CORK INSTITUTE OF TECHNOLOGY

Credit Card Fraud Detection using Machine Learning Techniques

For the module COMP9060 – Applied Machine Learning
Master of Science in Data Science and Analytics, Department of Mathematics, 17 May 2020

Name: Sumeet Shivgand
Student id: R00182850

1. Abstract:

Credit card frauds are easy and friendly targets. E-commerce and many other online sites have increased the online payment modes which increased the risk of online frauds. As there was increased in fraud rates, researchers started using machine learning techniques to detect and analyze the online transaction frauds. This paper discusses different methods for identifying fraud using machine learning and compares them using output metrics such as accuracy, precision and specificity. This will help to improve the security of card transactions in the future.

Keywords: credit card, machine learning, accuracy

2. Introduction:

Credit card fraud detection is a relevant problem that draws the attention of machine-learning and computational intelligence communities, where large number of automatic solutions have been proposed [1]. A credit card is a major problem that has become the most popular in online transaction and as well as daily purchase[2]. With the development of modern technology, credit card fraud is increasing significantly which results in the loss of billions of dollars worldwide each year. Most of the time it is difficult to identify the credit card fraud. Machine Learning is one of the most successful fraud identifications techniques. It uses classification and regression approach for recognizing fraud in credit card. There are two types of machine learning algorithm such as supervised and unsupervised learning algorithm. Supervised learning algorithm uses labeled transactions for training the classifier whereas unsupervised learning algorithm uses peer group analysis that groups customers according to their profile and identifies fraud based on customers spending behavior[3]. Many machine learning algorithms have been presented for credit card detection, but we are going to includes Decision Tree and Support Vector Machine. The given dataset is imbalanced, so we are using two resampling methods such as over sampling and under sampling. This report examines the performance of above algorithms based on their ability to classify whether the transactions are authorized or fraudulent and then compares them. The comparison is based on accuracy, specificity and precision. The result showed that Decision tree algorithm showed better accuracy and precision than SVM.

3. Research:

In this section, we are going to outline a specific topic of research that we will incorporate into our study. We are focusing on various machine learning methods like decision tree and support vector machine. Then, we are performing feature scaling like min-max scaler, standard scaler and robust scaler on column. Also, our dataset is imbalanced as the number of observations per class is not equally distributed. We will be using various approaches to handle imbalanced data like random under sampling, random over sampling and SMOTE.

4. Methodology:

We know that all fraudulent transaction follows a similar pattern and then using any pattern recognition system such as decision tree or SVM we can classify transactions as fraudulent whose working is explained below. Also, we are going to explain feature scaling and imbalanced data approaches.

4.1. Feature Scaling:

It is a technique which is used to normalize the range of independent variables or features of data. It is performed during data pre-processing to handle highly varying magnitudes or values or units.

4.1.1. The Standard Scaler

The Standard Scaler is one of the most widely used scaling algorithms out there. It assumes that your data follows a Gaussian distribution (Gaussian distribution is the same thing as Normal distribution)

The mean and the standard deviation are calculated for the feature and then the feature is scaled based on:

$$(xi - mean(x))/stdev(x)$$

The idea behind Standard Scaler is that it will transform your data, such that the distribution will have a mean value of 0 and a standard deviation of 1.

If the data is not normally distributed, it’s not recommended to use the Standard Scaler**.**

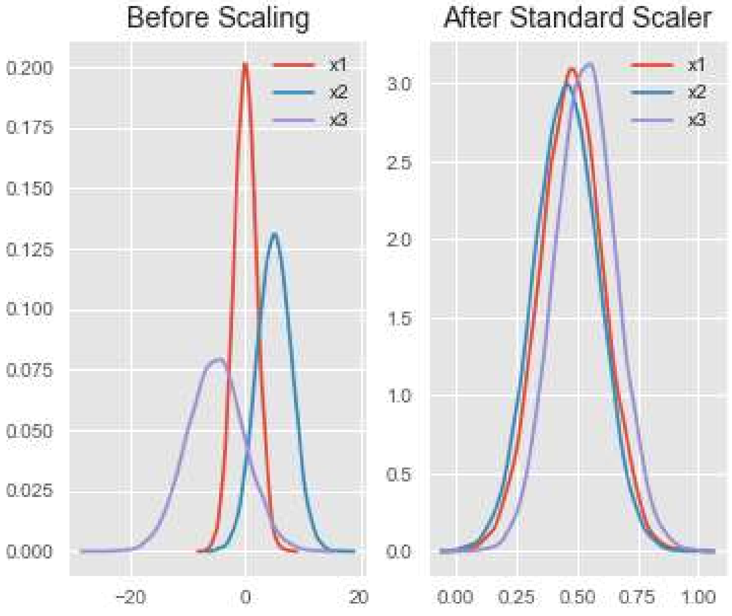

Fig.1 Standard Scaling

4.1.2. The Min-Max Scaler

The Min-Max Scaler uses the following formula for calculating each feature:

$$(xi–min(x))/ (max(x)–min(x))$$

It transforms the data so that it's now in the range you specified. You specifiy the range by passing in a tuple to the feature_range parameter.

Note that, by default, it transforms the data into a range between 0 and 1 (-1 and 1, if there are negative values). It can be used as an alternative to The Standard Scaler or when the data is not normally distributed.

It uses the *min* and *max* values, so it's very sensitive to outliers.

Be careful not to use it when your data has noticeable outliers.

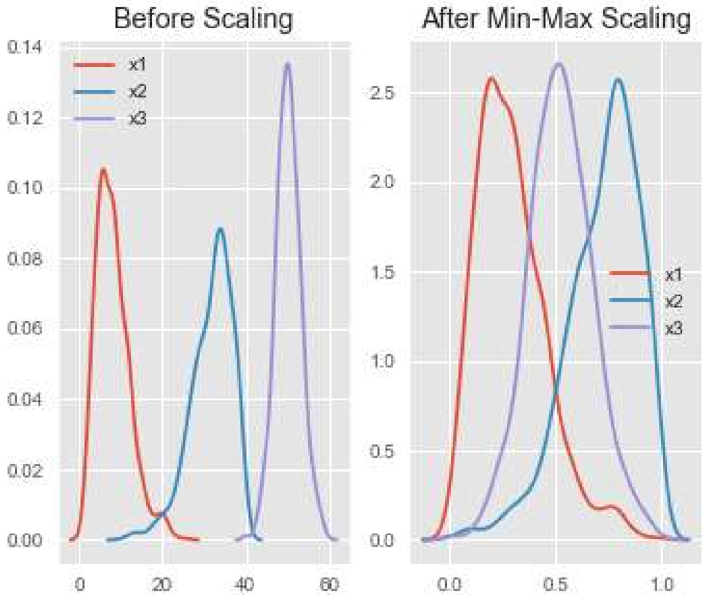

Fig.2 Min-Max Scaling

4.1.3. The Robust Scaler

The Robust Scaler uses statistics that are robust to outliers:

$$(xi–Q1(x))/(Q3(x)–Q1(x))$$

For each feature.

It removes the median and uses the *interquartile range.* Q1 and Q3 represent quartiles.

The **IQR** is the range between the 1st quartile and the 3rd quartile.

This usage of interquartiles means that they focus on the parts where the bulk of the data is. This makes them very suitable for working with outliers.

Notice that after Robust scaling, the distributions are brought into the same scale and overlap, but the outliers remain outside of bulk of the new distributions.

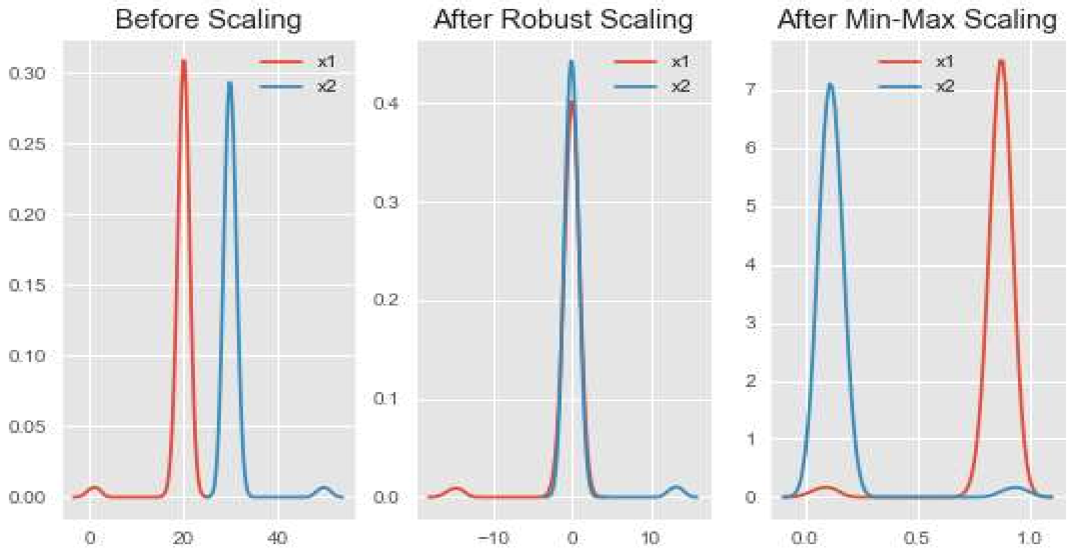

Fig.3 Robust Scaling

4.2. Resampling Methods

4.2.1. Random Oversampling

Random oversampling involves randomly duplicating examples from the minority class and adding them to the training dataset.

This technique can be effective for those machine learning algorithms that are affected by a skewed distribution and where multiple duplicate examples for a given class can influence the fit of the model. This might include algorithms that iteratively learn coefficients, like artificial neural networks that use stochastic gradient descent. It can also affect models that seek good splits of the data, such as support vector machines and decision trees

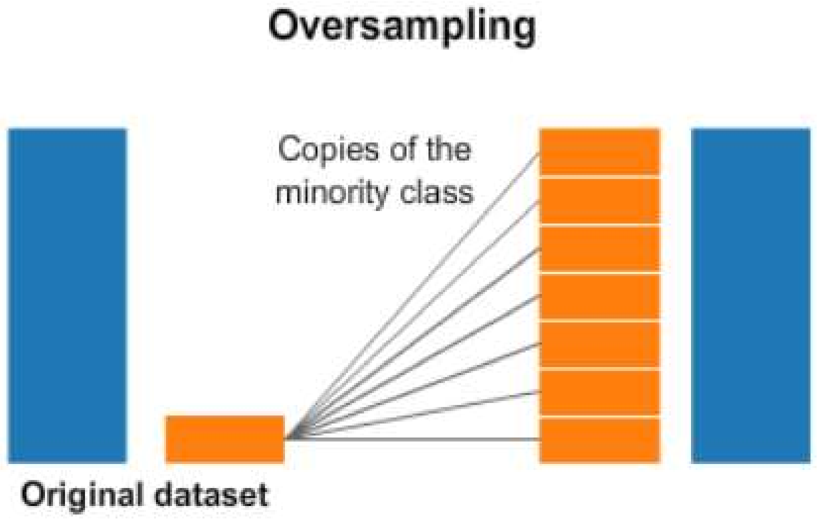

Fig.4 Over Sampling

4.2.2. Random Undersampling

Random undersampling involves randomly selecting examples from the majority class to delete from the training dataset.

This has the effect of reducing the number of examples in the majority class in the transformed version of the training dataset. This process can be repeated until the desired class distribution is achieved, such as an equal number of examples for each class.

This approach may be more suitable for those datasets where there is a class imbalance although a enough examples in the minority class, such a useful model can be fit.

A limitation of undersampling is that examples from the majority class are deleted that may be useful, important, or perhaps critical to fitting a robust decision boundary.

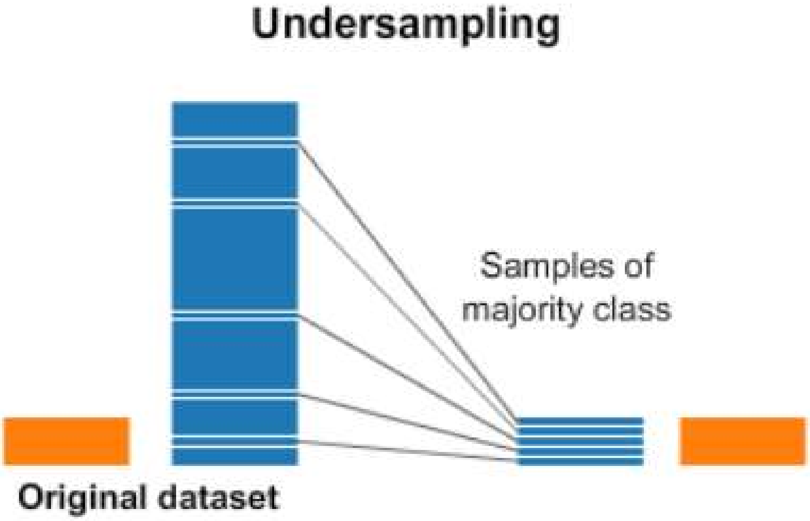

Fig.5 Under Sampling

4.2.3. SMOTE

This technique is followed to avoid overfitting which occurs when exact replicas of minority instances are added to the main dataset. A subset of data is taken from the minority class as an example and then new synthetic similar instances are created. These synthetic instances are then added to the original dataset. The new dataset is used as a sample to train the classification models.

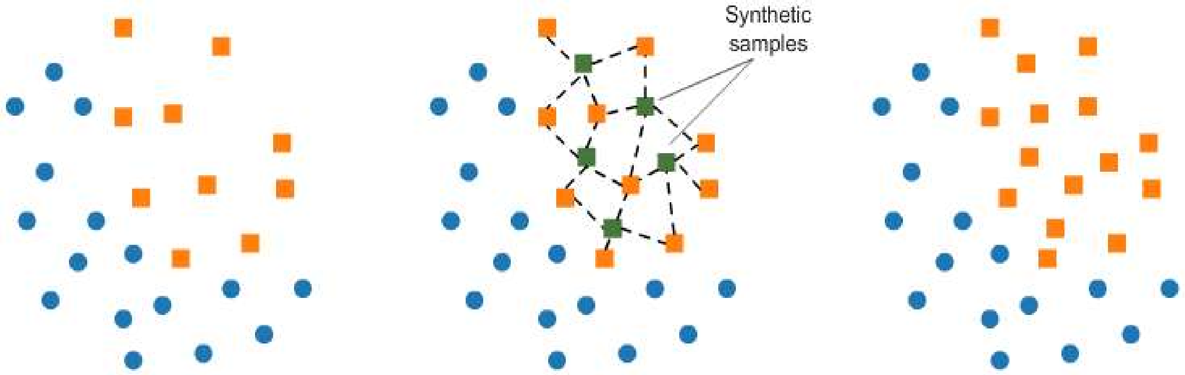

Fig.6 SMOTE

4.3. Machine Learning:

4.3.1. Decision Tree:

Decision trees are non-parametric supervised machine learning algorithm which is widely used in classification and regression tasks. This method sorts the data into branch-like segments that build an inverted tree with a root node, internal nodes, and leaf nodes[4]. It can handle large data sets efficiently without imposing a complex parametric structure. It is a computational tool for classification and prediction. A tree comprises of internal nodes which denote a test on an attribute, each branch denotes an outcome of that test and each leaf node (terminal node) holds a class label [5]. It recursively divides a dataset either using the depth of the first greedy approach or the breadth of the first greedy approach and then stops when all the elements have been assigned to a class. The best partition is the one in which the subsets should not overlap, i.e. they are clearly disjointed to the maximum amount.

The decision tree uses ID3 technique to build decision tree by considering entropy of dataset. Entropy is used to measure the amount of uncertainty in the data set. The criteria for dividing the design of the decision tree are decided by the calculation of the entropy of each attribute. The entropy of the different state can be calculated as follows.

$$H(p1, p2 \ldots\ldots p_s) = \sum_{i=1}^{s}(pilog\left(\frac{1}{pi}\right))$$

Where *P1, P2, …Ps* are the probabilities of the attributes of dataset.

The entropy of each attribute in dataset is calculated and gain is found by subtracting entropy of entire dataset with entropy of splitting attribute[6]. The attribute which as highest gain is selected as root node and accordingly decision tree is designed. The ID3 calculates the gain of a split as follows.

$$\mathrm{Gain}(\mathrm{D}, \mathrm{S}) = \mathrm{H(D)} - \sum_{i=1}^{s} p(Di)H(Di)$$

4.3.2. Random Forest:

Random forest is a supervised machine learning algorithm based on ensemble learning. The random forest algorithm works in a similar way and uses multiple algorithm i.e. multiple decision trees, resulting in a forest of trees, hence the name "Random Forest" [7]. The random forest algorithm can be used for both regression and classification tasks.

Our data sets are quite imbalanced, which can lead to problems during the learning process. Several approaches have been proposed to deal with imbalance in the context of Random Forests including resampling techniques[8].

Advantages of using random forest

- The random forest algorithm is not biased and depends on multiple trees where each tree is trained separately based on the data, therefore biasedness is reduced overall.
- It's a very stable algorithm. If a new data point is introduced in the dataset then it doesn't affect the overall algorithm rather affect the only a single tree.
- It works well when dataset has both categorical and numerical features.

4.3.3. XGBoost

XGBoost is an ensemble learning method. Sometimes, it may not be enough to rely upon the results of just one machine learning model. Ensemble learning offers a systematic solution to combine the predictive power of multiple learners. The resultant is a single model which gives the aggregated output from several models.

XGBoost was a huge improvement on existing gradient boosted tree methods. It enables regular gradient boosting, parallel processing, tree pruning and allows customized objective and evaluation functions. We 're not going to go into detail about working with XGBoost and comparing it to other tree gradient boosting methods (e.g. GBM). There are many parameters that need to be tuned when using XGBoost. We did the parameter tuning by an extensive grid search. At first, we choose a learning rate for which the model makes predictions relatively fast (high learning rate). Then we optimize the number of trees used in the ensemble using cross validation.

5. Model Evaluation:

The three models are run on credit card fraud dataset with combination of SMOTE, Random Over Sampling and Random Over Sampling. The confusion matrix tells how the tuples in training and testing models are correctly classified. Performance of all learning algorithms used for fraud detection in credit card transactions are compared in table 1. The models are evaluated based on parameters such as precession, recall, accuracy and F1 score.

Techniques	Accuracy	Precision	Recall	f1-Score
Decision tree with SMOTE	97.76	0.44	0.65	0.52
Random forest with SMOTE	99.90	0.80	0.71	0.75
XGBoost with SMOTE	99.90	0.78	0.74	0.76
Decision tree with RandomOverSampling	99.88	0.77	0.59	0.67
Random forest with RandomOverSampling	99.92	0.96	0.65	0.77
XGBoost with RandomOverSampling	99.92	0.92	0.68	0.78
Decision tree with RandomUnderSampling	90.54	0.02	0.85	0.03
Random forest with RandomUnderSampling	97.34	0.06	0.79	0.11
XGBoost with RandomUnderSampling	95.88	0.04	0.79	0.07

Table 1. Comparison of Machine Learning Techniques

From the above table, the Random forest XGBoost with Random Over Sampling gives the highest accuracy whereas Decision tree and XGBoost with Random Under Sampling gives low accuracy as compared to others. High detection rate (Precision) is offered by Random forest XGBoost with Random Over Sampling. On the other hand, Decision tree, Random forest and XGBoost with Random Under Sampling provides low detection rate.

6. Conclusion and Future work

Although there are several fraud detection techniques available today, but none can completely detect all frauds. This is because the very small number of transactions are fraudulent in nature from the total transactions. The major drawback of all the techniques is that they will not be guaranteed to give the same result in all the environments rather they will give good result with data and poor result with other type of data. There are some techniques Artificial Neural Network and Naïve Bayesian Network which have high detection rate and high accuracy, but they are very expensive to train. Since, we have imbalance data, so we apply the SMOTE to balance the dataset, where we found that the classifiers were performing better than before. We finally observed that Random forest and XGBoost are the algorithms that gave better results.

A solution to these gaps by creating a hybrid of various techniques that are already used to detect fraud in order to remove their limitations and improve performance.

References:

[1] "Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3784–3797, Aug. 2018, doi: 10.1109/TNNLS.2017.2736643.

[2] "A Research on Credit Card Fraudulent Detection System," *Int. J. Recent Technol. Eng.*, vol. 8, no. 2, pp. 5029–5032, Jul. 2019, doi: 10.35940/ijrte.B1083.078219.

[3] S. K. Shirgave, C. J. Awati, R. More, and S. S. Patil, "A Review On Credit Card Fraud Detection Using Machine Learning," vol. 8, no. 10, p. 5, 2019.

[4] D. Bruno, G. Rodrigues, Nator Junior Carvalho Da Costa, A. A. A. Saraiva, J. V. M. Sousa, and N. M. Fonseca Ferreira, "Learning model for fraud detection in credit cards," 2018, doi: 10.13140/RG.2.2.20238.82245.

[5] Y. Jain and S. Jain, "A Comparative Analysis of Various Credit Card Fraud Detection Techniques," vol. 7, no. 5, p. 6, 2019.

[6] S. Patil, V. Nemade, and P. K. Soni, "Predictive Modelling For Credit Card Fraud Detection Using Data Analytics," *Procedia Comput. Sci.*, vol. 132, pp. 385–395, 2018, doi: 10.1016/j.procs.2018.05.199.

[7] V. Jonnalagadda, "Credit card fraud detection using Random Forest Algorithm," p. 5, 2019.

[8] E. Altendorf, P. Brende, J. Daniel, and L. Lessard, "Fraud Detection for Online Retail using Random Forests," p. 5.