Credit card fraud detection system

*Project report submitted in partial fulfillment of the requirement for the degree of*

Bachelor of Technology

By

Swarnendu Sengupta (1410110456)

Under supervision

of

Dr. Madan Gopal

http://snu.edu.in/images/snu_logo.jpg

DEPARTMENT OF ELECTRICAL ENGINEERING

SCHOOL OF ENGINEERING

SHIV NADAR UNIVERSITY

(DECEMBER, 2017)

Candidate Declaration

I hereby declare that the thesis entitled “Credit card fraud detection system” submitted for the B Tech Degree program. This thesis has written in my own words. I have adequately cited and referenced the original sources.

(Signature)

(Swarnendu Sengupta)

(1410110456)

Date: \_\_\_\_\_\_\_\_\_\_

CERTIFICATE

It is certified that the work contained in the project report titled “Credit card fraud detection system,” by “Swarnendu Sengupta,” has been carried out under my/our supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor(s)

Name(s)- Dr. Madan Gopal

Department(s) of Electrical Engineering

School of Engineering

Shiv Nadar University

December, 2017

Table of Contents

[1. Introduction 8](#_Toc500776952)

[2. Literature review 9](#_Toc500776953)

[3. Present Investigation / Methodology 13](#_Toc500776954)

[3.1. Dataset 13](#_Toc500776955)

[3.2. Visualization 13](#_Toc500776956)

[4. Theory 18](#_Toc500776957)

[4.1. Over-sampling 18](#_Toc500776958)

[4.2. Under sampling 18](#_Toc500776959)

[4.3. SMOTE 18](#_Toc500776960)

[4.4. GMM 18](#_Toc500776961)

[4.5. Decision Tree 19](#_Toc500776962)

[4.6. Ensemble learning 19](#_Toc500776963)

[5. Process 23](#_Toc500776964)

[6. Results 29](#_Toc500776965)

[7. Appendix 32](#_Toc500776966)

[7.1. Appendix I 32](#_Toc500776967)

[7.2. Appendix II 33](#_Toc500776968)

[7.3. Appendix III 33](#_Toc500776969)

[7.4. Appendix IV 34](#_Toc500776970)

[7.5. Appendix V 34](#_Toc500776971)

[7.6. Appendix VI 35](#_Toc500776972)

[8. References 36](#_Toc500776973)

[9. Acknowledgement 38](#_Toc500776974)

List of figures:-

Figure 3.1. The dataset………………………………………………………………………..…12

Figure 3.2. The above graph represents the linear distribution of hour1 vs hour2………………..13

Figure 3.3. Amount vs OUTPUT………………………………………………………………….13

Figure 3.4. Amount vs Total………………………………………………………………………..14

Figure 3.5. Area of the world taken into consideration……………………………………………14

Figure 3.6. Area wise frauds………………………………………………………………………15

Figure 3.7. No of transactions per hour with the amount of money spent in frauds………………..15

Figure 3.8. Total transactions that happened during the total 24 hours……………………………16

Figure 3.9. Density of frauds done………………………………………………………………...16

Figure 4.1. Ensemble learning – stacking…………………………………………………………20

Figure 4.2. Summary of the ensemble learning…………………………………………………...21

Figure 5.1. The ROC curve for training dataset…………………………………………………...23

Figure 5.2. The ROC curve for testing dataset…………………………………………………….23

Figure 5.3. Bias-Variance trade-off and optimum model selection……………………………….25

Figure 5.4. Description on how to find the best model and bias-variance…………………………25

Figure 5.6. Error vs model complexity……………………………………………………………26

Figure 6.1. AUC with and without SMOTE……………………….......…………………………29

List of tables:-

Table 6.1. Training dataset on different sampling techniques……………………………………..29

Table 6.2. Testing dataset on different sampling techniques……………………………………...29

Table 6.3. Ensemble learning techniques on training dataset……………………………………...30

Table 6.4. Ensemble learning techniques on testing dataset………………………………………30

Abstract

This project is a work towards combination of various techniques like adversarial learning and ensemble learning in detection of credit card frauds in the banking sector. This project reflects the use of ensemble learning over adaptive learning algorithms like adversarial learning using clustering algorithms like GMM and Fuzzy k-means. The reason for the choice of adversarial learning is that it is adaptive to the changing nature of the mindset of the fraudster. But, it has been shown that not always does this have a positive effect on the work being done. It tends to fail when there is a higher balance in the class and is effected by noise and some transactions which though legitimate are beyond the normal thought process of the person.

Thus using adversarial learning didn’t give a fulfilling and correct answer always. The search for other type of learning algorithms led to random forest and from there to its father, ensemble learning. Ensemble learning in literature and also in industry have proven to be very effective in detection of anomalies in the field of medicine and pharmaceuticals where a small change can lead to drastic effect. Since it is widely used there, it was tried on this type of dataset where the imbalance ratio was 3:100, i.e. in every 100 samples, 3 are from the minority group. The move towards ensemble learning proved to be fruitful as it provided the results that was expected and improved on many other qualities which lacked in the adversarial learning part.

The layout of the report is as follows:- there is a formal introduction section which describes the problem statement, next is the literature review section where the papers referred to have been stated in brief along with some visualization. This is followed by the theory of each of the techniques used and their justification. The process of obtaining the results is stated in details. The report is ended by the appendix where the important formula and steps to get to a particular algorithm is stated in details. This report is terminated by giving references and acknowledging the pillars of strength behind the success of this project.

# Introduction

The problem statement chosen is finding an efficient algorithm which would help in detection of credit card frauds. People around the globe are transacting online through their banks and the main mode of payments have essentially become credit cards. But there is a downside to this also.

As the number of credit card users have increased over the years, so has the type and number of crimes committed in this sector. Therefore, it has become necessary that some strict and stringent actions be taken to find out (1) the source of the fraudsters and (2) develop an appropriate and accurate algorithm so as make this process faster and feasible.

Credit card fraud detection is essentially a classification problem. The main objective is to correctly classify the credit card transactions as legitimate and fraudulent. Some problems that are there with this kind of data are the following:-

1. Non-availability of real-world and real-time database
2. Small datasets per user which are not enough to decode the usage pattern of the users
3. Highly biased data, i.e. the number of frauds in much lower than the number of legitimate transactions

Due to the above mentioned causes, accuracy is not the measure of correctness of detection of transaction. A different metric was taken which indicates the best model for this kind of detection: namely AUC of ROC curves. AUC or Area Under Curve of ROC (Receiver Operating Characteristics) is a measure to determine the diagnostic ability of a binary classifier system as its discrimination threshold is varied. The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. If a model exhibits an AUC of 1, then it is the best system and is worthless if it exhibits anything below or equal to 0.5.

# Literature review

Many techniques and work has been done and published in this sector. People have tried out different techniques like kNN, SVM, neural networks, bagging, etc. Some of the papers have reported many different techniques like FraudMiner, adversarial learning[1,2] which gave pretty good results. Some papers have suggested manipulating the number of samples by using different sampling techniques like undersampling, oversampling, ROSE (Random Over Sampling), SMOTE ( Synthetic Minority Over Sampling Technique). With the above mentioned techniques an attempt was made to bring balance in the dataset. The problem with the earlies technique that used normal machine learning techniques without manipulating the data saw the fraud data points as noise and regarded in the same manner.

Tanmay Behra[4] has used a hybrid approach for credit card fraud detection. He along with Panigrahi have used fuzzy clustering along with neural network for fraud detection. They worked on the base that every card holder has a particular type of spending behavior, which creates an activity profile for the user.

Almost all the fraud detection methodologies capture the behavioral patterns as rules and check for any violation in subsequent transactions. However, the target of a reliable FDS is to learn the behavior of the users dynamically so as to minimize the losses. Thus, it is required to develop a FDS which can detect frauds effectively as well as learn the dynamic change of users and fraudsters’ behavior over time.

But to accurately do so one needs to have a very vast user database where the number of transactions per user is in hundreds if not higher.

Saeideh Alimolaei [5] suggested the use of fuzzy theory to understand the customer behavior. The author then fixed a ruleset for the fuzzy system which consisted of 120 if-then rules. Based on the rules given to the fuzzy system, he then used the same system to find the output results of the transactions that were classified into two categories based on the output results, one is normal and other one suspicious. Though the accuracy achieved was not very high, but his paper was a point which showed us that fuzzy with proper logic could lead to higher accuracy and on combination with other machine learning techniques might give better accuracy as well as higher AUC.

Later, as machine learning evolved with time people started to move onto a more responsive and elaborate method which is ensemble learning. In this technique of machine learning, there are three things that mainly comprise of ensemble learning:-

1. Bagging
2. Boosting
3. Ensemble learning

Bagging or bootstrapping aggregating is a way to decrease the variance of a prediction by generating additional data for training on the original dataset using combinations using repetitions to produce multisets for the same cardinality or size as the original size. By increasing the size of the dataset model’s predictive force might not improve but there is a decrease in variance, thereby narrowly tuning the prediction to expected outcome.

Boosting is a two-step approach, where one first uses subsets of the original data to produce a series of averagely performing models and then “boosts” their performance by combining them together using a certain cost function ( in case of classification it is mostly a majority voting). Unlike bagging, in the classical boosting the subset creation is not random and depends on the performance of the previous models: every new subset contains the elements thet were(likely to be) misclassified by the previous models.

Stacking is a similar to boosting: there needs to several models that need to be applied to the original data. The difference here is, however, that there isn’t any empirical formula for the weight function, rather there is a meta-level approach and another model/approach to estimate the input together with outputs of every model to estimate the weights or, in other words, to determine what models perform well and what badly given these input data.

Zareapoor [3] states the use of bagging ensemble technique over other and gives a comparative study between bagging algorithms using decision trees, Naïve Bayes, kNN and Support Vector Machines using Radial Basial Function. In the paper he partitioned the dataset such that he could have 4 different datasets, each having same fraudulent dataset but the number of legitimate transactions were made different in order to accommodate datasets with different fraud rates. They further proved that instead of using old machine learning techniques bagging classifier had better fraud catching rate, lower false alarm rate and higher Matthews Correlation Coefficient.

Mustafa [6] exploited the probability distribution of data sets to put in the two approaches namely Gaussian and Poisson distributions to generate and sample the minority and majority class examples [7]. The distribution based balancing is a two-step process. In the first step a probability distribution is learned and in the second step same number of examples is sampled from classes, hence a balanced data set is formed. This balancing is achieved by oversampling the class with less than b examples, undersampling classes with more than b examples and finally get the fully balanced data set. In addition, we are interested in hybrid ensemble approach featured with data resampling. Ensemble learning combines multiple weak learners to obtain the strong learner with better predictive performance. Bagging [8] and AdaBoost [9] are two common methods for forming ensembles. Boosting reduces bias and variance while bagging reduces variance in error. Furthermore, hybrid ensemble learning methods are proposed to attain better generalization. For example, MultiBoosting [10] combines AdaBoost with wagging [11], a variant of bagging.

Rojarath‘s paper [12] proposes the methodology for improving the performance of the classification model, over several methods. The accuracy values obtained through experiments permit the evaluation of each method's performance. They proposed a concept that brought Ensemble learning to model classification, in order to improve performance through majority voting, called M-Ensemble learning. The improved Ensemble learning approach is divided into two main formats of combined methods, namely the 3-Ensemble model (combining odd number methods, such as Naïve Bayes, Decision Tree, and Multilayer Perceptron); and the 4-Ensemble model (combining even number methods, such as Naïve Bayes, Decision Tree, Multilayer Perceptron, and K-Nearest Neighbor). The most improved classification model resulted from the improved 3-Ensemble method, with an accuracy value of 83.13%, compared with the Multilayer Perceptron based model classification and the 4-Ensemble model, which yielded accuracy values of 80.67% and 81.86%, respectively.

Tsai [13] developed a novel hybrid financial distress model based on combining the clustering technique and classifier ensembles. In addition, single baseline classifiers, hybrid classifiers, and classifier ensembles are developed for comparisons. In particular, two clustering techniques, Self-Organizing Maps (SOMs) and k-means and three classification techniques, logistic regression, multilayer-perceptron (MLP) neural network, and decision trees, are used to develop these four different types of bankruptcy prediction models. As a result, 21 different models are compared in terms of average prediction accuracy and Type I & II errors. By using five related datasets, combining Self-Organizing Maps (SOMs) with MLP classifier ensembles performs the best, which provides higher predication accuracy and lower Type I & II errors. It was found that SOM with the weighted voting based classifier ensembles outperformed all others.

# Present Investigation / Methodology

## Dataset

Following the ongoing trend, the work progressed in the direction of ensemble learning on the Kaggle dataset. This dataset was collected from UCSD FICO data mining contest of 2009 which had 100,000 data points and 18 variables. The dataset looks like:-

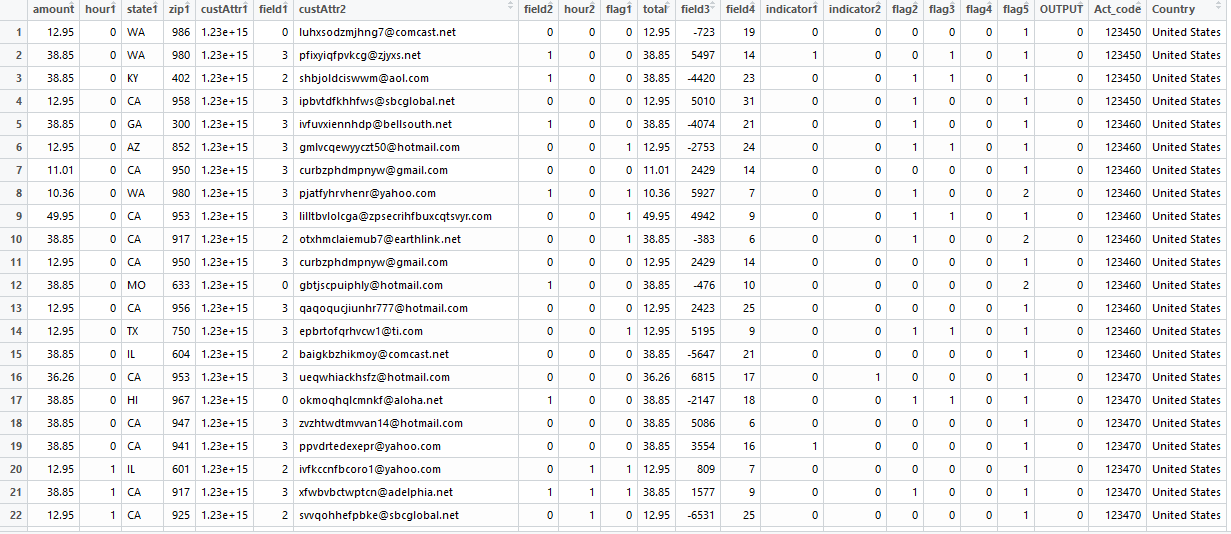


Figure 3.1. The dataset

The attributes of the dataset are as follows:-

1. amount & total – the amount transacted
2. hour1 & hour2 – time of transaction
3. state1 & zip1 – location of the transaction
4. custAttr1, custAttr2 & Act\_code – customer specific information
5. OUTPUT – defines the class 0 represents legitimate and 1 represents fraud
6. Country – the country where the transactions happened

## Visualization

The primary step to any machine learning problem is data visualization. Here the tool used was Tableau to visualize and inherit some information if possible from the data. Some visuals with the inference drawn are shown below.

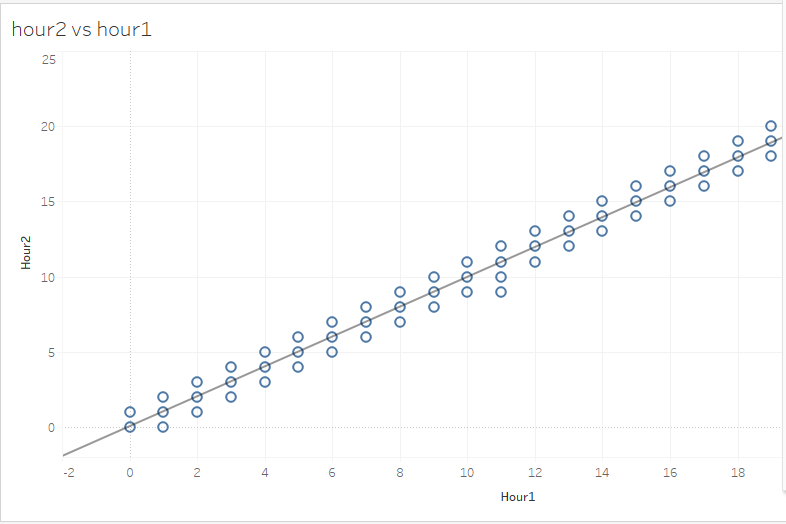


Figure 3.2. The above graph represents the linear distribution of hour1 vs hour2.

The equation of the line is hour2 = 0.99309\*hour1+0.0512494. Thus it can be said that barring one point (not shown in the graph), hour2 and hour1 are pointing towards the same thing. But this subtle difference could be important for the models to differentiate, so they are not touched upon.

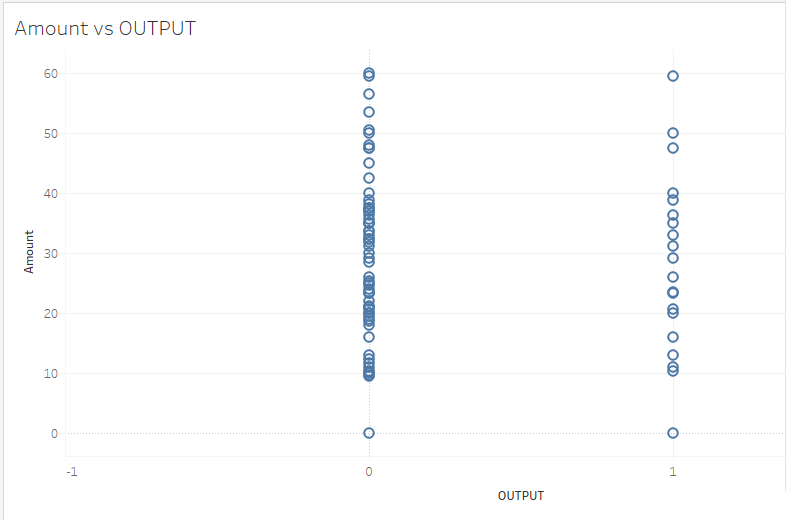


Figure 3.3. Amount vs OUTPUT

As the figure suggests, output is not dependent on the amount transferred. So this also has to be a feature.

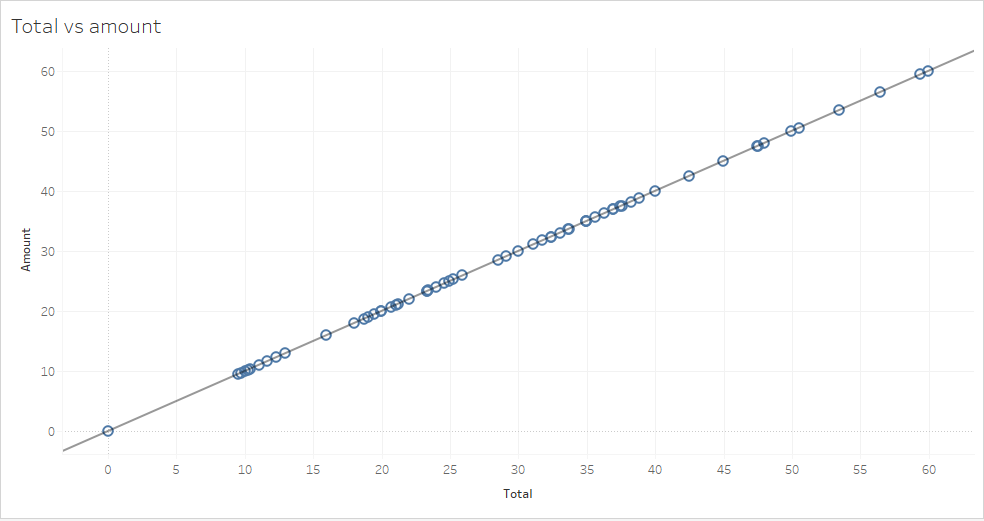


Figure 3.4. Amount vs Total

As the figure suggests, amount and output are the same. Thus one of these variables can be removed while considering making models.

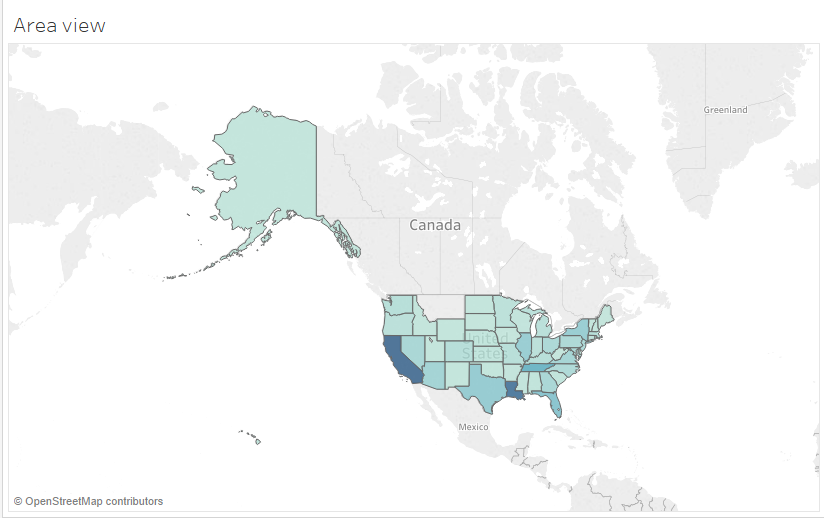


Figure 3.5. Area of the world taken into consideration.

The color coding shows that CA and LA lead in number of fraudulent transactions. Thus transactions made in those areas have to be put through more minute and precise procedure.

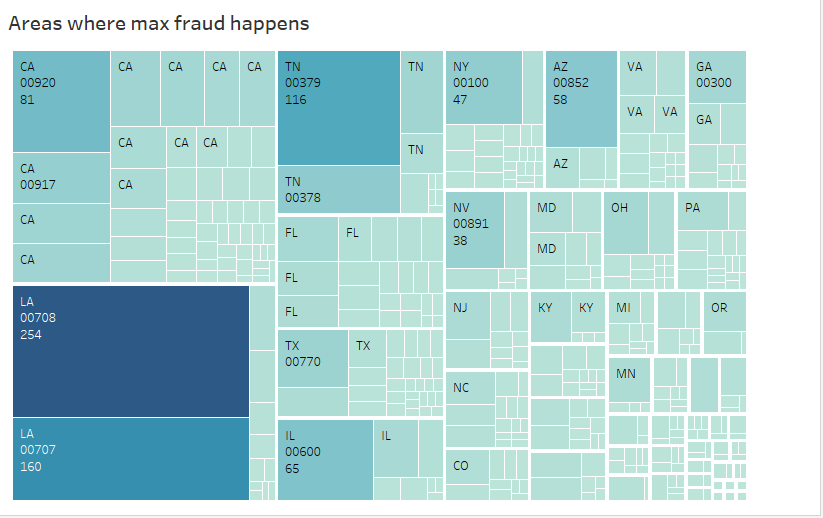


Figure 3.6. Area wise frauds.

It will help banks in those areas to scrutinize their transactions at a lower risk and better accuracy

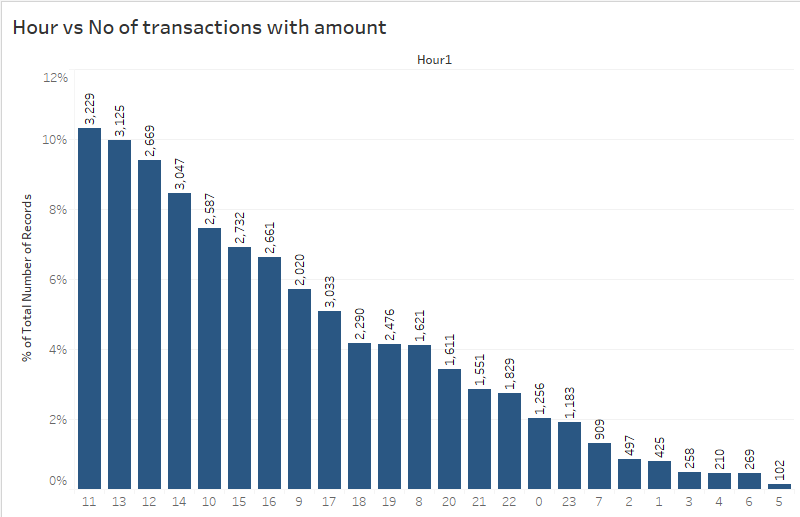


Figure 3.7. No of transactions per hour with the amount of money spent in frauds.

It can be observed that most fraudulent transactions happened during the middle of the day when most normal (legitimate) transactions happened.

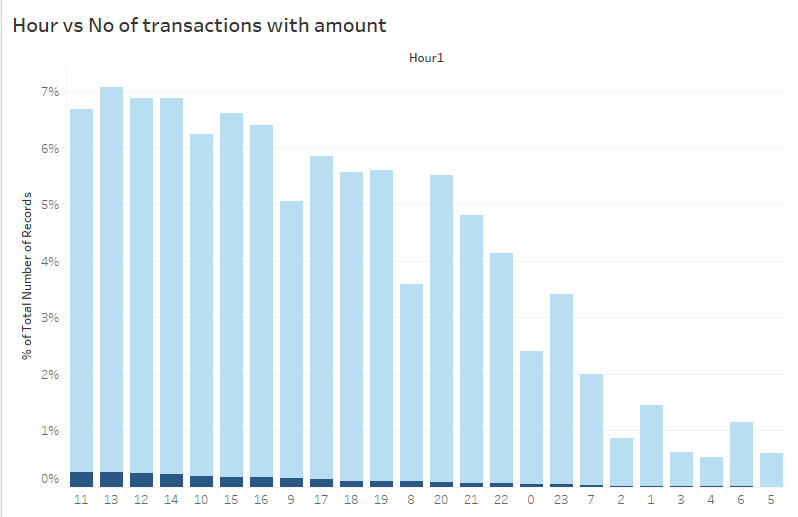


Figure 3.8. Total transactions that happened during the total 24 hours.

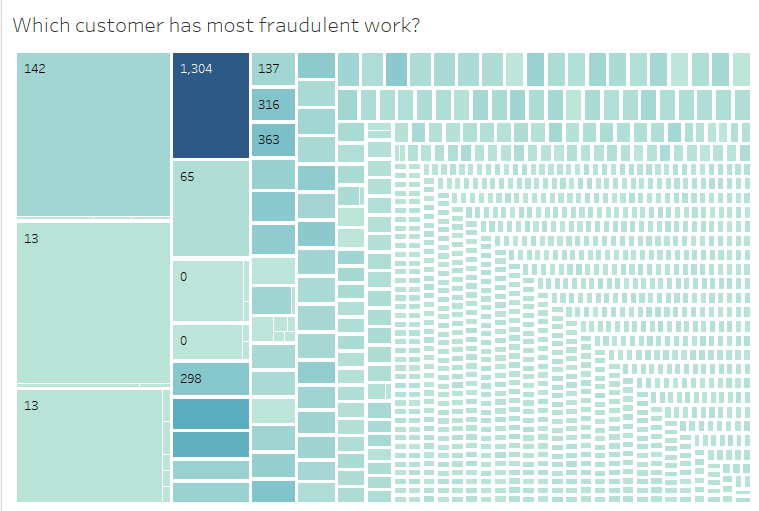


Figure 3.9. Density of frauds done.

The color shows the amount whereas the size of the box show the number of fraudulent transactions done. The people who have high amount but low number of transactions are the customers whose accounts and all transactions need to be put under scanner.

# Theory

## Over-sampling

It tries to balance class distribution by randomly replicating minority class instances. Several authors agree that this method can increase the likelihood of occurring overfitting, since it makes exact copies of existing instances. The method followed is a heuristic approach.

## Under sampling

It is a non-heuristic method that aims to balance class distribution through the random elimination of majority class examples. Its major drawback is that it can discard potentially useful data, which could be important for the induction process.

## SMOTE

This algorithm is used to amplify the minority class by over-sampling it. It adds synthetic data to the original dataset without replacement. A sample from the dataset is taken and k nearest neighbors are taken into account. To create the synthetic data point, a vector is considered which is between one of those k neighbors and the current data point. Their difference is then multiplied by a random number between 0 and 1. This is then added to the data point under consideration. This causes the selection of a random point along the line segment between two specific features. This approach effectively forces the decision region of the minority class to become more general. The algorithm is in Appendix A.

For the purpose of this project, Oversampling, Under-sampling, SMOTE and a combination of oversampling and under-sampling was used. Further, the modified dataset that the various sampling techniques produced were trained using classification trees or decision trees.

## GMM

The basic assumption of GMM is that it assumes that the data is distributed into K Gaussian clusters. The main job is to find out the mean and variance of each cluster along with population distribution between the clusters

Again,

Here, is the mixing coefficient of the mixtures

Now, let us have another variable z which has two parameters i, k. z(i,k) will be 1 iff GMM component k produces i, else 0.

Now the log likelihood function becomes,

ASSUMPTION-X is incomplete. Presence of z(i,k) is also there.

## Decision Tree

The simple algorithm which governs this is: first the single variable is found which best splits the data into two groups. The data is separated and then this process is applied separately to each sub-group till either it reaches a minimum size or until no improvement can be made. The “best” split point is selected by using some criteria like Gini index, information gain etc. on each variable. They are then sorted and the attribute with the highest gain is set at the root and then this process is followed recursively.

Gini index is defined as

, where is the probability of occurrence of jth target variable.

## Ensemble learning

There exist a different type of learning, which is a combination of many different types of learning algorithms. Mainly ensemble learning can be classified into three main sub categories:-

1. Bagging
2. Boosting
3. Stacking

Bagging is also referred to as bootstrap aggregation. To understand bagging, there is a need to understand bootstrapping. Bootstrapping is a sampling technique in which ‘n’ observations or rows are chosen out of the original dataset of ‘n’ rows as well. But the key is that each row is selected with replacement from the original dataset so that each row is equally likely to be selected in each iteration. Thus, there is a possibility of having multiple bootstrapped samples from the same data. Once these multiple bootstrapped samples are extracted, decision trees are grown for each of these bootstrapped samples and use the majority vote or averaging concepts to get the final prediction. This is how bagging works.

One important thing to note here is that it’s done mainly to reduce the variance. Now, random forest actually uses this concept but it goes a step ahead to further reduce the variance by randomly choosing a subset of features as well for each bootstrapped sample to make the splits while training.

Boosting is a sequential technique in which, the first algorithm is trained on the entire dataset and the subsequent algorithms are built by fitting the residuals of the first algorithm, thus giving higher weight to those observations that were poorly predicted by the previous model. It relies on creating a series of weak learners each of which might not be good for the entire dataset but is good for some part of the dataset. Thus, each model actually boosts the performance of the ensemble.

It’s really important to note that boosting is focused on reducing the bias. This makes the boosting algorithms prone to overfitting. Thus, parameter tuning becomes a crucial part of boosting algorithms to make them avoid overfitting. When one weak learner is used, the misclassified data is also included in the dataset used for the next set of learners with higher preference, thus ensuring that as the time progresses the boosting covers the whole dataset and takes care of all the anomalies.

In stacking, multiple layers of machine learning models are placed one over another where each of the models passes their predictions to the model in the layer above it and the top layer model takes decisions based on the outputs of the models in layers below it.

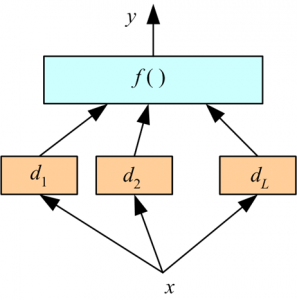


Figure 4.1. Ensemble learning - stacking

Here, there are two layers of machine learning models:

* Bottom layer models (d1, d2, d3) which receive the original input features(x) from the dataset.
* Top layer model, f() which takes the output of the bottom layer models (d1, d2, d3 ) as its input and predicts the final output.

Here, we have used only two layers but it can be any number of layers and any number of models in each layer. Two of the key principles for selecting the models:

1. The individual models fulfill particular accuracy criteria.
2. The model predictions of various individual models are not highly correlated with the predictions of other models.

The top layer model can also be replaced by many other simpler formulas like:

* Averaging
* Majority vote
* Weighted average

All the data can be summarized in the table below.

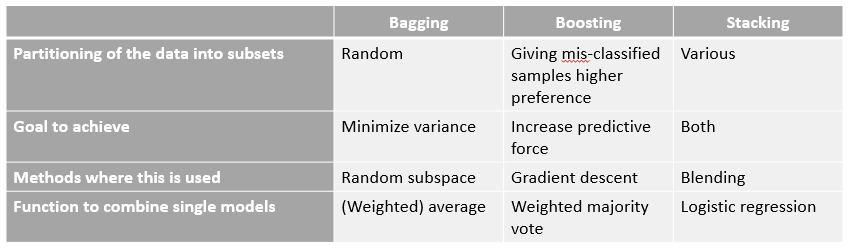


Figure 4.2. Summary of the ensemble learning [Source: <https://i.stack.imgur.com/RFfqb.png>]

# Process

Mary Frances Zeager, Aksheetha Sridhar, Nathan Fogal, Stephen Adams, Donald E. Brown, and Peter A. Beling of University of Virginia [2] came up with the solution of using clustering method followed by SMOTE to analyze and find the area under the ROC curve.

The algorithm proposed was the following:-

1. Divide the dataset into n sequentially ordered sub-datasets
2. for i=1 to n:
   1. Split the dataseti into training, testing and validation sets
   2. Train logistic regression model using training sets
   3. Test model using testing set to calculate confusion matrix
   4. Fit a Gaussian Mixture Model with 3 components
   5. Assign each transaction to the components/strategy it most likely belongs to
   6. Best\_strategy – strategy that provides the adversary with the highest false negative rate
   7. Best\_fraud – subset of the fraudulent transactions that utilize the best\_strategy
   8. Implement SMOTE on the Best\_fraud data to create synthetic fraudulent data that exemplifies Best\_stategy
   9. Append this synthetic data to dataseti+1
3. Test logistic regression model on validation set
4. Return AUC

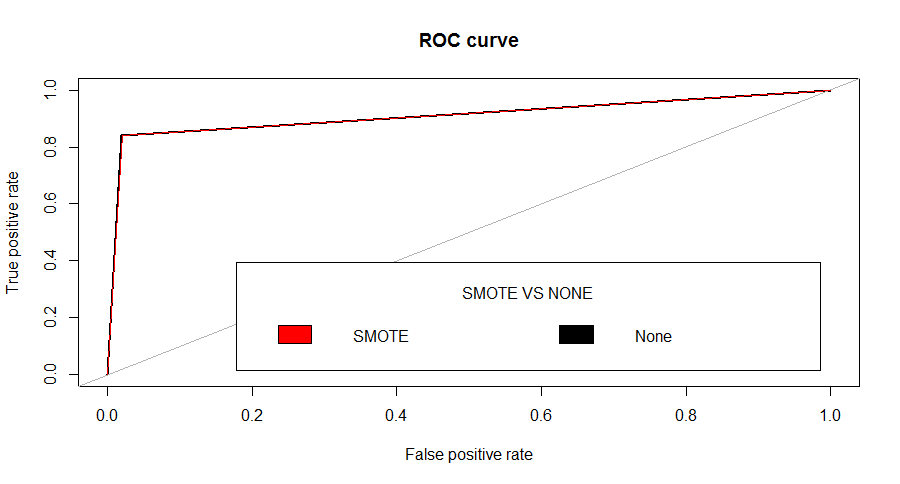
The authors noticed that when they compared the algorithm’s output with to that without SMOTE, the AUC of the ROC curves were increasing significantly. Also the value of the AUC of ROC curves with SMOTE were either equal to or more than that of the one without SMOTE. He chose 3 components as he considered segmenting the transaction amount into low, medium and high bins. Thus all the other continuous variables suffered the same fate.

After this the focus shifted to more advanced higher level of machine learning techniques which included ensemble learning. At first, the original data was split into two parts: - training which was 60% and testing which was 40%. The training set had 58,383 legitimate transactions and 1616 fraudulent transactions. Then under-sampling was performed to balance the two types of transactions. After under-sampling was done, the dataset contained 1684 legitimate transactions and 1616 transactions. It can be seen that due to this, there is considerable loss of information. This information would have been useful for determining the subtle differences. Then decision tree was used to train the model. The decision tree was set to return probability that the data was a fraudulent case. Then the decision tree was used to predict the training AUC and also the threshold that it decided to set for maximizing the AUC. After finding this, the same activity was done using the testing dataset.

After under-sampling was performed, over-sampling was performed. In this the minority samples are randomly duplicated and hence in some cases lead to over-fitting. It was assumed that there was no over-fitting. Again the same process was repeated. In this case, the number of minority samples increased to 58617 and the number of majority samples were stagnant at 58383. The same process was followed with was with the under-sampled case.

While doing both, the number of samples were 30114 legitimate and 29886 fraud. After applying SMOTE, the dataset changed to having 30029 legitimate cases and 29970 fraud cases. At last all these were compared with the basic dataset without any kind of alteration done to the dataset. The number of legitimate cases were 58383 and fraudulent cases were 1616. After each sampling, decision tree was applied to it and then training and test matrices were calculated along with the ROC curves and the AUC for each curve.

It was found that applying SMOTE and without SMOTE, the results were similar in case of training dataset. In the training dataset, applying no sampling proved to be much better than applying any sampling. However, if it was to be compared with the results of testing dataset, applying SMOTE proved to be of higher benefit to the model. Hence for further analysis on this problem, the training data was first put through SMOTE and then further analysis was done.

Figure 5.1. The ROC curve for training dataset

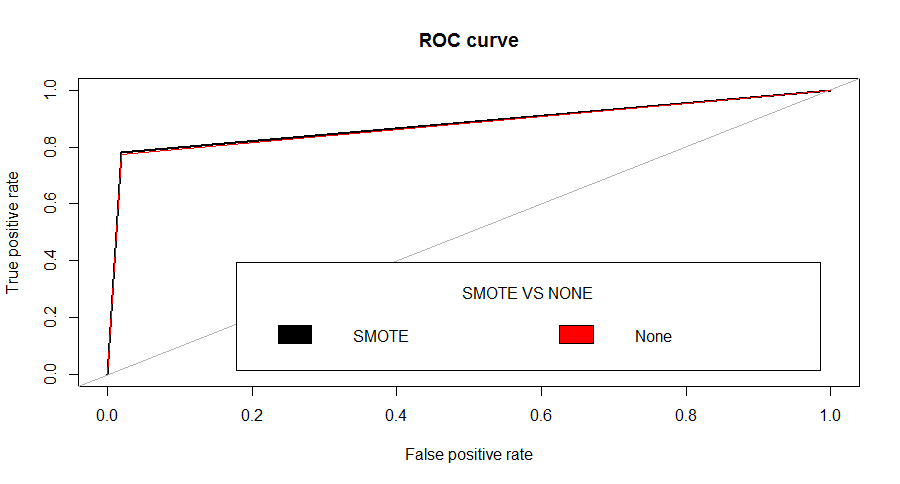


Figure 5.2. The ROC curve for testing dataset

After the training data was modified using SMOTE, it was inserted into the ensemble learning model. Now, ensemble learning has three parts, namely, bagging, boosting and stacking. For bagging, there are a few algorithms that are very popularly used in machine learning, one is C5.0 and the other is CART along with random forest. They have their own advantages and disadvantages. Zareapoor [3] has used C4.5 which is an earlier version on C5.0. However, in this project CART, C5.0 and random forest have been used and their results have been compared.

In boosting, AdaBoost and GBM are the most commonly and widely used. So in this project they has been used. It would also help us comparing the two types of boosting procedures.

In stacking, many different types of stacks can be used. The stacks are basically made up of two or more layers. The base layer contains some algorithms which are called base learners. The output of these base learners is fed to the learner at the top which is called the meta-learner. In case of classification, the output of the base-learners is usually probability of occurrence of a class. These output probabilities then are treated as a classification model for the meta-learner who has to make decisions based on these inputs provided to it.

There are a basic few considerations that need to be taken while considering which all algorithms to take as base learners. The considerations are as follows:-

1. All the classifiers should have very low correlation amongst them.
2. The accuracy of each of the models should be high.

The algorithms considered for ensemble learning are:-

1. GBM or gradient boosting machine
2. RF or random forest
3. Neural network
4. GLM or general logistic model
5. CART or classification and regression tree
6. AdaBoost
7. C5.0

The first four had been used to create the ensemble learning model where the first three were used to create the base learners and the last one, i.e. the fourth was used to create the meta-learner or the top level learner. To know the best configuration of each model’s parameters were varied according to the error metric of accuracy. At a certain point after which the model was overfitting, the metric of the model was chosen as the ideal point.

Some examples of metric are:-

1. Number of trees in Random forest, CART and C5.0
2. Number of neurons in neural network
3. Number of bags in AdaBoost

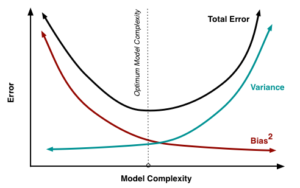


Figure 5.3. Bias-Variance trade-off and optimum model selection

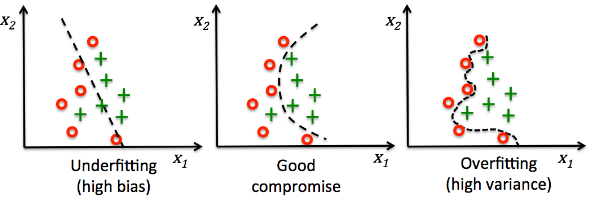


Figure 5.4. Description on how to find the best model and bias-variance

Pseudo-codes of each of the algorithms used can be found in the appendix. The error that was seen was not only of the training but also of the testing. The figure below gives a better understanding on the problem at hand. It also provides us with the solution. If there is a high bias, then train the model on a more complex model. If there is a high variance then either get more data or simplify the model.

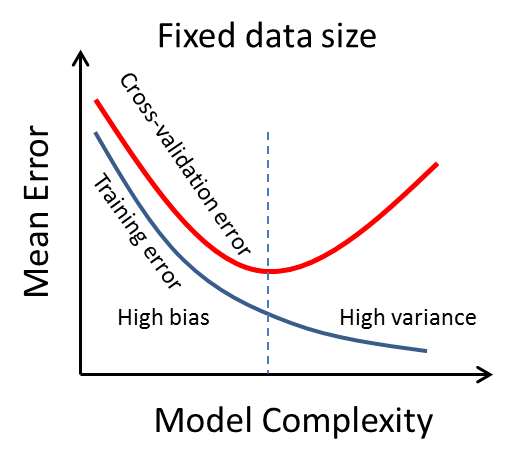


Figure 5.6. Error vs model complexity

Two stacked models were made. One had RF, GBM and neural network as the base and the other without the neural network as it least contributed to the ensemble learning in the stack mode. The meta-learner was GLM. Ensemble 3 has RF as base learner and GLM as meta-learner.

# Results

The graph of the adversarial learning where the black represents the AUC without SMOTE and red represents the AUC with SMOTE.

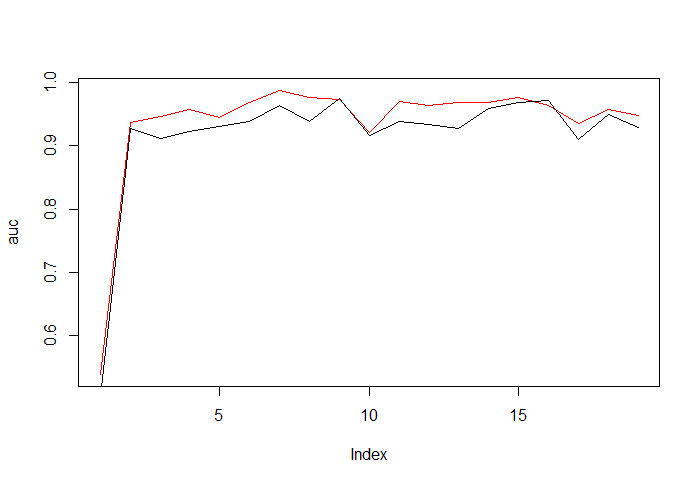


Figure 6.1. AUC with and without SMOTE

The results for doing different sampling techniques followed by applying basic CART (classification and regression tree):-

Table 6.1. Training dataset on different sampling techniques

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Techniques | Accuracy | Precision | Recall | F-score | AUC | Threshold |
| Over-sampling | 96.56 | 0.342 | 0.356 | 0.348 | 0.7139 | 0.79 |
| Under-sampling | 96.55 | 0.342 | 0.355 | 0.348 | 0.7135 | 0.83 |
| Both | 96.88 | 0.341 | 0.406 | 0.371 | 0.7139 | 0.853 |
| SMOTE | 97.89 | 0.269 | 0.839 | 0.406 | 0.9097 | 0.0938 |
| Nothing | 97.90 | 0.274 | 0.842 | 0.412 | 0.9111 | 0.0197 |

Table 6.2. Testing dataset on different sampling techniques

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Techniques | Accuracy | Precision | Recall | F-score | AUC | Threshold |
| Over-sampling | 79.675 | 0.624 | 0.077 | 0.138 | 0.5324 | 0.79 |
| Under-sampling | 97.115 | 0.309 | 0.423 | 0.358 | 0.7026 | 0.975 |
| Both | 96.855 | 0.316 | 0.374 | 0.342 | 0.7043 | 0.8525 |
| SMOTE | 97.88 | 0.255 | 0.779 | 0.384 | 0.8799 | 0.0938 |
| Nothing | 97.88 | 0.258 | 0.775 | 0.388 | 0.8775 | 0.0197 |

This result shows that SMOTE may not have a better accuracy, but the area under the ROC curve is higher than any other sampling used. The results showing the output of different ensemble learning models.

Table 6.3. Ensemble learning techniques on training dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Techniques | Accuracy | Precision | Recall | F-score | AUC |
| GBM | 98.44 | 0.9957 | 0.9883 | 0.992 | 0.9325 |
| RF | 99.74 | 0.9983 | 0.9990 | 0.99870 | 0.99971 |
| Neural Network | 97.95 | 0.9986 | 0.9807 | 0.9896 | 0.7989 |
| GLM | 97.87 | 0.9978 | 0.9807 | 0.9892 | 0.7989 |
| CART |  |  |  |  |  |
| AdaBoost | 97.89 | 0.027 | 1 | 0.052 | 0.6353 |
| C5.0 | 98.14 | 0.821 | 0.394 | 0.532 | 0.6958 |
| Ensemble1 | 99.748 | 0.9982 | 0.99916 | 0.99871 | 0.99975 |
| Ensemble2 | 99.748 | 0.9982 | 0.99916 | 0.99871 | 0.99975 |
| Ensemble 3 | 99.748 | 0.9982 | 0.99916 | 0.99871 | 0.99975 |

Table 6.4. Ensemble learning techniques on testing dataset

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Techniques | Accuracy | Precision | Recall | F-score | AUC |
| GBM | 97.9 | 0.9951 | 0.9834 | 0.9892 | 0.8322 |
| RF | 98.0225 | 0.9941 | 0.9857 | 0.9899 | 0.8528 |
| Neural Network | 99.75 | 0.9981 | 0.9792 | 0.9885 | 0.7785 |
| GLM | 97.68 | 0.9973 | 0.9791 | 0.9882 | 0.7792 |
| CART |  |  |  |  |  |
| AdaBoost | 97.90 | 0.719 | 1 | 0.836 | 0.9983 |
| C5.0 | 97.98 | 0.753 | 0.339 | 0.468 | 0.66789 |
| Ensemble1 | 98.03 | 0.9941 | 0.9857 | 0.9899 | 0.85277 |
| Ensemble2 | 98.03 | 0.9941 | 0.9857 | 0.9899 | 0.85277 |
| Ensemble 3 | 98.03 | 0.9941 | 0.9857 | 0.9899 | 0.85277 |

# Appendix

### Appendix I

Algorithm SMOTE (T,N,k)

Input: Number of minority class samples T; Amount of SMOTE N%; Number of nearest neighbors k

Output :(N/100 )\*T synthetic minority class samples

1. (∗If N is less than 100%, randomize the minority class samples as only a random percent of them will be SMOTEd. ∗)
2. if N< 100
3. then Randomize the T minority class samples
4. T =( N/ 100) ∗ T
5. N = 100
6. Endif
7. N=(int)(N/100 )(∗The amount of SMOTE is assumed to be in integral multiples of 100. ∗)
8. k= Number of nearest neighbors
9. numattrs= Number of attributes
10. Sample[ ][ ]: array for original minority class samples
11. newindex: keeps a count of number of synthetic samples generated, initialized to 0
12. Synthetic[][]:array for synthetic samples(∗Compute k nearest neighbors for each minority class sample only.∗)
13. for i ←1toT
14. Compute k nearest neighbors for i, and save the indices in the nnarray
15. Populate(N,i,nnarray)
16. endfor

Populate( N, i, nnarray)(∗Function to generate the synthetic samples.∗)

1. while N=0
2. Choose a random number between 1 and k, call it nn. This step chooses one of the k nearest neighbors of i.
3. for attr←1tonumattrs
4. Compute: dif=Sample[nnarray[nn]][attr]−Sample[i][attr]
5. Compute: gap= random number between 0 and 1
6. Synthetic[newindex][attr]=Sample[i][attr]+gap∗dif
7. Endfor
8. newindex++
9. N=N−1
10. Endwhile
11. return

(∗End of Populate.∗)

### Appendix II

GBM algorithm

1. Fit a model to the data, F_1(x) = y
2. Fit a model to the residuals, h_1(x) = y - F_1(x)
3. Create a new model, http://5047-presscdn.pagely.netdna-cdn.com/wp-content/uploads/2017/01/latex_1.png

It’s not hard to see how we can generalize this idea by inserting more models that correct the errors of the previous model. Specifically,

F(x) = F_1(x) \mapsto F_2(x) = F_1(x) + h_1(x) \dots \mapsto F_M(x) = F_{M-1}(x) + h_{M-1}(x)  
where F_1(x)is an initial model fit to y

Since we initialize the procedure by fitting F_1(x), our task at each step is to find h_m(x) = y - F_m(x).

### Appendix III

Random forest algorithm

1. Randomly select “k” features from total “m” features.Where k << m
2. Among the “k” features, calculate the node “d” using the best split point.
3. Split the node into daughter nodes using the best split.
4. Repeat 1 to 3 steps until “l” number of nodes has been reached.
5. Build forest by repeating steps 1 to 4 for “n” number times to create “n” number of trees.

### Appendix IV

CART

* 1. Find each predictor’s best split.
  2. For each continuous and ordinal predictor, sort its values from the smallest to the largest. For the sorted predictor, go through each value from top to examine each candidate split point (call it v, if x ≤ v, the case goes to the left child node, otherwise, goes to the right.) to determine the best. The best split point is the one that maximize the splitting criterion the most when the node is split according to it. The definition of splitting criterion is in later section. For each nominal predictor, examine each possible subset of categories (call it A, if x∈ A, the case goes to the left child node, otherwise, goes to the right.) to find the best split.
  3. Find the node’s best split. Among the best splits found in step 1, choose the one that maximizes the splitting criterion.
  4. Split the node using its best split found in step 2 if the stopping rules are not satisfied.

### Appendix V

C5.0

This algorithm has a few base cases.

* All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
* None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
* Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

In pseudocode, the general algorithm for building decision trees is:[[4]](https://en.wikipedia.org/wiki/C4.5_algorithm" \l "cite_note-4)

1. Check for the above base cases.
2. For each attribute a, find the normalized information gain ratio from splitting on a.
3. Let a\_best be the attribute with the highest normalized information gain.
4. Create a decision node that splits on a\_best.
5. Recur on the sublists obtained by splitting on a\_best, and add those nodes as children of node.

### Appendix VI

GMM

1. We know
2. And,
3. Now is the probability generated by with k(th) normal distribution.
4. gets the value of 0/1. So lies between 0 and 1. Hence it is also a soft classifier.
5. Start the EM algorithm by K means
6. Get an estimate of
7. Run EM algorithm on the modelling equations governed by the factors in step 2.
8. Get model which is a mixture of Gaussian distributions
9. Calculate the likelihood that the instance belongs to each cluster by applying appropriate formula for each cluster.
10. Normalize the likelihoods to obtain the probabilities of membership in each cluster
11. Assign each cluster membership based on the highest probability
12. Thus data is assigned to one of the K clusters by applying GMM.

# References

1. K. R. Seeja and Masoumeh Zareapoor, “FraudMiner: A Novel Credit Card Fraud Detection Model Based on Frequent Itemset Mining,” The Scientific World Journal, vol. 2014, Article ID 252797, 10 pages, 2014. doi:10.1155/2014/252797
2. Adversarial Learning in Credit Card Fraud Detection, Mary Frances Zeager, Aksheetha Sridhar, Nathan Fogal, Stephen Adams, Donald E. Brown, and Peter A. Beling
3. Application of Credit Card Fraud Detection: Based on Bagging Ensemble Classifier, Masoumeh Zareapoor, Pourya Shamsolmoali, International Conference on Intelligent Computing, Communication & Convergence (ICCC-2014), doi: 10.1016/j.procs.2015.04.201
4. Credit Card Fraud Detection: A Hybrid Approach Using Fuzzy Clustering & Neural Network , Tanmay Kumar Behera, Suvasini Panigrahi 2015 Second International Conference on Advances in Computing and Communication Engineering, DOI 10.1109/ICACCE.2015.33
5. S. Alimolaei, "An intelligent system for user behavior detection in Internet Banking," 2015 4th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), Zahedan, 2015, pp. 1-5.  
   doi: 10.1109/CFIS.2015.7391642
6. G. Mustafa, Z. Niu, A. Yousif and J. Tarus, "Distribution based ensemble for class imbalance learning," Fifth International Conference on the Innovative Computing Technology (INTECH 2015), Galcia, 2015, pp. 5-10. doi: 10.1109/INTECH.2015.7173365
7. Bermejo, Pablo, Jose A. Gmez, and Jose M. Puerta. Improving the performance of Naive Bayes multinomial in e-mail foldering by introducing distribution-based balance of datasets. Expert Systems with Applications 38.3 (2011): 2072-2080.
8. Leo Breiman (1996). Bagging predictors. Machine Learning. 24(2):123-140.
9. Freund, Yoav, and Robert E. Schapire. Experiments with a new boosting algorithm. ICML. Vol. 96. 1996.
10. Webb, Geoffrey I.:Multiboosting: A technique for combining boosting and wagging. Machine learning 40.2 (2000): 159-196.
11. Bauer, Eric, and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine learning 36.1-2 (1999): 105-139.
12. A. Rojarath, W. Songpan and C. Pong-inwong, "Improved ensemble learning for classification techniques based on majority voting," 2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing, 2016, pp. 107-110.  
    doi: 10.1109/ICSESS.2016.7883026
13. Combining cluster analysis with classifier ensembles to predict financial distress, [Chih-FongTsai](http://www.sciencedirect.com/science/article/pii/S1566253511000807" \l "!), <https://doi.org/10.1016/j.inffus.2011.12.001>,
14. EM ALGORITHMS OF GAUSSIAN MIXTURE MODEL AND HIDDEN MARKOV MODEL, Guorong Xuan Wei Zhang Peiqi Chai, Department of Computer Science, Tongji University

# Acknowledgement

First and foremost, I have to thank my supervisor Dr. Madan Gopal. Without his assistance and dedicated involvement in every step throughout the process, this project would have never been accomplished. I would like to thank you very much for your support and understanding over these past months.

Getting through my dissertation required more than academic support, and I have many, many people to thank for listening to and, at times, having to tolerate me over the past three years. I cannot begin to express my gratitude and appreciation for their friendship. Ashish Kushwaha, Naveenkumar Marati and Peehu Raj have been unwavering in their personal and professional support during the time I spent at the University.

Most importantly, none of this could have happened without my family. To my parents and my friends– it would be an understatement to say that, as a family, we have experienced some ups and downs in the past few months. Every time I was ready to quit, you did not let me and I am forever grateful. This dissertation stands as a testament to your unconditional love and encouragement.