MLSL1 Individual Assignment

Report by **Yash Srivastava**

PGID: **12010060**

Business Understanding

Objective: To Maximize the timely deliveries for Fedex
Constraints: Minimize delay

Dataset Discription: We are provided with 6 months of data of all the deliveries sent by Fedex for year 2008. The deliveries are classified as 0 and 1 where 0 is when the package was delivered on time and vice versa.

We need to build a prediction model which will classify weather the package will be delivered on time or the package will be delayed.

Data Understanding and Data Preparation

Null Values: Actual_Shipment_Time = 2%, Planned_TimeofTravel < 1%, Shipment_Delay = 2%.

Dropped these null values as they were very low compared to the overall data.

Variable Selection: From the EDA we have gained some understanding regarding the choice of variables.

Year – This was dropped since all the data is for the same year.

Month – This was dropped because there wasn't any significance variation for deliveries for these 6 months.

Carrier_Num – This was dropped due to insignificance.

Shipment_Delay – This was dropped as it will provide perfect classification and the resulting model will be overfitted.

Source, Destination – This was dropped as due to insignificance.

Planned_TimeofTravel – Dropped this variable due high correlation with Distance.

Distribution of Class variable Delivery_Status

0 - 2804359

1 – 718214

As we can see there is significant class imbalance in the data set. We have more number of timely deliveries compared to deliveries that were delayed.

Feature Engineering:

Actual_Planned – Created this variable which will capture weather Actual_Shipment_Time was less than or greater than Planned_Shipment_Time

Type – Created this variable which will capture weather the deliveries were made on the same day or not.

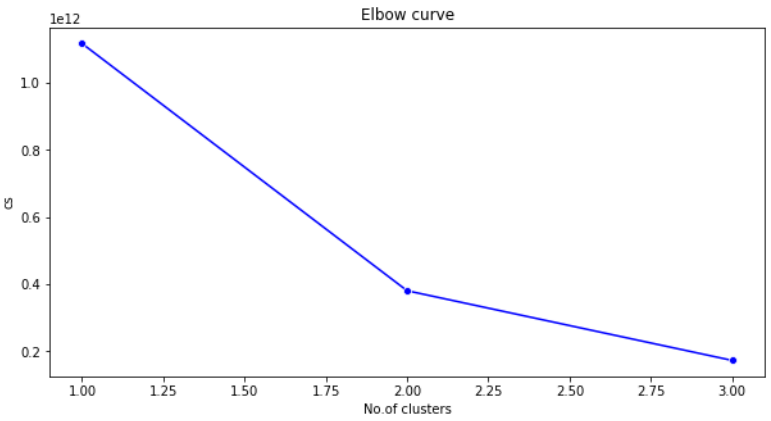
Carrier_Name – Binned this variables as High, Medium and low based on the total volume of deliveries sent.

Hour – Created this variable to find the shipment hour when the package was supposed to be shipped.

Remaing all variables were kept as it is in the dataframe.

Created dummy variable using 1 0 encoding for: **Actual_Planned, Type, Carrier_Name, and Hour**

Feature Selection using Unsupervised Learning: We have used K-means clustering to find homogenious groups in the dataset.



As seen from the elbow curve, the optimal number of custers is 2.

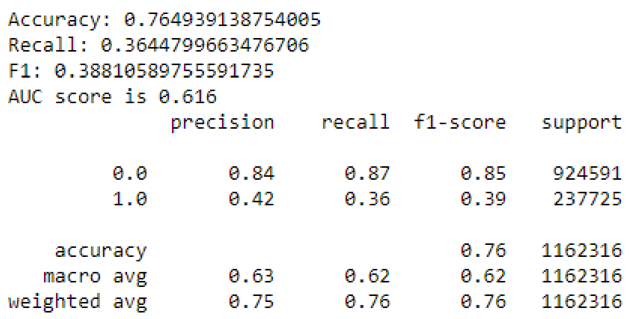
Ran K-means algorithm will K=2 on the dataset.

The resukting output was put back in the dataframe as new variable **'Clusters'**

Modelling

<u>Technique used</u>: Decesion Tree

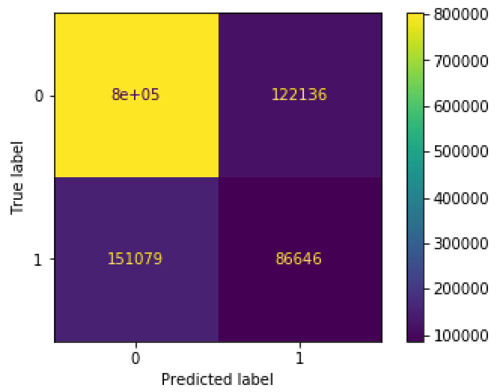
Iteration 1: Ran the decesion tree classifier on the entire dataset which was prepared. The classifier ran with default parameters to reach a fully grown tree.



```
Accuracy: 0.764939138754005
Recall: 0.3644799663476706
F1: 0.38810589755591735
AUC score is 0.616
              precision    recall  f1-score   support

         0.0       0.84      0.87      0.85    924591
         1.0       0.42      0.36      0.39    237725

    accuracy                           0.76   1162316
   macro avg       0.63      0.62      0.62   1162316
weighted avg       0.75      0.76      0.76   1162316
```



<u>Validation Outputs</u>

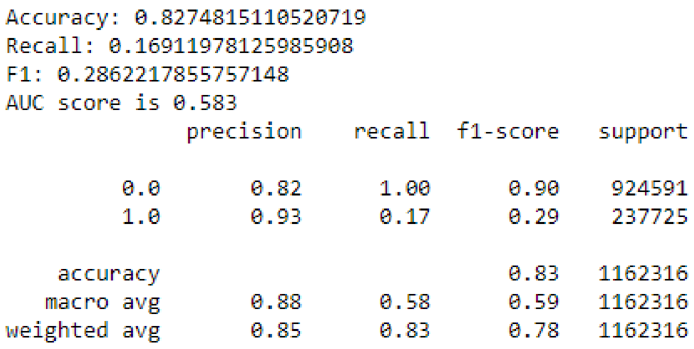
As we can see from the model outputs that the accuracy is very low. The model is able to successfully classify only 76% of the testing observations.

The recall and F1 score is satisfactory as the data has a massive class imbalance.

The model is able to predit only 36% of True negatives.

Iteration 2: Using grid search cross validation technique to for tunning the hyperparameters of the decesion tree classifier.

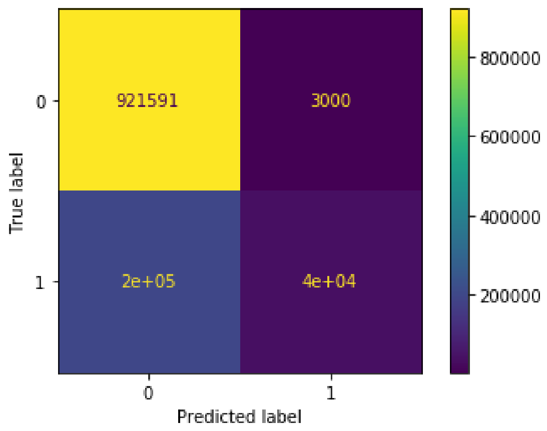
Best Parameters - {'criterion': 'gini', 'max_depth': 15, 'min_samples_leaf': 6, 'min_samples_split': 6, 'splitter': 'best'}



```
Accuracy: 0.8274815110520719
Recall: 0.16911978125985908
F1: 0.2862217855757148
AUC score is 0.583
              precision    recall  f1-score   support

         0.0       0.82      1.00      0.90    924591
         1.0       0.93      0.17      0.29    237725

    accuracy                           0.83   1162316
   macro avg       0.88      0.58      0.59   1162316
weighted avg       0.85      0.83      0.78   1162316
```



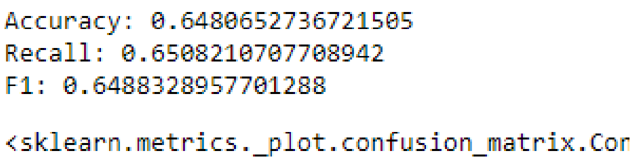
<u>Validation Outputs</u>

As we can see from the model outputs that the accuracy has significantly increased. The model is now able to successfully classify 83% of the testing observations.

The recall and F1 score is low as the data has a massive class imbalance.

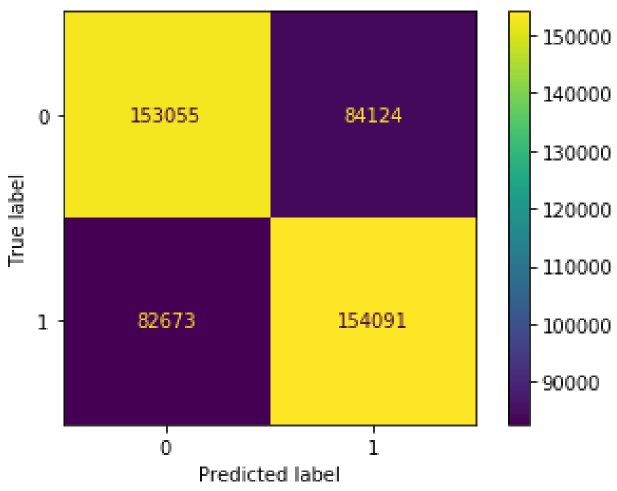
The model is able to predit only 17% of True negatives.

Let us try to treat class imbalace by using Oversampling and Undersampling:

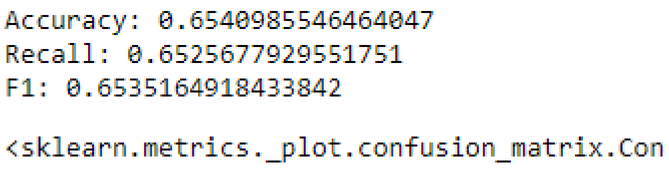


```
Accuracy: 0.6480652736721505
Recall: 0.6508210707708942
F1: 0.6488328957701288

<sklearn.metrics._plot.confusion_matrix.Cor
```

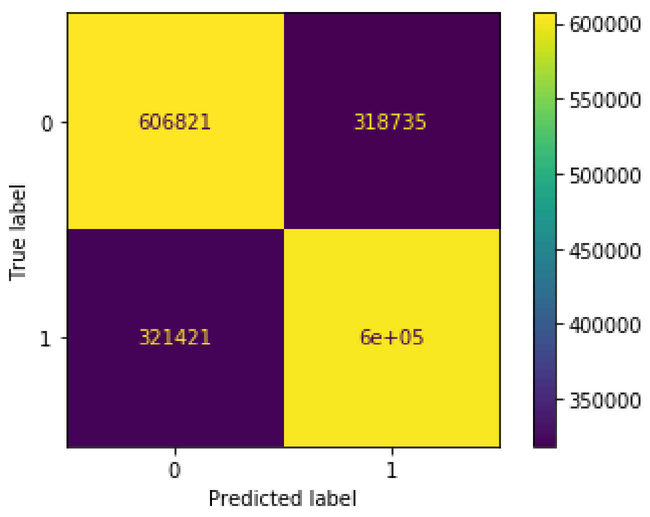


Undersampled



```
Accuracy: 0.6540985546464047
Recall: 0.6525677929551751
F1: 0.6535164918433842

<sklearn.metrics._plot.confusion_matrix.Con
```



Oversampled

As we can see from the ouputs both undersampled and oversampled results in a significant decrease in model accuracy.

However we are now able to predict more number of true positives but our type 2 error has significantly increased were the model is predicting that the deliveries will be on time but actually they were delayed.

Therefore we cannot use these sampling techniques on this data.

Note* - There are more sampling techniques to treat class imbalances but in the interest of time we have used only these two as of now.

Inference: From the above inferences we select model 2 as the optimum model. The model however is biased towards negatives which is result of class imbalance.