**CREDIT CARD FRAUD DETECTION USING**

**Synthetic Minority Oversampling Technique [SMOTE] Technique.**

**INTRODUCTION:**

The crime of credit card fraud begins when someone either steals a credit or debit card, or fraudulently obtains the card number and other account information necessary for the card to be used successfully. In general fraud detection is very challenging because fraudsters are coming up with new and innovative ways of detecting fraud in this digital world, so it is difficult to find a pattern that we can detect. Fraud Detection is a common problem in Machine Learning

**Traditional fraud detection with rules-based systems**

**Drawbacks of using rules-based systems**

Rules based systems have their limitations:

1. Fixed thresholds per rule to determine fraud

2. Limited to yes/no outcomes

3. Fail to capture interaction between features

**Use machine learning for fraud detection!**

1. Machine learning models adapt to the data, and thus can change over time

2. Uses all the data combined rather than a threshold per feature

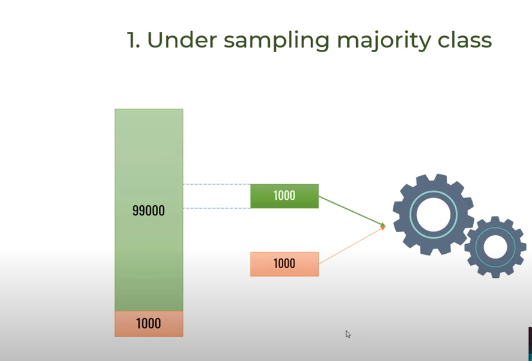
3. Can give a score, rather than a yes/no

4. Will typically have a beer performance and can be combined with rules

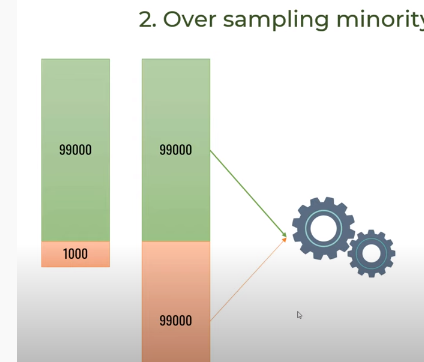
When you are training your model with training dataset for fraud transaction you will often find that you will have 10,000 good transaction and only one will be fraud this create imbalance in your dataset. We will get 99% accuracy; reason is majority of transaction is not fraud. But the other hand we will have to care about the forefront transaction although accuracy is 99%, because this simple not telling us what fraud is? So, this kind of imbalance create a lot of issues. To solve this type of problem can be tackle with the help of some techniques for handling imbalance data

**TECHNIQUES TO HANDLE IMBALANCE DATA**

1. **UNDER SAMPLING MAJORITY CLASS**

Let’s suppose we have 99,000 sample (Not Fraud) belonging to class A and 1000 sample(Fraud) belonging to class B .To tackle that imbalance we have to take randomly 1000 samples from 99,000 samples [class A=>Not Fraud] and discard remaining samples, and then combine with 1000 samples [class B =>Fraud ] and train your model.But the problem is throwing away so much data.

1. **Over Sampling the Minority Class by Duplication**

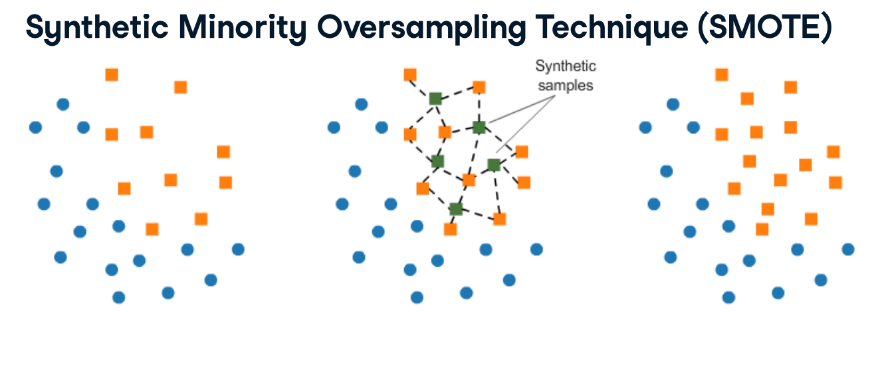
In this technique, we simply duplicate the minor class [Class B=>Fraud] transaction 99 times and you get 99 000 transaction and then you train the model.

1. **Over Sampling Minority Class using SMOTE**

SMOTE (synthetic minority oversampling technique) is one of the most commonly used oversampling methods to solve the imbalance problem. It aims to balance class distribution by randomly increasing minority class examples by replicating them.

SMOTE synthesises new minority instances between existing minority instances

In python, Model is available **Imblearn** which can be used for SMOTE.

SMOTE is slightly more sophisticated than just copying observations. We apply SMOTE to our credit card data set.

**Logistic Regression:**

Logistic Regression is an algorithm that can be used for regression as well as classification task, but it is widely used for classification tasks. The response variable that is binary belongs to one of the classes. It is used to predict categorical variables with the help of dependent variables. We can build and evaluate our logistic regression model using python’s scikit-learn package.

**Classification Report:**

A classification report will give the following results,

**Accuracy:** Accuracy is a ratio of correctly predicted observation to the total observations True Positive: The number of correct predictions that the occurrence is positive. True Negative: Number of correct predictions that the occurrence is negative.

**F1- Score:** It is the weighted average of precision and recall

**Precision and Recall:** Precision is the fraction of relevant instances among the retrieved instances, while recall is the fraction of relevant instances that have been retrieved over the total number of instances. They are basically used as the measure of relevance.

**Confusion matrix:**

A confusion matrix is a matrix (table) that can be used to measure the performance of an machine learning algorithm, usually a supervised learning one. Each row of the confusion matrix represents the instances of an actual class and each column represents the instances of a predicted class.

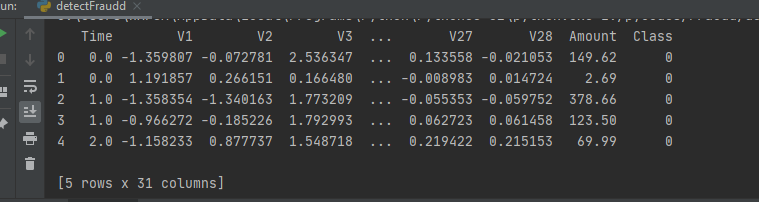
**Pipeline:** Pipelines are nothing but an object that holds all the processes that will take place from data transformations to model building.

**DATASET:**

This project uses the Credit Card Fraud Detection dataset from **Kaggle.**

European Dataset. The datasets contain transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

Features **V1, V2, … V28** are the principal components obtained with PCA, the only features which have not been transformed with PCA are '**Time**' and '**Amoun**t'. Feature '**Time**' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature '**Amount**' is the transaction Amount, this feature can be used for example-dependant cost-senstive learning. Feature '**Class**' is the response variable and it takes value 1 in case of fraud and 0 otherwise.



**Methodology:**

Step1: Import the Libraries

Step2: Load the dataset

Step3: Apply data preprocessing on the loaded dataset.

Step4: Define the resampling method using SMOTE technique

Step5: plot the original data and also plot the resampled data.

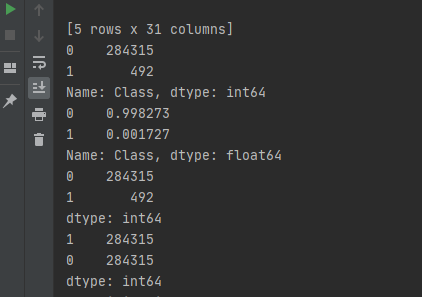
Step6: Create the training and testing sets 70% for training and 30% for Testing.

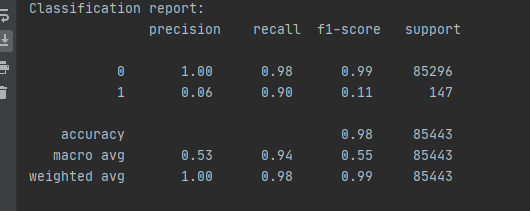
Step5: Fit a logistic regression model to our data

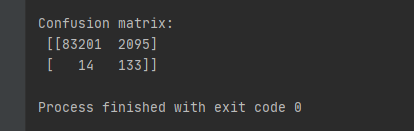
Step6: Show the classification report and confusion matrix

Step7: Define the pipeline, tell it to combine SMOTE with the Logistic Regression model.

**Result:**





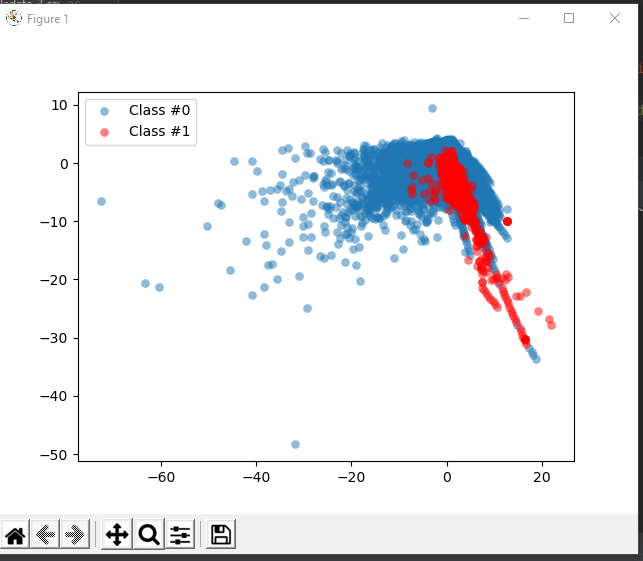


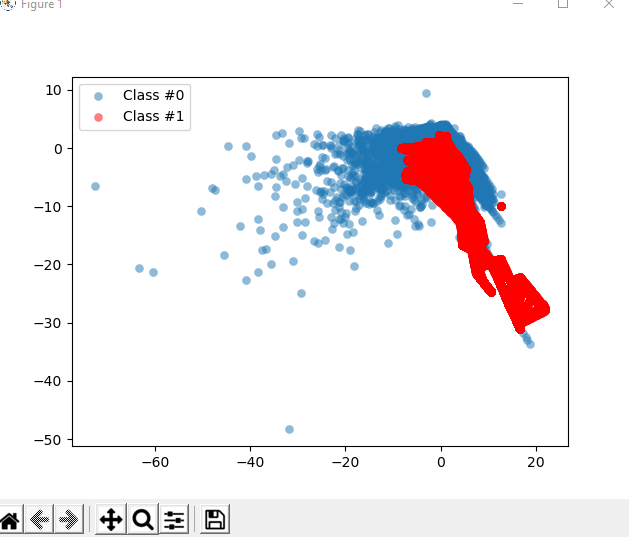
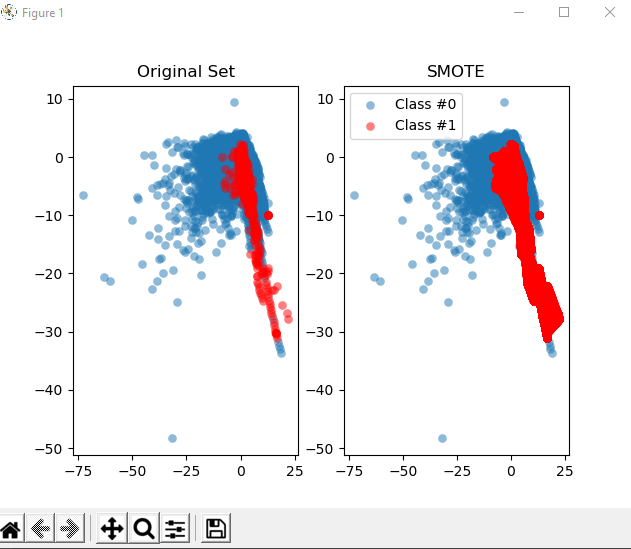
**Plots**

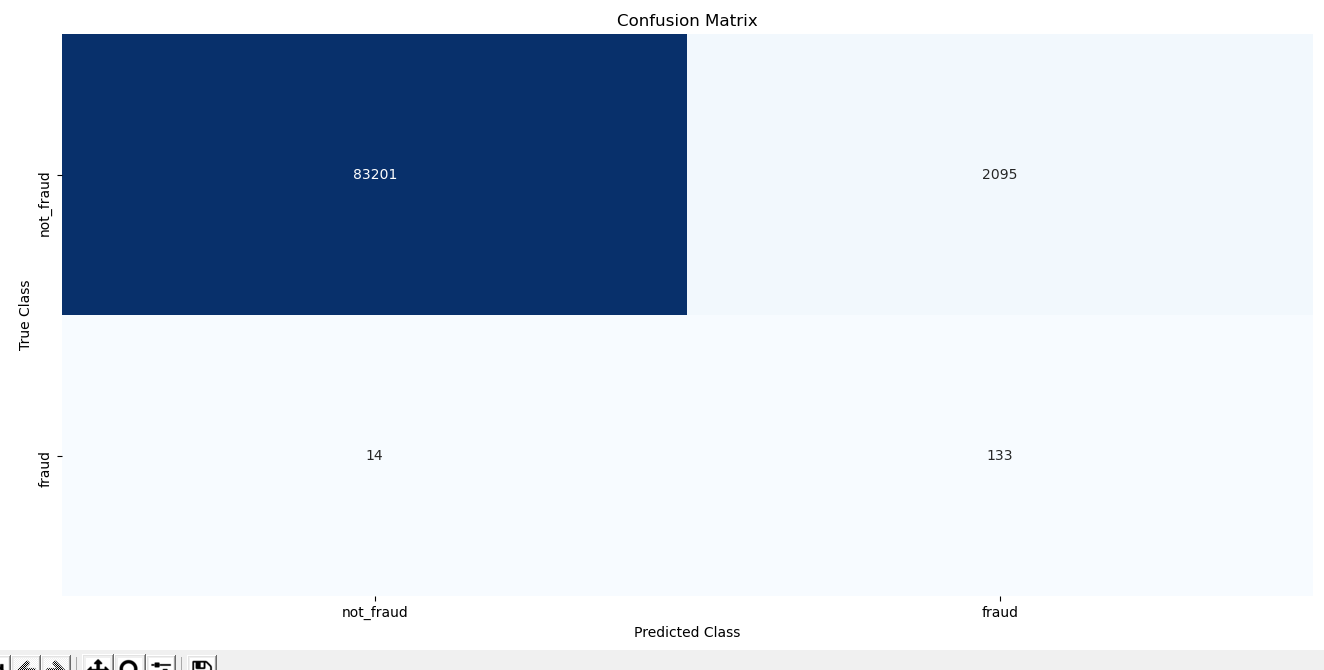
Class# 0=>not fraud

Class #1 => Fraud

* **Original dataset scatter plot.**



* **After Applying SMOTE Technique:**
* **Comparison Plot:**
* **Confusion Matrix heat Map**



**Conclusion:**

We are getting fewer false positives, also we are catching a higher percentage of fraud cases. We are only using our test data to calculate the model results on. Few observations available to look at in the confusion matrix because we are using 30% of our dataset. Accuracy is about 99%, for not fraud transaction, to reduce the accuracy of not fraud transaction we simple use smote oversampling technique.by increasing the number of fraud transactions. We have reduced the accuracy to 98% as compared to previous model but the recall value of minority class has also improved to 92 %, this is a good model.