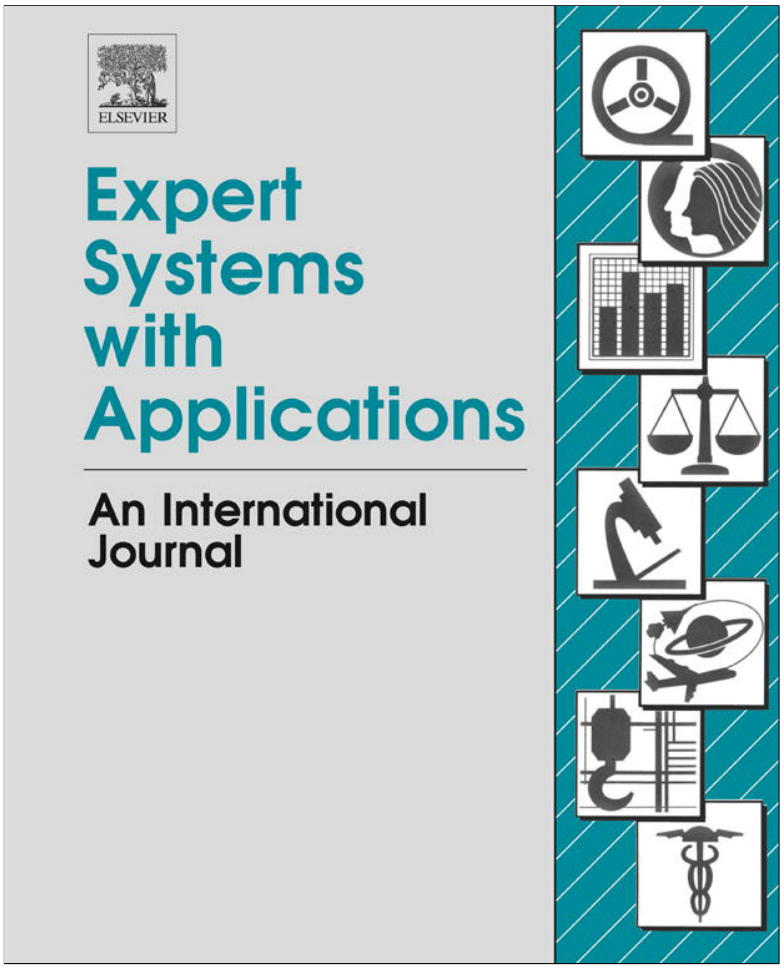

Expert Systems with Applications 40 (2013) 5916–5923

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

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

A cost-sensitive decision tree approach for fraud detection

Yusuf Sahin [a,*], Serol Bulkan [b], Ekrem Duman [c]

[a] Department of Electrical & Electronics Engineering, Marmara University, Kadikoy, 34722 Istanbul, Turkey
[b] Department of Industrial Engineering, Marmara University, Kadikoy, 34722 Istanbul, Turkey
[c] Department of Industrial Engineering, Ozyegin University, Cekmekoy, 34794 Istanbul, Turkey

ARTICLE INFO

Keywords:
Cost-sensitive modeling
Credit card fraud detection
Decision tree induction
Classification
Variable misclassification cost

ABSTRACT

With the developments in the information technology, fraud is spreading all over the world, resulting in huge financial losses. Though fraud prevention mechanisms such as CHIP&PIN are developed for credit card systems, these mechanisms do not prevent the most common fraud types such as fraudulent credit card usages over virtual POS (Point Of Sale) terminals or mail orders so called online credit card fraud. As a result, fraud detection becomes the essential tool and probably the best way to stop such fraud types. In this study, a new cost-sensitive decision tree approach which minimizes the sum of misclassification costs while selecting the splitting attribute at each non-terminal node is developed and the performance of this approach is compared with the well-known traditional classification models on a real world credit card data set. In this approach, misclassification costs are taken as varying. The results show that this cost-sensitive decision tree algorithm outperforms the existing well-known methods on the given problem set with respect to the well-known performance metrics such as accuracy and true positive rate, but also a newly defined cost-sensitive metric specific to credit card fraud detection domain. Accordingly, financial losses due to fraudulent transactions can be decreased more by the implementation of this approach in fraud detection systems.

1. Introduction

Fraud can be defined as wrongful or criminal deception aimed to result in financial or personal gain. The two main mechanisms to avoid frauds and losses due to fraudulent activities are fraud prevention and fraud detection systems. Fraud prevention is the proactive mechanism with the goal of disabling the occurrence of fraud. Fraud detection systems come into play when the fraudsters surpass the fraud prevention systems and start a fraudulent transaction. A review of fraud domains and detection techniques can be found in Bolton and Hand (2002), Kou, Lu, Sirwongwattana, and Huang (2004), Phua, Lee, Smith, and Gayler (2005), Sahin and Duman (2010). One of the most well-known fraud domains is the credit card systems. Credit card frauds can be made in many ways such as simple theft, application fraud, counterfeit cards, never received issue (NRI) and online fraud (where the card holder is not present). In online fraud, the transaction is made remotely and only the card's details are needed. Because of the international availability of the web and ease with which users can hide their location and identity over internet transactions, there is a rapid growth of committing fraudulent actions over this medium.

There are many previous studies done on credit card fraud detection. The general background of the credit card systems and non-technical knowledge about this fraud domain can be learned from Hanagandi, Dhar, and Buescher (1996) and Hand and Blunt (2001), respectively. The most commonly used fraud detection methods in this domain are rule-induction techniques, decision trees, Artificial Neural Networks (ANN), Support Vector Machines (SVM), logistic regression, and meta-heuristics such as genetic algorithms. These techniques can be used alone or in collaboration using ensemble or meta-learning techniques to build classifiers. Most of the credit card fraud detection systems are using supervised algorithms such as neural networks (Brause, Langsdorf, & Hepp, 1999; Dorronsoro, Ginel, Sanchez, & Cruz, 1997; Juszczak, Adams, Hand, Whitrow, & Weston, 2008; Quah & Sriganesh, 2008; Schindeler, 2006; Shen, Tong, & Deng, 2007; Stolfo, Fan, Lee, Prodromidis, & Chan, 1997; Stolfo, Fan, Lee, Prodromidis, & Chan, 1999; Syeda, Zhang, & Pan, 2002; Prodromidis, Chan, & Stolfo, 2000); decision tree techniques like ID3, C4.5 and C&RT (Chen, Chiu, Huang, & Chen, 2004; Chen, Luo, Liang, & Lee, 2005; Mena, 2003; Wheeler & Aitken, 2000); and SVM (Gartner Reports, 2010; Leonard, 1993).

Credit card fraud detection is an extremely difficult, but also popular problem to solve. There comes only a limited amount of data with the transaction being committed. Also, there can be past transactions made by fraudsters which also fit a pattern of normal (legitimate) behavior (Aleskerov, Freisleben, & Rao, 1997). Furthermore,

* Corresponding author. Tel.: +90 533 5566217; fax: +90 216 3480292.
E-mail addresses: ysahin@marmara.edu.tr (Y. Sahin), sbulkan@marmara.edu.tr (S. Bulkan), ekrem.duman@ozyegin.edu.tr (E. Duman).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.05.021

the problem has many constraints. First of all, the profiles of normal and fraudulent behaviors change constantly. Secondly, the development of new fraud detection methods is made more difficult by the fact that the exchange of ideas in fraud detection, especially in credit card fraud detection is severely limited due to security and privacy concerns. Thirdly, data sets are not made available and the results are often censored, making them difficult to assess. Even, some of the studies are done using synthetically generated data (Brause et al., 1999; Dorronsoro et al., 1997). Fourthly, credit card fraud data sets are highly skewed sets. Lastly, the data sets are also constantly evolving making the profiles of normal and fraudulent behaviors always changing (Bolton & Hand, 2002; Kou et al., 2004; Phua et al., 2005; Sahin & Duman, 2010). So, credit card fraud detection is still a popular challenging and hard research topic. Visa reports about credit card frauds in European countries state that about 50% of the whole credit card fraud losses in 2008 are due to online frauds (Ghosh & Reilly, 1994). Many papers reported huge amounts of losses in different countries (Bolton & Hand, 2002; Dahl, 2006; Schindeler, 2006). Thus new approaches improving the classifier performance in this domain have both financial implications and research contributions. Defining a new cost-sensitive approach is one of the best ways for such an improvement due to the characteristics of the domain.

Although traditional machine learning techniques are generally successful in many classification problems, having a high accuracy or minimizing the misclassification errors is not always the goal for the classifier developed. In the applications of real-world machine-learning problem domains, there are various types of costs involved and Turney defined nine main types of costs (Turney, 2000). However, most of the machine-learning literature does not take any of these costs into account, only a few of the remainings take the misclassification cost into consideration. Turney also stated that the cost of misclassification errors occupies a unique position in their taxonomy (Turney, 2000). Nevertheless, according to the Technological Roadmap of the ML-netII project (European Network of Excellence in Machine Learning), cost-sensitive learning is stated to be one of the most popular topics in the future of machine learning research (Saitta, 2000; Zhou & Liu, 2006). Thus, improving classifier performance of a fraud detection system by building cost-sensitive classifiers is the best way to enable recovery of large amounts of financial losses. Besides, the customer loyalty and trust will also be increased. Also, cost-sensitive classifiers have been shown to be effective in addressing the class imbalance problem (Thai-Nghe, Gantner, & Schmidt-Thieme, 2010; Zhou & Liu, 2006).

Most of the past studies work on constant misclassification cost matrices or cost matrices composed of a number of constant heterogeneous misclassification costs; however, each false negative (FN) has a unique misclassification cost inherent to it. Accordingly, each FN should be prioritized in some way to show the misclassification cost difference. For example, a fraudulent transaction with a larger transaction amount or a larger usable card limit should be more important to detect than one with a smaller amount or usable card limit. A constant cost matrix or a combination of constant cost matrices cannot depict this picture. So, this study is one of the pioneers to take such cases into account while working with classification problem under variable misclassification costs. This is one of the gaps in the literature of credit card fraud detection aimed to be filled by this study.

In this study, a new cost-sensitive decision tree induction algorithm that minimizes the sum of misclassification costs while selecting the splitting attribute at each non-terminal node of the tree is developed and the classification performance is compared with those of the traditional classification methods, both cost-insensitive and cost-sensitive with fixed misclassification cost ratios, such as traditional decision tree algorithms, ANN and SVM. The results show that this cost-sensitive decision tree algorithm outperforms the existing well-known methods on our real-world data set in terms of the fraudulent transactions identified and the amount of possible losses prevented.

In credit card fraud detection, the misclassification costs and the priorities of the frauds to be identified differ depending on the individual records. As a result, the common performance metrics such as accuracy, True Positive Rate (TPR) or even Area Under Curve are not suitable to evaluate the performance of the models because they accept each fraud as having the same priority regardless of the amount of that fraudulent transaction or the available usable limit of the card used in the transaction at that time. A new performance metric which prioritizes each fraudulent transaction in a meaningful way and checks the performance of the model in minimizing the total financial loss should be used. Fraudsters generally deplete the usable limit of a credit card once they get the opportunity of committing fraudulent transactions using the card. Accordingly, the financial loss of a fraudulent transaction can be assumed as the available limit of the card before the transaction instead of the amount of the transaction. So, the performance comparisons of the models over the test set are done over the newly defined cost-sensitive performance metric Saved Loss Rate (SLR) which is the saved percentage of the potential financial loss that is the sum of the available usable limits of the cards from which fraudulent transactions are committed. To show the correctness of our argument, True Positive Rate (TPR) values for the performance of the models are also given in performance comparisons of the models.

The rest of this paper is organized as follows: Section 2 gives a review of the cost-sensitive approaches in machine learning and Section 3 gives some insights to the structure of credit card data. Section 4 gives the details of the newly developed cost-sensitive decision tree algorithm. Section 5 gives the results and a short discussion about the results and Section 6 concludes the study.

2. Cost sensitive approaches in machine learning

There are different methods used to take cost sensitivity into account while building up classifier models. The first one builds up cost-sensitive classifier models by changing the training data distributions by oversampling or undersampling so that the costs of the data in the set are conveyed by the appearance of the examples. Some studies tried to overcome misclassification cost problem by stratification; and duplicating or discarding examples when the data set is imbalanced (Japkowicz, 2000; Kubat & Matwin, 1997). However, these researchers assume that the cost matrix entries are fixed numbers instead of record-dependent values. Researchers such as Domingos tried to build up mechanisms like MetaCost to convert cost-insensitive classifiers to cost-sensitive ones (Domingos, 1999; Elkan, 2001.

According to some studies, oversampling is effective in learning with imbalanced data sets (Japkowicz & Stephen, 2002; Japkowicz et al., 2000; Maloof, 2003). However, oversampling increases the training time and because it makes copies of examples of the minor class/classes, it may result in overfitting problem (Chawla, Bowyer, & Kegelmeyer, 2002; Drummond & Holte, 2003). Unlike oversampling, undersampling tries to decrease the number of examples of major class/classes so that a balance is achieved in the distribution of training set data with respect to classes. Some studies have shown that undersampling is good at handling the imbalanced data problem (Drummond & Holte, 2003; Japkowicz & Stephen, 2002; Japkowicz et al., 2000; Maloof, 2003).

The second method to take cost sensitivity into account while building up classifier models is adjusting the threshold toward inexpensive classes to make misclassification of expensive class

examples harder for minimizing misclassification cost (Langford & Beygelzimer, 2005; Maloof, 2003; Sheng & Ling, 2006; Zhou & Liu, 2006). Oversampling, undersampling and adjusting the threshold make no changes to the algorithms and can be used with almost all algorithms (Ma, Song, Hung, Su, & Huang, 2012). However, the former two make changes on the inputs of the modeling algorithms while the latter make changes on the outputs of the model built by the algorithms (Zhou & Liu, 2006). Just like adjusting the threshold, increasing the learning rate for expensive class used in the learning algorithm, if there is, enables the model to learn the examples with higher costs more than the ones with lower costs (Kukar & Kononenko, 1998; Wan, Wang, & Ting, 1999).

The last method to take cost sensitivity into account is modifying a cost-insensitive learning algorithm or defining a new cost-sensitive learning algorithm. If the algorithm is a decision tree-based one, this can be done via either making the splits in a cost-sensitive manner or pruning the tree in a cost-sensitive manner or applying an additional cost adjustment function. While many researchers uses different heuristics to grow cost-sensitive decision trees (Breiman, Friedman, Olshen, & Stone, 1984; Brodley, 1995; Draper, Brodley, & Utgoff, 1994), some used different techniques to prune traditionally grown decision trees using misclassification costs (Bradford, Kunz, Kohavi, Brunk, & Brodley, 1998; Knoll, Nakhaeizadeh, & Tausend, 1994).

3. Structure of credit card data

The credit card data used in this study are taken from a bank's credit card data warehouses with the required permissions. The past data in the credit card data warehouses are used to form a data mart representing the card usage profiles of the customers. The data in the data mart is used to form the training set used in the modeling phase and the test set used for testing the trained models.

The original data of the time period, 12 months, used to form the training set have about 22 million credit card transactions. The distribution of this data with respect to being normal or fraudulent is highly skewed. The 12 months period that is used to build our sample included 978 fraudulent records and about 22 million normal ones with a ratio of about 1:22,500. So, to enable the models to learn both types of profiles, we used stratified sampling to under sample the legitimate records to a meaningful number. We tried samples with different legitimate/fraud ratios. However, all the data belonging to the time period of next 6 months which includes about 11344000 transactions where 484 of them are frauds is directly included in the test set. All the transactions in the test set are scored by the classifier methods. The data distributions of the training and test set are given in Table 1.

The number of transactions for each card differs from one to another; but each transaction record is of the same fixed length and includes the same fields. Hand and Blunt give a description of the characteristics of credit card data (Hand & Blunt, 2001).

Though some of the customers may have more than one credit card, each card is taken as a unique profile because customers with more than one card generally use each card in a different customer profile with a different purpose. Every card profile consists of variables each of which discloses a behavioral characteristic of the card usage. These variables may represent the transaction patterns of the cards with respect to location, time or type of the place where the transaction takes place. Fraud detection systems use classifier models to detect fraudulent activities by identifying significant deviations from the given card usage profiles. These variables are derived not only from the transaction itself but also from the past transaction history of the card. We will be content with mentioning about the type of variables used, but regarding the privacy, confidentiality and security concerns, we are not allowed to talk on the full list of the variables. These variables have one of the five main variable types: all transactions statistics, regional statistics, merchant type statistics, time-based amount statistics and time-based number of transactions statistics. Some of the variables can be stated as Transaction Type, Merchant Category Code, POS Entry Mode, PIN Entry Capability, Card Type, Card Domain and Card Usage Country.

The variables in all transactions statistics type disclose the general card usage profile of the cards by their card holders in general. The variables in regional statistics type give the spending habits of the card holder with respect to geographical regions. The variables belonging to merchant type statistics show the usage of the card in different merchant categories by their card holders. The variables in time-based statistics types identify the usage profile of the cards with respect to usage amounts versus time ranges or usage frequencies versus time ranges. While evaluating a new transaction from a credit card, any deviations from the normal usage profile of the card identified by these variables can give a signal of the fraudulent usage. So, each of these variables are calculated for all transactions for each card and included in the training set.

Table 1
Data distribution w.r.t. classes.

Sets		# of records	
		Record count in population	Record count in sets
Training set	Normal	~22000000	8802
	Fraud	978	978
Test set	Normal	13644000	13644000
	Fraud	484	484

4. Cost-sensitive decision tree approach

One of the biggest problems of modeling a real world classification problem is the imbalanced data distribution problem where identifying the records belonging to the minority class is much more important than identifying the ones belonging to the majority class as in the case of credit card fraud detection. An effective way of overcoming the problem is the cost-sensitive modeling where misclassifying a minority class record costs more than misclassifying a majority class record.

In this paper, the details of a cost-sensitive decision tree induction algorithm developed to identify fraudulent credit card transactions are given. In the well-known decision tree algorithms, the splitting criteria are either insensitive to costs and class distributions or the cost is fixed to a constant ratio such that the cost of classifying a fraudulent transaction as normal (a false negative – FN) is "n" times the cost of classifying a normal transaction as fraudulent (a false positive – FP). Moreover, in these algorithms, misclassification cost is taken into consideration during the pruning process, not induction process. There are some previous studies done on cost-sensitive tree induction where the cost of misclassification depends only on the class (Drummond & Holte, 2000a; Drummond & Holte, 2000b; Ling, Sheng, & Yang, 2006; Liu, 2009) or on the individual example itself (Duman & Özçelik, 2011; Ling, Yang, Wang, & Zhang, 2004). Best of our knowledge, this is the first work to explore the specific combination of application of cost-sensitive decision tree induction algorithms on credit card fraud detection where misclassification costs are varying.

In credit card transactions, each fraudulent transaction creates a different cost. So, using a fixed misclassification cost for every fraudulent transaction does not fit our problem. Accordingly, we use a different cost for each transaction which is inherent in itself. Fraudsters generally spend all the available usable limit of a credit

card in the following transactions after they obtained the possibility to commit a transaction using the card unless they are not detected as they commit this first fraudulent one. They generally manage to achieve this in four or five transactions on the average (Duman & Özçelik, 2011). So, identifying a fraudulent transaction as legitimate actually costs as much as the available usable limit of the card used in that transaction. Hence, the misclassification cost of a fraudulent record is defined as the available usable limit of the card used in the transaction instead of the amount of the transaction or a predefined fixed amount of cost.

Moreover, this assumption makes a difference among the cost and the priority of each fraudulent transaction. In other words, detecting a fraudulent transaction committed with a card having a larger available usable limit saves more than a fraudulent one committed with a card having a smaller available usable limit. So, the priority of the detection of the first fraudulent transaction is higher than that of the second one. Thus, each false negative has a different misclassification cost and the performance of the model should be evaluated over a cost-sensitive metric such as the newly defined metric SLR which is percentage of the total amount of saved available usable limits instead of the metrics based on the number of frauds detected as given in (1).

$$SLR = \frac{\sum_{j=1}^{k}(C_{FN})_j}{\sum_{i=1}^{f}(C_{FN})_i}, \quad where\ \mathrm{k} = \text{number of frauds detected},$$
$$\mathrm{f} = \text{total number of frauds}, (\mathrm{C_{FN}})_{\mathrm{J}} = \text{misclassification cost of } \mathrm{FN_j} \qquad (1)$$

The classical decision tree models are not applicable to the case of the variable misclassification cost depending on the individual transaction. Thus, we developed a new cost-sensitive decision tree algorithm where the splitting criteria of the decision tree learning algorithm are influenced by changes in the individual misclassification costs.

The cost matrix used in our algorithm is given in Table 2. For the misclassification cost of normal transactions (C_{FP}), our algorithm assumes a fixed misclassification cost which is found by some ad hoc procedures and interviews with the bank staff with domain expertise. The algorithm takes the available usable limit of the card, used in the transaction, before the transaction as the misclassification cost for the fraudulent ones (C_{FN}).

The new cost-sensitive decision-tree learning algorithm defined here selects the splitting variable of a node, if a split is possible, based on the reduction of the total misclassification cost instead of reduction of impurity. We assume that FP is actually a normal transaction falsely classified as fraudulent, and FN is actually a fraudulent transaction falsely classified as legitimate. At the beginning, all the transactions in the training set are assigned to the root node of the tree. First of all, the cost of the node is calculated. In a decision tree, all the transactions in a node can be classified as either fraudulent or legitimate. So, both the total misclassification cost in the case of assigning the transactions of the node as fraudulent (C_P) and the total misclassification cost in the case of assigning the transactions as normal (C_N) are calculated. For the calculation of C_P and C_N, we used four different methods: CS – Direct Cost, CS – Class Probability, CS – Gini and CS – Information Gain.

Table 2
Cost matrix used.

Actual value	Predicted Value	
	Positive (fraudulent)	Negative (Legitimate)
Positive (Fraudulent)	True Positive (TP) (misclassification cost = 0)	False Negative (FN) (misclassification cost = C_{FN})
Negative (Normal)	False Positive (FP) (misclassification cost = C_{FP})	True Negative (TN) (misclassification cost = 0)

In CS – Direct Cost method, we do not integrate any impurity measure in the cost calculation function and find the best split by only using the reduction in total expected misclassification cost alone. This cost measure is inspired from Ling et al. (2004), Zubek and Dietterich (2002), Greiner, Grove, and Roth (2002). Instead of using an impurity measure to find the splitting variable, this method chooses the variable which enables the biggest reduction in the total misclassification cost. The total misclassification cost in the case of assigning the transactions as normal, C_N, is calculated to be the sum of the available usable limit of each fraudulent records ($(C_{FN})_i$) in that node. Assigning a legitimate transaction as fraud only results in an observation cost which is same for each legitimate transaction (C_{FP}). So, in this method, only the misclassification costs are used for tree induction and classification. Assume that there are "f" fraudulent records and "n" normal (legitimate) records those falling into a node where "N" ($N = f + n$) gives the total number of records in this node, C_P and C_N can be calculated as given below in (2) and (3):

$$C_N = \sum_{i=1}^{f}(C_{FN})_i \qquad (2)$$

$$C_P = n * C_{FP} \qquad (3)$$

Unlike CS – Direct Cost method where only the expected total misclassification cost is used in finding the optimum split for the current node, if there is any, regardless of the class distribution of the data or impurity measure in the node; the traditional decision tree induction techniques use the class distribution or a measure of impurity in some way to find the split for the next level. So, to add the effect of the class distribution and impurity in the node to the cost-sensitive splitting mechanism in the following methods, we modify the misclassification cost calculations used in searching for the split in a way inspired from the well-known traditional decision tree algorithms ID3, C5.0 and C&RT.

These traditional decision tree methods use impurity measures to choose the splitting attribute and the split value/s. ID3 (Prodromidis et al., 2000) uses entropy and information gain while the successor, C5.0 uses gain ratio and C&RT (Wheeler & Aitken, 2000) uses Gini index for impurity measurement. For a two-class case, expected information (entropy) and Gini Index can be calculated as given in (4) below:

$$a.\ Entropy = \sum_{i=1}^{2}(-p_i * log_2(p_i))$$
$$b.\ Gini = 1 - \sum_{i=1}^{2}(-p_i)^2$$
$$where\ p_i = \text{relative frequency of class } i \qquad (4)$$

In CS – Class Probability method, the relative frequency of the classes (class probabilities) are integrated in the cost calculation functions to add the effect of the class distributions to the node costs. As the relative frequency of a class in the node gets larger, there will be more records in the node belonging to this class. Thus, the records in this node would be classified to this class if a relevant impurity measure was used as in decision tree methods ID3 and C5.0. Because we decide on the label of the node according to the misclassification cost, we should multiply the cost of a class with the relative frequency of the other class so that we will decrease the cost of the class with higher relative frequency more. So, we will decrease the cost of choosing the class having higher frequency. By the way, we favor the class with higher frequency in the node. C_P and C_N in CS – Class Probability method can be calculated as given below in (5) and (6):

$$C_N = \left(\sum_{i=1}^{f}(C_{FN})_i\right) * \left(\frac{f}{n+f}\right) \qquad (5)$$

$$C_P = \left(\frac{n}{n+f}\right) * n * C_{FP} \tag{6}$$

In CS – Gini method, the square of class probabilities are integrated in the cost calculation functions to add the effect of the class distributions to the node costs in a different way inspired from the Gini index impurity measure used in C&RT. We multiply the cost of a class with the square of the relative frequency of the other class so that we will decrease the cost of the class with higher relative frequency more as in CS – Class Probability method. C_P and C_N in CS – Gini method can be calculated as given below in (7) and (8):

$$C_N = \left(\sum_{i=1}^{f}(C_{FN})_i\right) * \left(\frac{f}{n+f}\right)^2 \tag{7}$$

$$C_P = n * C_{FP} * \left(\frac{n}{n+f}\right)^2 \tag{8}$$

In CS – Information Gain method, negative logarithm of relative class frequencies are integrated in the cost calculation functions to add the effect of the class distributions to the node costs in a different way inspired from the information gain impurity measure used in ID3. Because logarithms of relative frequencies are non-positive values, we multiply with minus one to make it non-negative. C_P and C_N in CS – Information Gain method can be calculated as given below in (9) and (10):

$$C_N = -log\left(\frac{n}{n+f}\right) * \left(\sum_{i=1}^{f}(C_{FN})_i\right) \tag{9}$$

$$C_P = -log\left(\frac{f}{n+f}\right) * n * C_{FP} \tag{10}$$

After calculating the misclassification costs for each case, the case with the smaller cost is chosen as the misclassification cost of the node as given in (11). The transactions in the node are assigned to the class with the smaller total expected misclassification cost (N = Normal, F = Fraud). Because assigning a fraud as fraud and assigning a legitimate as legitimate have a zero misclassification cost, they are not included in the misclassification cost calculations. The node is labeled with the class label having minimum total misclassification cost as given in (12).

$$C_{Node} = min(C_N, C_F) \tag{11}$$

$$Label_{Node} = If\ (C_N < C_F) then\ N, else\ F \tag{12}$$

After finding the misclassification costs for each class, the class probabilities of being fraudulent or normal are found as given below in (13) and (14). Because the classification algorithms are based on misclassification costs, the probability of being in one class is larger when the misclassification cost of the other class is larger. Thus, the class which results in a smaller rate of misclassification cost is chosen as the class of the node. The larger the misclassification cost of a class in a node, the smaller the probability of the records in that node to belong in that class. So, there is an inverse relationship between the misclassification cost associated with a class and the probability (P) of that class.

$$P_{N-Node} = C_F/(C_N + C_F) \tag{13}$$

$$P_{F-Node} = C_N/(C_N + C_F) \tag{14}$$

Starting from the root node, each node is checked for the most suitable variable to be used in the split of the node, if a split is possible, to decrease the total misclassification cost as much as possible. The approach in splitting a node according to the variable type is as follows: Multi-split is used for the categorical variables, and binary-split is used for the numerical (range) variables. After finding the cost of each child node (C_{CN}) as mentioned above, the total cost (C_T) of child level after the split is found as given in (15) (assuming m child nodes after the split). Instead of using a formula like Gain Ratio, we prefer to directly divide to the number of child nodes after the split because (Liu, 2009; Sheng et al., 2006) state that such an approach only overcomes the disadvantages of information gain but also settles down the practical issue confronting gain ratio.

$$C_T = \left(\sum_{i=1}^{m}(C_{CN})_i\right)/m \tag{15}$$

The sum of the costs of the child nodes are divided by the number of child nodes after the split so that there will not be a bias to select the variables resulting in more split nodes than the ones resulting in fewer split nodes. If the total cost of the child level is less than the cost of the parent node; thus, there is a reduction in the misclassification cost, then this split is a candidate to be used. Every possible split for each input variable is used in the search for the candidate splits for the best cost reduction, and the split which gives the best cost reduction in the child level is chosen as the split for the node. If there is no candidate split which results in a cost reduction, or the number of transactions in a node is below the minimum number of transactions allowed, the parent node is assigned to be a leaf node. By the misclassification cost calculation of the nodes, not only the class of the node, but also the probabilities of the transactions in the node to be fraudulent and normal are also found.

5. Results and discussions

In real-world examples, most of the credit card operations management departments have a limited number of staff to monitor the fraud alerts. So, many fraud detection systems should show their best performance in a fixed number of fraud alerts. In our case, our data supplier bank can only check 8% of all transactions. So, we sort the records in the test set according to their fraud probabilities given by the classifier models and compare the performance of the models in the top 8% risky transactions in the test set.

Because the cost of each fraudulent record is different, each fraud should be prioritized according to its cost. Thus, detecting a fraud with a higher cost should become more important than detecting one with a lower cost. Accordingly, the performance of the model should be evaluated according to the misclassification costs which means that the common performance metrics such as accuracy or precision (or True Positive Rate – TPR) are not suitable to evaluate the performance of models where varying misclassification costs is in question such as this case. That is why the performance comparisons of the models over the test set are done over Saved Loss Rate (SLR) which depicts the saved percentage of the potential financial loss which is the available usable limit of the cards from which fraudulent transactions are committed. To show the correctness of our argument, TPR values for the performance of the models are also given with SLR values.

In this study, the models which show the best performance among the ones developed using the same method with different parameters are chosen and their performances are compared with the performance of the models built using the cost-sensitive decision tree algorithm defined in this study. Accordingly, six models are selected among the ones built using traditional decision tree algorithms implemented in SPSS PASW Modeler. These are the models built by using C5.0, CART, CHAID, CHAID with a fixed cost ratio of 5–1 (misclassifying a fraudulent record costs five times the misclassification of a legitimate one), Exhaustive CHAID (an extension of CHAID which performs merging and testing of predictor variables in detail), and Exhaustive CHAID with a cost ratio of 5–1.

Table 3
Statistical analysis of performances of ANN models.

Model			N	Mean	Std. dev.	Std. error mean
Group statistics						
SLR	Dynamic		10	86.89	2.85986	0.90437
	Quick		10	87.60	1.32077	0.41767
TPR	Dynamic		10	90.62	0.83373	0.26365
	Quick		10	90.60	0.54365	0.17192

		Independent samples test								
		Levene's test for equality of variances		*t*-test for equality of means						
		F	Sig.	*t*	df	Sig. (2-tailed)	Mean difference	Std. error difference	95% Confidence interval of the difference	
									Lower	Upper
SLR	Equal variances assumed	16.819	0.001	−0.713	18.000	0.485	−0.71000	0.99615	−2.80284	1.38284
	Equal variances not assumed			−0.713	12.672	0.489	−0.71000	0.99615	−2.86773	1.44773
TPR	Equal variances assumed	4.547	0.047	0.064	18.000	0.950	0.02000	0.31475	−0.64126	0.68126
	Equal variances not assumed			0.064	15.482	0.950	0.02000	0.31475	−0.64906	0.68906

Among the ANN models, the best two performances belong to the models built using the dynamic and quick networks implemented in SPSS PASW Modeler. In the quick method, a single feed-forward back propagation neural network is trained. By default, the network has one hidden layer containing max $(3 * (n_i + n_o)/20)$ neurons, where n_i is the number of input neurons and n_o is the number of output neurons. The network is trained using the back-propagation method. In the dynamic method, again a single feed-forward back propagation neural network is trained; however, the topology of the network changes during training, with neurons added to improve performance until the network achieves the desired accuracy. There are two stages to dynamic training: finding the topology and training the final network. The performances of Quick and Dynamic methods on test set are statistically compared over different 10 execution results for each one. There has been no statistically meaningful difference found between the performances of the two ANN methods with respect to both TPR and SLR. The details of the analysis are given in Table 3.

Also one model with the best performance among the models built using SVM methods is selected. The performances of all the chosen models are given in Table 4.

Among the chosen models built by using the traditional methods, the ANN models show the best performance in terms of frauds caught or TPR, and one in terms of SLR. However, three of the cost-sensitive decision tree model outperforms all the other models both in terms of TPR and SLR.

Our CS – Direct Cost method which only uses misclassification cost to build the tree shows the worst performance. Although previous studies stated that such approaches using only expected total misclassification cost outperforms many traditional cost-insensitive methods (Ling et al., 2004), our results given in Figs. 1 and 2 show that we can not only use misclassification cost for classification, but should depict the impact of class distributions or impurity of data in some way to our cost calculations.

Cost-sensitive decision tree models built using such a combination shows the best performances with respect to both TPR and SLR as given in Figs. 1 and 2. Though the TPR performances are close and the frauds caught difference is very small for the ANN models and the cost-sensitive decision tree models, there is a huge difference in the SLR performance of the models which means that there is a huge difference in the amount of financial loss recovered by these models. Again, CS – Direct Cost shows the worst performance w.r.t both TPR and SLR because it does not take any class distribution or impurity measure into account.

By comparing the performances of cost-sensitive decision tree models and the other models as given in Figs. 1 and 2, we can clearly state that the cost-sensitive ones except CS – Direct Cost saved much more financial resources than the others. Financial institutions generally worry about the total financial loss or recovery instead of the number of fraudulent transactions detected. So, the models built using these cost-sensitive approaches will satisfy their needs in total recovery amount. Furthermore, these cost-sensitive models also outperform the traditional classifiers in number of fraudulent transactions detected.

When the performance figures in Figs. 1 and 2 are examined, it is seen that though TPR performances of Dynamic and Quick models seem to be similar on average, SLR performances of the models are different and Quick models recover more on average. It is the same when the performance of Dynamic-Worst model and the performance of the C&RT model is compared. Although there is a 0.5% difference in their TPR performances, there is a 1% difference in the reverse way in their SLR performance which means a huge amount of money. Because this metric directly depicts the financial amounts recovered, the better performance means the more probable financial loss prevented. This shows that TPR is not a suitable performance metric for this problem and SLR or another metric based on the misclassification cost should be used in such classification problems.

Table 4
Performances of models.

Model	TP	TPR	SLR
Dynamic_Average	439	90.6	86.9
Dynamic_Best	445	91.9	90.7
Dynamic_Worst	433	89.5	83.7
Quick_Average	439	90.6	87.6
Quick_Best	443	91.5	89.6
Quick_Worst	433	89.5	86.0
C5.0	435	90.0	85.0
C&RT	431	89.0	84.7
CHAID	435	89.9	84.7
Exhaustive CHAID	435	89.9	84.7
SVM (Polynomial)	402	83.1	78.3
CS – Direct Cost ($C_{FP} = 30$)	361	74.6	73.3
CS – Class Probability ($C_{FP} = 50$)	446	92.1	94.9
CS – Gini ($C_{FP} = 5$)	449	92.8	95.8
CS – Information Gain ($C_{FP} = 25$)	448	92.6	95.2

C_{FP} = Cost of false positive.

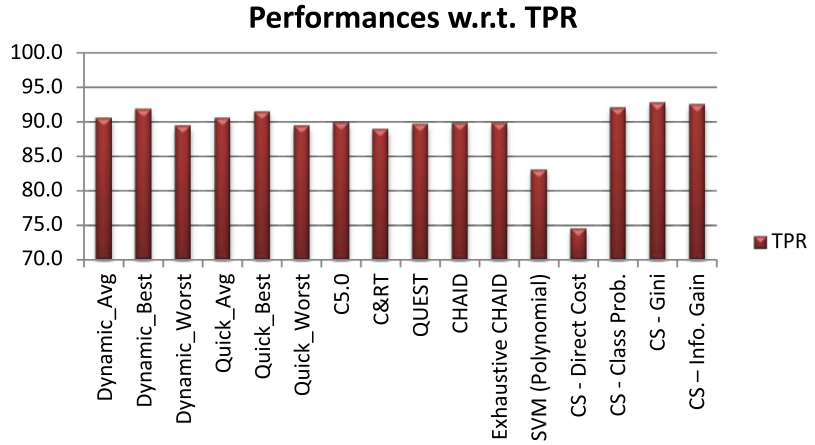

Fig. 1. Performances of models w.r.t. True Positive Rate (TPR).

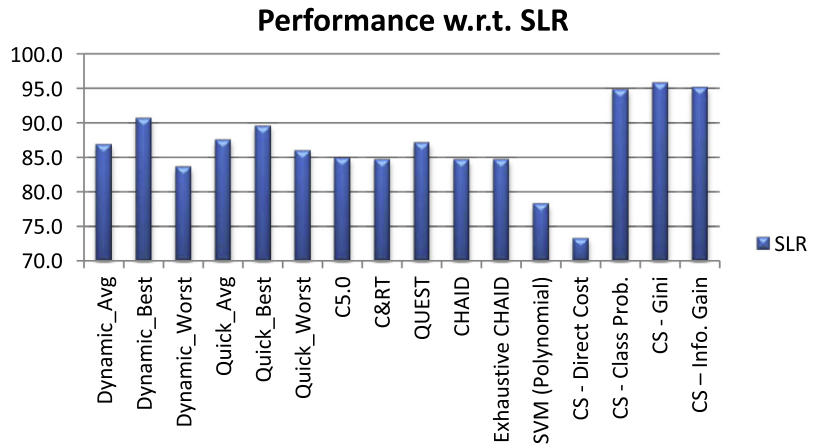

Fig. 2. Performances of models w.r.t. Saved Loss Rate (SLR).

6. Conclusions

In this work, we have developed and implemented a number of cost-sensitive decision tree approaches to be used in credit card fraud detection and show that it outperforms the models built using the traditional data mining methods such as decision trees, ANN and SVM. We propose a new approach for making the splits by taking variable misclassification costs depending on each individual record into account. The performances of the models are compared on a real-world data set showing that the models can be readily implemented in real-world systems.

We evaluated the proposed algorithm on a real world data set and demonstrated the conclusions which can be used as guidelines for working with a classification problem where variable misclassification cost depending on each individual example should be taken into account. We revealed that the well-known performance metrics such as accuracy and TPR are not suitable for this kind of problems, and developed a new performance metric for the credit card fraud detection problem which is the percentage of available usable limits saved.

The performances of classifiers built using CS – Direct Cost method on real-world test set indicate that we cannot use misclassification cost without incorporating the class distribution or an impurity measure in cost calculations. However, using our cost-sensitive approaches which incorporate such an information in the cost calculations make a significant improvement in the classification performance both with respect to TPR and the newly defined domain specific metric SLR over the best existing methods. We believe that this new metric will be a widely accepted and adopted in the future research studies in credit card fraud detection.

These performance improvements indicate many research contributions and managerial implications. First of all, research on classification in domains with imbalanced data such as fraud detection, specifically credit card fraud detection or medical diagnosis where misclassification costs highly differ among classes should focus on cost-sensitive classification to build classifiers which can prioritize minority class instances with big misclassification costs. Although there are many studies on cost-sensitive modeling in medical diagnosis, our study, as long as we know, is one of the pioneers to combine cost-sensitive modeling using decision trees and credit card fraud detection. Furthermore, it will serve as a guide for further studies in cost-sensitive classification. Secondly, by implementing such algorithms in real world fraud detection systems, financial institutions can save huge amounts of money when the financial losses due to fraud are considered. An improvement in classification performance less than 1% may result in millions of dollar savings. The performance difference among cost-sensitive models and traditional models in our test data corresponds to a huge amount. Such a performance improvement in larger banks with many customers will result in more savings. Accordingly, financial institutions are in a search for such improvements. Also, the loyalty and trust of customers in financial institutions will be enriched by the way.

Acknowledgment

This study was supported through funds provided by Scientific Research Unit of Marmara University under Project No: "FEN-C-DRP-211009-0320."

References

Aleskerov, E., Freisleben, B., & Rao, B. (1997). CARDWATCH: A neural network based data mining system for credit card fraud detection. *Computational Intelligence for Financial Engineering*, 220–226.

Bolton, R. J., & Hand, D. J. (2002). Statistical fraud detection: A review. *Statistical Science, 28*(3), 235–255.

Bradford, J. P., Kunz, C., Kohavi, R., Brunk, C., & Brodley, C. E. (1998). Pruning decision trees with misclassification costs. In *Proceedings of 10th European conference on machine learning* (pp. 131–136). Berlin.

Brause, R., Langsdorf, T., & Hepp, M., (1999). Neural data mining for credit card fraud detection. In *Proceedings of the 11th IEEE international conference on tools with artificial intelligence*.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification and regression trees*. Wadsworth International Group.

Brodley, C. E. (1995). Automatic selection of split criterion during tree growing based on node location. In *Proceedings of 12th international conference on, machine learning (ICML-95)* (pp. 73–80).

Chawla, N., Bowyer, L., & Kegelmeyer, W. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research, 16*, 321–357.

Chen, R., Chiu, M., Huang, Y., & Chen, L. (2004). Detecting credit card fraud by using questionnaire-responded transaction model based on SVMs. In *Proceedings of IDEAL2004* (pp. 800–806). Exeter, UK.

Chen, R.-C., Luo, S.-T., Liang, X., & Lee, V. C. S. (2005). Personalized approach based on SVM and ANN for detecting credit card fraud. In *Proceedings of the IEEE international conference on neural networks and brain* (pp. 810–815). Beijing, China.

Dahl, J. (2006). Card fraud. *Credit Union Magazine*.

Domingos, P. (1999). MetaCost: A general method for making classifiers cost-sensitive. *Knowledge Discovery and Data Mining*, 155–164.

Dorronsoro, J. R., Ginel, F., Sanchez, C., & Cruz, C. S. (1997). Neural fraud detection in credit card operations. *IEEE Transactions on Neural Networks, 8*(4), 827–834.

Draper, B., Brodley, C. E., & Utgoff, P. (1994). Goal-directed classification using linear machine decision trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 16*, 888–893.

Drummond C., & Holte, R. C. (2000). Explicitly representing expected cost: An alternative to roc representation. In *Proc. ACM SIGKDD, int'l conf. knowledge discovery and data mining* (pp. 198–207).

Drummond, C., & Holte, R. C. (2000). Exploiting the cost (in)sensitivity of decision tree splitting criteria. In *Proceedings of the 17th international conference on machine learning* (pp. 239–246).

Drummond, C., & Holte, R. C. (2003). C4.5, class imbalance, and cost sensitivity: why under sampling beats over-sampling. In *Proc. int'l conf. machine learning, workshop learning from imbalanced data sets II*.

Duman, E., & Özçelik, M. H. (2011). Detecting credit card fraud by genetic algorithm and scatter search. *Expert Systems with Applications, 38*, 13057–13063.

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceeding of the seventeenth international joint conference of artificial intelligence* (pp. 973–978). Seattle: Morgan Kaufmann.

Gartner Reports, from the World Wide Web: http://www.gartner.com, May 10, 2010.

Ghosh, S., & Reilly, D. L., (1994). Credit card fraud detection with a neural-network, In *27th Hawaii international l conference on information systems, vol. 3 (2003)* (pp. 621–630).

Greiner, R., Grove, A., & Roth, D. (2002). Learning cost-sensitive active classifiers. *Artificial Intelligence Journal, 139*(2), 137–174.

Hanagandi, V., Dhar, A., & Buescher, K. (1996). Density-based clustering and radial basis function modeling to generate credit card fraud scores. In *Proceedings of the IEEE/IAFE conference* (pp. 247–251).

Hand, D. J., & Blunt, G. (2001). Prospecting gems in credit card data. *IMA Journal of Management Mathematics, 12*, 173–200.

Japkowicz, N. (2000). The class imbalance problem: significance and strategies. In *Proceedings of the 2000 international conference on, artificial intelligence (ICAI'2000)* (pp. 111–117).

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis Journal, 6*, 429–450.

Juszczak, P., Adams, N. M., Hand, D. J., Whitrow, C., & Weston, D. J. (2008). Off-the-peg and bespoke classifiers for fraud detection. *Computational Statistics & Data Analysis, 52*(9), 4521–4532.

Knoll, U., Nakhaeizadeh, G., & Tausend, B. (1994). Cost-sensitive pruning of decision trees. In F. Bergadano & L. De Raedt (Eds.). *ECML 1994 LNCS* (Vol. 784, pp. 383–386). Heidelberg: Springer.

Kou, Y., Lu, C.-T., Sirwongwattana, S., & Huang, Y.-P., (2004). Survey of fraud detection techniques. In *Proceedings of the 2004 IEEE international conference on networking, sensing and control* (pp. 749–754). Taipei, Taiwan.

Kubat, M., & Matwin, S. (1997). Addressing the curse of imbalanced training sets: One-sided selection. In *Proceedings of the fourteenth international conference on machine learning* (pp. 179–186).

Kukar, M., & Kononenko, I. (1998). Cost-sensitive learning with neural networks, In *Proceedings of the 13th European conference on artificial intelligence* (pp. 445–449). Brighton, UK.

Langford, J., & Beygelzimer, A. (2005). Sensitive error correcting output codes. *Lecture Notes in Computer Science, 3559*, 21–49.

Leonard, K. J. (1993). Detecting credit card fraud using expert systems. *Computers & Industrial Engineering, 25*, 103–106.

Ling, C. X., Yang, Q., Wang, J., & Zhang, S. (2004). Decision trees with minimal costs. In *Proceedings of 2004 international conference on machine learning (ICML'2004)*.

Ling, C. X., Sheng, V. S., & Yang, Q. (2006). Test strategies for cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering, 18*(8), 1055–1067.

Liu, X. (2009). A benefit-cost based method for cost-sensitive decision trees. In *Proceedings of 2009 WRI global congress on intelligent systems* (pp. 463–467).

Liu, X. (2009). A benefit-cost based method for cost-sensitive decision trees. *Global Congress on Intelligent Systems*, 463–467.

Ma, G.-Z., Song, E., Hung, C.-C., Su, L., & Huang, D.-S. (2012). Multiple costs based decision making with back-propagation neural networks. *Decision Support Systems, 52*(3), 657–663.

Maloof, M. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. In *Proceedings of the ICML'03 workshop on learning from imbalanced data sets* (p. 63).

Mena, J. (2003). *Investigate data mining for security and criminal detection*. Amsterdam: Butterworth-Heinemann.

Phua, C., Lee, V., Smith, K., & Gayler, R. (2005). A comprehensive survey of data mining-based fraud detection research. *Artificial Intelligence Review*.

Prodromidis, A. L., Chan, P. K., & Stolfo, S. J. (2000). Meta-learning in distributed data mining systems: issues and approaches. In H. Kargupta & P. Chan (Eds.), *Advances of distributed data mining*. AAAI Press. Chapter 3.

Quah, J. T., & Sriganesh, M. (2008). Real-time credit card fraud detection using computational intelligence. *Expert Systems with Applications, 35*(4), 1721–1732.

Sahin, Y., & Duman, E., (2010). An overview of business domains where fraud can take place, and a survey of various fraud detection techniques. In *Proceedings of the 1st international symposium on computing in science and engineering* (pp. 542–549). Aydin, Turkey.

Saitta, L. (Ed.). (2000). *Machine learning – a technological roadmap*. Amsterdam, Netherlands: University of Amsterdam.

Schindeler, S. (2006). Fighting card fraud in the USA. In *Credit control* (pp. 50–56). House of Words Ltd.

Shen, A., Tong, R., & Deng, Y. (2007). Application of classification models on credit card fraud detection. In *International conference on service systems and service management*. Chengdu, China.

Sheng, V. S., & Ling, C. X. (2006). Thresholding for making classifiers costsensitive. In *Proceedings of the national conference on artificial intelligence* (Vol. 21(1), pp. 476–481). AAAI Press.

Sheng, V. S., et al. (2006). Cost-sensitive test strategies. In *Proceedings of the twenty-first national conference on, artificial intelligence (AAAI-06)*.

Stolfo, S. J., Fan, D. W., Lee, W., Prodromidis, A. L., & Chan, P. K. (1997). Credit card fraud detection using metalearning: Issues and initial results. In *AAAI workshop on ai approaches to fraud detection and risk management* (pp. 83–90). Menlo Park, CA: AAAI Press.

Stolfo, S. J., Fan, D. W., Lee, W., Prodromidis, A. L., & Chan, P. K. (1999). Cost-based modeling for fraud and intrusion detection: Results from the JAM project. In *Proceedings of the DARPA information survivability conference and exposition* (pp. 130–144). New York: IEEE Computer Press.

Syeda, M., Zhang, Y., & Pan, Y. (2002). Parallel granular neural networks for fast credit card fraud detection. In *Proceedings of the 2002 IEEE international conference on fuzzy systems* (pp. 572–577).

Thai-Nghe, N., Gantner, Z., & Schmidt-Thieme, L. (2010). Cost-sensitive learning methods for imbalanced data. In *The 2010 International joint conference on neural networks (IJCNN)* (pp. 1–8, 18–23).

Turney, P.D. (2000). Types of cost in inductive concept learning. In *Workshop on cost-sensitive learning at the seventeenth international conference on machine learning*. Stanford University, California.

Wan, C., Wang, L., & Ting, K. M. (1999). Introducing cost-sensitive neural networks. In *Proceedings of the second international conference on information, communications and signal processing* (pp. 1–4). Singapore: IEEE.

Wheeler, R., & Aitken, S. (2000). Multiple algorithms for fraud detection. *Knowledge-Based Systems, 13*(2/3), 93–99.

Zhou, Z. H., & Liu, X. Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering, 18*(1), 63–77.

Zubek, V. B., Dietterich, T. G. (2002). Pruning improves heuristic search for cost-sensitive learning. In *Proceedings of the nineteenth international conference on machine learning* (pp. 27–34). Sydney, Australia.