Credit Card Fraud Detection

Dilip Kumar Gangwar **Suman Kumar** **V.N.Dwaraka Mai**

MT17012 **MT17060** **MT17067**

Abstract:

The users of credit card are increasing rapidly and along with it, credit card fraud cases have also increased. There are billions of dollars lost every year due to these fraudulent credit card transactions. In this report, we have used the Dataset taken from Kaggle containing total 2,84,807 transactions. The dataset is unbalanced due to less amount of fraudulent transactions. So with the help of machine learning techniques, we can train the model and classify the transactions into fraudulent or non-fraudulent ones. The dataset is made balanced and machine learning techniques like Logistic Regression, Decision tree, Random Forest, Support vector machine are applied, each having the recall as 94%, 82.1%, 82.1%, 46.6% respectively.

Keywords: Logistic Regression, Decision Trees, Random Forest, SVM, SMOTE

Introduction:

With the increase in e-commerce and online transactions, the use of credit cards is also increasing. At the same time, the increase in credit card fraud transaction is also seen. The major types of fraud cases are card not present, lost or stolen card, fake cards. Credit card fraud detection problem is important mainly due to two reasons. Firstly, identifying fraudulent transactions manually is not feasible as there are millions of transactions taking place every day. Secondly, the profile of fraudulent and normal transactions changes every day as attackers get more familiar with new techniques of doing fraud transactions. So credit card frauds can be smart.

But generally, they often follow a certain pattern. Machine learning algorithms can be trained in these patterns and they can identify immediately if any abnormal activity is observed. Fraudsters can be smart, they frequently change their methods to trick the system. But they often use bits and pieces of their previous patterns. So if we can develop a system to identify such patterns we can react accordingly before there is much damage.

Credit card fraud detection can be seen as a classification problem. That is we can classify the transaction into one of the classes, fraud or non-fraud. Machine learning models are able to analyse the given data and develop a predicting model. But the challenging task here is the amount difference is very large, non-availability of the balanced dataset and most importantly the dataset is not available in its raw form as it contains some sensitive data.

Hypothetically, we assume that fraudulent transactions are of big amounts. But after analysing the dataset, it turned out to be completely opposite to this hypothesis. Most of the transactions which turned out to be fraudulent are having very fewer amounts. So if we are able to identify such critical aspects, we will be able to create the best classifier model.

Dataset:

Machine learning algorithms generally work best if the dataset is balanced or nearly balanced. If data of any one class has significantly larger instances than other classes then it is a problem. When a model is trained on an unbalanced dataset, the model tends to be biased to the classes with more no of instances. That is the model learns more about the class with more instances, thus predicts well for this class/es and will not be able to learn more about the class with fewer instances, thereby will not be so effective for the prediction

The dataset we used is the transactions of European credit card holders in September 2013. The dataset is available at https://www.kaggle.com/mlg-ulb/creditcardfraud The dataset contains 2,84,807 transactions. Out of these, only 492 transactions are fraudulent transactions and rest 284315 transactions are non-fraudulent transactions. The dataset is largely unbalanced. We need to balance this dataset in order to build a better classifier.

The credit card dataset contains highly sensitive data. So to hide this data, the dataset is PCA transformed. The dataset contains 31 features, Time, Amount, Class, V1 to V28. The features Time, Amount and Class are not PCA transformed, features V1 to V28 are PCA transformed.

Methodology:

The dataset is unbalanced. We approached the problem in 3 steps. Firstly, the classifiers are applied on unbalanced dataset (Fig. 1(i)) and metrics value is reported. Secondly, the classifiers are applied on dataset after the application of standard scaler (Fig. 1(ii)). Thirdly, the classifiers are applied on balanced dataset that we got after the application of SMOTE (Fig. 1(iii)).

For balancing the unbalanced data, there exist some techniques like Undersampling, Oversampling, SMOTE. We applied the SMOTE technique.

SMOTE stands for Synthetic Minority Oversampling Technique which statistically increases the number of cases of the class with less number of instances. This technique creates new instances by considering the already existing instances in that class in the given input. By applying SMOTE the number of instances of the majority class will remain unchanged. Only the minority class instances will be increased to the number of instances in the majority class.

SMOTE technique will not simply copy the already available instances to increase their count. The algorithm considers feature space of the desired class and the nearest neighbours of the class. Then it creates new data items by combining features of the desired class with features of its neighbours.

We have first separated the test dataset from the original dataset and then applied SMOTE on only training part so that test data has only original data and no synthesised data. The given dataset was labelled dataset. That is we can apply supervised learning techniques.

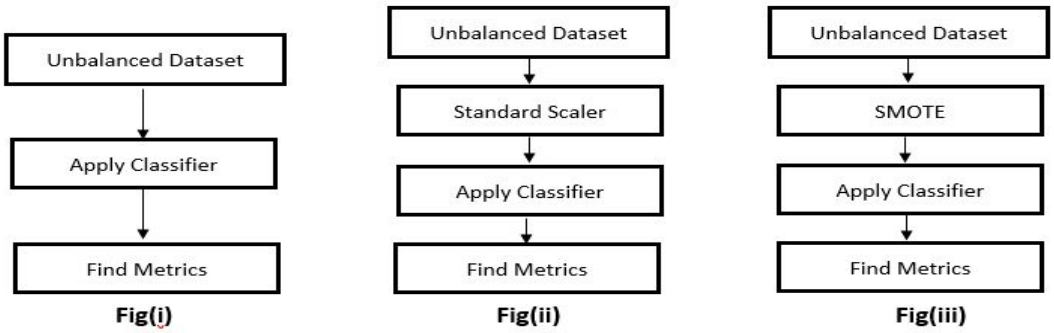

Fig 1: (i) Applying classifier on unbalanced dataset (ii) Applying Standard Scaler before classification (iii) Applying SMOTE technique to balance dataset

Models:

We have applied three classifiers: Logistic Regression, Decision forest and Random Forest. A brief description of each classifier is presented below.

1. **Logistic Regression:**

   It is a statistical machine learning technique. This method is best suitable and mostly used for binary classification problems. The function used in logistic regression is a sigmoid function. A sigmoid function is an S-shaped curve which takes real values as input and maps them into values within range 0 and 1.

   $$\sigma(x) = 1 / (1 + e ^ {-x})$$

   Logistic regression calculates the probability of each class. That is given some input, X, Logistic regression calculates the probability of the input belonging to each of the class, Y. This can be written as:

   $$P(X) = P(Y \mid X)$$

   Logistic regression is a linear method. We use the logistic function to transform the predictions into binary values (0 or 1).

2. **Decision Trees:**

   Decision trees belong to supervised machine learning category. This algorithm is used for both regression and classification problems. The concept of decision trees is to train the model on some decision rules so that the model can predict the desired values on the test data.

   Decision Tree algorithm provides a solution to any problem by constructing a tree from the learning rules. In the resulting tree, internal nodes represent the attributes and the leaf nodes represent the class labels.

   In order to develop the best decision tree, the best attributes should be at the root node or close to the root node. Now at every level, the dataset is split into subsets in such a way that each subset will have the same kind of or same range of values for the attribute at that level. This process continues till all the branches have leaf nodes.

   The major challenge faced in creating a decision tree is to identify which attribute should be present at the root level. For this, we need to perform feature selection methods. For our model, the Gini index was the best approach.

3. **Random Forest:**

   Random forest is another supervised learning technique and as the name indicates it generates multiple, random trees from the given data. The forest built is an ensemble of decision trees.

   In simpler words, a random forest creates multiple decision trees and then combine these trees to get more accurate and more stable prediction results.

4. **SVM:**

   The support vector machines classify the data based on the decision plane which define the boundaries of the classes.

   Generally, the decision planes are not so easily defined. Sometimes data might be in such a way that the margin between two classes is really less. In such cases we cannot decide a decision plane perfectly.

   When the data is not clearly separable, we accept some relaxation in defining

the boundary of the classes. This relaxation is called as support vectors. The margin between support vectors and the decision plane should be maximum in order to clearly distinguish between two classes. At the same time, the penalty should be less, that is minimum relaxation.

We choose the decision plane with the maximum margin with support vectors and also which results in less penalty by considering less relaxation.

Experiments and Results:

Data Visualization:

The dataset which we obtained from Kaggle is PCA transformed due to confidentiality issues to 31 features. So we don't know the nature of the attributes and their values or ranges. So to get an idea about the dataset we have visualized the data using Univariate, Bivariate and Multivariate exploration of data.

Fig. 2, Fig. 3, Fig. 4, Fig. 5 are some of the graphs after visualisation of the data.

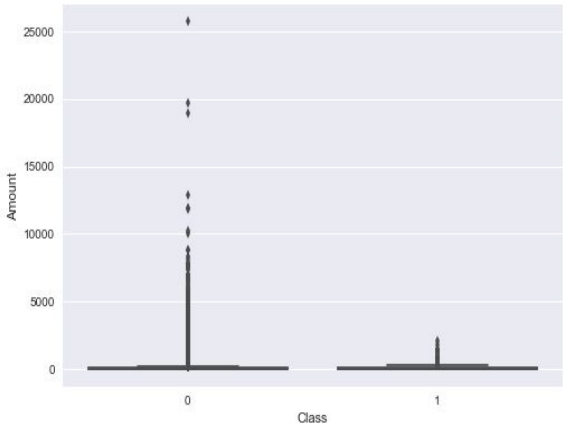

Fig 2

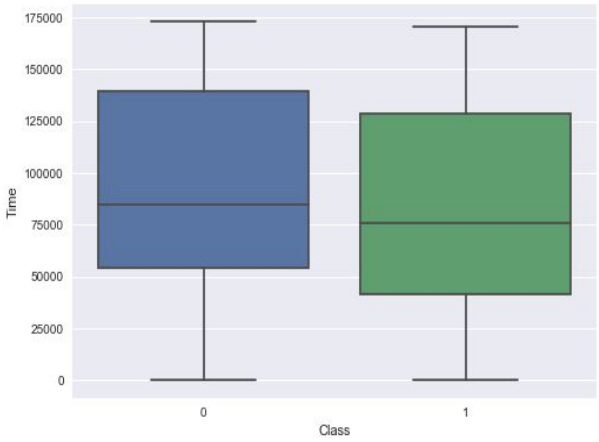

Fig 3

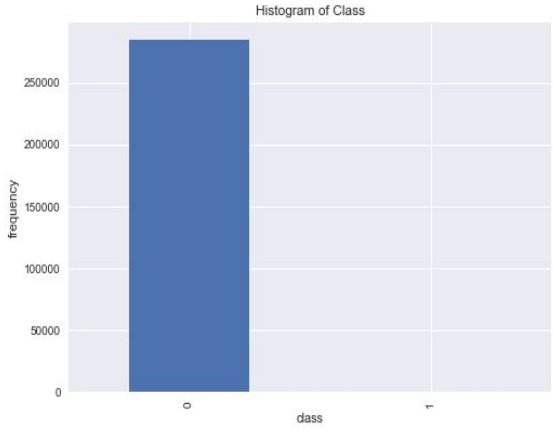

Fig 4

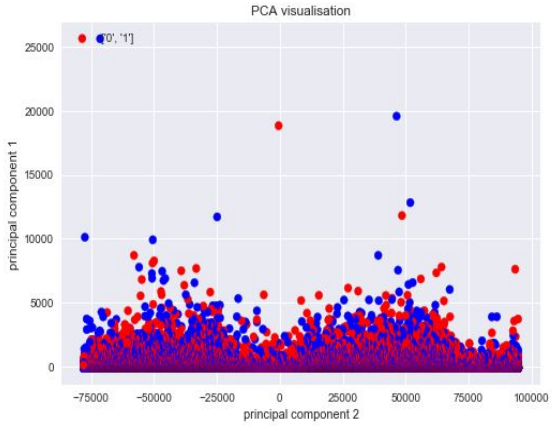

Fig 5

All these figures are of bivariate visualization because we have plotted the graphs using two attributes and the bars or points shows how one attribute varies with other attributes.

Fig 2 depicts that in non-fraudulent transactions, the amounts are in every range that is from smaller amounts to larger amounts but in fraudulent transactions, the amount range is very less.

Fig 3 depicts that the time taken to complete a transaction in non-fraudulent cases is comparatively higher than the fraudulent cases. Both minimum and maximum times observed in fraudulent transactions are lower than then the minimum and maximum times observed in non-fraudulent transactions.

Fig 4 shows that the data is highly imbalanced. The first bar shows the non-fraudulent instances and the second bar represents the fraudulent instances. Since the difference between these two classes is very high, the 2nd bar is not visible at all.

Fig 5 shows that the two classes are non-separable. Hence, we have chosen to solve this problem using classification techniques instead of clustering or anomaly detection techniques.

Table 1: Results

Data Type	Classifiers	Accuracy	Recall	Precision	F1-Score	AUC-ROC
Unbalanced Data	Logistic Regression	99.8	56.4	74	64	78.2
	Decision Tree	99.9	80.1	74.3	77.1	90
	Random Forest	99.9	74.2	92.5	82.4	87.1
	SVM	99.8	36.6	75.5	49.3	68.3
Standard Scaler	Logistic Regression	99.9	63.3	87.6	73.5	81.6
	Decision Tree	99.9	80.1	74.3	77.1	90
	Random Forest	99.9	74.2	92.5	82.4	87.1
	SVM	99.8	36.8	75.9	49.5	68.7
SMOTE	Logistic Regression	**97.5**	**94**	**6.3**	**11.8**	**95.7**
	Decision Tree	99.7	82.1	45.3	58.4	91
	Random Forest	99.9	82.1	85.5	83.8	91.0
	SVM	99.8	46.6	65.5	54.4	69.3

Table 1: Results

Metrics:

We used the following metrics: Accuracy, Precision, Recall, F1-Score and AUCROC value. The dataset is highly unbalanced as the number of fraudulent transactions is way too less compared to non-fraudulent transactions. Our aim here is to classify the fraud transaction more accurately i.e true positive should increase but it will it will be acceptable if some non-fraudulent transactions are classified as fraudulent ones i.e., increase in false positive is acceptable up to some limit.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

$$\text{AUCROC} = \frac{\text{TPR} + \text{FPR}}{2}$$

Accuracy is not a good performance metric for unbalanced data. Thus, we have focused on improving recall rather than accuracy.

All the evaluation metric values for all the four classifiers are given in the following table. Even though we have included other evaluation metrics we have chosen the classifer with high recall as the best classifier.

Results:

In the case of the unbalanced dataset, decision tree gave the maximum recall and logistic regression and random forest were not able to produce better results. After applying standard scaling there is an increase in recall of logistic regression. But the other two approaches results were not improved. After applying SMOTE, logistic regression outperformed all other techniques.

Conclusion:

In this report, we applied four classifiers Logistic Regression, SVM, Decision Tree, Random Forest. We evaluated all the models on the basis of recall instead of accuracy since the data was unbalanced.

The final recall values of all the classifiers after applying SMOTE are taken into consideration to determine the best classifier. The classifiers logistic regression, SVM, decision trees and random forest obtained 94%, 46.6%, 82.1% and 82.1% respectively as the recall values on the balanced dataset that is obtained after applying SMOTE.

So from the above results, we can infer that Logistic regression worked well in classifying fraudulent or non-fraudulent transactions.

References:

1. https://medium.com/coinmonks/handling-imbalanced-datasets-predicting-credit-card-fraud-544f5e74e0fd

2. http://rikunert.com/SMOTE_explained

3. https://towardsdatascience.com/dealing-with-imbalanced-classes-in-machine-learning-d43d6fa19d2

4. J. O. Awoyemi, A. O. Adetunmbi and S. A. Oluwadare, "Credit card fraud detection using machine learning techniques: A comparative analysis," *2017 International Conference on Computing Networking and Informatics (ICCNI), Lagos, 2017, pp. 1-9.*

5. https://medium.com/@martinpella/logistic-regression-from-scratch-in-python-124c5636b8ac

6. https://stats.stackexchange.com/questions/227088/when-should-i-balance-classes-in-a-training-data-set