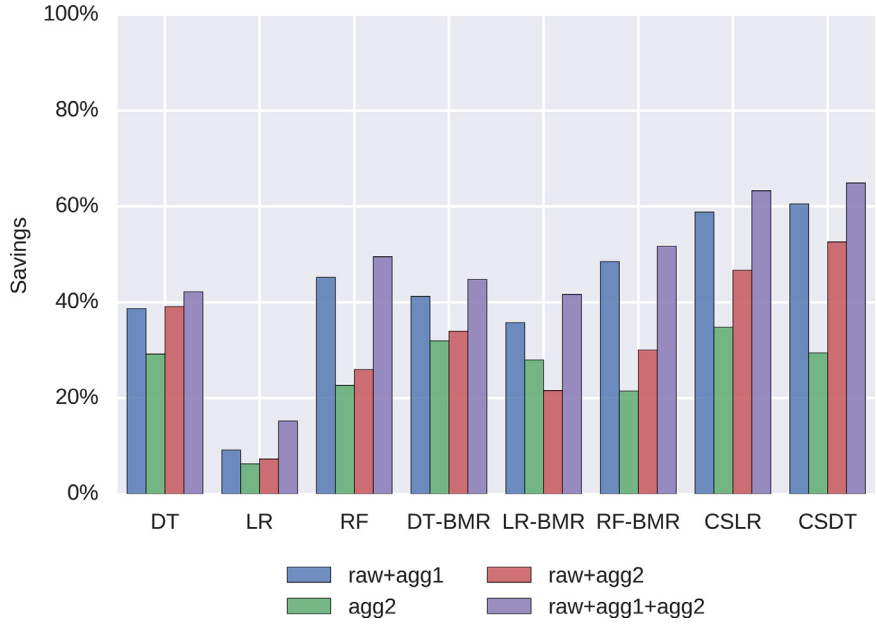

Fig. 5. Comparison of the extended aggregated (*agg*2) set of features. It is observed, that when the proposed expanded aggregated features are combined with the raw and aggregated features, an increase in savings of 16.4% is made.

Afterwards, we evaluate the results when using our proposed expanded aggregated features (*agg*2). In Fig. 5, the results are shown. It is observed that when comparing the $raw + agg1$ and $raw + agg2$, the results of the new features are worse than the traditional aggregated features. However, when the proposed expanded aggregated features are combined with the raw and aggregated features, an increase of 16.4% in savings is made. Therefore, there is a need to use the proposed extended aggregated features with the standard aggregated features in order to improve the results of the algorithms. Moreover, in all cases the use of the raw and both aggregated set of features perform better than using only the raw and aggregated features.

Then, we evaluate the results of the periodic set of features. In Fig. 6, 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 with both sets of aggregated features that an increase in savings is found.

Finally, in Fig. 7, we compared 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 standard aggregated features give an average increase in savings of 201%. Moreover, by introducing the extended aggregated features, the savings increase on average by 252% compared with the models trained using only the raw features. As previously shown, in order to improve the results, these new features need to be combined with the standard

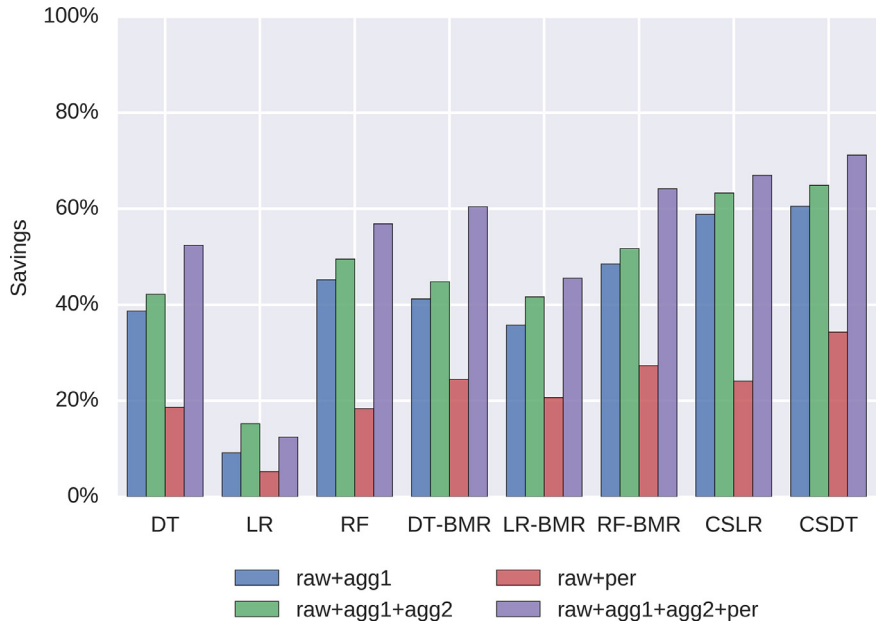

Fig. 6. Comparison of the proposed periodic (*per*) set of features. It is observed, that when the new set of features are combined with both aggregated sets of features, an additional increase of savings of 16.4% is made.

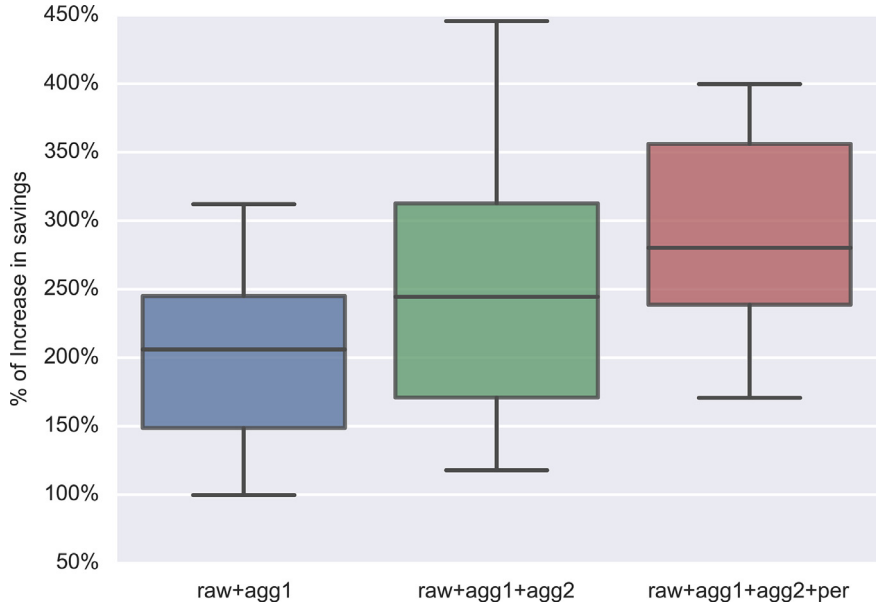

Fig. 7. Comparison of the average increase in savings when introducing each set of features compared with the results of using only the raw set of features.

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.

6. 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 extend the current approaches to analyze the consumer behavior by first proposing a new set of features that enable complex relations of the data to be developed. Then, we proposed a 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 252% and 287%, respectively.

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 no. AFR-PhD-5942749).

Appendix A. Von Mises distribution

In this section we show the calculation of the periodic mean and periodic standard deviation of the von Mises distribution. The von Mises distribution, also known as the periodic normal distribution, is a distribution of a wrapped normal distributed variable across a circle (Fisher, 1995). The von Mises distribution of a set of examples $D = \{t_1, t_2, \cdots, t_N\}$ is defined as

$$D \sim vonmises\left(\mu_{vM}, \frac{1}{\sigma_{vM}}\right), \tag{A.1}$$

where μ_{vM} and σ_{vM} are the periodic mean and periodic standard deviation, respectively, and are calculated as follows

$$\mu_{vM}(D) = 2\tan^{-1}\left(\frac{\sum_{t_j \in D}\sin(t_j)}{\left(\sqrt{\left(\sum_{t_j \in D}\cos(t_j)\right)^2 + \left(\sum_{t_j \in D}\sin(t_j)\right)^2} + \sum_{t_j \in D}\cos(t_j)\right)}\right), \tag{A.2}$$

and

$$\sigma_{vM}(D) = \sqrt{ln\left(\frac{1}{\left(\frac{1}{N}\sum_{t_j \in D}\sin(t_j)\right)^2 + \left(\frac{1}{N}\sum_{t_j \in D}\cos(t_j)\right)^2}\right)}, \tag{A.3}$$

respectively (Bishop, 2006).

References

American Institute of CPAs (2011). International financial reporting standards (IFRS). *Technical Report*. American Institute of CPAs.

Bachmayer, S. (2008). Artificial immune systems, *Vol. 5132*, pp.119–131. doi:10.1007/11823940. http://www.springerlink.com/index/rq58w1v614933838.pdf.

Bhattacharyya, S., Jha, S., Tharakunnel, K., & Westland, J. C. (2011). Data mining for credit card fraud: A comparative study. *Decision Support Systems, 50*(3), 602–613. doi:10.1016/j.dss.2010.08.008. http://linkinghub.elsevier.com/retrieve/pii/S0167923610001326.

Bishop, C. M. (2006). Pattern recognition and machine learning. *Information science and statistics*: 4. Springer. doi:10.1117/1.2819119. http://www.library.wisc.edu/selectedtocs/bg0137.pdf.

Bolton, R., & Hand, D. J. (2001). Unsupervised profiling methods for fraud detection. *Credit scoring and credit control VII*. http://onlinelibrary.wiley.com/doi/10.1002/cbdv.200490137/abstract http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.24.5743&rep=rep1&type=pdf.

Bolton, R. J., Hand, D. J., Provost, F., & Breiman, L. (2002). Statistical fraud detection: a review. *Statistical Science, 17*(3), 235–255. doi:10.1214/ss/1042727940. http://projecteuclid.org:80/Dienst/getRecord?id=euclid.ss/1042727940/.

Brause, R., Langsdorf, T., & Hepp, M. (1999). Neural data mining for credit card fraud detection. In *Proceedings of the 11th international conference on tools with artificial intelligence* (pp. 103–106). doi:10.1109/TAI.1999.809773. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=809773.

Correa Bahnsen, A., Aouada, D., & Ottersten, B. (2014a). Example-dependent cost-sensitive logistic regression for credit scoring. In *Proceedings of the 2014 13th international conference on machine learning and applications* (pp. 263–269). Detroit, USA: IEEE. doi:10.1109/ICMLA.2014.48.

Correa Bahnsen, A., Aouada, D., & Ottersten, B. (2015). Example-dependent cost-sensitive decision trees. *Expert Systems with Applications, 42*(19), 6609–6619. doi:10.1016/j.eswa.2015.04.042. http://linkinghub.elsevier.com/retrieve/pii/S0957417415002845.

Correa Bahnsen, A., Stojanovic, A., Aouada, D., & Ottersten, B. (2013). Cost sensitive credit card fraud detection using bayes minimum risk. In *Proceedings of the 2013 12th international conference on machine learning and applications* (pp. 333–338). Miami, USA: IEEE. doi:10.1109/ICMLA.2013.68. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6784638.

Correa Bahnsen, A., Stojanovic, A., Aouada, D., & Ottersten, B. (2014b). Improving credit card fraud detection with calibrated probabilities. In *Proceedings of the fourteenth siam international conference on data mining* (pp. 677–685).Philadelphia, USA http://linkinghub.elsevier.com/retrieve/pii/S095741741400089X.

Dal Pozzolo, A., Caelen, O., Le Borgne, Y.-A., Waterschoot, S., & Bontempi, G. (2014). Learned lessons in credit card fraud detection from a practitioner perspective. *Expert Systems with Applications, 41*(10), 4915–4928. doi:10.1016/j.eswa.2014.02.026.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the seventeenth international joint conference on artificial intelligence* (pp. 973–978). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.514.

European Central Bank (2014). *Technical Report*. European Central Bank.

Fisher, N. I. (1995). *Statistical analysis of circular data: 9* p. 277. Cambridge University Press, Cambridge, UK.

Hand, D. J., Whitrow, C., Adams, N. M., Juszczak, P., & Weston, D. J. (2007). Performance criteria for plastic card fraud detection tools. *Journal of the Operational Research Society, 59*(7), 956–962. doi:10.1057/palgrave.jors.2602418. http://www.palgrave-journals.com/doifinder/10.1057/palgrave.jors.2602418.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction* (2nd). Stanford, CA: Springer.

Hulse, J. V., & Khoshgoftaar, T. M. (2007). Experimental perspectives on learning from imbalanced data. In *Proceedings of the international conference on machine learning*.

Jha, S., Guillen, M., & Christopher Westland, J. (2012). Employing transaction aggregation strategy to detect credit card fraud. *Expert Systems with Applications, 39*(16), 12650–12657. doi:10.1016/j.eswa.2012.05.018. http://dx.doi.org/10.1016/j.eswa.2012.05.018.

Krivko, M. (2010). A hybrid model for plastic card fraud detection systems. *Expert Systems with Applications, 37*(8), 6070–6076. doi:10.1016/j.eswa.2010.02.119. http://linkinghub.elsevier.com/retrieve/pii/S0957417410001582.

Maes, S., Tuyls, K., Vanschoenwinkel, B., & Manderick, B. (2002). Credit card fraud detection using Bayesian and neural networks. In *Proceedings of NF 2002*. http://www.personeel.unimaas.nl/K-Tuyls/publications/papers/Maenf02.pdf.

Mahmoudi, N., & Duman, E. (2015). Detecting credit card fraud by Modified Fisher Discriminant Analysis. *Expert Systems with Applications, 42*(5), 2510–2516. doi:10.1016/j.eswa.2014.10.037. http://linkinghub.elsevier.com/retrieve/pii/S0957417414006617.

Minegishi, T., & Niimi, A. (2011). Proposal of credit card fraudulent use detection by online-type decision tree construction and verification of generality. *International Journal for Information Security Research (IJISR), 1*(4), 229–235. http://www.infonomics-society.org/IJISR/Proposal of Credit Card Fraudulent Use Detection by Online_type Decision Tree Construction and Verification of Generality.pdf

Panigrahi, S., Kundu, A., Sural, S., & Majumdar, A. (2009). Credit card fraud detection: A fusion approach using Dempster Shafer theory and Bayesian learning. *Information Fusion, 10*(4), 354–363. doi:10.1016/j.inffus.2008.04.001. http://linkinghub.elsevier.com/retrieve/pii/S1566253509000141.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830. http://dl.acm.org/citation.cfm?id=2078195.

Provost, F., Fawcett, T., & Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms.. In *Proceedings of the fifteenth international conference on machine learning* (pp. 445–453). Morgan Kaufmann.

Sahin, Y., Bulkan, S., & Duman, E. (2013). A cost-sensitive decision tree approach for fraud detection. *Expert Systems with Applications, 40*(15), 5916–5923. doi:10.1016/j.eswa.2013.05.021. http://linkinghub.elsevier.com/retrieve/pii/S0957417413003072.

Sánchez, D., Vila, M., Cerda, L., & Serrano, J. (2009). Association rules applied to credit card fraud detection. *Expert Systems with Applications, 36*(2), 3630–3640. doi:10.1016/j.eswa.2008.02.001. http://linkinghub.elsevier.com/retrieve/pii/S0957417408001176.

Tasoulis, D., & Adams, N. (2008). Mining information from plastic card transaction streams. In *Proceedings in 18th international conference on computational statistics*. http://www2.imperial.ac.uk/~dtasouli/papers/TasoulisAWH2008.pdf.

Van Vlasselaer, V., Bravo, C., Caelen, O., Eliassi-Rad, T., Akoglu, L., Snoeck, M., & Baesens, B. (2015). APATE: A Novel Approach for Automated Credit Card Transaction Fraud Detection using Network-Based Extensions. *Decision Support Systems, 75*, 38–48. doi:10.1016/j.dss.2015.04.013. http://linkinghub.elsevier.com/retrieve/pii/S0167923615000846.

Weston, D. J., Hand, D. J., Adams, N. M., Whitrow, C., & Juszczak, P. (2008). Plastic card fraud detection using peer group analysis. *Advances in Data Analysis and Classification, 2*(1), 45–62. doi:10.1007/s11634-008-0021-8. http://www.springerlink.com/index/10.1007/s11634-008-0021-8.

Whitrow, C., Hand, D. J., Juszczak, P., Weston, D. J., & Adams, N. M. (2008). Transaction aggregation as a strategy for credit card fraud detection. *Data Mining and Knowledge Discovery, 18*(1), 30–55. doi:10.1007/s10618-008-0116-z. http://www.springerlink.com/index/10.1007/s10618-008-0116-z.