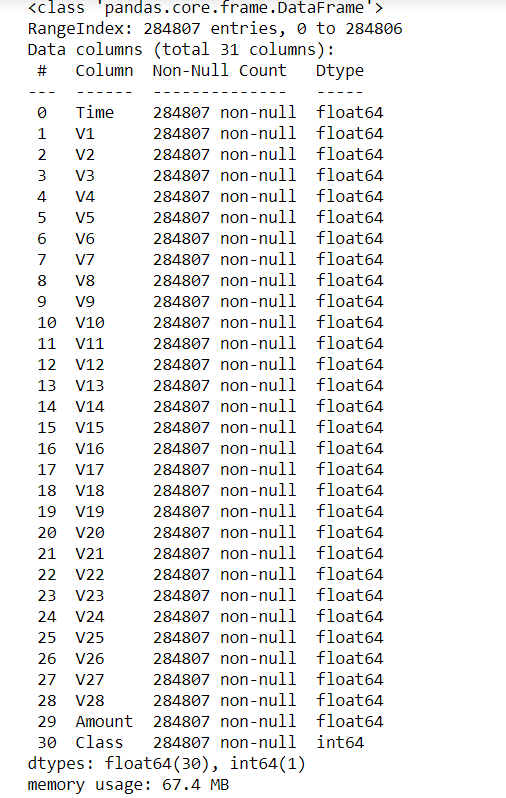
**Credit Card Fraud Detection Documentation**

1. **Motivation:**
   1. The reason for picking up this problem is that in daily life there is more number of fraudulent intruders interacting like customers and using their money.
   2. To avoid these scenarios we need some system to be developed in identifying the fraudulent behaviors.
   3. It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.
2. **Problem Statement:**
   1. The dataset contains transactions made by credit cards in September 2013 by European cardholders.
   2. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions.
   3. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.
   4. It contains only numerical input variables which are the result of a PCA transformation.
   5. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data.
   6. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'.
   7. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset.
   8. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning.
   9. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.
   10. Given the class imbalance ratio, we recommend measuring the accuracy using the Area Under the Precision-Recall Curve (AUPRC).
3. **Proposed Approach:**
   1. This project **proposes** an intelligent **credit card fraud detection** model for **detecting fraud** from highly imbalanced and anonymous **credit card** transaction **datasets**.
   2. The class imbalance problem is handled by finding legal as well as **fraud** transaction patterns for each customer by using frequent itemset mining.
   3. The most commonly techniques **used fraud detection** methods are Naïve Bayes (NB), Support Vector Machines (SVM), K-Nearest Neighbor **algorithms** (KNN).
   4. But here we have used Logistic regression and Random Forest Classifier for better result.
4. **Database:**
   1. We have fetched the dataset from Kaggle.com as excel sheet.
   2. The link to the dataset is <https://www.kaggle.com/mlg-ulb/creditcardfraud>
5. **Experiments:**
   1. After receiving the dataset, we have done the basic steps,
      1. Data Wrangling
      2. Data Cleaning & Pre-processing
      3. EDA
      4. Modeling

**Data Wrangling:**

**Data preprocessing** is a **data** mining **technique** which is used to transform the raw **data** in a useful and efficient format.

**As this dataset doesn't carry any missing values and null values, so data wrangling looks very easy here. Will proceed with the other findings in the upcoming steps.**



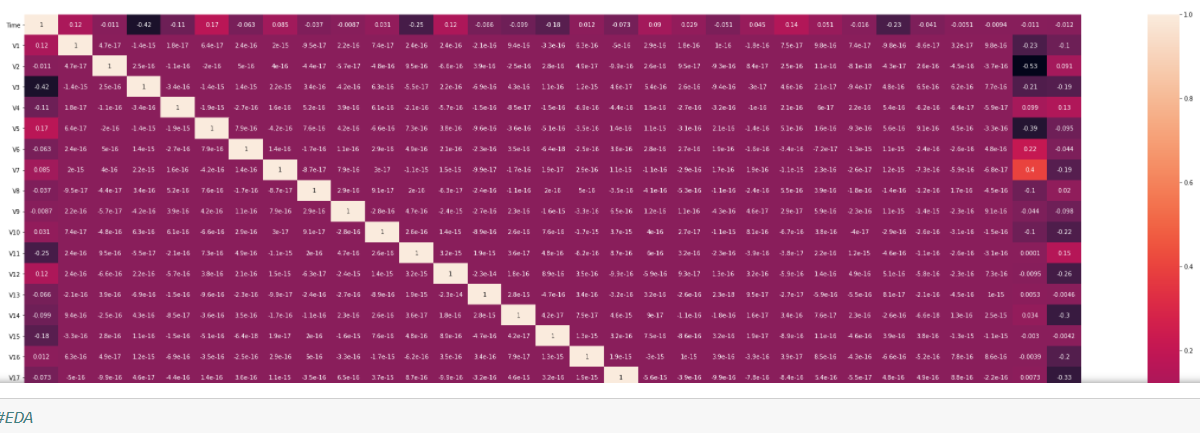
**EDA:**

In EDA, the exploratory data analysis will be done,

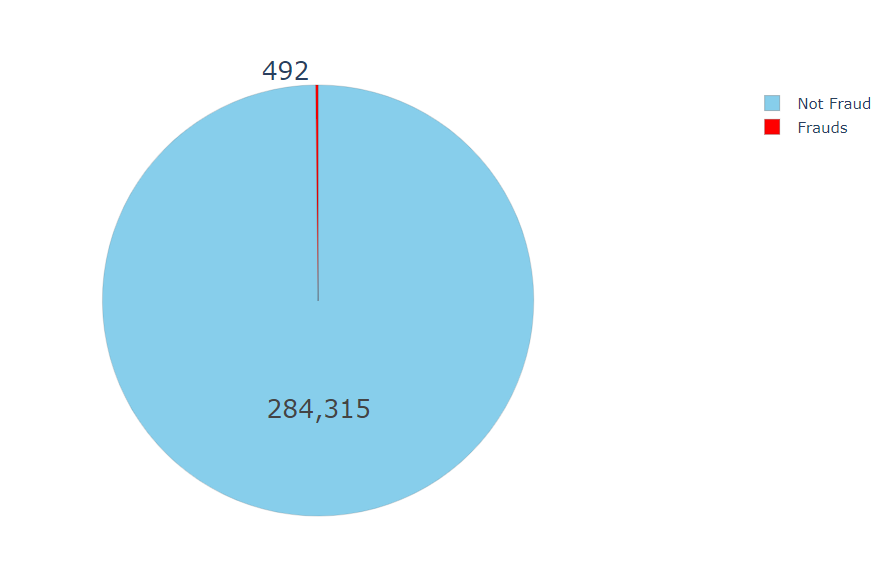
SNS Plot for class distributions:



Correlation of data in Heat map :

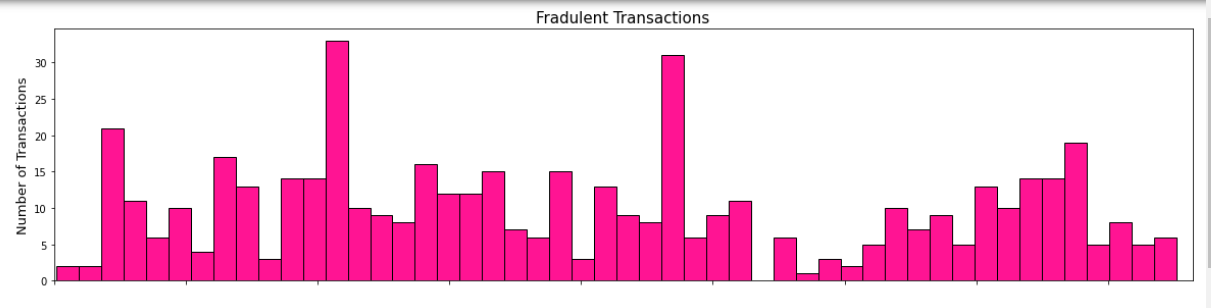


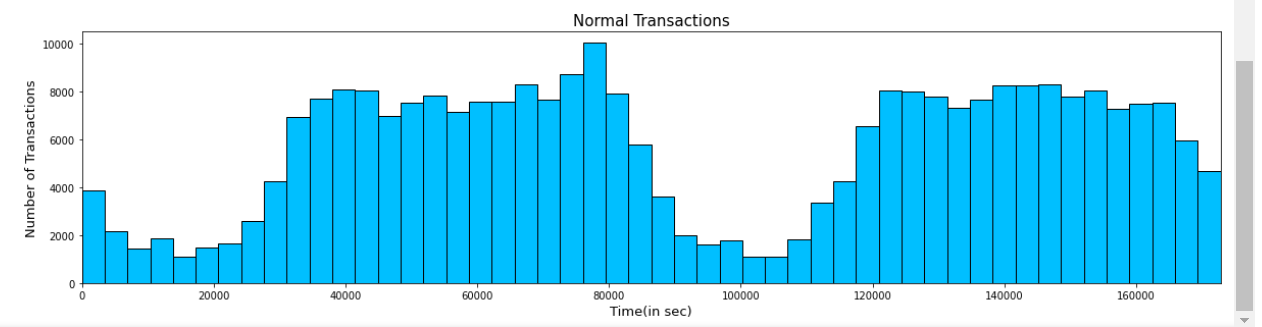
Pie Chart for exploring Fraudulent and non-fraudulent Information:

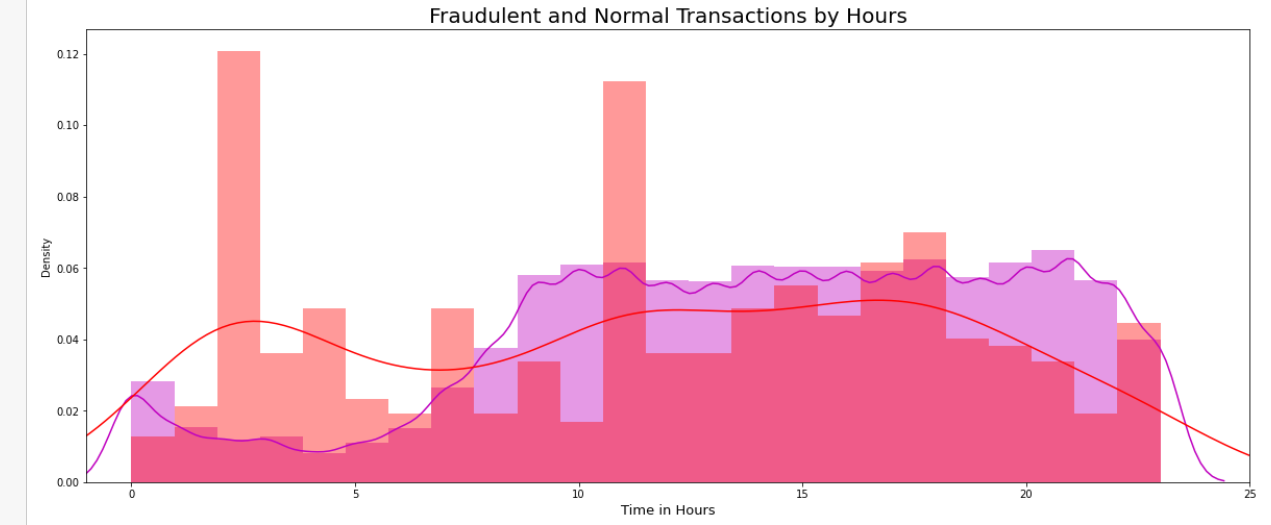


OBSERVATIONS:

* This dataset has 492 frauds out of 284,315 transactions. Thus, the dataset is highly unbalanced, the positive class (frauds) account for 0.173% of all transactions.
* Most of the transactions are non-fraud which is obvious. If we use this data for our predictive models and analysis, our algorithms will probably overfit to the non-fraudulent transactions and will answer in non-fraudulent all the time which can result in actual frauds to slip by!
* Note that our task is not to find the obvious, rather we have to find the anomalies and signs of fraud! Thus, we will take care of this imbalance during preprocessing.
* There are 113 fraud transactions for just one dollar and 27 fraud transaction for 99.99 dollars. Also, there are 27 fraud transaction for zero amount.
* The reason for zero transaction can be the Zero Authorization which is an account verification method for credit cards that is used to verify a cardholders information without charging the consumer.
* In Fraudulent transactions, there were slight peak at 40000 and 100000 time period
* In Normal transactions, there were slight down at 20000 and 100000 time period which is not very useful data for now





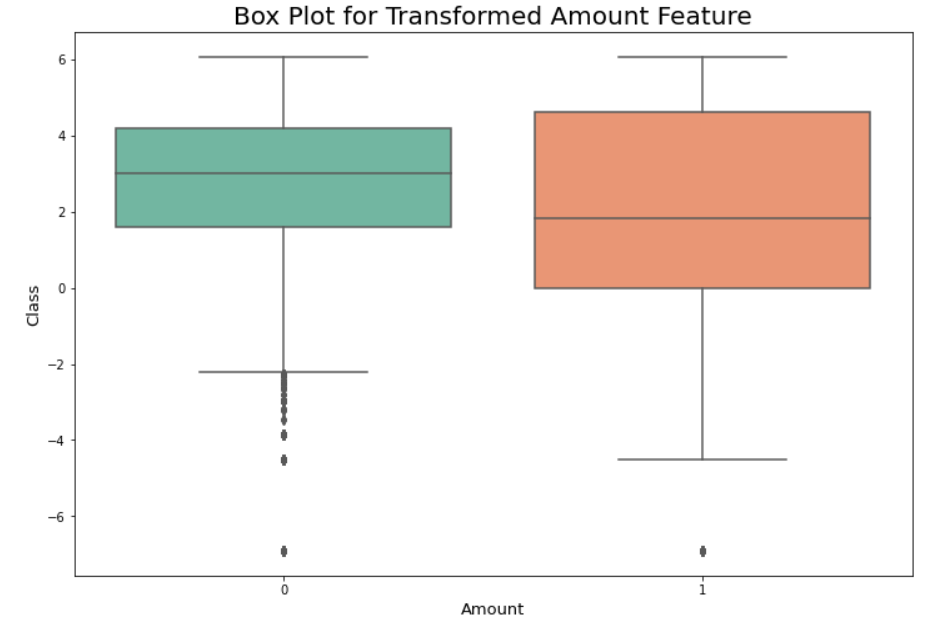


**Data Cleaning & Pre-processing**

* For some of the features we can observe a good selectivity in terms of distribution for the two values of Class: V4, V11 have clearly separated distributions for Class values 0 and 1, V12, V14, V18 are partially separated, V1, V2, V3, V10 have a quite distinct profile, whilst V20-V28 have similar profiles for the two values of Class and thus not very useful in differentiation of both the classes.
* In general, with just few exceptions (Time and Amount), the features distribution for legitimate transactions (values of Class = 0) is centered around 0, sometime with a long queue at one of the extremities. In the same time, the fraudulent transactions (values of Class = 1) have a skewed (asymmetric) distribution.:

**Outlier Removal**

* As we already saw that amount column has a extreme outliers so it necessary to remove them as they can effect the model's performance. We will used Interquartile range to detect outliers which removes anything below the lower limit (25 percentile) and anything above upper limit (75 Percentile).
* Note that, the data we have for fraudulent cases is very low so we wanna keep our cutoff a bit high so as avoid removing much of the fraud cases. Here, as the data is skewed (kind of exponential) so having high cutoff will help us. Let's take the cutoff value as 5.0 instead of 1.5 which is usually used.



**Handling Class Imbalance**

* Imbalanced data is a problem in supervised learning problems which can result is high bias towards majority class. As we have already seen that this data is severly imbalanced so to balance it we can use various techniques such as:

1. Oversampling
2. Undersampling
3. SMOTE

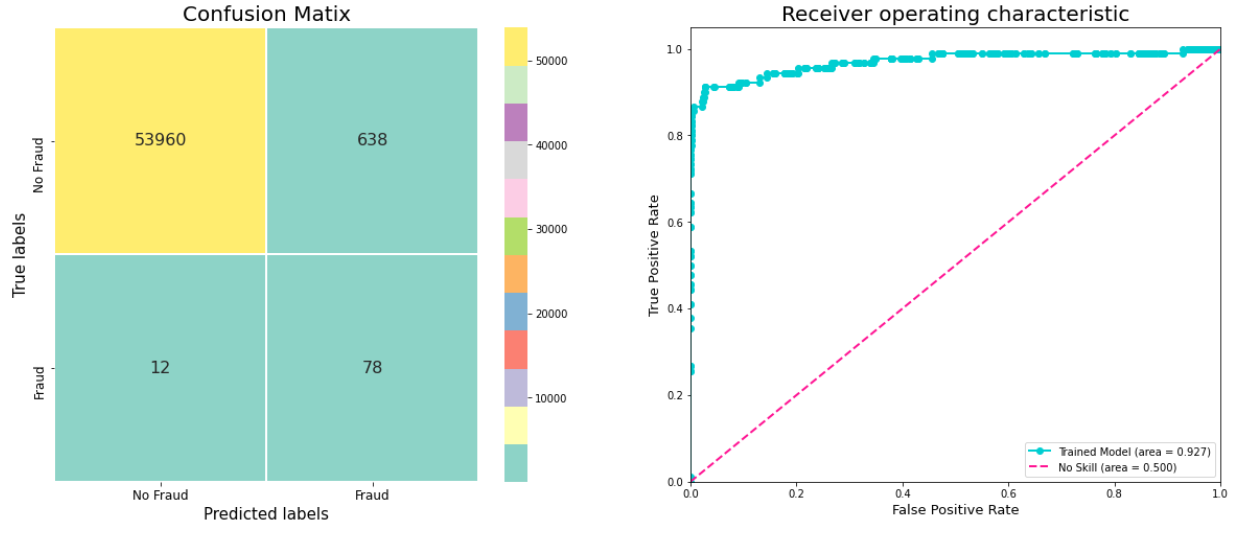
* Out of all these three SMOTE is the most effective so we will go with it, In this technique, instead of simply duplicating data from the minority class, we synthesize new data from the minority class. This is a type of data augmentation for tabular data can be very effective. This approach to synthesizing new data is called the Synthetic Minority Oversampling Technique, or SMOTE for short.
* We can clearly see that now the data is completely balanced so let's use some visualization technique to visualize this data.
* Note that, as our data has a lot of columns and humans can only understand 3D so we will use Dimensionality reduction technique to reduce our data to 3D and then plot it. So, let's get started!
* As our data is imbalanced so we will not use train\_test\_split and instead we will use stratified split which will take the representative of respective populations i.e Fraud Transactions and Normal Transactions.

**Modeling:**

* In this section, we will finally apply models and classify whether a certain transaction done a particular time is fraud or genuine. Thus, this is a binary classification problem.
* One thing we should keep in mind that we might get very high accuracy but we should focus on optimizing out f1\_score and recall as we want to perform better on fraud cases as they are the most important.

**Logistic Regression:**

* Let's start off with a simple model like Logistic Regression. Note that I will be doing cross validation using Randomized search as the data is very huge and we will do this cross validation after splitting to avoid Data Leakage as discussed above.
  + 1. Accuracy: 95.35%
    2. Precision: 0.07
    3. Recall: 0.91
    4. f1 Score: 0.12



**Random Forest Classifier:**

* Now, let's try something which can take account of complex relationships. There are many such models but Random forest is bit better as it is a ensemble model and focuses on reducing variance i.e overfitting without much effecting the bias which is all we want. Also, this algorithm works in time complexity, O(d.n.log(n)) where d is the number of features.
* I have shown the best parameters after GridsearchCV and not the whole process itself as it is very time consuming and takes forever so you can try it yourself.
  + 1. Accuracy :0.99943
    2. AUC : 0.82778
    3. Precision : 1.00000
    4. Recall : 0.65556
    5. F1 : 0.79195

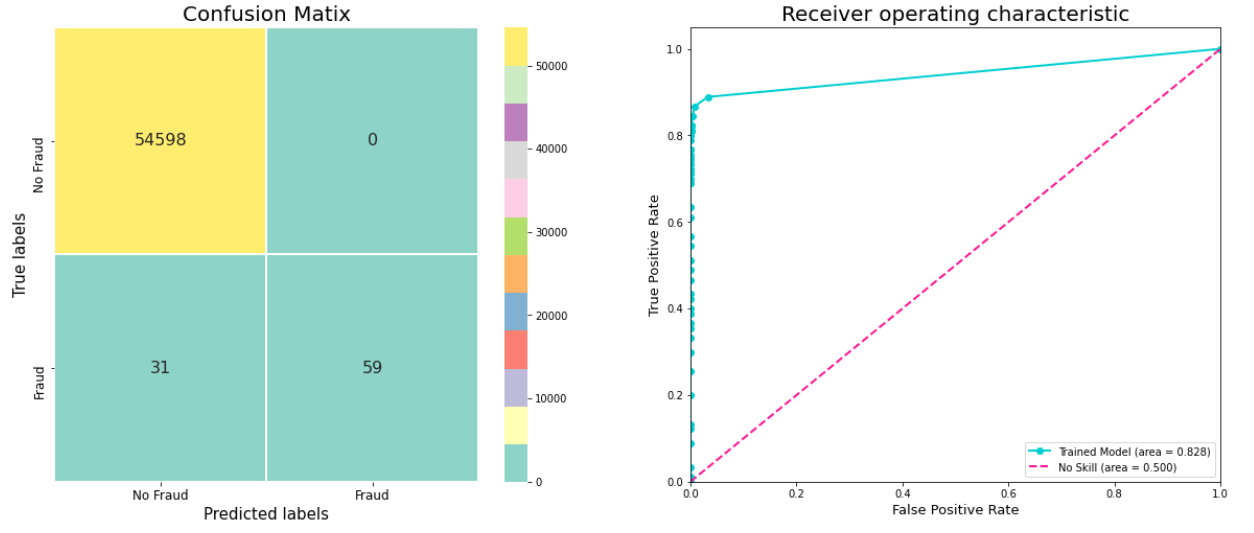
**Results:**

**Logistic Regression:**

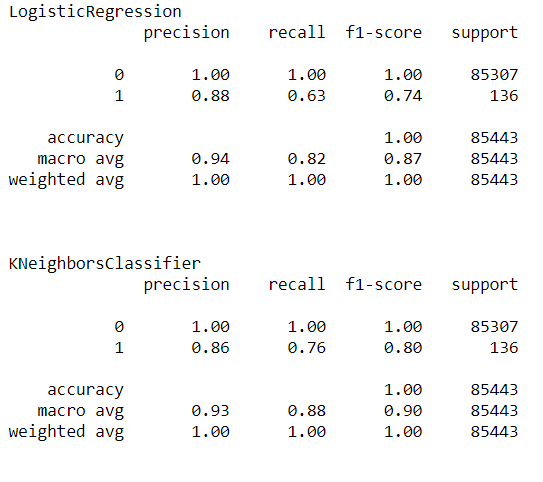
* Accuracy: 95.35%
* Precision: 0.07
* Recall: 0.91
* f1 Score: 0.12

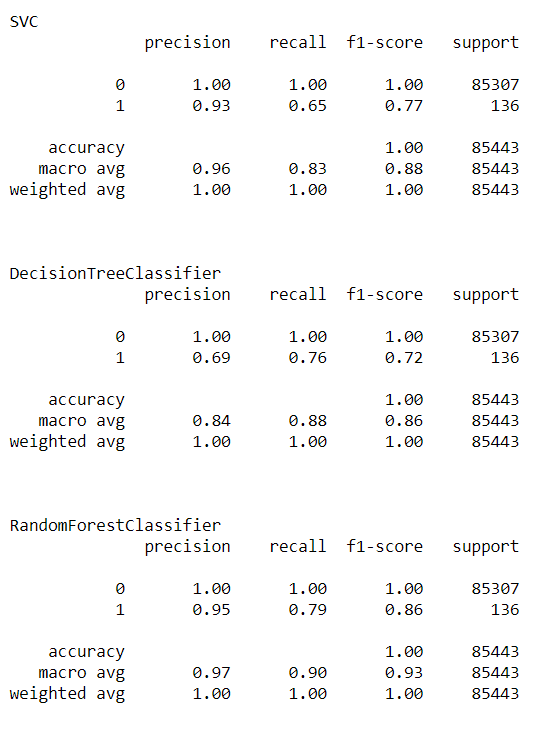
**Random Forest Classifier:**

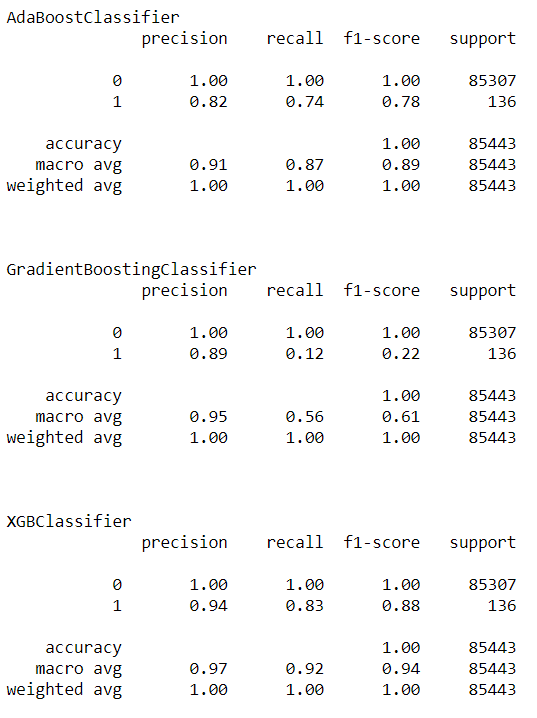
* Accuracy : 0.99943
* AUC : 0.82778
* Precision : 1.00000
* Recall : 0.65556
* F1 : 0.79195

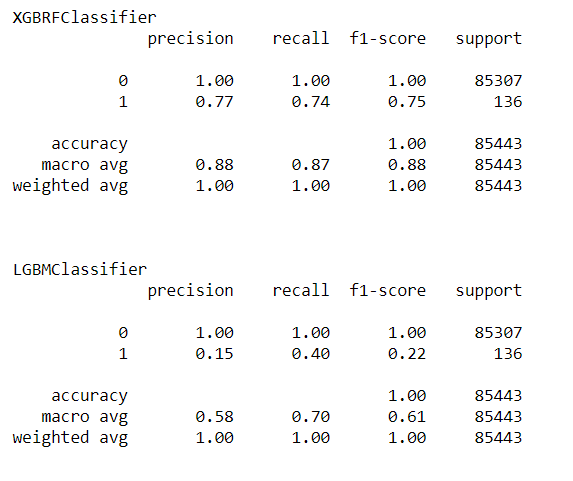
****

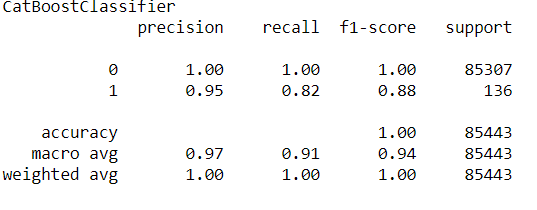
**Results of ML Algorithms Used:**



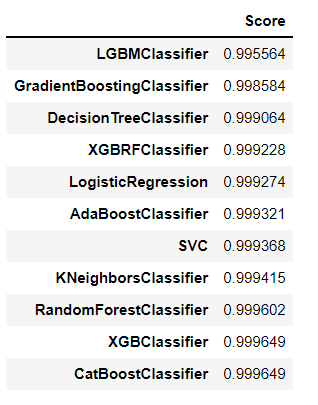




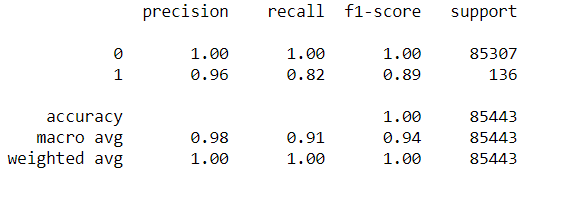




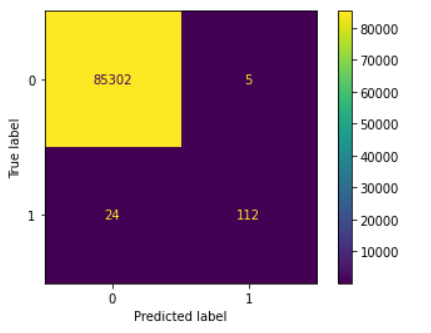
**Scores of Different Algorithms Used:**



**XGB Scores:**

****

**Confusion Matrix:**

****

**CV Scores:**



**Conclusion:**

* Logistic regression did really well on the fraud class giving almost 95% accuracy and algorithm does well on the fraud class, predicting 78 fraud cases are correct but 12 cases as wrong so we still have a room for improvement. It we applied model to the imbalanced data instead then it results would be pathetic. You can try that for yourself.
* Precision and accuracy are good but the recall is pretty low due to which the model is not performing well on fraudulent data.
* From all the data, XGB performed better with precision score and the CV Score has good Accuracy, Better Precision whereas Recall and f1 scores are pretty low.

**Future Work:**

* As the recall is bit lower will be working on various ML method to improve the recall and f1 scores.