Report

Advanced Business Intelligence

And

Data Visualization

Alok Mondal

Professor:

Madan Radhakrishnan

Cergy/Fr

Credit card Fraud detection: Over sampling and Neural Networks

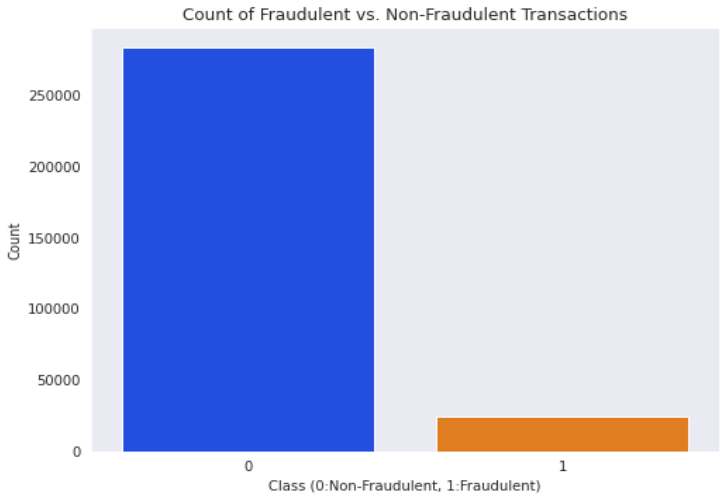
1. Introduction

Credit cards are commonly used thanks to the expansion of the web business and therefore the development of portable smart gadgets. Cardless transactions are presently ruling the net world. As an example, online transaction with a Mastercard/Visa is more popular, especially all Mastercard/Visa transactions are done by web payment gateways, e.g., PayPal and Alipay. The Mastercard/Visa has made a web transaction easier and more convenient. The employment of credit cards is widely increased, because of this fraud appears as a significant problem within the Mastercard/Visa card business. However, there's a growing trend of transaction fraud leading to great losses of cash once a year. It is estimated that losses are going to be increased each year at double-digit rates by 2020. Since Mastercard/Visa card isn't required within the online transaction environment and also the card's data is enough to end payment, it's easier to conduct a fraud than before. Mastercard/Visa card fraud has become a top barrier to the event of e-commerce and has a dramatic impact on the economy. Fraud detection could be a process of observing the transaction behavior of a cardholder so on recognize whether an incoming transaction is completed by the cardholder or others. There are two kinds of techniques for fraud detection: Misuse detection and Anomaly detection. Misuse detection uses some classification techniques to decide if an incoming transaction may be a fraud or not. Normally, such a method has to give some thought to the present varieties of frauds to create models by learning different fraud patterns. Anomaly detection is to construct the profile of normal transaction behavior of a cardholder hooked into his/her historical transactional data, and judge a recent transaction as potential fraud if it deviates from the normal transaction behavior. However, this method requires enough successive sample data to characterize the cardholder’s normal transaction behavior.

2. Dataset Preview

The dataset has taken from Kaggle and the link here. This data set is uploaded in order to get the insights of Credit card Defaulters based on the respective attributes. Dataset contains 307510 rows and 122 columns including target columns.

There were 282686 non-fraudulent transactions (91.927%) and 24825 fraudulent transactions (8.073%).



3. Business Challenge

Detecting fraud transactions is of great importance for any credit card company. Our challenge for well-known dataset to detect potential frauds so that customers are not charged for items that they did not purchase. The most complicated part of the solution was to achieve good accuracy for users who have made only a few transactions. We could apply the regular model, which is good for users with a rich transaction history, but it would give worse scores if there is a lack of historical data (for example, a new user). So, the goal is to build a classifier that tells if a transaction is a fraud or not.

4. Data Preprocessing

Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, lacking in certain behaviours or trends, and is likely to contain many errors.

Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing.

As the dataset is too large, we divide the dataset into 3 part (object, Integer, float) based on its datatype and analyse it. Data is cleansed through processes such as filling in missing values or deleting rows with missing data, smoothing the noisy data, or resolving the inconsistencies in the data. Smoothing noisy data is particularly important for ML datasets, since machines cannot make use of data they cannot interpret. Data can be cleaned by dividing it into equal size segments that are thus smoothed (binning), by fitting it to a linear or multiple regression function (regression), or by grouping it into clusters of similar data (clustering). Data inconsistencies can occur due to human errors (the information was stored in a wrong field). Duplicated values should be removed through deduplication to avoid giving that data object an advantage (bias).

Reducing data is a part of data preprocessing. There are various methods to reduce data. For example, once a subset of relevant attributes is chosen for its significance, anything below a given level is discarded. Encoding mechanisms can be used to reduce the size of data as well. If all original data can be recovered after compression, the operation is labelled as lossless.

5. Methods have used for Credit Card Fraud Detection Tools

5.1 Logistic Regression

Logistic regression uses an approach which is functional to find the binary response probability based on several features. It uses the sigmoid function which is a nonlinear function to find the parameters which fit the best. The sigmoid function (sigma) along with the input (x) to the sigmoid function are given below:

$$\sigma(x) = \frac{1}{(1 + l^{-x})}$$

$$x = w_0 z_0 + w_1 z_1 + \cdots + w_n z_n$$

The best coefficients w and input data which is a vector z are multiplied together multiply each element. These add up to get a number which finally determines a classification score of the target class. If the value of sigmoid comes out to be lesser than 0.5, then it is considered to be 0; otherwise it's a 1.

5.2 K-Nearest Classifier

K-nearest neighbors' classifier uses similarity measures like Euclidean, Minkowski, or Manhattan distance and is a learning method which is instance based. Minkowski distance is suited well for categorical variables whereas Euclidean and Manhattan distance work well with variables which are continuous. In this paper we use Euclidean distance in the k nearest neighbors classifier. The Euclidean distance (Di j) between two input vectors (Xi, X j) is given as:

$$D_{ij} = \sqrt{\sum_{k=1}^{n} (X_{ik} - X_{jk})^2} \text{ k=1,2,. . . ,n}$$

5.3 Decision Tree

A decision tree is a supervised learning technique that has a pre-defined target variable and is most often used in classification problems. This tree can be applied to either categorical or continuous input & output variables. The training process resembles a flow chart, with each internal (non-leaf) node a test of an attribute, each branch is the outcome of that test, and each leaf (terminal) node contains a class label. The uppermost node in the tree is called the root node.

In the decision process, the sample (population) is split into two or more sub-populations sets of maximal, which is decided by the most significant splitter or differentiator in the input variables.

The ultimate goal is to create a predictive model that can take observations about a sample (the branches) and make accurate conclusions about the sample's target value (the leaves).

5.4 Gaussian Naive Bayes

Based on Bayesian theory, Naive Bayes is a statistical approach which uses the highest probability to make decisions. Bayesian probability uses known values to estimates probabilities which are unknown. In this, logic and prior knowledge are applied to statements that are uncertain. This technique uses the conditional independence assumption among the features in the data. The conditional probabilities and of fraud and non-fraud classes are used in the naive Bayes classifier.

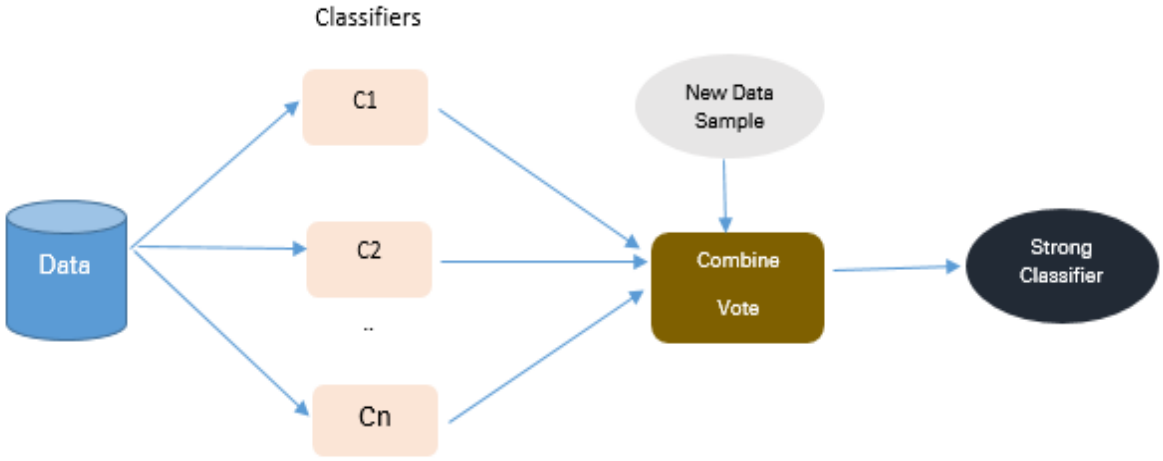
$$P(c_i|f_k) = \frac{P(f_k|c_i) * P(c_i)}{P(f_k)}$$
$$P(f_k|c_i) = \prod_{i=1}^{n} P(f_k|c_i),\ k = 1, 2, \ldots, n$$

5.5 Multi-Layer Perceptron

Multi-Layer Perceptron contains more than single neuron's linear layer. The simplest example can be of a 3-layer system which has the first layer as the input layer, last layer as the output layer and middle layer as the hidden layer. Input data is fed to the input layer and output data is collected from the output layer. Hidden layers can be increased as much as we want. Feed Forward network is the most typical neural network. The ultimate goal is to approximate f() which is some function. For a classifier y = f(x) that gives an output value y for every input x, the multilayer perceptron finds an approximation to the classifier by defining mapping and learning the best parameters for it. Many functions can be chained together in an MLP. Each layer in hidden layer performs a transformation of a linear sum of inputs which can be represented as y = f(WxT + b), where W represents the weights in the layer, x is an input vector which can also be the output of the previous layer, b is the bias vector and f is some activation function. Activation functions are functions which describe the relationship between the input and the output in a non-linear manner. The class score for each input is given by the output of the network. The performance is measured using a loss function. If the predicted and actual class does not correspond then the loss increases. To tackle the problem of overfitting and underfitting, an optimizer is used.

5.6 Algorithmic Ensemble Techniques

The main objective of ensemble methodology is to improve the performance of single classifiers. The approach involves constructing several two stage classifiers from the original data and then aggregate their predictions



5.6.1 CatBoost Classifier

CatBoost is a high-performance open source library for gradient boosting on decision trees. So, CatBoost is an algorithm for gradient boosting on decision trees. It is a readymade classifier in scikit-learn's conventions terms that would deal with categorical features automatically. It can easily integrate with deep learning frameworks like Google's TensorFlow and Apple's Core ML. It can work with diverse data types to help solve a wide range of problems (described later) that businesses face today. It is developed by Yandex researchers and engineers, and is used for search, recommendation systems, personal assistant, self-driving cars, weather prediction and many other tasks.

5.6.2 XGBClassifier

The XGBoost stands for eXtreme Gradient Boosting, which is a boosting algorithm based on gradient boosted decision trees algorithm. XGBoost applies a better regularization technique to reduce overfitting, and it is one of the differences from the gradient boosting. The 'xgboost' is an open-source library that provides machine learning algorithms under the gradient boosting methods. The xgboost.XGBClassifier is a scikit-learn API compatible class for classification.

5.6.3 LGBMClassifier

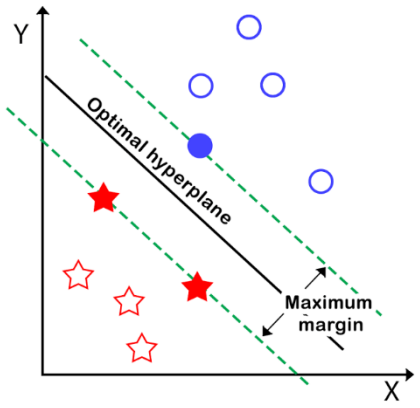
LightGBM, short for Light Gradient Boosted Machine, is a library developed at Microsoft that provides an efficient implementation of the gradient boosting algorithm. The primary benefit of the LightGBM is the changes to the training algorithm that make the process dramatically faster, and in many cases, result in a more effective model.

6. Proposed Work

6.1. Over Sampling- Support vector machine

The aim of an SVM is to fit a hyperplane between data points in space, i.e. support vectors, such that the samples are separated by the largest gap possible. Classification occurs by determining which side the gap new data points fall on. SVM or Support Vector Machine algorithm tries to draw a hyperplane between two classes to separate them. There can be many hyperlanes but SVM's purpose is finding maximum margin or finding maximum distance between data points from each classes.

This data points are nearest data points to hyperplane from each classes. Let's look image below it explains better.



6.2 Random Forest Classifier

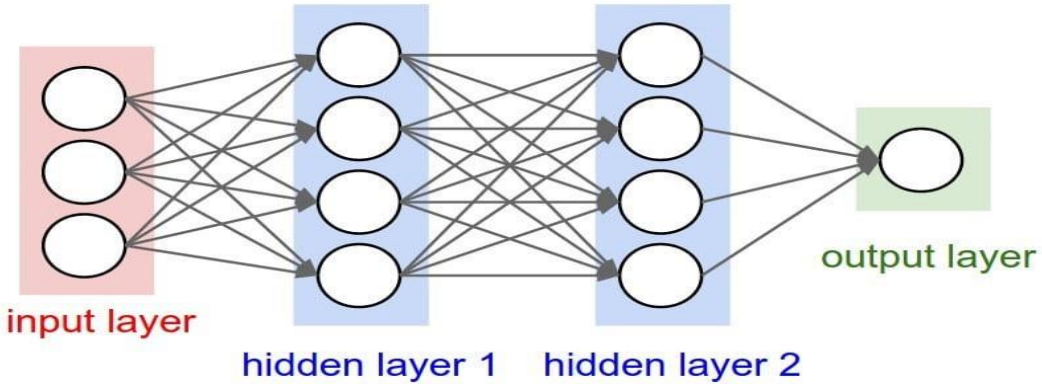
Random forest is one of the supervised learning algorithms. It is used both for classification and regression. Random forest algorithm selects random samples from the given dataset. It then constructs a decision tree for each and every sample and then gets a prediction from each of the decision trees. A vote is performed for each predicted result and the decision with the most votes is selected as the final prediction.

6.3 Deep Neural Network

Deep Neural Networks (DNN) are one of the main tools which are used in machine learning. "Neural" part of their name is called like that because these systems try to learn things like the human brain. Replicated networks contain some kind of neurons and these neurons create a network by connecting each other. These networks have a capacity of learning, storing and finding out relationships between data like a human!

For example, they can learn to identify images that contain cars by analyzing example images. So after learning phase is completed if you ask for algorithm 'Is it a car?' by giving it an image, the algorithm can answer your question because it identified other cars images and learned how a car looks like.

Neural Networks has input and output layers like others but most of the cases they also have hidden layer s, and usually, we can say how 'deep' our algorithm according to the number of hidden layers.



7. Performance Evaluation

The experiments are evaluated using 4 basic metrics - True Negative (TN), True Positive (TP), False Negative (FN) and False Positive (FP). The performance of the 7 methods is compared based on their accuracy, precision, f1 score and recall. True positives are the cases which are actually positive and are also classified as positive. Similarly, True negatives are the cases which are actually negative and are classified as negative. False positives are the cases which are actually negative but are classified as positives. Similarly, False negatives are the cases which are actually positive but are classified as negative.

Accuracy:

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

Precision:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positives}$$

Recall:

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

F1-Score:

Weighs both precision and recall into on measure

$$F1 - Score = \frac{2 \times Precision \times Recall}{Precision\ \times Recall} = \frac{2\ \times TP}{2 \times TP + FP + FN}$$

Accuracy is the measure of number of correct predictions divided by the total number of predictions. Precision is the ratio of positive predictions to the total number of positive classes predicted. Recall is the ratio of positive

8. Result

The result has given in the below charts is selecting all the features of the dataset.

	Model Name	*Train Accuracy*	*Test Accuracy*	*Precision*	*Recall*	*F1 score*
0	*Logistic Regression*	*0.92*	*0.92*	*0.00*	*0.00*	*0.00*
1	*K Neighbors Classifier*	*0.92*	*0.92*	*0.12*	*0.01*	*0.02*
2	*Gaussian NB*	*0.91*	*0.91*	*0.06*	*0.01*	*0.01*
3	*Decision Tree Classifier*	*1.00*	*0.85*	*0.13*	*0.16*	*0.15*
4	*Random Forest Classifier*	*1.00*	*0.92*	*1.0*	*0.0*	*0.0*
5	*MLP Classifier*	*0.92*	*0.92*	*0.0*	*0.0*	*0.0*
6	*Linear SVC*	*0.9*	*0.9*	*0.08*	*0.02*	*0.03*

Selecting 20 best features

Feature selection methods from this category basically assign score metrics for each feature by the help of various statistical calculations and then filter out the most irrelevant ones.

Filter methods are usually applied as a preprocessing step.

Estimate mutual information for a discrete target variable.

SelectKBest" class is actually a more general approach compared to other classes, since it takes an additional scoring function parameter which states which function to use in feature selection. So, we can think it as a kind of wrapper. We used mutual_info_classif inside this object.

mutual_info_classif method basically utilize the mutual information. It calculates mutual information value for each of independent variables with respect to dependent variable, and selects the ones which has most information gain. In other words, it basically measures the dependency of features with the target value. The higher score means more dependent variables.

Here we use K-best and mutual_info_classif select method to find out the top 20 best features.

	Feature	Score
19	FLAG_MOBIL	0.065781
22	FLAG_CONT_MOBILE	0.064850
20	FLAG_EMP_PHONE	0.043878
11	NAME_EDUCATION_TYPE	0.043774
13	NAME_HOUSING_TYPE	0.041062
27	REGION_RATING_CLIENT_W_CITY	0.039689
26	REGION_RATING_CLIENT	0.039036
52	FLAG_DOCUMENT_3	0.033848
4	FLAG_OWN_REALTY	0.030956
10	NAME_INCOME_TYPE	0.023887
12	NAME_FAMILY_STATUS	0.022450
25	CNT_FAM_MEMBERS	0.020013
16	DAYS_EMPLOYED	0.016007
8	AMT_ANNUITY	0.012324
37	EXT_SOURCE_2	0.012013
38	EXT_SOURCE_3	0.011803
28	WEEKDAY_APPR_PROCESS_START	0.009486
3	FLAG_OWN_CAR	0.008728
2	CODE_GENDER	0.008503
36	ORGANIZATION_TYPE	0.008255

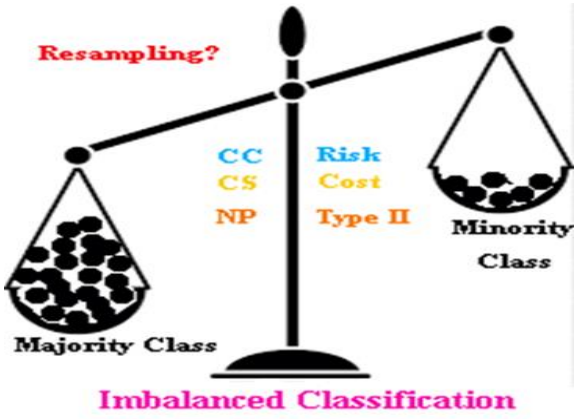
Here I have applied the best 20 features to improve my accuracy on different methods and the result are here:

	Model Name	*Train Accuracy*	*Test Accuracy*	*Precision*	*Recall*	*F1 score*
0	*Logistic Regression*	*0.92*	*0.92*	*0.00*	*0.00*	*0.00*
1	*Gaussian NB*	*0.92*	*0.92*	*0.00*	*0.00*	*0.00*
2	*K Neighbors Classifier*	*0.92*	*0.91*	*0.12*	*0.01*	*0.02*
3	*Decision Tree Classifier*	*1.00*	*0.85*	*0.12*	*0.14*	*0.12*
4	*Random Forest Classifier*	*1.0*	*0.92*	*0.32*	*0.01*	*0.01*
5	*MLP Classifier*	*0.92*	*0.92*	*0.0*	*0.0*	*0.0*

6	*Linear SVC*	*0.76*	*0.76*	*0.05*	*0.12*	*0.08*

Data Imbalance-Over Sampling

One of the common issues found in datasets that are used for classification is imbalanced classes issue. Imbalanced data typically refers to a classification problem where the number of observations per class is not equally distributed. It usually reflects an unequal distribution of classes within a dataset.



If there are two classes, then balanced data would mean 50% points for each of the class. In our case 50:1 ratio between the fraud and non-fraud classes which is highly imbalance. There are 3 major techniques are there to eliminate the imbalance problem.

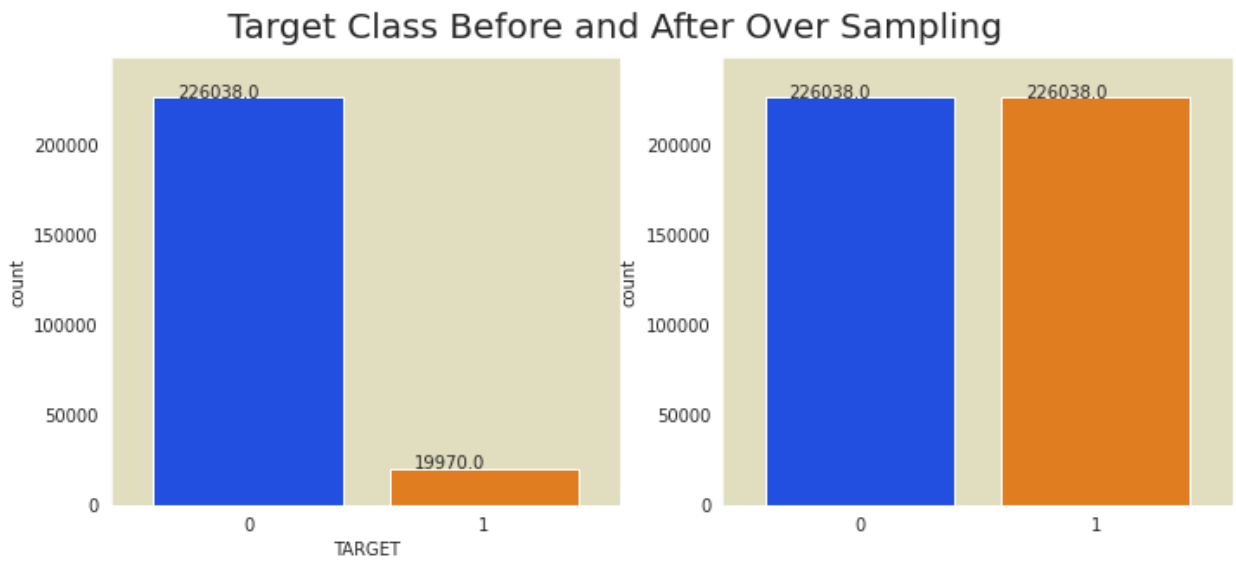
1. Random Over Sampling

2. Random Under Sampling

3. SMOTE (Synthetic Minority Over-sampling Technique).

Since under sampling may discard the useful information which could be important for building good classifiers, we choose random over sampling and SMOTE (Synthetic Minority Oversampling Technique). Random over sampling increases the number of instances in the minority class by randomly replicating them in order to present a higher representation of the minority class in the sample. Implementing Random over sampling on this dataset helps to the balance the labels (more no fraud than fraud transactions).

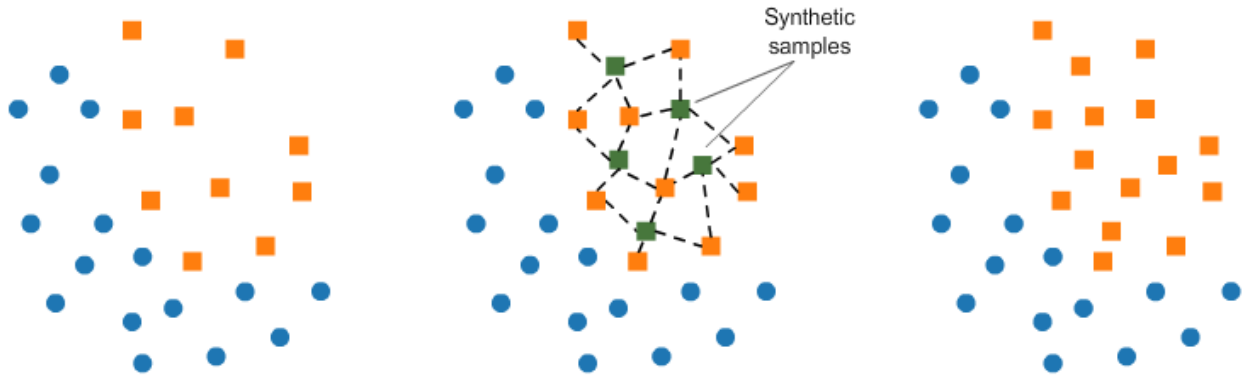
```
Before Oversampling: Counter ({0: 226038, 1: 19970})
After Oversampling: Counter ({0: 226038, 1: 226038})
```



The chart has shown about oversampling method

	Model Name	*Train Accuracy*	*Test Accuracy*	*Precision*	*Recall*	*F1 score*
0	*Gaussian NB*	*0.50*	*0.90*	*0.07*	*0.02*	*0.03*
1	*Decision Tree Classifier*	*1.00*	*0.86*	*0.14*	*0.15*	*0.14*
2	*K Neighbors Classifier*	*0.92*	*0.74*	*0.09*	*0.26*	*0.14*
3	*Logistic Regression*	*0.57*	*0.59*	*0.11*	*0.58*	*0.18*
4	*Random Forest Classifier*	*1.0*	*0.92*	*0.34*	*0.0*	*0.01*
5	*MLP Classifier*	*0.5*	*0.92*	*0.0*	*0.0*	*0.0*
6	*Linear SVC*	*0.51*	*0.97*	*0.13*	*0.08*	*0.15*

Smote Sampling on Different Model



SMOTE is a technique that generates new observations by interpolating between observations in the original dataset. Implementing SMOTE on our imbalanced dataset helped us with the imbalance of our labels (more no fraud than fraud transactions).

```
Before SMOTE: Counter ({0: 226038, 1: 19970})
After SMOTE: Counter ({0: 226038, 1: 226038})
```

	Model Name	*Train Accuracy*	*Test Accuracy*	*Precision*	*Recall*	*F1 score*
0	*DecisionTreeClassifier*	*1.00*	*0.84*	*0.13*	*0.18*	*0.15*
1	*GaussianNB*	*0.58*	*0.82*	*0.12*	*0.21*	*0.15*
2	*KNeighborsClassifier*	*0.89*	*0.68*	*0.09*	*0.34*	*0.15*
3	*LogisticRegression*	*0.58*	*0.61*	*0.11*	*0.56*	*0.18*
4	*Random Forest Classifier*	*1.0*	*0.92*	*0.0*	*0.0*	*0.0*
5	*MLPClassifier*	*0.5*	*0.92*	*0.0*	*0.0*	*0.0*
6	*LinearSVC*	*0.5*	*0.9*	*0.09*	*0.03*	*0.04*

After applying Random over sampling and SMOTE, there is a drastic decrease in the accuracy of the model. Accuracy is not the best metric to use when evaluating imbalanced datasets as it can be misleading instead use f1-score, precision/recall score or confusion matrix. After applying SMOTE we observe precision/recall of minority class increases.

Ensemble Method

In order to increase the performance of the model we use ensemble techniques. Here we use 3 ensembling models.

1.CatBoostClassifier

2.BaggingClassifier

3.AdaBoostClassifier

	Model Name	*Train Accuracy*	*Test Accuracy*	*Precision*	*Recall*	*F1 score*
0	*Cat Boost Classifier*	*0.92*	*0.92*	*0.37*	*0.01*	*0.01*
1	*XGB Classifier*	*0.92*	*0.92*	*0.55*	*0.00*	*0.00*
2	*LGBM Classifier*	*0.41*	*0.37*	*0.10*	*0.82*	*0.17*

DNN Model building

Here we will build a 5-layer deep neural networking using the Sequential model in **Keras**. Specifically,

```
model_Neural = Sequential([
    Dense(units=20, input_dim = X_train.shape[1], activation='relu'),
    Dense(units=24,activation='relu'),
    Dropout(0.5),
    Dense(units=20,activation='relu'),
    Dense(units=24,activation='relu'),
    Dense(1, activation='sigmoid')
])
```

For the 1st hidden layer, 'input_dim' is the number of input variables. 'units' is the number of nodes or neurons in each layer. We use Rectified Linear Unit (ReLU) as an activation function for the hidden layers. ReLU normally performs better than Sigmoid and Hyperbolic Tangent functions when building deep neural networks. This is because Sigmoid and Tanh tends to saturate when the input value is either too large or too small. In addition, they only show a high gradient around their mid-points, such as 0.5 for sigmoid and 0 for tanh.

We use the Sigmoid function in the output layer for a binary classification problem.

DNN model evaluation

```
model_Neural.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
model_Neural.fit(X_train, Y_train, batch_size=30, epochs=10)
```

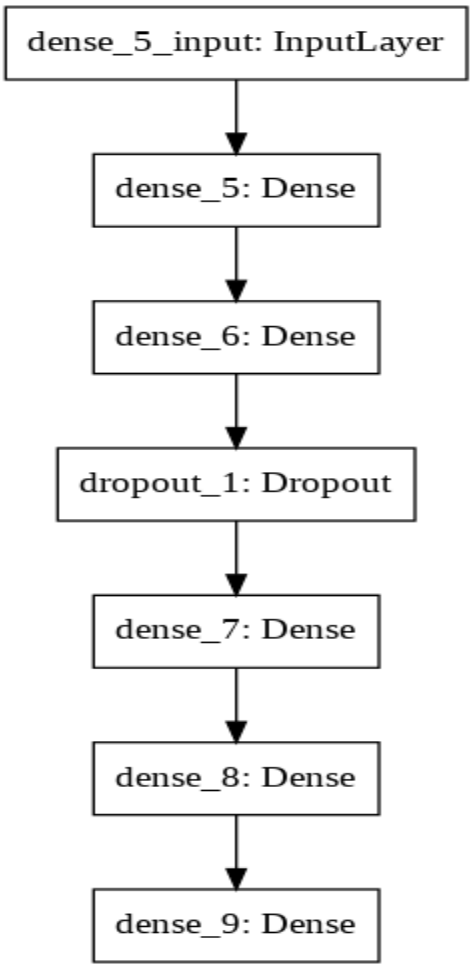
Note above we use 'binary_crossentropy' as the loss function and 'Adam' to update network weights. 'Adam' is a popular algorithm to achieve good results fast in the deep learning field.

The model weights are updated every 30 samples. for clear with the concept of epoch and batch. An epoch represents one time that the entire training set is fed through the network. A batch defines the number of samples to iterate through before updating the internal model parameters. At the end of the batch, the predictions are compared to the expected output, and error is calculated. Using this error, the optimizer improves the model by moving down the error gradient. Now let's evaluate the model. Specifically,

```
1922/1922 [==============================] - 3s 2ms/step - loss: 0.2816
- accuracy: 0.9211
Test Accuracy: 92.11%
Test Loss: 0.2815752923488617
```

Remember we use 'accuracy' as metrics. The model is found with 92.11% accuracy

The final plot:



Conclusion

This report successfully investigates the performance of Logistic Regression, Gaussian NB, Decision trees, K nearest neighbors, Multi-Layer Perceptron, Random Forests, DNN and Ensemble method in determining fraudulent credit card transactions. So far we tried using various performance metrics like accuracy, F1-Score, Precision-Recall curve for different techniques like over-sampling of the minority class, under-sampling of majority class and SMOTE (Synthetic Minority Over-Sampling Technique).

- ✓ EDA helpful to understand the overview of the dataset.
- ✓ 9 different classifier models are trained on real life dataset and their performances are evaluated based on various parameters and metrics.
- ✓ Feature Selection reduce the training time and do not effect of model performance.
- ✓ SMOTE reduce accuracy of the model but it increases the overall performance of the dataset.
- ✓ In 3 baseline classification model decision trees perform better compare to other two models.
- ✓ Ensembling learning techniques increase the overall performance
- ✓ Random forest classifier always provides good accuracy.
- ✓ DNN also provided little bit good accuracy comparing ML other methods.

Based on our evaluation metrics we found that undersampling of majority class resulted in poor performance when compared to Over-Sampling techniques. It's still hard to pick a winner here. Comparing the performance of the different classifiers used, it is clear that linear SVM produces the best performance over others model in oversampling method.

References:

1. *https://www.researchgate.net/publication/335526336_Credit_Card_Fraud_Detection_Using_Autoencoder_Model_in_Unbalanced_Datasets*
2. *Predictive Modelling For Credit Card Fraud Detection Using Data Analytics- Suraj Patil*, Varsha Nemade, PiyushKumar Soni- www.sciencedirect.com*
3. *Credit Card Fraud Detection using Machine Learning Algorithms- Vaishnavi Nath Dornadulaa*, Geetha Sa- www.sciencedirect.com*
4. *Credit Card Fraud Detection using Pipeling and Ensemble Learning --Siddhant Baggaa, Anish Goyala, Namita Guptab, Arvind Goyal- www.sciencedirect.com*
5. *https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/*
6. *https://towardsdatascience.com/credit-card-fraud-detection-9bc8db79b956*
7. *https://medium.com/@Nethone_/a-beginners-guide-to-machine-learning-in-payment-fraud-detection-prevention-360c95a9ca54*
8. *https://medium.com/@cdabakoglu/artificial-neural-network-with-keras-d858f82f90c5*
9. *https://www.kaggle.com/nareshbhat/fraud-detection-feature-selection-over-sampling*
10. *https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/*
11. *https://spd.group/machine-learning/credit-card-fraud-detection-case study/#Credit_Card_Fraud_Detection_Algorithm*