A Review on Novel Approach to Handle Imbalanced Credit Card Transactions

Sudhansu R. Lenka[1], Bikram K. Ratha[2], Biswaranjan Nayak[3]
[1] *Utkal University, Research Scholar, Bhubaneswar, 751004, India*
[2] *Utkal University, Reader & Head, Bhubaneswar, 751004, India*
[3] *Trident Academy of Technology, Bhubaneswar, 751024, India*

Abstract

Recently, most of the people are using credit and debit cards for every purchase and payment. So the excessive usage of these cards attracts the illicit to implement different techniques to create fraudulent activities against these transactions. As a result, each year billions of dollars are lost due to ineffectiveness of the fraud detection system. Credit card transactions are highly imbalanced, since most of them are genuine and very few are fraudulent. These imbalance transactions lead to a huge challenge for the machine learning and data mining algorithms. A single algorithm cannot accelerate the performance of the model, so ensemble of classifiers is the approach to handle such an issue. In this paper, we first provide different approaches to detect frauds, the methods to evaluate the performance and the challenges faced by the fraud detection model. Second, we present a comprehensive review on the imbalanced problem, state of the art on ensembles techniques, assessment measures to evaluate the algorithm performance and finally, we perform different comparison tests among the ensemble-based methods.The ensemble-based methods are classified into different categories to handle imbalanced fraudulent transactions where each method is grouped based on their working principle. The comparison tests of different methods have shown that the performance of the detection model can be improved by integrating random undersampling approaches with bagging or boosting methods. Additionally, the results justifies that ensemble-based methods are worthwhile in integrating the pre-processing techniques before learning the classifier.

Keywords – *Bagging, Boosting, Cost-sensitive learning, Ensembles, Imbalance data set, Credit card fraud.*

I. INTRODUCTION

Financial fraud is one of the major problems in the financial industries and government organizations. Most of the financial transactions take place either through online or offline mode and this accelerates the fraudulent activities through credit card transactions. Credit card fraud may take place in different ways, like stolen card fraud, Card not Present (CNP) fraud and application fraud [1]. In stolen card fraud, the fraudster tries to misuse another person's card without the owner's knowledge. In CNP fraud, fraudster needs only the card information to commit fraud. Through application fraud, fraudster tries to receive a card by giving false personal information to the bank.

Machine learning (ML) techniques are the suitable approaches to detect the credit card fraudulent transactions [3]. To improve the performance of the Fraud Detection System (FDS), it must satisfy certain characteristics. First, it must not block too many genuine transactions, second, it must deal with class imbalance problem and finally, it must be an automated system to detect fraudulent patterns. Credit card fraud detection has many challenges, some of these are: imbalanced dataset, publicly unavailability of real dataset and dynamic behaviour of the fraudsters. In this paper, we survey on the imbalanced nature of the credit card transactions and state-of-the-art solutions discussed to solve the issues. A data set is said to be imbalanced if the number of fraudulent transactions are much less than legitimate transactions. The imbalanced data stream significantly reduces the performances of most of the machine learning algorithms. When the classifier deals with the imbalanced data sets, it achieves high predictive accuracy for the majority class instances, while predict low accuracy for the minority class instances. Another important issue with the imbalanced data set is the performance evaluation metrics. Normally, the performance of the machine algorithms is measured using metrics like, overall predictive accuracy and error rate. But in case of imbalanced data sets these two parameters are inappropriate, since the prior probabilities of the majority and minority class instances are unequally distributed [9]. Many techniques have been proposed to address imbalanced class issue. These techniques can be categorized into three groups. The algorithm level approaches modify the existing algorithms, to correctly classify the minority (or positive) samples [10]-[12]. Data level techniques are pre-processing steps where the data samples are rebalanced to reduce the effects of imbalanced class distribution [13]-[15]. Finally, cost sensitive methods combine both the algorithm and data level approaches to reduce the misclassification cost during learning phase [16], [17]. An ensemble of classifiers is another approach to solve the imbalanced problems [28], [29].

Ensembles are proposed to increase the prediction level of a single classifier by training multiple classifiers on the same data set and integrating each individual outputs to a single class level. While integrating different classifiers, we aim to maintain their consistencies with the training data inorder to achieve better accuracy. In this paper, we propose an ensemble with varying data, where the classifiers are trained with different data set.

This remaining part of the paper is organized as follows. In Section II, we presents the related work, which includes descriptions about credit card FDS, major challenges in the detection model, different solutions to address this problem and different sampling methods. In Section III, we review the state of the art on ensemble-based learning to handle imbalanced distribution. Different assessment metrics for imbalanced distribution are discussed in Section IV. In Section V, we present the experimental observation for different ensemble-based learning algorithms along with their corresponding parameters and statistical tests to evaluate the algorithms. Results and experimental analysis of the different algorithms are discussed in Section VI. Finally, a conclusion and future work is drawn in Section VII.

II. RELATED WORKS

A. *Credit Card Fraud Detection*

Fraud detection is to identify that the incoming transaction belongs to fraudulent or genuine category by using a set of trained credit card transactions [4]. An efficient FDS should be accurate and cost-effective, i.e. the amount of cost needed to check the transaction behaviour should not be more than the loss due to fraud [2]. ML techniques [3] can more efficiently predict the incoming transactions as genuine or fraudulent. An automated detection model implements Expert Driven or Data Driven approaches to analyse the credit card transactions.

The Expert Driven approach works by following the feedbacks of the fraud investigators. Using their feedbacks the rules are defined and the model predicts the nature of the incoming transactions by following the rules. Expert Driven approaches are easier to develop and understand but they are applicable to a specific domain.

The FDS implementing the Data Driven approaches can detect the fraudulent patterns using supervised or unsupervised ML techniques. The advantages of such systems are: it can detect the fraudulent patterns by using all the features, can deal with large data stream and can predict the frauds implementing new strategies. However, the disadvantages are the system needs enough training data and in some cases the investigators may not understand the reason of an alert.

B. *Performance Measure for Fraud Detection*

*T*he performance of the fraud detection model can be evaluated by using area under the ROC curve (AUC) metrics [5]. The value of AUC metrics can be explained in the form of probability that the classifier assigns higher ranks to fraudulent transactions than legitimate transactions [6]. Average Precision (AP) is another ranking approach mostly implemented in fraud detection model [106]. Along with these measures, cost-based measures [7], [8] also frequently used in FDS. Cost-based measures represent the amount of monetary loss due to frauds in the form of cost matrix. But in the cost matrix the amount of maximum or minimum loss for a particular problem may change over time [18]. To handle this issue, a normalised cost [19] is proposed to access the monetary loss.

C. *Major Challenges in FDS*

The challenges to be considered while designing a fraud detection model are: i) Class Imbalance, ii) Concept Drift, iii) Small Sample Size, iv) Class Overlapping and v) Small Disjuncts.

- Class Imbalance: Credit card transactions data are highly imbalanced, since the numbers of fraudulent transactions are normally less than 10% of the total transactions [20]. Learning from imbalanced data sets is a major issue in case of supervised learning. The degree of class imbalance distribution reduces the performance of the classification. Classifier provides highly imbalanced degree of accuracy i.e. it achieves nearly 100% accuracy for classifying the majority class samples and for minority class the accuracy reaches 0-10% for instance [27]. So, a traditional machine learning algorithm have a bias towards the majority class instances because the rules correctly predict the majority instances, while ignoring the minority class instances and treating those data stream as noise. Two main techniques to handle class imbalance problems are: sampling methods and cost-based methods [21].

Sampling methods rebalance the class distribution in the training sample before implementing the classification. Sampling methods follows two rebalance approaches; either undersamples the majority class instances in the training set, or oversamples the training samples by duplicating the minority class instances [22]. SMOTE [14] is an oversampling approach which generates synthetic training samples from minority class by interpolation.

Cost-based methods assign different costs for misclassification and more cost is assigned to the misclassification of minority class [18]. In case of credit card FDS, the misclassification cost is proportional to the transaction amount [7], [8] and assigns larger cost to false negative cases (i.e. the classifier may generate false alarm but never take risk to misclassify a fraud as genuine.

In such imbalanced problem domains the classifier should provide high degree of accuracy for the minority class samples without neglecting the accuracy of the majority class samples. The

traditional evaluation metrics like the overall accuracy and the error rate does not provide necessary information due to imbalanced class distribution. Therefore, the performance of the classifiers in such imbalanced data sets can be evaluated using the assessment metrics like Receiver Operating Characteristics (ROC) curve, precision recall curves and cost curves. In Section 4 of this paper has discussed in detail about this evaluation metrics.

- Concept Drift: In FDS, the behaviour of the legitimate users as well as fraudster's changes with time and this principle is known as concept drift [40]. In financial transactions, the card holder behaviour can be analysed from two features i.e. transaction amount and the frequency of transaction. These two features does not remain fixed for a particular card holder, it varies with time due to the users life style and the availability of resources. Concept drift phenomenon mainly refers to supervised learning scheme, where the relation between the input vectors and targeted output varies with time [40]. So, the fraud detection system can handle concept drift issues by using adaptive learning algorithms. These algorithms can be used to update the detection model inorder to handle new fraudulent techniques [40].

When the data sets are characterized by both concept drift and imbalanced class distribution, then it can be handle by integrating ensemble approaches and resampling techniques [23], [24]. Another solution to overcome such problem is oversampling the minority class instances with time [24] and undersampling majority class.

- Small sample size: Due to inadequate number of fraudulent samples, it is difficult to detect pattern uniformity in minority class samples. In the paper [33], it was shown that the misclassification rate in imbalanced class distribution can be reduced if the number of minority class instances is representative i.e. keeping imbalanced ratio fixed. So, the patterns of minority class (or positive) can be learned in a better way despite theimbalance class distribution. But in reality the data set the ratio does not remain fixed. The work of [34] shown that by increasing the training sample size, reduces the misclassification rate of the imbalance class distribution.
- Class overlapping or class separability: Class overlapping is one of the major problems in imbalance class distribution. It is also known as class separability and it corresponds to the degree of separability between the classes of the data set [35]. So the classifier finds difficult to separate the minority class instances from the majority class. In case of financial transactions, overlapping of data means when the model treated the fraudulent transaction as genuine and vice-versa. This is because the fraudsters are implementing new approaches which are very close to genuine approach, so the detection model treats the fraudulent transaction as genuine [80]. Therefore, a fraud detection model to handle overlapping data issues must implement a suitable classifier and proper methods as in [36]. From the experimental results of [25], it is concluded that the class imbalance distribution can be handled, but when the classes are highly overlapped then the number of correctly classified minority class samples reduces.

D. Tackling Imbalanced Data sets

To deal with the imbalanced data sets, a large number of techniques have been proposed. As mentioned in the introduction, these techniques can be categorized into three groups: Algorithm level approaches, Data level approaches and Cost-sensitive learning approaches.

- Algorithm level approach: These approaches are used to modify the existing machine learning algorithms and biasing the learning phase towards the minority class instances [41]-[43]. This method requires a good understanding of the classifiers and identifies the reasons for its failure in mining the imbalanced data distributions. The standard way to handle the imbalance class distribution problem is to select an appropriate inductive bias. For example, we can handle such situations in decision trees algorithm either by adjusting the probability estimation at the leaf nodes [10], or by developing new pruning techniques [11]. Similarly for SVMs, to address this problem we can assign different penalty for different classes [42], or modify the class boundary according to the kernel function [64]. For association rule mining, different minimum supports are assigned to different classes due to skewed data set [41].
- Data level approach: It is a pre-processing approach which rebalances the data set before classifying the instances. In FDS, most of the researchers proposed data level balancing techniques by implementing oversampling and undersampling approaches[81],[91]. Balancing can be done either by replicating minority class instances (oversampling) or by removing the majority class instances (undersampling). But randomly using such resampling techniques may leads to removal of useful samples or it may generate irrelevant new samples. Hence, new advanced techniques were proposed in the research to keep the ratio of classes intact and/ or generate new samples according to the given class distribution [45].
- Costsensitive learning approach: In cost sensitive learning scheme, when a classifier misclassifies an instance then as a penalty different costs are assigned. These costs are represented in the form of a matrix, called cost matrix C [44]. Let the entry C (i, j) in the matrix denotes the cost of predicting an instance of class 'i' as class 'j'. Similarly, it denotes C (+,-) when the classifier predicts a fraud transactions as genuine and C (-, +) for the reverse case. But in case FDS, the model must give more emphasis on fraudulent instances as

compared to genuine. That means, it must assigns higher costs for misclassifying a fraudulent instances (False Negative) than genuine instances (False Positive), i.e. C (+, -) > C (-, +) and for correct predictions zero cost is assigned, i.e. C (+, +) = C (-, -) = 0. But in real-world problems it is very difficult to assign the actual values in the cost matrix.

E. Data Preprocessing Methods

In some of the papers it has been shown that rebalancing the class distribution at the processing step is usually a constructiveapproach [13], [46]. These resampling techniques rebalance the data set before the implementation of the classifier algorithms. Resampling techniques can be implemented in three levels. In undersampling approaches a new subset of data samples are generated by removing majority class instances. In oversampling methods a superset of the original data set is created by replicating some instances of the minority class and finally, hybrid resampling approaches which combines both the above sampling methods. There are several approaches existing under these categories, but we will focus only on the methods that can be implemented in combination with ensemble-based classifiers.

- Random undersampling: It is a resampling approach the class distributions are balanced by randomly removing the majority class (or genuine) instances. But one of the major disadvantages of this approach is that we may eliminate the useful information which could be used in the next step.
- Random oversampling: Similarly, in this approach the class distribution are balanced by randomly replicating the minority class instances, but it may leads to overfitting [47], since it may generate the duplicateminority class instances.
- Synthetic Minority Oversampling Technique (SMOTE): It is another oversampling approach, which creates new minority class instances and adding those samples near the instances of the same class. In FDS, SMOTE is anotherbetter approach to handle imbalanced class distribution [5].
- Modified Synthetic Minority Oversampling Technique (MSMOTE) [48]: In this modified version of SMOTE, the algorithm divides the minority class samples into three groups, safe, border and latent noise samples. The data samples are grouped based on the distances among all samples. In this case, when a new sample is generated, then the nearest neighbour is selected from the same group. For safe data samples, the algorithm randomly selects an instance from the K-Nearest Neighbors (KNN), for border instances it only selects the nearest neighbour and for latent noise data samples, it does not select any instances from the data set [49].
- Selective Preprocessing of Imbalanced Data (SPIDER): In this approach, the original data set is created by integrating the oversampled minority (fraudulent) class instances with the correctly classified majority (genuine) class instances [50]. It works in two phases, first identifies the instances that are misclassified using KNN algorithm.In the second phase, it rebalances the data set by using three different options like, weak, relabel or strong [49]. If it comes under weak category, it rebalances the data set by adding morerare class instances; for relabel category, it increases the rare class instances as well as relabels the majority class instances; and finally for strong category, rebalances is done by adding more number of minority class (or fraudulent) instances.

III. STATE OF THE ART SOLUTION TECHNIQUES TO HANDLE IMBALANCED DATA SETS

In this section, the researchers have discussed a set of classical learning algorithms to construct sets of classifier with the modality of classifiers properly complement each other. Then a new taxonomy has been utilized that is based on ensemble methods keeping in mind to handle imbalanced data sets. So, for structure representation, we have divided the explanation in two segments as:

A. Descriptions of ensemble of classifiers with different techniques.
B. Different approachesto handle class imbalance problem with ensemble of classifiers.

A. Descriptions of ensemble of classifiers with different techniques

Actually, the primary intension of ensemble methodology is to increase the performance of a classifier by incorporating a set of classifiers and combining their predictions to procure a new classifier which may able to outperform all of them. Here, the main objective is to generate multiple classifiers from the original data and for test cases their predictions can be integrated to generate a single class level. The main objective of combining several classifiers in ensemble methodology is to improve the overall predictions, since each of the classifiers are trained on different parts of the input instances, so the misclassified instances may not be same for all [97]. As per the researchers work specified in [28], [51] and, [52], ensemble-based classifiers means combining similar types of classifiers and that can be represented as a multiple classifier systems. But here, the focus is laid on ensembles whose classifiers are formed by modifying the original data-set.

In [53] and [54], the researchers studied on different classifiers to develop an ensemble-based model using the concepts of bias-variance decomposition and related ambiguity decomposition [55].Actually, a bias can be defined as a measure which can correctly generalize a test instances, similarly the variance can be defined as a measure of the extent to which the classifier's prediction is sensitive to the trained data. Since the variance causes overfitting, so the performance of the ensembles can

be improved by reducing the variance of the classifiers. In contrast, ambiguity decomposition shows that, by takingthe combination of several classifier predictions yields better performance on an average, than the method of selecting any one of the classifiers at random. However, these concepts are mainly applicable to regression problems because the output of such problems is real-valued and the mean squared error is computed as the loss function. But as per the researchers work specified in [56] and [57], these concepts could not find its strong foot hold. Also different researchers provide different assumptions as specified in [58], [59] for which no consensus arises for generalized loss functions [60].

In [61], [62], many researchers have shown that to form an ensemble, diversity among classifiers is an important criteria and another important functionality is that the base classifiers should be weak learners. Actually a learning classifier is termed as weak, when a slight modification in the data set reflects a large change in the induced model. Due to this reason most of the commonly used base classifiers implements tree induction algorithms.

In a weak learning algorithm, for construction of an ensemble different types of techniques can be implemented. The most commonly used ensemble learning methods are AdaBoost [30], [31] and Bagging [32] and their implementations have improved significantly in several classification problems [63].

- Bagging: Breiman [32] design an ensemble model using the concept of bootstrap aggregation. The model consists of several classifiers which gets trained by randomly taking the instances (with replacement) from the original trained data set, i.e. the main objective is to maintain the original data sample size intact. Since the classifiers get trained through different data-sets, as a result diversity is obtained. When a test sample is provided to each classifier, the majority or weighted vote is used to predict the class label. The pseudo code of Bagging is depicted in Fig. 1.

Algorithm 1 Bagging
Input: S: Training set; T: Number of iterations;
n: Bootstrap sized; I: Weak learner
Output: Bagged classifier:
$H(x) = sign\left(\sum_{t=1}^{T} h_t(x)\right)$ where $h_t \in [-1, 1]$ are the induced classifiers

1: **for** t = 1 to T **do**
2: $S_t \leftarrow$ RandomSampleReplacement (n, S)
3: $h_t \leftarrow I(S_t)$
4: **end for**

Fig 1: Bagging Algorithm

The modified Bagging techniques called pasting small votes which is specially designed to handle large data sets [67]. Due to smaller memory capacity the large data sets are partitioned into smaller data samples and each sample are provided to the classifiers as a training instance.It can be implemented in two different ways i.e. Rvotes which randomly generates the data subsets and Ivotes which creates consecutive data-sets based on the importance of the instances. The samples that show improvement in diversity are treated as important samples. The data set must consist of easy and difficult instances. The instances that are misclassified by the ensemble classifier are treated as difficult and these are detected by out-of-bag classifiers [32]. For better performance of the algorithm, the model includes the misclassified (or difficult) instances to the consecutivedata subset, as a result the easy instances gets lower priority to be included in the subset.

- Boosting: The researcher Schapire [32] has introduced Boosting in 1940, which is also known as ARCing, adaptive resampling and combining. In his work, he has proved that that a weak learner can be converted into a strong learner by using probably approximately correct (PAC) learning method. AdaBoost is one of the significant algorithm comes under this category, which is regarded as one of top ten data mining algorithms as discussed in [68]. The researchers have shown in [69] that apart from variance it reduces bias. In [70], the researchers have also shown that it supports SVM (Support Vector Machine) which helps to boost the margins. The functionality of AdaBoost is that it uses the entire data-set to train each classifier in a serial manner. The main objective of the algorithm is to correctly classify the misclassified instances. So, after each round the algorithm increases the priority of those misclassified instances inorder to correctly classify them in the next round.It gives more effort to the samples that are difficult to classify and the effort is measured in terms of weight. Initially equal weights are assigned to all the samples and subsequently the weights of misclassified instances increases, as a result the weights of correctly classified instances deceases. The AdaBoost weighting technique is same as resampling the data space using oversampling and undersampling approaches. Hence, it implements data-level approaches, which are very effective techniques to solve the class imbalance problem. In imbalanced data set, the traditional algorithms perform poorly on fraudulent instances due to bias error. So, AdaBoost has the ability to reduce the learning bias in class imbalance problem domains. The pseudocode of AdaBoost Algorithm is depicted in Fig. 2

Algorithm 2 AdaBoost

Input: Training set S = {x_i, y_i}, I = 1,.. N; and $y_i \in \{-1, +1\}$; T: Number of iterations; I: Weak learner

Output: Boosted classifier: $H(x) = \text{sign}(\sum_{t=1}^{T} \alpha_t h_t(x))$ where h_t, α_t are the induced classifiers (with $h_t(x) \in \{-1, 1\}$) and their assigned weights, respectively

1: $D_1(i) \leftarrow 1/N$ for i= 1,…, N
2: **for** t= 1 to N **do**
3: $h_t \leftarrow I(S, D_t)$
4: $\epsilon_t \leftarrow \sum_{i, y_i \neq h_t(x_i)} D_t(i)$
5: **if** $\varepsilon_t > 0.5$ **then**
6: T← t −1
7: **return**
8: **end if**
9: $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)$
10: $D_{t+1}(i) =: D_t(i).\, e^{(-\alpha_t h_t(x_i) y_i)}$ for i = 1,…, N
11: Normalize D_{t+1} to be a proper distribution
12: **end for**

Fig 2: Pseudocode for AdaBoost Algorithm

AdaBoost technique is further classified into two categories: AdaBoost.M1 and AdaBoost.M2 [31]. The AdaBoost M1 classification algorithm designed to handle multiclass imbalance problem domains using different weight changing mechanism. The AdaBoost M2 algorithm is also has the ablility to handle multiclass imbalance problems using base classifiers' confidence rates. Neither of these two algorithms can individually handle imbalance problems, rather they combinedly focus to correctly identify the majority class instances.

B.Different approaches to handle class imbalance problem with ensemble of classifiers

Recently, ensemble of classifiers has been proposed as one of the appropriate solution to the class imbalance problem which is discussed by the researchers in [49], [65] and [66]. We categorized the ensemble learning methods into two approaches; cost-sensitive boosting and data preprocessed ensembles. Cost-sensitive boosting techniques are equivalent to cost-sensitive learning, where costs are associated to boosting algorithms. On the other side, we kept the algorithms into one category that have a common characteristic, where all of them embed data preprocessing technique in ensemble learning methods. Therefore, in the second category boosting and bagging-based ensembles and hybrid ensembles are incorporated. The entire classifications are depicted in Fig. 3.

- Cost-sensitive Boosting: Cost-sensitive boosting algorithm implements the learning technique of AdaBoost by including the cost items into weight update formula (line 10 of Algorithm 2). AdaBoost is an accuracy-oriented algorithm, in case of imbalance class problems the learning (weighting) strategy is bias towards the majority class instances, since the entire accuracy of the algorithm depends upon it. In three different ways we can include the cost items into the weight update formula of AdaBoost. Based on these weight update formula, the cost-sensitive boosting algorithms are classified into AdaCost [79], AdaC1, AdaC2 and AdaC3 [71].

i. AdaCost: In this algorithm, weight update is done by adding a cost adjustment function Ø. If the instance is misclassified, then this function increase its weight more for an instance with a higher cost factor otherwise the weight is decreased. Assume C_i be the cost of misclassifying an instance i, so the cost function for positive class $Ø_+ = -0.5C_i + 0.5$ and for negative class $Ø_- = 0.5C_i + 0.5$. The weight function and α_t are recomputed by the following equations:

$$D_{t+1}(i) = D_t(i).e^{-\alpha_t y_i h_t(x_i) \varphi_{sign(h_t(x_i), y_i)}} \quad (1)$$

$$\alpha_t = \frac{1}{2} \ln \frac{1 + \sum_i D_t(i).e^{-\alpha_t y_i h_t(x_i) \varphi_{sign(h_t(x_i), y_i)}}}{1 - \sum_i D_t(i).e^{-\alpha_t y_i h_t(x_i) \varphi_{sign(h_t(x_i), y_i)}}} \quad (2)$$

ii. AdaC1: In this case, the costs (C_i) are embedded into the exponenet part of the equation:

$$D_{t+1}(i) = D_t(i).e^{-\alpha_t C_i y_i h_t(x_i)} \quad (3)$$

The update weight parameter $\alpha_t$is reevaluated as :

$$\alpha_t = \frac{1}{2} \ln \frac{1 + \sum_{i, y_i = h_t(x_i)} C_i D_t(i) - \sum_{i, y_i \neq h_t(x_i)} C_i D_t(i)}{1 - \sum_{i, y_i = h_t(x_i)} C_i D_t(i) + \sum_{i, y_i \neq h_t(x_i)} C_i D_t(i)} \quad (4)$$

where, $C_i \epsilon [0, +\infty]$

iii. AdaC2: In this algorithm, the costs also embedded with the weight update formula. But in this case, the costs are associated outside the exponent part in the following equation:

$$D_{t+1}(i) = C_i D_t(i).e^{-\alpha_t h_t(x_i) y_i} \quad (5)$$

α_t is recomputed as:

$$\alpha_t = \frac{1}{2} \ln \frac{\sum_{i, y_i = h_t(x_i)} C_i D_t(i)}{\sum_{i, y_i \neq h_t(x_i)} C_i D_t(i)} \quad (6)$$

iv. AdaC3: In this algorithm, the weight updation formula is changed by associating costs both inside and outside the exponent part of the equation:

$$D_{t+1}(i) = C_i D_t(i).e^{-\alpha_t C_i h_t(x_i) y_i} \quad (7)$$

Similarly, α_t is recomputed as:

$$\alpha_t = \frac{1}{2} \ln \frac{\sum_i C_i D_t(i) + \sum_{i, y_i = h_t(x_i)} C_i^2 D_t(i) - \sum_{i, y_i \neq h_t(x_i)} C_i^2 D_t(i)}{\sum_i C_i D_t(i) - \sum_{i, y_i = h_t(x_i)} C_i^2 D_t(i) + \sum_{i, y_i \neq h_t(x_i)} C_i^2 D_t(i)} \quad (8)$$

- Boosting-based Ensembles: The algorithms that embed techniques for data preprocessing in the boosting methods are included in this family. The algorithms such as SMOTEBoost [82], MSMOTBoost [48], RUSBoost [65] and DataBoost-IM [83] are included in this family.

i. SMOTEBoost and MSMOTBoost: Here, both the algorithms incorporate synthetic instances by utilizing SMOTE and MSMOTE data preprocessing techniques respectively and it is found that the weights of the new instances are proportional to the total number of instances in the new data-set. As a result, their weights always remains the same for each iterations and for all new data instances. At the other side, original data-set's instances weights are made normalized in such a manner so that they form a distribution with the new instances. After training the classifier, the weights of the original data-sets are updated and again sampling techniques are implemented to update the weights. As a result, the training data gets more diversity and which helps the ensemble learning classifier to outperforms.

ii. RUSBoost: Its working principle is similar to SMOTEBoost. The difference is that it randomly eliminates the majority class instances by implementing the under sampling approaches in each iteration. So, here, new weights to the instances are not required to be assigned. It works by simply normalizing the weights of the remaining instances in the new data-set with respect to the summation of weights. The remaining procedure is equivalent to SMOTEBoost.

iii. DataBoost-IM: It is a combination of AdaBoost.M1 algorithm and data synthesis algorithm. In this algorithm the difficult samples are identified as seeds and then it performs rebalancing procedure for both the classes.

- Bagging-based Ensembles: Bagging ensembles are widely applied to many class imbalance problems because of its simplicity and generalization ability. Actually, the specialty of a bagging algorithm is that it does not require updating the weights. Hence, in the algorithm, it neither requires a weight update formula nor any modifications in the computation of the algorithm. In this method, the focus is based on how effectively collect each bootstrap replica as shown in Step 2 of Algorithm 1 (Fig.1), i.e. to handle the class imbalance problem the algorithm must obtain a good classifier in each iteration without neglecting the significance of the diversity The four main algorithms in this family are OverBagging [72], UnderBagging [84], UnderOverBagging [72], and IIVotes [85].

i. OverBagging: To handle the class imbalance problems, in each bag, the classes of different instances are required to be considered when they are randomly drawn from the original data-set. Hence, instead of carry out random sampling for the whole data-set, before training each classifier an oversampling process can be carried out which is known as OverBagging and it can be done in two ways. In the first case, the new bootstrap can include all majority class instances and in the second case, resample them so that diversity can be increased. As it is proposed in [72], by utilizing SMOTE Bagging preprocessing algorithm all minority class instances can be taken care of.

ii. UnderBagging: In this approach, undersampling is used instead of oversampling which can be also it can be functioned in two ways i.e. i) it can be applied to a majority class and ii) in order to obtain a priori more diverse ensembles, a resampling with replacement of the minority class can also be utilized. UnderBagging is also used with altered names where the the same functional structure can be maintained which are proposed by researchers as Asymmetric Bagging in [73] and QuasBagging[74]. The researchers proposed roughly-balanced Bagging in [75] that possesses same approach as UnderBagging, but it does not utilize a totally balanced bag concept. Another approach proposed by researchers which is known as Partitioning in [76], [77]. It is also proposed as Bagging Ensemble Variation [78]) where, the instances of the majority classes are separated into disjoint data-sets and individual classifier is get trained with any of the bootstraps used.

iii. UnderOverBagging: It is similar to SMOTEBagging approach which is used to create each bag. Here, both oversampling and undersampling approaches are utilized. The first classifiers are get trained by using less number of instances with respect to the last instances so that diversity can be kept intact.

iv. Imbalanced IVotes (IIvotes): It integrates both the SPIDER data preprocessing technique along with IVotes. The advantage of this method is that the number of bags are not required to define, because when the out-of-bag error estimation not decreases, automatically the algorithm is stopped.

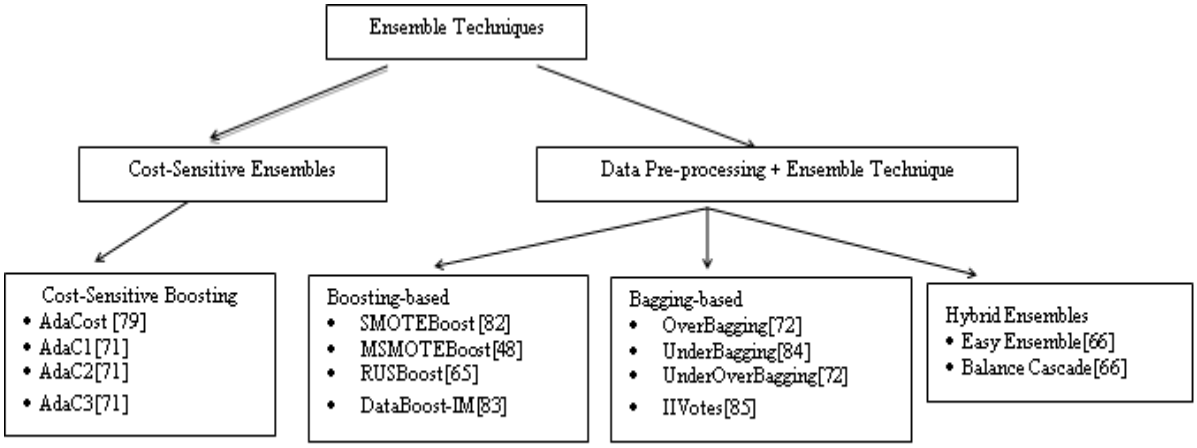

Fig.3. Classification of Ensemble techniques

- Hybrid Ensembles: As compared to other three category algorithms, these algorithms carry out two fold ensemble learning techniques, where boosting and bagging techniques are combined by utilizing through a preprocessing technique. The researchers proposed both EasyEnsemble and BalanceCascade algorithms as discussed in [66].

i. EasyEnsemble: In this method, after each AdaBoost iteration, the instances from the original data-set is kept intact without carrying out any operation, to train all the classifiers in parallel. The other case is that, this approach can be represented as UnderBagging techniqueif the base learner utilizes AdaBoost techniques. If the number of classifiers are kept fixed, then it will train less bags as compared to UnderBagging technique, where more classifiers will get chance to utilize each single bag.

ii. Balance Cascade: As this approach performs in a supervised manner, the classifiers are required to be sequentially trained. So, in each bagging iteration, after learning the AdaBoost classifier, the majority class examples which are correctly classified with a higher confidence factor by the current trained classifiers are then removed from the data-set, in turn they are not considered for further iterations.

IV. ASSESSMENT METRICS FOR IMBALANCED LEARNING

To assess the effectiveness of the imbalanced learning algorithms standardized evaluation measures must be used. In these problem domains, the performance of the classifier to identify minority class (or fraud) instances is usually very poor. To improve the performance, the learning objective must be: (1) balance the class distribution; and/or (2) improve accuracy for fraud instances. In this section, we present a brief description about major assessment metrics for imbalanced learning algorithms.

Table1. Confusion Matrix

	Predicted as Positive	Predicted as Negative
Actually Positive	True Positive (TP)	False Negative (FN)
Actually Negative	False Positive (FP)	True Negative (TN)

A. *Singular Assessment Metrics*

In classification problems, accuracy and error rate are very often used to measure the performance of the algorithms. A confusion matrix (Table 1.) gives information about actual and predicted results of a classifier. The information's in the confusion matrix are defined as:

$$True\ Positive\ Rate\ (TP_{rate}) = \frac{TP}{TP+FN}$$

$$True\ Negative\ Rate\ (TN_{rate}) = \frac{TN}{TN+FP}$$

$$False\ Positive\ Rate\ (FP_{rate}) = \frac{FP}{TN+FP}$$

$$False\ Negative\ Rate\ (FN_{rate}) = \frac{FN}{TP+FN}$$

Following the confusion matrix information's, accuracy and error rate are defined as:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (8)$$

$$Error\ Rate = 1 - Accuracy = \frac{FP+FN}{TP+FP+FN+TN} \quad (9)$$

But if the data sets are highly imbalanced, then accuracy and error rate are the not the right choice to measuring the performance of classifiers. Instead of these two metrics, other frequently used evaluation metrics to access imbalanced learning performance are precision, recall, F-measure and G-mean. These metrics are defined as:

$$Precision\ (P) = \frac{TP}{TP+FP} \quad (10)$$

$$Recall\ (R) = \frac{TP}{TP+FN} \quad (11)$$

$$F - Measure = \frac{(1+\beta)^2 . Recall . Precision}{\beta^2 . Recall + Precision} \quad (12)$$

Where, β is a parameter to adjust the relative importance of precision versus recall (usually, $\beta = 1$)

$$\Rightarrow F - Measure = \frac{2RP}{R+P} \quad (13)$$

$$G - mean = \sqrt{\frac{TP}{Tp+FN} \times \frac{TN}{TN+FP}} \quad (14)$$

Intuitively, precision is used to evaluatethe exactness (i.e., out of the retrieved positive instances, how many are really positive), whereas recall is used to evaluate the completeness (i.e. how many positive samples are predicted correctly). The two metrics behave just like accuracy and error, but both precision and recall are not sensitive to imbalance class distribution. From the above formulas it shows that precision (Eq.10.) is related to class distributions, while recall (Eq. 11.) is not. However, proper use of these two metrics helps to evaluate the performance of the classifier in imbalanced learning phase. F-measure metric (Eq. 12) is another approach to measure the performance of an algorithm by taking fraction of the weighted importance on recall or precision as determined by the β parameter. As a result, F-measure is a more effective metric to measure performance of a classifier as compared to accuracy metric, however it still remain sensitive to data distributions. G-mean metric (Eq. 14) is another way to measure the degree of inductive bias by taking the ratio positive accuracy and negative accuracy. Though F-Measure and G-Mean provides better results than accuracy, but still these two measures can't be used to compare the performance of different classifiers on same data set).

B. *Receiver Operating Characteristics (ROC) Curves:*

In order to avoid the above issues, ROC curve (Fig.4.) is another approach [86],[87] to evaluate the performance of classifiers by using the evaluation metrics, like, True Positive rate (TP_{rate}) and False Positive rate (FP_{rate}). The ROC graph is plotted based on TP rate and FP rate, and any point in the graph represents the performance of a classifier on a given data set. The ROC curve represents a graphical representation of the trade-offs between benefits (TP_{rate}) and costs (FP_{rate}) of a classification. The performance of the classifier is optimal when TP_{rate}= 1(maximum) and FP_{rate}=0 (minimum) (i.e. point A in the Fig.4). So the classifier performs well, if the point in the ROC space remains closer to point A. If a classifier whose corresponding ROC point lies on the diagonal (such as point E), such a classifier is known as random classifier which randomly guess the class labels for each unknown instance. The classifier that outputs a continuous numeric value can be represented by a set of points in ROC space and these points can be graphically represented by a ROC curve, as the curve L1 and L2 in Fig.4. To assess the performance of different classifiers, the area under the ROC curve (AUC) is used as an evaluation metrics [86], [87]. So the ROC curve that covers maximum AUC generates better performance as compared to the classifier associated with smaller AUC. So, in our example L2 provides better average performance than L1. The AUC measure is computed by the following formula:

$$AUC = \frac{1+TP_{rate}-FP_{rate}}{2} \quad (7)$$

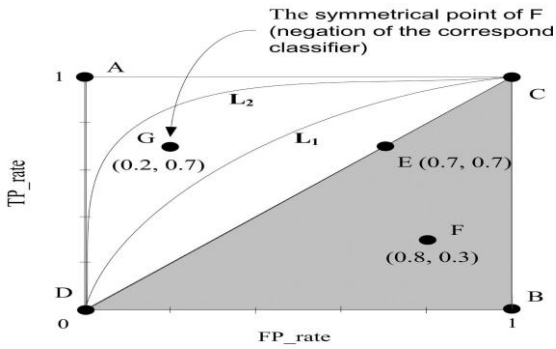

Fig 4: ROC curve representation

C. *Precision-Recall (PR) Curves*

Although ROC curves is a better approach to visualize the performance of a classifier, but it also have some limitations. It is observed that in case of highly imbalanced fraudulent transactions, the ROC curve provides an excessively optimistic representation of an algorithm's performance. To overcome such limitations, PR curve is a better approach to measure the performance of an algorithm's [88]. Using the information's of confusion matrix and the equations of precision and recall, the PR curve can be plotted by taking precision rate and recall rate. As compared to ROC curve, PR curve provides more information related to performance assessment of an algorithm under highly imbalanced data distribution. So, PR curves are mostly proposed in most of the recent research work for performance evaluations and comparisons of different classifiers [88], [89], and [90].

V. EXPERIMENTAL OBSERVATION

Here, we briefly discuss the different observations are made by different algorithms followed by a statistical analysis tests to show a proper comparison among the algorithms.

A. *Algorithms and Parameters*

We propose C4.5 decision tree generating algorithm to act as a baseline classifier in all ensembles [92]. Most of the ensembles methodologies discussed in [49] were used in

combination with C4.5. To deal with the imbalanced class distribution, C4.5 algorithm is most widely used [94] and has been included as the baseline classifier [68]. The configuration parameters for C4.5 are shown in Table-2. The classification tree algorithm, e.g. Hellinger distance tree [94] can be used to deal with the imbalanced fraud transactions. But, the tree reduces the performance due to the utilization of sampling techniques. To overcome such issues, SMOTE preprocessing techniques can be used to rebalance the data before the decision trees undergoes learning phase (see Section II-E).

Table 2. Parameters Specification for C4.5

Parameters	Value
Prune	True
Confidence level	0.25
Basic number of item-sets/leaf	2
Confidence	Laplace Smoothing

Ensemble Learning techniques are more robust than C4.5 in the imbalanced data set. These ensemble learning algorithms are group into two categories: the first category includes classic ensembles, such as Bagging, AdaBoost, AdaBoost.M1 and AdaBoost.M2, which are not applicable for imbalanced data sets; the other category include the algorithms like Cost-sensitive Boosting, Boosting-based, Bagging-based and Hybrid ensembles [49], which can deal with the skewed class distribution. In Section III, we have given brief descriptions about these algorithms.

So to give an overview idea, Table 3 summarizes the entire algorithm groups along with abbreviations and short descriptions. Table 4 shows the rest of the parameters necessary for the algorithms to execute.

B. Statistical Tests

In order to compare the performances of different learning algorithms, a statistical support is required [95]. For statistical comparisons, we need a set of nonparametric tests because it may happen that the initial condition does not get satisfied as a result the statistical report may lose its credibility.

To find the performance of different algorithms, two types of comparisons are made i.e. pairwise comparisons and multiple comparisons. Pairwise comparisons are made between a pair of algorithms and the better algorithm is chosen using Wilcoxon paired signed rank test [96]. Multiple comparisons are made between numbers of algorithms. Iman-Davenport test [98] is used to measure the statistical differences among the algorithm results. To find the best algorithm in the group, Holmpost-hoc test [37] can be used. Similarly, to find the distinct algorithm, n× n comparisons are required and this can be done using Shaffer post-hoc test [38]. The post-hoc methods can be used to determine rejection factor of the hypothesis of comparison at a particular level of significance (α). Besides this, it can also compute the p-value for each comparison, which determines the lowest level of significance of the hypothesis that leads to rejection [49]. From the two parameters we can analyse how far the two algorithms differ.

The performance of the algorithms can also be detected by computing the average ranking of the method. The rankings are determined by allocating rank to individual algorithm and first rank (value 1) is allocated to the best algorithm, second rank (value 2) is allocated to the second best, and it is followed on. At last the average ranking of a method is computed by taking the average of the ranks of all data sets used.

VI. RESULTS

The algorithms that we have discussed in the previous section will undergo different empirical comparisons. For the experiments we have use the dataset which contains credit card transactions of European cardholders in September 2013. This dataset is highly imbalanced which have 492 frauds out of 2,84,807 transactions [39]. To evaluate the performance of the ensemble learning algorithms by using, we divide the procedure into three phases:

A. No. of classifiers

In this phase, we focus on the configuration of the number of classifiers that is best suitable for the algorithms. We compare the performance of the algorithms by executing with both 10 and 40 number of classifiers as done in the research work [49]. All the families are executed with this configuration except non-ensemble, hybrids and IVotes methods.

Each of the algorithms is tested on both the configuration of classifiers using Wilcoxon signed-rank test. Table 5 shows the results of Wilcoxon tests of each family. In this case, "1" is appended to the algorithm abbreviation to represent ten classifiers and "4" for forty classifiers. It also shows the ranks of each method and the hypothesis is tested with a significance value of α= 0.05, it also compute the p-value which represent the difference between the two classifiers. The last field of the table depicts the configuration that is required for the next phase, determined as per either the hypothesis results or as per ranks.

Table 5 represents that, classic boosting methods (i.e. ADAB and M1) performs better with 40 classifiers, whereas M2 shows better results with 10. Similarly, most of the methods of classic bagging and bagging-based techniques perform better with 40 classifiers, except UOB method. But, the RUS method of boosting-based approaches outperform with 10 classifiers. The cost-sensitive method (C2) achieves a p-value of nearly 0.05 with 40 classifiers, which is evaluated method of utilized family. Just like this, SBAG and MBAG are getting selected with 40 classifiers due to their very low p-value.

B. Intrafamily Comparison

In the second phase, comparisons between the families are done. Each family has a pair of algorithms, so using different tests the best algorithms are selected and those methods can be used in the final phase. To determine the best method in a family, either we can use Wilcoxon signed-rank test for pairwise comparison; or, we can use Iman-Davenport test followed by Holm post-hoc if required. Since in our study we are considering five different families of algorithms, so this subsection is divided into five parts and in each part one family will be analysed:

- Non-ensemble Methods: The algorithms belong to this family are C45 (C4.5 decision tree) and SMT (C4.5 implementation on preprocessed data-sets like SMOTE). The algorithms performances are measured using Wilcoxon test and result is shown in Table 6. It is observed that SMT is better than C45 due to higher ranks and rejected null hypothesis with a p-value of 0.00039.
- Classic Ensembles: This family includes both boosting and bagging methods. The average rankings of the methods are evaluated using Iman-Davenport test. The method having highest average ranking is the worst performer. As a result, M14 will be chosen as the representative of the family for securing the lowest average ranking (shown in Fig.5).

Table 3. Algorithms used in the experimental study

Non-ensemble Classifiers		
Abbr.	**Techniques**	**Brief Description**
C45	C4.5	Classic C4.5 decision tree learning algorithm
SMT	SMOTE+C4.5	C4.5 applied on data-sets previously pre-processed with SMOTE
Classic Ensembles		
Abbr.	**Techniques**	**Brief Description**
ADAB	AdaBoost	Classic AdaBoost, without using confidences
M1	AdaBoost.M1	Multi-class AdaBoost, slightly different weight update
M2	AdaBoost.M2	Multi-class AdaBoost using confidence estimates
BAG	Bagging	Classic Bagging, resampling with replacement, bag size equal to original data-set size
Cost-sensitive Boosting Ensembles		
Abbr.	**Techniques**	**Brief Description**
C2	AdaC2	AdaBoost along with cost outside the exponent
Boosting-based Ensembles		
Abbr.	**Techniques**	**Brief Description**
RUS	RUSBoost	AdaBoost.M2 with random undersampling in each iteration
SBO	SMOTEBoost	AdaBoost.M2 with SMOTE in each iteration
MBO	MSMOTEBoost	AdaBoost.M2 with MSMOTE in each iteration
Bagging-based ensembles		
Abbr.	**Techniques**	**Brief Description**
UB	UnderBagging	Bagging with undersampling of the majority class, data-set size doubles the number of positive instances
UB2	UnderBagging2	Bagging with resampling of both classes (balance), data-set size doubles the number of positive instances
OB	OverBagging	Bagging with oversampling of the minority class, data-set size doubles the number of negative instances
OB2	OverBagging2	Bagging with resampling of both classes (balance), data-size doubles the number of negative instances
UOB	UnderOverBagging	Bagging where the number of instances of each bag varies (from undersampling to oversampling)
SBAB	SMOTEBagging	Bagging in case of individual bag's SMOTE quantity differs
MBAG	MSMOTEBagging	Bagging in case of individual bag's MSOMTE quantity differs
SPw	IIVotes weak	IVotes the SPIDER (weak) in each iteration
SPr	IIVotes relabel	IVotes with SPIDER (relabel) in each iteration
SPs	IIVotes strong	IVotes with SPIDER (strong) in each iteration
Hybrid Ensembles		
Abbr.	**Techniques**	**Brief Description**
EASY	EasyEnsemble	UB2 but learning each bag with AdaBoost
BAL	BalanceCascade	Similar to EASY but eliminating majority class instances in individual bagging iteration

Table 4. Parameter specification for the algorithms to be used in the experiment

Algorithm	**Parameters**

SMOTE	No. of Neighbors(k=5), Quantity = Balance amount Distance = Heterogeneous Value Difference Metric
C2	Cmin =1, Cmaj =1/IR
SBO, MBO	Use SMOTE's configuration
SPIDER	Number of Neighbors k=5
EASY BAL	Number of Bags =4 AdaBoost Iterations = 10

Table 5. Determination of number of classifiers

Comparison	R+	R-	Hypothesis (α=0.05)	p-value	Selected
ADAB4 vs. ADAB1	616.0	373.0	Not Rejected	0.17791	ADAB4
M14 vs. M11	558.0	432.0	Not Rejected	0.58135	M14
M24 vs. M21	487.0	503.0	Not Rejected	0.94587	M21
BAG4 vs. BAG1	792.5	197.5	Rejected for BAG4	0.00063	BAG4
C24 vs. C21	680.5	309.5	Not Rejected	0.05341	C24
RUS4 vs. RUS1	301.0	689.0	Rejected for RUS1	0.01934	RUS1
SBO4 vs. SBO1	493.5	496.5	Not Rejected	0.80592	SBO1
MBO4 vs. MBO1	504.0	486.0	Not Rejected	0.93076	MBO4
UB4 vs. UB1	834.5	155.5	Rejected for UB4	0.00009	UB4
UB24 vs. UB21	753.5	236.5	Rejected for UB24	0.00404	UJB24
OB4 vs. OB1	502.0	488.0	Not Rejected	0.95909	OB4
OB24 vs. OB21	601.0	389.0	Not Rejected	0.26104	OB24
UOB4 vs. UOB1	385.0	605.0	Not Rejected	0.13926	UOB1
SBAG4 vs. SBAG1	625.5	364.5	Not Rejected	0.11482	SBAG4
MBAG4 vs. MBAG1	643.5	346.5	Not Rejected	0.07690	MBAG4

R+ corresponds to the execution with 40 and R- to 10 classifiers.

Table 6. Wilcoxon tests for non-ensembles methods

Comparison	R+	R-	Hypothesis (α=0.05)	p-value	Selected
SMT vs. C45	798.5	191.5	Rejected for SMT	0.00039	SMT

R+ ranks in favour of SMT and R- in favour of C45.

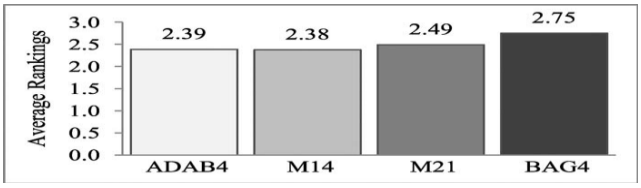

Fig5: Projection of average rankings of classic ensembles.

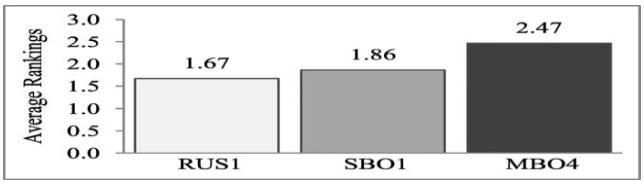

Fig 6: Projection of average rankings of boosting-based ensembles.

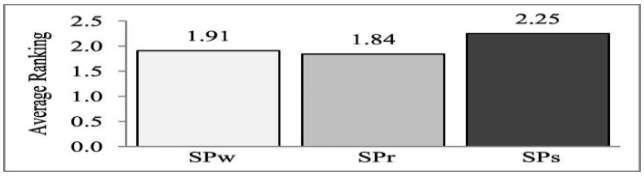

Fig7: Projection of average rankings of IIVotes-based ensembles.

Table 7. Test to reduce bagging-bsed ensembles trough Wilcoxon tests

Compari-son	R+	R-	Hypothesis (α=0.05)	p-value	selected
UB24 vs. UB4	458.5	531.5	Not Rejected	0.63516	UB4
OB24 vs. OB4	913.0	77.0	Rejected for OB24	1.06E-06	OB24
MBAG4 vs. SBAG4	285.0	705.0	Rejected for SBAG4	0.01065	SBAG4

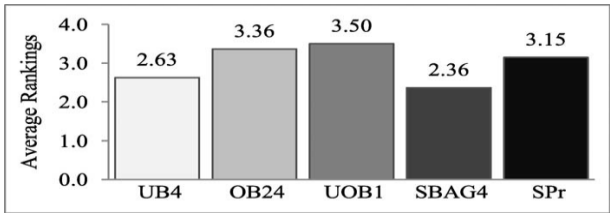

Fig 8:Projection of average rankings of bagging-based ensembles.

Table 8. Comparison between SBAG4 and UB4

Comparison	R+	R-	Hypothesis (α=0.05)	p-value
SBAG4 vs. UB4	492	497	Not Rejected	0.94224

R+ denotes rank of SBAG4 and R- denotes rank of UB4.

Table 9. Wilcoxon test for hybrid methods

Comparison	R+	R-	Hypothesis (α=0.05)	p-value	Selected
BAL vs. EASY	418.5	571.5	Not Rejected	0.3356	EASY

R+ are ranks for BAL and R- for EASY.

- Boosting-based Ensembles: It includes methods like RUSBoost, SMOTEBoost and MSMOTEBoost. Here, also the average ranking methods are evaluated based on the test (shown in Fig.6). RUS1 method is considered as the control algorithm (or best algorithm) due to lowest average ranking.
- Bagging-based Ensembles: A Bagging-based ensemble contains similar types of approaches, i.e. UB/UB2, OB/OB2, and SBAG/MBAG. So, instead of evaluating each individual method, we will first select the appropriate method from each pair using Wilcoxon test. In the next phase, evaluation is done to find the overall best performer using Iman-Davenport test. The results of Wilcoxon tests are shown in Table-7. Among UnderBagging methods, the undersampling of majority class instances (UB4) is selected due to its higher ranks. Similarly, in case of OverBagging, OB2 performs better than OB because of its higher ranks than OB. In synthetic oversampling methods, SMOTEBagging outperforms MSMOTE. So the methods that are selected in the first phase are UB4, OB24 and SBAG4.

Bagging-based approaches also include IIVotes ensemble methods which integrate SPIDER processing technique with IVotes. This approach includes three methods i.e. IIVotes weak (SPw), IIVotesrelabel (SPr) and IIVotesStrong (SPs). Fig. 7 shows the computation of average ranking of these methods using Iman-Davenport test, where SPr obtains highest average ranks (i.e. 1.84). Finally, all the selected methods in first phase (i.e. UB4, OB24, SBAG4 and SPr) are compared using the same test. Fig 8 shows the computational results of the test. From the figure we conclude that the average ranking of SBAG4 is very close to that of UB4. So to draw a significant difference between these two methods, Wilcoxon test is carried out (Table 8). From this test we conclude that SBAG4 performs overall slightly better than UB4 and so SBAG4 will be treated as the representative of this family.

- Hybrid Ensembles: This family includes two approaches, Easy and Bal. Wilcoxon signed-rank test is utilized to determine the best hybrid method. The result of the test is depicted in Table- 9.

C. Interfamily Comparison

In the previous two phases, we have selected the best method from each family, so now in this phase the globally best method will be selected. Table 10 shows the selected methods from each family along with their average rankings. The rankings are computed using Iman-Davenport test, and the Table 10 shows that the ranks of SBAG4 and RUS1 are very close to each other. So inorder to achieve the best algorithm, a pairwise comparison is done through Wilcoxon test (Table 11). Therefore, SBAG4 is regarded as the best algorithm of the hierarchical analysis where the ranks are considered as the vital ones.

Table 10. Summary of average rankings of the representatives of individual family

Family	**Method**	**Abbr.**	**Avg. Ranking**
Non-ensemble	SMOTE	SMT	4.01
Classic	AdaBoost.M2(N=40)	M24	4.76
Cost-sensitive	AdaC2(N=40)	C24	3.58
Boosting-based	RUSBoost(N=10)	RUS1	2.68
Bagging-based	SMOTEBagging(N=40)	SBAG4	2.45
Hybrids	EasyEnsemble	Easy	3.51

Table 11. Comparison between SBAG4 and RUS1 through Wilcoxon tests

Comparison	**R+**	**R-**	**Hypothesis (α=0.05)**	**p-value**
SBAG4 vs. RUS1	527.5	462.5	Not Rejected	0.71717

R+ are ranks for SBAG4 and R- for RUS1.

VII. CONCLUSION and FUTURE WORK

In this review paper we present a brief description about credit card FDS, the major challenges faced by the system and provides a set of solutions to tackle the imbalanced credit card transactions. This survey paper also provides a comprehensive study on the ensembles learning algorithms to address this problem. Several researchers have, integrated different methodologies to enhance the induced classifiers along with class imbalance through utilization of varieties of ensemble learning algorithms. Due to absence of a specific framework approach to classify each one of them, here we present a new classification that is based on four different families which possess their own ensemble learning algorithms as well as techniques to address the imbalance fraudulent transactions. We have performed a hierarchical analysis which was followed by nonparametric statistical tests. We have also done comparison between classic ensemble approaches and non-ensemble approaches.

Finally, we conclude that our study on the imbalanced credit card transactions along with their issues, state-of-the-art solutions to tackle this problem and the different assessment measures to evaluate the performance of the detection model will act as comprehensive resource in the future. It has been observed that ensemble-based algorithms provide better results by utilization of data pre-processing techniques along with trained single classifier. So, this ensemble learning approaches can be implemented for fraud detection problem to obtain better practical results as compared to other approaches.

REFERENCES

[1] Raymond Anderson. The Credit Scoring Toolkit: Theory and Practice for Retail Credit Risk Management and Decision Automation. Oxford University Press, 2007.

[2] Jon TS Quah and M Sriganesh. Real-time credit card fraud detection using computational intelligence. Expert Systems with Applications, 35(4):1721–1732, 2008.

[3] Christopher M Bishop et al. "Pattern recognition and machine learning", volume 4, springer New York, 2006.

[4] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, and Bernard Manderick,"Credit card fraud detection using bayesian and neural networks", In Proceedings of the 1st international naiso congress on neuro fuzzy technologies, 2002.

[5] Dal Pozzolo, O. Caelen, Y.-A. Le Borgne, S. Waterschoot, and G. Bontempi. Learned lessons in credit card fraud detection from a practitioner perspective. Expert Systems with Applications, 41(10):4915–4928, 2014.

[6] D. Hand. "Measuring classifier performance: a coherent alternative to the area under the roc curve", Machine learning, 77(1):103–123, 2009.

[7] C. Bahnsen, D. Aouada, and B. Ottersten," Example-dependent cost-sensitive decision trees", Expert Systems with Applications, 2015.

[8] N. Mahmoudi and E. Duman. "Detecting credit card fraud by modified fisher discriminant analysis", Expert Systems with Applications, 42(5):2510–2516, 2015.

[9] Bharatheesh, T.L. &Iyengar, S.S. (2004). Predictive Data Mining for Delinquency Modeling. Retrieved 18 July, 2012.

[10] J. R. Quinlan, "Improved estimates for the accuracy of small disjuncts," Mach. Learn., vol. 6, pp. 93–98, 1991.

[11] Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown," in Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, New York, 2001, pp. 204–213.

[12] G. Wu and E. Chang, "KBA: kernel boundary alignment considering imbalanced data distribution," IEEE Trans. Knowl. Data Eng., vol. 17, no. 6, pp. 786–795, Jun. 2005.

[13] G. E. A. P. A. Batista, R. C. Prati, and M. C. Monard, "A study of the behavior of severalmethods for balancingmachine learning training data,"SIGKDDExpl. Newslett., vol. 6, pp. 20–29, 2004.

[14] N. V. Chawla, K.W. Bowyer, L. O. Hall, andW. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002.

[15] N. V. Chawla, N. Japkowicz, and A. Kolcz, Eds., Special Issue Learning Imbalanced Datasets, SIGKDD Explor. Newsl., vol. 6, no. 1, 2004.

[16] N. Chawla, D. Cieslak, L. Hall, and A. Joshi, "Automatically countering imbalance and its empirical relationship to cost," Data Min. Knowl. Discov., vol. 17, pp. 225–252, 2008.

[17] Freitas, A. Costa-Pereira, and P. Brazdil, "Cost-sensitive decision trees applied to medical data," in Data Warehousing Knowl. Discov. (Lecture Notes Series in Computer Science), I. Song, J. Eder, and T. Nguyen, Eds., Berlin/Heidelberg, Germany: Springer, 2007, vol. 4654, pp. 303–312.

[18] C. Elkan. The foundations of cost-sensitive learning. In International Joint Conference on Artificial Intelligence, volume 17, pages 973–978. Citeseer, 2001.

[19] C. Whitrow, D. J. Hand, P. Juszczak, D. Weston, and N. M. Adams, Transaction aggregation as a strategy for credit card fraud detection, Data Mining and Knowledge Discovery, 18(1):30–55, 2009.

[20] M. Krivko, "A hybrid model for plastic card fraud detection systems", Expert Systems with Applications, 37(8):6070–6076, 2010.

[21] H. He and E. A. Garcia," Learning from imbalanced data", Transactions on Knowledge and Data Engineering, 21(9):1263–1284, 2009.

[22] Dal Pozzolo, O. Caelen, and G. Bontempi," When is undersampling effective in unbalanced classification tasks?", In Machine Learning and Knowledge Discovery in Databases. Springer, 2015.

[23] G. Ditzler and R. Polikar. Incremental learning of concept drift from streaming imbalanced data. Transactions on Knowledge and Data Engineering, 25(10):2283–2301, 2013.

[24] J. Gao, B. Ding, W. Fan, J. Han, and P. S. Yu. Classifying data streams with skewed class distributions and concept drifts. Internet Computing, 12(6):37–49, 2008.

[25] R.C. Prati, G.E.A.P.A. Batista, Class imbalances versus class overlapping: an analysis of a learning system behavior, in: Proceedings of the Mexican International Conference on Artificial Intelligence, April 2004, pp. 312–321.

[26] Sheskin, Handbook of Parametric and Nonparametric Statistical Procedures, 2nd ed. London, U.K.: Chapman & Hall/CRC, 2006.

[27] K. Woods, C. Doss, K. Bowyer, J. Solka, C. Priebe, and W.Kegelmeyer, "Comparative Evaluation of Pattern Recognition Techniques for Detection of Microcalcifications in Mammography," Int'l J. Pattern Recognition and Artificial Intelligence, vol. 7, no. 6, pp. 1417-1436, 1993.

[28] R. Polikar, "Ensemble based systems in decision making," IEEE Circuits Syst. Mag., vol. 6, no. 3, pp. 21–45, 2006.

[29] L. Rokach, "Ensemble-based classifiers," Artif. Intell. Rev., vol. 33, pp. 1–39, 2010.

[30] R. E. Schapire, "The strength of weak learnability," Mach. Learn., vol. 5, pp. 197–227, 1990.

[31] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," J. Comput. Syst. Sci., vol. 55, no. 1, pp. 119–139, 1997.

[32] L. Breiman, "Bagging predictors," Mach. Learn., vol. 24, pp. 123–140, 1996.

[33] N. Japkowicz and S. Stephen, “The class imbalance problem: A systematic study,” Intell. Data Anal., vol. 6, pp. 429–449, 2002.
[34] Japkowicz, N. and S. Stephen, "The class imbalance problem: A systematic study”, Intelligent data analysis, 2002. 6(5): p. 429-449.
[35] Nguyen, G.H., A. Bouzerdoum, and S.L. Phung, "Learning Pattern Classification Tasks with Imbalanced Data Sets". 2009.
[36] Chen, Rong-Chang, Luol, Shu-Ting, Lee, Vincent C.S., ,”Personalized approach based on SVM and ANN for detecting credit card fraud”, Proceedings of the IEEE International Conference on Neural Networks and Brain, pp. 810–815,2005.
[37] S. Holm, “A simple sequentially rejective multiple test procedure,”Scand. J. Stat., vol. 6, pp. 65–70, 1979.
[38] J. P. Shaffer, “Modified sequentially rejective multiple test procedures,” J. Am. Stat. Assoc., vol. 81, no. 395, pp. 826–831, 1986.
[39] https://www.kaggle.com/mlg-ulb/credit card fraud.
[40] Gama, Joao, Bifet, Albert, Pechenizkiy, Mykola, Bouchachia, Abdelhamid, “A survey on concept drift adaptation”, ACM Comput. Surv, 2013.
[41] Liu, Y. Ma, and C. Wong, “Improving an association rule based classifier,” in Principles of Data Mining and Knowledge Discovery (Lecture Notes in Computer Science Series 1910), D. Zighed, J. Komorowski, and J. Zytkow, Eds., 2000, pp. 293–317.
[42] Y. Lin, Y. Lee, and G.Wahba, “Support vector machines for classification in nonstandard situations,” Mach. Learn., vol. 46, pp. 191–202, 2002.
[43] R. Barandela, J. S. S´anchez, V. Garc´ıa, and E. Rangel, “Strategies for learning in class imbalance problems,” Pattern Recog., vol. 36, no. 3, pp. 849–851, 2003.
[44] Zhou, Z.-H., Liu, X.-Y., "On multi-class cost-sensitive learning",Comput. Intell. 26(3), 232–257, 2010.
[45] Chawla,N.V., Bowyer,K.W., Hall, L.O.,Kegelmeyer,W.P." Smote: synthetic minority over-sampling technique", J.Artif. Intell. Res. 16, 321–357, 2002.
[46] Fern´andez, S. Garc´ıa, M. J. del Jesus, and F. Herrera, “A study of the behaviour of linguistic fuzzy-rule-based classification systems in the framework of imbalanced data-sets,” Fuzzy Sets Syst., vol. 159, no. 18, pp. 2378–2398, 2008.
[47] Drummond and R.C. Holte. C4.5, "class imbalance, and cost sensitivity: why under-sampling beats over-sampling", In Workshop on Learning from Imbalanced Datasets II, 2003.
[48] S. Hu, Y. Liang, L. Ma, and Y. He, “MSMOTE: Improving classification performance when training data is imbalanced,” in Proc. 2nd Int. Workshop Comput. Sci. Eng., 2009, vol. 2, pp. 13–17.
[49] M. Galar, A. Fern´andez, E. Barrenechea, H. Bustince, and F. Herrera, Member,"A Review on Ensembles for the Class Imbalance Problem: Bagging, Boosting, and Hybrid-Based Approches", IEEE Transactions on systems, man and cybernetics, 2011.
[50] J. Stefanowski and S. Wilk, “Selective pre-processing of imbalanced data for improving classification performance,” in Data Warehousing and Knowledge Discovery, I.-Y. Song, J. Eder, and T. Nguyen, Eds., 2008, pp. 283–292.
[51] T. K. Ho, J. J. Hull, and S. N. Srihari, “Decision combination in multiple classifier systems,” IEEE Trans. Pattern Anal.Mach. Intell., vol. 16, no. 1,pp. 66–75, Jan. 1994.
[52] T. K. Ho, “Multiple classifier combination: Lessons and next steps,”in Hybrid Methods in Pattern Recognition, Kandel and Bunke, Eds.Singapore: World Scientific, 2002, pp. 171–198.
[53] N. Ueda and R. Nakano, “Generalization error of ensemble estimators,”in Proc. IEEE Int. Conf. Neural Netw., 1996, vol. 1, pp. 90–95.
[54] S. Geman, E. Bienenstock, and R. Doursat, “Neural networks and the bias/variance dilemma,” Neural Comput., vol. 4, pp. 1–58, 1992.
[55] Krogh and J. Vedelsby, “Neural network ensembles, cross validation, and active learning,” in Proc. Adv. Neural Inf. Process. Syst., 1995, vol. 7, pp. 231–238.
[56] G. Brown, J. Wyatt, R. Harris, and X. Yao, “Diversity creation methods: A survey and categorization,” Inf. Fusion, vol. 6, no. 1, pp. 5–20, 2005.
[57] L. I. Kuncheva, “Diversity in multiple classifier systems,” Inf. Fusion, vol. 6, no. 1, pp. 3–4, 2005.
[58] B. Kong and T. G. Dietterich, “Error-correcting output coding corrects bias and variance,” in Proc. 12th Int.Conf.Mach. Learning, 1995, pp. 313–321.
[59] R. Kohavi and D. H. Wolpert, “Bias plus variance decomposition for zero-one loss functions,” in Proc. 13th Int. Conf. Mach. Learning, 1996.
[60] G. M. James, “Variance and bias for general loss functions,” Mach.Learning, vol. 51, pp. 115–135, 2003.
[61] K. Tumer and J. Ghosh, “Error correlation and error reduction in ensemble classifiers,” Connect. Sci., vol. 8, no. 3–4, pp. 385–404, 1996.
[62] X. Hu, “Using rough sets theory and database operations to construct a good ensemble of classifiers for data mining applications,” in Proc. IEEE Int. Conf. Data Mining, 2001, pp. 233–240.
[63] N. C. Oza and K. Tumer, “Classifier ensembles: Select real-world applications,” Inf. Fusion, vol. 9, no. 1, pp. 4–20, 2008.
[64] G. Wu and E. Y. Chang, "Class-boundary alignment for imbalanced dataset learning",Proc. ICML’03 Workshop on Learning from Imbalanced Data Sets, Washington, DC (August 2003).
[65] Seiffert, T. Khoshgoftaar, J. Van Hulse, and A. Napolitano, “Rusboost:A hybrid approach to alleviating class imbalance,” IEEE Trans. Syst.,Man, Cybern. A, Syst., Humans, vol. 40, no. 1, pp. 185–197, Jan. 2010.
[66] X.-Y. Liu, J. Wu, and Z.-H. Zhou, “Exploratory undersampling for classimbalance learning,” IEEE Trans. Syst., Man, Cybern. B, Appl. Rev,vol. 39, no. 2, pp. 539–550, 2009.
[67] L. Breiman, “Pasting small votes for classification in large databases and on-line,” Mach. Learn., vol. 36, pp. 85–103, 1999.
[68] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D.J. Hand, and D. Steinberg, “Top 10 algorithms in data mining,” Knowl.Inf. Syst., vol. 14, pp. 1–37, 2007.
[69] J. Friedman, T. Hastie, and R. Tibshirani, “Additive logistic regression: a statistical view of boosting,” Ann. Statist., vol. 28, pp. 337–407, 1998.
[70] Rudin, I. Daubechies, and R. E. Schapire, “The dynamics of AdaBoost: Cyclic behavior and convergence of margins,” J. Mach. Learn. Res., vol. 5,pp. 1557–1595, 2004.
[71] Y. Sun, M. S. Kamel, A. K.Wong, and Y.Wang, “Cost-sensitive boosting for classification of imbalanced data,” Pattern Recog., vol. 40, no. 12, pp. 3358–3378, 2007.
[72] S.Wang and X. Yao, “Diversity analysis on imbalanced data sets by using ensemble models,” in IEEE Symp. Comput. Intell. Data Mining, 2009, pp. 324–331.
[73] Tao, X. Tang, X. Li, and X. Wu, “Asymmetric bagging and random subspace for support vector machines-based relevance feedback in image retrieval,” IEEE Trans. Pattern Anal. Mach. Intell., pp. 1088–1099, Jul. 2006.
[74] Chang, B. Li, G. Wu, and K. Goh, “Statistical learning for effective visual information retrieval,” in Proc. Int. Conf. Image Process., 2003, vol. 3, no. 2, pp. 609–612.
[75] S. Hido, H. Kashima, and Y. Takahashi, “Roughly balanced bagging for imbalanced data,” Stat. Anal. Data Min., vol. 2, pp. 412–426, 2009.
[76] P. K. Chan and S. J. Stolfo, “Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection,” in Proc. 4th Int. Conf. Knowl. Discov. Data Mining (KDD-98), 1998, pp. 164–168.
[77] R. Yan, Y. Liu, R. Jin, and A. Hauptmann, “On predicting rare classes with SVM ensembles in scene classification,” in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., 2003, vol. 3, pp. 21–24.
[78] Li, “Classifying imbalanced data using a bagging ensemble variation (BEV),” in Proc. 45th Annual Southeast Regional

Conference (Association of Computing Machinery South East Series 45). New York: ACM, 2007, pp. 203–208.
[79] W. Fan, S. J. Stolfo, J. Zhang, and P. K. Chan, "Adacost: Mis-classification cost-sensitive boosting," presented at the 6th Int. Conf. Mach. Learning, pp. 97–105, San Francisco, CA, 1999.
[80] Kim,Min-jung,Kim,Taek-soo,"A Neural Classifier with Fraud Density Map for Effective Credit Card Fraud Detection".pp.378–383, 2002.
[81] Chen, Rong-Chang,"A new binary support vector system for increasing detection rate of credit card fraud", Int .J. Pattern Recognit .Artif.Intell.20 (2), 227–239, 2006.
[82] N.V. Chawla, A. Lazarevic, L. O. Hall, and K.W.Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in Proc. Knowl. Discov. Databases, 2003, pp. 107–119.
[83] H. Guo and H. L. Viktor, "Learning from imbalanced data sets with boosting and data generation: The DataBoost-IM approach," SIGKDD Expl. Newsl., vol. 6, pp. 30–39, 2004.
[84] R. Barandela, R. M. Valdovinos, and J. S. S´anchez, "New applications of ensembles of classifiers," Pattern Anal. App., vol. 6, pp. 245–256, 2003.
[85] J. Błaszczy´nski,M. Deckert, J. Stefanowski, and S.Wilk, "Integrating selective pre-processing of imbalanced data with ivotes ensemble," in Rough Sets and Current Trends in Computing (Lecture Notes in Computer Science Series 6086),M. Szczuka,M. Kryszkiewicz, S. Ramanna, R. Jensen, and Q. Hu, Eds. Berlin/Heidelberg, Germany: Springer-Verlag, 2010, pp. 148–157.
[86] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Data Mining Researchers," Technical Report HPL-2003-4, HP Labs, 2003.
[87] T. Fawcett, "An Introduction to ROC Analysis," Pattern Recognition Letters, vol. 27, no. 8, pp. 861-874, 2006.
[88] J. Davis and M. Goadrich, "The Relationship between Precision-Recall and ROC Curves," Proc. Int'l Conf. Machine Learning,pp. 233-240, 2006.
[89] R. Bunescu, R. Ge, R. Kate, E. Marcotte, R. Mooney, A. Ramani, and Y. Wong, "Comparative Experiments on Learning Information Extractors for Proteins and Their Interactions," Artificial Intelligence in Medicine, vol. 33, pp. 139-155, 2005.
[90] P. Singla and P. Domingos, "Discriminative Training of Markov Logic Networks," Proc. Nat'l Conf. Artificial Intelligence, pp. 868-873, 2005.
[91] Hofmann, Markus, "A comprehensive survey of methods for overcoming the class imbalance problem in fraud detection", 2012.
[92] J. R. Quinlan, C4.5: Programs for Machine Learning, 1st ed. San Mateo, CA: Morgan Kaufmann Publishers, 1993.
[93] C.-T. Su and Y.-H. Hsiao, "An evaluation of the robustness of MTS for imbalanced data," IEEE Trans. Knowl. Data Eng., vol. 19, no. 10, pp. 1321–1332, Oct. 2007.
[94] A. Cieslak and N. V. Chawla, "Learning decision trees for unbalanced data," in Machine Learning and Knowledge Discovery in Databases (Lecture Notes in Computer Science Series 5211),W. Daelemans, B. Goethals, and K. Morik, Eds., 2008, pp. 241–256.
[95] S. Garc´ıa, A. Fern´andez, J. Luengo, and F. Herrera, "A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability," Soft Comp., vol. 13, no. 10, pp. 959–977, 2009.
[96] F.Wilcoxon, "Individual comparisons by ranking methods," Biometrics Bull., vol. 1, no. 6, pp. 80–83, 1945.
[97] J. Kittler, M. Hatef, R. Duin, and J. Matas, "On combining classifiers,"IEEE Trans. Pattern Anal. Mach. Intell., vol. 20, no. 3, pp. 226–239, Mar.1998.