Credit Card Fraud Detection using Novel Learning Strategy

Bavadharani. G
Department of CSE
TRP Engineering College
Trichy , India

Deepika. B, Divya. K, Sathya. R
Department of CSE
TRP Engineering College
Trichy , India

***Abstract*— Credit card fraud detection with online shopping sector is performed here. A Credit card transactional data was simulated, trained and predicted and then the transaction blocking rules will check those details. The fault phase is data driven; this is purely data driven and adopts a classifier or another statistical model to estimate the probability for each feature vector being a fraud. This entire process is handled by investigator. With the investigated user information we perform the fraud detections. Investigators call cardholders and, after having verified , assign the label "genuine" or "fraudulent" to the alerted transaction, and return this information to the FDS. In the following, we refer to these labeled transactions as feedbacks and use the term alert–feedback interaction to describe this mechanism yielding supervised information in a real-world FDS.**

Keywords— Data mining ,Credit card fraud detection ,Naïve Bayesian classification ,Finger print image ,One Time Password.

I. INTRODUCTION

CREDIT card fraud detection is a relevant problem that draws the attention of machine-learning and computational intelligence communities, where a large number of automatic solutions have been proposed. In fact, this problem appears to be particularly challenging from a learning perspective, since it is characterized at the same time by class imbalance], namely, genuine transactions far outnumber frauds, and concept drift, namely, transactions might change their statistical properties over time. These, however, are not the only challenges characterizing learning problems in a real-world fraud-detection system (FDS). In a real-world FDS, the massive stream of payment requests is quickly scanned by automatic tools that determine which transactions to authorize. Classifiers are typically employed to analyze all the authorized transactions and alert the most suspicious ones. Alerts are then inspected by professional investigators that contact the cardholders to determine the true nature (either genuine or fraudulent) of each alerted transaction. By doing this, investigators provide a feedback to the system in the form of labeled transactions, which can be used to train or update the classifier, in order to preserve (or eventually improve) the fraud-detection performance over time. The vast majority of transactions cannot be verified by investigators for obvious time and cost constraints. These transactions remain unlabeled until customers discover and report frauds, or until a sufficient amount of time has elapsed such that non disputed transactions are considered genuine. Thus, in practice, most of supervised samples are provided with a substantial delay, a problem known as verification latency. The only recent supervised information made available to update the classifier is provided through the alert– feedback interaction. Most papers in the literature ignore the verification latency as well as the alert–feedback interaction, and unrealistically assume that the label of each transaction is regularly made available to the FDS, e.g., on a daily basis However, these aspects have to be considered when designing a real-world FDS, since verification latency is harmful when concept drift occurs, and the alert–feedback interaction is responsible of a sort of sample selection bias (SSB) that injects further differences between the distribution of training and test data. Another important difference between what is typically done in the literature and the real-world operating conditions of Fraud-Detection System (FDS) concerns the measures used to assess the fraud-detection performance.

Most often, global ranking measures, like the area under the ROC curve (AUC), or cost-based measures, used, but these ignore the fact that only few alerts can be controlled everyday, and that companies are very concerned of the precision of the generated alerts. The main contributions of this paper are as follows.

1) We describe the mechanisms regulating a real-world FDS, and provide a formal model of the articulated classification problem to be addressed in fraud detection.

2) We introduce the performance measures that are considered in a real-world FDS.

3) Within this sound and realistic model, we propose an effective learning strategy for addressing the above challenges, including the verification latency and the alert–feedback interaction. This learning strategy is tested on a large number of credit card transactions. This paper is organized as follows. We first detail the operating conditions of a real-world FDS in Section II, and then in Section III model the articulated fraud-detection problem and present the most suitable performance measures.

In particular, we deem that it is most appropriate to assess the number of detected fraudulent transactions (or cards) over the maximum number of transactions (or cards) that investigators can check. The main challenges raising when training a classifier for fraud-detection purposes are then discussed in Section IV.

Section V introduces the proposed learning strategy, which consists in separately training different classifiers from feedbacks and delayed supervised samples, and then aggregating their predictions. This strategy, inspired by the different nature of feedbacks and delayed supervised samples,or cost based measures are used is shown to be particularly effective in FDS using sliding window or ensemble of classifiers. We validate our claims in experiments (Section VI) on more than 75 million e-commerce credit card transactions acquired over three years, which are also analyzed to observe the impact of class imbalance and concept drift in real-world transaction streams.

Our work builds upon], which is significantly extended by describing in detail the real-world operating conditions of an FDS and by analyzing the SSB introduced by the alert– feedback interaction. Furthermore, the experimental section has been largely updated and completed by presenting additional analysis over two large data sets.

II. EXISTING SYSTEM

Layers of Controls in an FDS

A. Terminal

The terminal represents the first control layer in an FDS and performs conventional security checks on all the payment requests]. Security checks include controlling the PIN code (possible only in case of cards provided with chip), the number of attempts, the card status (either active or blocked), the balance available, and the expenditure limit. In case of online transactions, these operations have to be performed in real time (response has to be provided in a few milliseconds), during which the terminal queries a server of the card issuing company. Requests that do not pass any of these controls are denied, while the others become transaction requests that are processed by the second layer of control.

B.Transaction-Blocking Rules

Transaction-blocking rules are if-then (-else) statements meant to block transaction requests that are clearly perceived as frauds. These rules use the few information available when the payment is requested, without analyzing historical records or cardholder profile. An example of blocking rule could be ―IF internet transactions AND unsecured Web site THEN deny the transaction.‖1 In practice, several transaction-blocking rules are simultaneously executed, and transactions firing any of these rules are blocked (though cards are not deactivated). Transaction blocking rules are manually designed by the investigator and, as such, are expert-driven components of the FDS. To guarantee real-time operations and avoid blocking many genuine transactions, blocking rules should be: 1) quick to compute 2) very precise, namely, should raise very few false alarms. All transactions passing blocking rules are finally authorized. However, the fraud-detection activity continues after having enriched transaction data with aggregated features used to compare the current purchase against the previous ones and the cardholder profile. These aggregated features include, for instance, the average expenditure, the average number of transactions in the same day, or the location of the previous purchases. The process of computing aggregated features is referred to as feature augmentation and is described in Section II-B. Augmented features and current transaction data are stacked in a feature vector that is supposed to be informative for determining whether the authorized transaction is fraudulent or genuine. The following layers of the FDS operate on this feature vector.

C. Scoring Rules

Scoring rules are also expert-driven models that are expressed as if-then (-else) statements. However, these operate on feature vectors and assign a score to each authorized transaction: the larger the score, the more likely the transaction to be a fraud. Scoring rules are manually designed by investigators, which arbitrarily define their associated scores. An example of scoring rule can be ―IF previous transaction in a different continent AND less than 1 h from the previous transaction THEN fraud score = 0.95.‖ Unfortunately, scoring rules can detect only fraudulent strategies that have already been discovered by investigators, and that exhibit patterns involving few components of the feature vectors. Moreover, scoring rules are rather subjective, since different experts design different rules.

D. Data Driven Model (DDM)

This layer is purely data driven and adopts a classifier or another statistical model to estimate the probability for each feature vector being a fraud. This probability is used as the fraud score associated with the authorized transactions. Thus, the DDM is trained from a set of labeled transactions and cannot be interpreted or manually modified by investigators. An effective DDM is expected to detect fraudulent patterns by simultaneously analyzing multiple components of the feature vector, possibly through nonlinear expressions. Therefore,the DDM is expected to find frauds according to rules that go beyond investigator experience, and that do not necessarily correspond to interpretable rules. This paper focuses on this component of the FDS and proposes a strategy to design, train, and update the DDM to improve fraud-detection performance. Transactions associated with feature vectors that have either received a large fraud score or a high probability of being a fraud generate alerts. Only a limited number of alerted transactions are reported to the investigators, which represent the final layer of control.

E.Investigators

Investigator are professionals experienced in analyzing credit card transactions and are responsible of the expert-driven layers of the FDS. In particular, investigators design transaction-blocking and scoring rules. Investigators are also in charge of controlling alerts raised by the scoring rules and the DDM, to determine whether these correspond to frauds or false alarms. In particular, they visualize all the alerted transactions in a case management tool, where all the information about the transaction is reported, including the assigned scores/probabilities, which in practice indicate how risky each transaction is. Investigators call cardholders and, after having verified, assign the label ―genuine‖ or ―fraudulent‖ to the alerted transaction, and return this information to the FDS. In the following, we refer to these labeled transactions as feedbacks and use the term alert–feedback interaction to describe this mechanism yielding supervised information in a real-world FDS. Any card that is found victim of a fraud is immediately blocked, to prevent further fraudulent activities. Typically, investigators check all the recent transactions from a compromised card, which means that each detected fraud can potentially generate more than one feedback, not necessarily corresponding to alerts or frauds. In a real-world FDS, investigators can only check few alerts per day as this process can be long and tedious. Therefore, the primary goal of a DDM is to return precise alerts, as investigators might ignore further alerts when too many false alarms are reported.

Features Augmentation

Any transaction request is described by few variables such as the merchant ID, cardholder ID, purchase amount, date, and time. All transaction requests passing the blocking rules are entered in a database containing all recent authorized transactions, where the feature-augmentation process starts. During feature augmentation, a specific set of aggregated features associated with each authorized transactions is computed, to provide additional information about the purchase and better discriminate frauds from genuine transactions. Examples of aggregated features are the average expenditure of the customer every week/month, the average number of transactions per day or in the same shop, the average transaction amount, and the location of the last purchases.. show that additional.

informative features can be extracted from the social networks connecting the cardholders with merchants/shops. Aggregated features are very informative, as they summarize the recent cardholder activities. Thus, they allow to alert transactions that are not suspicious by themselves but might be unusual compared with the shopping habits of the specific cardholder. Features augmentation can be computationally expensive, and aggregated features are often precomputed offline for each cardholder on the basis of historical transactions. Aggregated features are stacked with the transaction data in the feature vector.

Supervised Information Investigator

Feedbacks are the most recent supervised information made available to the FDS, but represent only a small fraction of the transactions processed every day. Additional labeled transactions are provided by cardholders that directly dispute unauthorized transactions .The timing of disputed transactions can vary substantially, since cardholders have different habits when checking the transcript of credit card sent by the bank. Moreover, checking disputed transactions entails some necessary administrative procedures that might introduce substantial delays. All other transactions remain unlabeled: these can be either genuine transactions or frauds that were missed by the FDS and ignored by the cardholders. However, after a certain number of days have passed without cardholder dispute, all the unreported transactions are considered genuine by default, and inserted in the training set of the DDM. Overall, there are two types of supervised information: 1) feedbacks provided by investigators that are limited in number but refer to recent transactions and 2) delayed supervised transactions, which are the vast majority for which the labels become available after several days (e.g., one month). This latter includes both disputed and non disputed transactions.

System Update

spending behavior evolves and fraudsters continuously design new attacks, and thus their strategies also change over time. It is then necessary to constantly update the FDS to guarantee satisfactory performance. Expert-driven systems are regularly updated by investigators who add ad hoc (transaction-blocking or scoring) rules to counteract the onset of new fraudulent activities and remove those rules liable of too many false alerts. However, investigators cannot modify the DDM, since it is not interpretable and can be only updated (e.g., retrained) on the basis of recent supervised information, as shown in Fig. 1. This operation typically requires a large number of labeled transactions; therefore, though investigators steadily provide feedbacks during the day, the classifier is usually updated/re-trained only once ,notably at the end of the day, when a sufficient number of feedbacks are available.

III. PROBLEM FORMULATION

Here, we model the classification problem to be addressed in a real-world FDS, providing a formal description of the alert–feedback interaction and presenting suitable performance measures.The proposed learning strategy (Section V) and our
experiments (Section VI) are built upon this model. Let xi denote the feature vector associated with the I th authorized transaction and ∈{+ ,−} be the corresponding class, where + denotes a fraud and – a genuine transaction. To cope with the time-variant nature of the transaction stream, a classifier K is updated (or newly retrained) every day. In particular ,we denotebyKt−1 the classifier that is trained on supervised transactions available up to day t−1. The classifier Kt−1 is then used to process the set of transactions T t that have been authorized at day t. We denote by PKt−1(+|xi) the posterior of Kt−1, namely, the probability for xi to be a fraud according to Kt−1. Investigators check only few high risk transactions. Thus, we model alerts as the k-most risky transactions, namely At ={ xi ∈Tt s.t. r(xi)≤k} (1) where r(xi) ∈{ 1,...,|Tt|} is the rank of xi according to PKt(+|xi), and k > 0 is the maximum number of alerts that can be checked by investigators. 2 As discussed in Section II-A.5, investigators contact the cardholders and provide supervised samples to the FDS in the form of feedbacks. In particular, feedbacks include all recent transactions from the controlled cards, which we model as Ft ={ (xi, yi) s.t. xi

is from cards(At)} (2) where cards(At) denotes the set of cards having at least a transaction in At. The number of feedbacks, i.e.,|Ft|, depends on the number of transactions associated with the k controlled cards. After a certain verification latency, the labels of all the transactions are provided to the FDS, since, as discussed in Section II-C, non disputed transactions are considered genuine. For the sake of simplicity, we assume a constant verification latency of δ days, such that at day t, the labels of all the transactions authorized at day t−δ are provided. We refer to these as delayed supervised samples Dt−δ ={ (xi, yi), xi ∈Tt−δ}. (3) Note that Ft−δ ⊂Dt−δ since transactions at day t−δ obviously include those that have been alerted. Fig. 2 illustrates the different types of supervised information available in an FDS. It is worth mentioning that, despite our formal description includes several aspects and details that have been so far ignored in the fraud-detection literature, this is still a simplified model. In fact, alerts in a real-world FDS are typically raised online while transactions are being processed, without having to rank all transactions in Tt. Similarly, the delayed supervised couples do not come all at once, as each disputed transactions might take less (or possibly more) than δ days. Notwithstanding, we deem that our formulation takes into account the aspects of a real-world FDS that are the most important ones from a learning perspective, which include alerts, alter–feedback interaction, and verification latency. We further comment that in principle, since the classifier analyzes each feature vector xi independently, it does not alert cards receiving several risky transactions until any of these enters in the pool of the alerts (1). However, these situations are particularly relevant for investigators, and can be handled either by suitable scoring rules or feature augmentation, adding for instance , a component that keeps track of the scores of recent transactions. Fraud-detection performance can be conveniently assessed in terms of the alert precision Pk(t) , which is defined as Pk(t)= |TPk(t)| k (4) where TPk(t) ={ (xi, yi) such that xi ∈ At, yi = +}. Thus , P k(t) is the proportion of frauds in the alerts At. Though the classifier independently processes each feature vector, the alert precision would be more realistically measured in terms of cards rather than authorized transactions. In fact, multiple transactions in at from the same card should be counted as a single alert, since investigators check all the recent transactions when contacting cardholders. This implies that k depends on the maximum number of cards that the investigators can control. In this context, it is more informative to measure the detection performance at the card level, such that multiple fraudulent transactions from the same card count as a single correct detection. Thus, we introduce CPk, the card precision, as the proportion of fraudulent cards detected in the k cards controlled by the investigators CPk(t)= |C+ t | k (5) where C+ t denotes the set of fraudulent cards correctly detected at day t, namely, fraudulent cards having reported at least one alert. To correctly account for those days where less than k cards are fraudulent, we define the normalized
CPk(t) as

NCPk(t)=
CPk(t) (t)
with (t) = 1 if γt ≥kγ t k if γt < k
where (t) is the maximum value of CPk(t) and γt is the number of fraudulent cards at day t. From (6), we have that NCPk(t) takes values in the range [0,1], while CPk(t) is in [0,1] when γt > k and in [0,(γ t/k)] otherwise. For example, if at day t, we have correctly detected 40 fraudulent cards (|C+ t |=40) out of the k =100 cards checked by investigators, and the overall number of fraudulent cards is 50 (γt = 50), then CPk(t) =0.4 while NCPk(t) =(0.4)/(0.5)=0.8. Note that, since (t) does not depend on the specific classifierKt−1 adopted, when algorithm ―A‖ is better than algorithm ―B‖ in terms of CPk, ―A‖ is also better than ―B‖ in terms of NCPk. Moreover, because of verification latency, the number of fraudulent cards in day t (i.e., γt) can be only computed after few days, and therefore NCPk cannot be computed in real time. Thus, we recommend using CPk for assessing the running performance, while NCPk for backtesting, e.g., when testing different FDS configurations, as in Section VI-F.

IV RELATED WORKS

A. Data-Driven Approaches in Credit Card Fraud Detection Both supervised [8], [12], [15] and unsupervised [11], [14],
[62] methods have been proposed for credit card fraud detection purposes. Unsupervised methods consist in outlier/ anomaly detection techniques that consider as a fraud any transaction that does not conform with the majority. Remarkably, an unsupervised DDM in an
FDS can be directly configured from unlabeled transactions. A well- known method is peer group analysis [65], which clusters customers according to their profile and identifies frauds as transactions departing from the typical cardholder's behavior (also see [52]). The typical cardholder's behavior has also been modeled by means of self-organizing maps [51], [54], [71]. Supervised methods are by far the most popular in fraud detection, and exploit labeled transactions for training a classifier. Frauds are detected by classifying feature vectors of the authorized transactions or possibly by analyzing the posterior of the classifier [10].Several classification algorithms have been tested on credit card transactions to detect frauds, including neural networks [1], [12], [28], logistic regression [41], association rules [56], support vector machines [66], modified Fisher discriminant analysis [47], and decision trees [6], [24], [55]. Several studies have reported random forest (RF) to achieve the best performance [8], [20], [23], [63], [66]: this is one of the reasons why we adopt RFs in our experiments.

Performance Measure for Fraud Detection

The typical performance measure for fraud-detection problems is the AUC [23], [24], [63]. AUC can be estimated by means of the Mann–Whitney statistic [48] and its value can be interpreted as the probability that a classifier ranks frauds higher than genuine transactions [37]. Another ranking measure frequently used in fraud detection is average precision [23], which corresponds to the area under the precision– recall curve.While these measures arewidelyused in detection problems, cost-based measures have been specifically designed for fraud-detection purposes. Cost-based measures [6], [47], [55] quantify the monetary loss of a fraud by means of a cost matrix that associates a cost with each entry of the confusion matrix. Elkan [29] shows that a cost matrix may be misleading because the minimum/maximum loss of the problem can change over time. To avoid this problem, normalized cost [66] or savings [6] are used to assess the performance with respect to the maximum loss. We argue that performance measures should also account for the investigators availability, as they have to check all the alerts raised by the FDS. Given the limited time investigators have, only a few alerts can be verified every day, and thus an effective FDS should provide investigators a small number of reliable alerts. This is the reason why we have introduced the alert-precision measures described in Section III.

V. PROPOSED LEARNING STRATEGY

It is important to stress that feedbacks (Ft) and delayed samples (Dt−δ) are very different sets of supervised samples. The first difference is quite evident: Ft provides recent up-to-date information while Dt−δ might be already obsolete for training a classifier that is meant to analyze transactions that will be authorized the next day. The second difference concerns the percentage of frauds in Ft and Dt−δ: while the class proportion in Dt−δ is heavily skewed toward the genuine class (see the proportions of frauds in Table I), the number of frauds in Ft actually depends on the detection performance of Kt−1, and high precision values might even result in Ft skewed toward frauds. The third, and probably the most subtle, difference is that supervised couples in Ft are not independently drawn, but are instead transactions from cards selected by Kt−1 as those that are most likely to have been frauded. As such, Ft is affected by SSB and any classifier trained on Ft would in principle learn how to label transactions that are most likely to be fraudulent.Thus, this might not be in principle precise on the vast majority of genuine transactions. Our intuition is that feedbacks and delayed samples are representative of two different classification problems, and thus they have to be separately handled. Therefore, our learning strategy consists in training a classifier exclusively on feedbacks (i.e., Ft) and a classifier exclusively on delayed supervised samples (i.e., Dt), and by aggregating their posterior probabilities when definingPKt(+|xi) to determine which transactions to alert. In the following, we detail the proposed learning strategy, where adaptation is performedaccordingto a passive approach and the classifier is updated everyday on a batch containing the latest supervised couples available, either feedbacks or delayed samples. As in Section III, we consider a constant verification latency of δ days. In particular, to process the transactions authorized at day t + 1, we rely on Q days of feedbacks {Ft,...,Ft−(Q−1)}, andM days of delayed supervised samples {Dt−δ,...,Dt−(δ+M−1)}, and these latter obviously include the feedbacks received in the same days (i.e., Fi ⊂ Di, i ≤ t − δ). Our learning strategy, which is detailed in Algorithm 1, consists in separately training a classifier Ft on feedbacks Ft =TRAIN({Ft,...,Ft−(Q−1)}) (7) and a classifier on delayed supervised samples Dt =TRAIN({Dt−δ,...,Dt−(δ+M−1)}) (8) and to detect frauds by the aggregation classifier At, whose posterior probability is defined as PAt(+|x)=αPFt(+|x)+(1−α)PDt(+|x) (9) where 0 ≤ α ≤ 1 is the weight parameter that balances the contribution of Ft and Dt. Thus, the posterior probability of the classifier Kt, which alerts the transactions authorized at day t +1, is given by (9). The parameters Q and M that, respectively, define how many days of feedbacks and delayed supervised samples are used for training our classifiers have to be defined considering the overall number of feedbacks and the percentage of frauds. The training set of Ft approximately contains Q·|Ft|samples (a different number of feedbacks might be provided everyday) and this has to be a sufficiently large number to train a classifier addressing quite a challenging classification problem in high dimensions. However, Q cannot be made arbitrarily large, not to include old feedbacks. Similar considerations hold when setting M, the considered number of days containing delayed transactions, which have to includea sufficient number of frauds. Note that it is nevertheless possible to include in the training set of Ft feedbacks received before δ days (Q ≥ δ) and in particular in our experiments we used Q =δ+M. The rationale behind the proposed learning strategy is twofold. At first, by training a classifier (7) exclusively on feedbacks, we guarantee larger relevance to these supervised samples, which would be otherwise outnumbered by the delayed supervised samples. Second, we alert only those transactions that both Ft and Dt consider most probably frauds: this follows from the fact that, in practice, because of the large number of transactions processed everyday, alerts correspond to values of PAt that are very close to one. Let us recall that Ft, thus also At, is affected by SSB due to alert–feedback interaction. The only training samples that are not affected by SSB are the delayed supervised samples that, however, might be obsolete because of concept drift.

A. Implementation of the Proposed Learning Strategy In our experiments, we implement the proposed learning strategy in two different scenarios, which correspond to two mainstream approaches for learning Dt. In the former, Dt is

Algorithm 1 Proposed Learning Strategy Require: M and Q, i.e., the number of days of delayed samples and feedbacks to use, respectively; Ft and Dt classifiers previously trained. Tt+1 ← transactions at day t +1. for each transaction x ∈Tt+1 do compute PFt(+,x) compute PDt(+,x) compute PAt(+,x) as in (9) rank Tt+1 according to PAt(+,·), generate alerts At. if update the classifier then Ft+1 ← feedbacks from cards alerted in At. Ft+1 ← TRAIN({Ft+1,...,Ft−Q}) Dt+1−δ ← transactions authorized at t +1−δ Dt+1 ←

TRAIN({Dt+1−δ,...,Dt−(δ+M)}) return Ft,Dt and At defined as in (9).

a sliding window classifier as in [62] and [63], which we denote by WD t , while in the latter, Dt is an ensemble of classifiers similar to

[36] and [23], which we denote by ED t . Both the classifiers WD t and ED t are trained on delayed samples {Dt−δ,...,Dt−(δ+M−1)}.

However, while WD t employs a unique model to this purpose, ED t is an ensemble of M classifiers {M1,M2,...,MM} where each individual classifierMi is trained on delayed samples of a different day, i.e., Dt−δ−i,i =0,...,M−1. The posteriorPED t (+|x) is obtained by averaging the posterior probabilities of the individual classifiers, i.e., PED t (+|x)=(M i PMi(+|x))/(M). In the sliding window case, the proposed learning strategy consists in analyzing the posterior of the classifier AW t , which aggregates Ft and WD t , i.e., PAW t (+|x) =α PFt(+|x) + (1 − α)PWD t (+|x) as in (9). The benchmark to compare against AW t is the classifier Wt, which is trained on all the supervised transactions referring to the same time interval (thus

mixing delayed samples and feedbacks): {Ft,...,Ft−(δ−1),Dt−δ,...,Dt−(δ+M−1)}. Similarly, in the ensemble case, the proposed learning strategy consists in analyzing the posterior of the classifier AE t , which is obtained by aggregating the posteriors of Ftand ED t , i.e., PAE t (+|x) = αPFt (+|x)+(1−α)PED t (+|x), as in (9). The benchmark to compare against AE t is the classifier Et, whose individuals are {M1,M2,...,MM,Ft}, and whose posterior PEt(+|x) is estimated by averaging the posterior probabilities of all its individuals, i.e., PEt(+|x) = (M i PMi(+|x)+PFt(+|x))/(M +1). In both aggregations AW t and AE t , we setα = 0.5 to give equal contribution to the feedback and delayed classifier, as better discussed in Section VI-F. For all the base classifiers involved (i.e., Ft,WD t ,Wt,Mi,i = 1,...,M), we adopt RF

[13] having 100 tree each. Each tree is trained on a balanced bootstrap sample, obtained by randomly undersampling the majority class while preserving all the minority class samples in the corresponding training set. In this way,each tree is trained onrandomlyselected genuine transactions and on the

TABLE I DATA SETS

same fraud examples. This under sampling strategy allows one to learn trees with balanced distribution and to exploit many subsets of the majority class. At the same time, the training times of these classifiers are reasonably low. A drawback of undersampling is that we potentially remove relevant training samples from the data set, though this problem is mitigated by the fact that we learn 100 different trees for each base classifier.

VI CONCLUSION

The majority of works addressing the fraud-detection problem in credit card transactions (see [5], [23], [63]) unrealistically assume that the class of each transaction is immediately provided for training the classifier. Here we analyze in detail thereal-world working conditions of FDS and providea formal description of the articulated classification problem involved. In particular, we have described the alert–feed back interaction, which is the mechanism providing recent supervised samples to train/update the classifier. We also claim that, in contrast to traditional performance measures used in the literature, in a real-world FDS, the precision of the reported alerts is probably the most meaningful one, since investigators can check only few alerts. Our experiments on two vast data sets of real-world transactions show that, in order to get precise alerts, it is mandatory to assign larger importance to feedbacks during the learning problem. Not surprisingly, feedbacks play a central role in the proposed learning strategy, which consists intraining a classifier on feedbacks and a classifier on delayed supervised samples, and then aggregating their posteriors to identify alerts. Our experiments also show that solutions that lower the influence of feedbacks in the learning process (e.g., classifiers that mix feedbacks and delayed supervised samples or that implement instance weighting schemes) are often returning less precise alerts. Future work concerns the study of adaptive and possibly nonlinear aggregation methods for the classifiers trained on feedbacks and delayed supervised samples. We also expect to further increase the alert precision by implementing a learning to rank approach [46] that would be specifically designed to replace the linear aggregation of the posterior probabilities. Finally, a very promising research direction concerns semisupervised learning methods [16], [39] for exploiting in the learning process also few recent unlabeled transactions.

REFERENCES

[1] E. Aleskerov, B. Freisleben, and B. Rao, ―CARDWATCH: A neural network based database mining system for credit card fraud detection,‖ in Proc. IEEE/IAFE Computat. Intell. Financial Eng., Mar. 1997, pp. 220–226.

[2] C. Alippi, G. Boracchi, and M. Roveri, ―A just-in-time adaptive classification system based on the intersection of confidence intervals rule,‖ Neural Netw., vol. 24, no. 8, pp. 791–800, 2011

[3] C. Alippi, G. Boracchi, and M. Roveri, ―Hierarchical change-detection tests,‖ IEEE Trans. Neural Netw. Learn. Syst., vol. 28, no. 2, pp. 246–258, Feb. 2016.

[4] C. Alippi, G. Boracchi, and M. Roveri, ―Just-in-time classifiers for recurrent concepts,‖ IEEE Trans. Neural Netw. Learn. Syst., vol. 24, no. 4, pp. 620–634, Apr. 2013.

[5] B. Baesens, V. Van Vlasselaer, and W. Verbeke, Fraud Analytics Using Descriptive, Predictive, and Social Network Techniques: A Guide to Data Science for Fraud Detection. Hoboken, NJ, USA: Wiley, 2015.

[6] C. Bahnsen, D. Aouada, and B. Ottersten, ―Example-dependent cost-sensitive decision trees,‖ Expert Syst. Appl., vol. 42, no. 19, pp. 6609–6619, 2015.

[7] C. Bahnsen, D. Aouada, A. Stojanovic, and B. Ottersten, ―Detecting credit card fraud using periodic features,‖ in Proc. 14th Int. Conf. Mach. Learn. Appl., Dec. 2015, pp. 208–213.

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

[9] Bifet and R. Gavalda, ―Learning from time-changing data with adaptive windowing,‖ in Proc. SDM, vol. 7. 2007, pp. 443–448.

[10] R. Bolton and D. Hand, ―Statistical fraud detection: A review,‖ Stat. Sci., vol. 17, no. 3, pp. 235–249, 2002.

[11] R. J. Bolton and D. J. Hand, ―Unsupervised profiling methods for fraud detection,‖ in Credit Scoring Credit Control VII. London, U.K.: Imperial College London, 2001, pp. 235–255.

[12] R. Brause, T. Langsdorf, and M. Hepp, ―Neural data mining for credit card fraud detection,‖ in Proc. Tools Artif. Intell., Jul. 1999, pp. 103–106.

[13] L. Breiman, ―Random forests,‖ Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[14] M. Carminati, R. Caron, F. Maggi, I. Epifani, and S. Zanero, BankSealer: A Decision Support System for Online Banking Fraud Analysis and Investigation, Berlin, Germany: Springer, 2014, pp. 380–394.

[15] P. K. Chan, W. Fan, A. L. Prodromidis, and S. J. Stolfo, ―Distributed data mining in credit card fraud detection,‖ IEEEIntell.Syst. Their Appl., vol. 14, no. 6, pp. 67–74, Nov. 1999.

[16] O. Chapelle, B. Scholkopf, and A. Zien, ―Semi-supervised learning,‖ IEEE Trans. Neural Netw., vol. 20, no. 3, pp. 542–542, 2009.

[17] N. Chawla, K. Bowyer, L. O. Hall, and W. P. Kegelmeyer, ―SMOTE: Synthetic minority over-sampling technique,‖ J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[18] S. Chen and H. He, ―Towards incremental learning of nonstationary imbalanced data stream: A multiple selectively recursive approach,‖ Evolving Syst., vol. 2, no. 1, pp. 35–50, 2011.

[19] Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, ―Sample selection bias correction theory,‖ in Algorithmic Learning Theory.

[20] Berlin, Germany: Springer, 2008, pp. 38–53. Dal Pozzolo, G. Boracchi, O. Caelen, C. Alippi, and G.

[21] Bontempi, ―Credit card fraud detection and concept-drift adaptation with delayed supervised information,‖ in Proc. Int. Joint Conf. Neural Netw., 2015, pp. 1–8. streams,‖ in Proc. Int. Joint Conf. Neural Netw., 2014, pp. 588–594.