CLARUSWAY
WAY TO REINVENT YOURSELF

الأكاديمية السعودية الرقمية
SAUDI DIGITAL ACADEMY

FD

Fraud Detection Project

G4

Table of Contents

Introduction

Credit Card Fraud:

Credit card fraud is an inclusive term for fraud committed using a payment card, such as a credit card or debit card.

The purpose may be to obtain goods or services or to make payments to another account, which is controlled by a criminal.

Types of Credit Card Fraud

Application fraud
When someone opens credit accounts in your name

Account takeover
When someone hijacks your account to access funds

Skimming
When someone copies your credit card info on a skimmer

Lost or stolen cards
When someone takes your card to make purchases

Deepchecks

Deepchecks provides comprehensive support for your testing requirements, from examining data integrity and assessing distributions to testing and validating your machine learning models and data. It also enables you to do so with minimal effort.

Dataset Description

- The dataset contains transactions made by credit cards in September 2013 by European cardholders.

- This dataset presents transactions that occurred in two days, where it has **492 frauds** out of **284,807** transactions.

- The dataset is **highly unbalanced**, the positive class (frauds) account for 0.17% of all transactions.

Dataset Description

- Attributes of the dataset in detail, we have 31 variables in our datasets only 3 feature is known which is :

- **Time**: contains the seconds elapsed between each transaction and the first transaction in the dataset
- **Amount**: the transaction Amount, can be used for example dependent cost sensitive learning
- **Class**: The target variable and it takes value 1 in case of fraud and 0 otherwise

Data Type (All data type are numerical):

- **Unnamed Features** : [' 'V1 - V28 ‘] , 28 Features
- **Named Features** : ['Time', 'Amount’, 'Class’] , 3 Features

Data Cleaning

Duplicates :

- Number of duplicated values: 1081

Missing values:

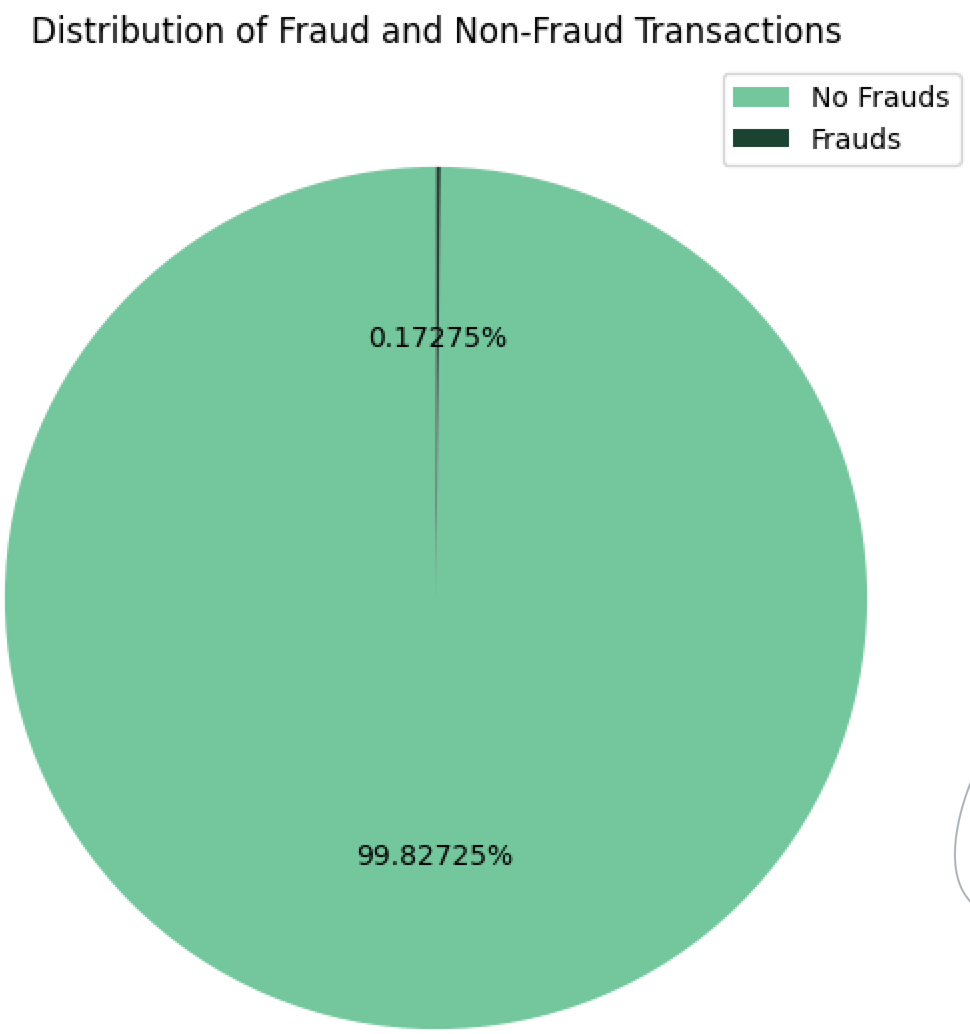
- No missing values

Outliers:

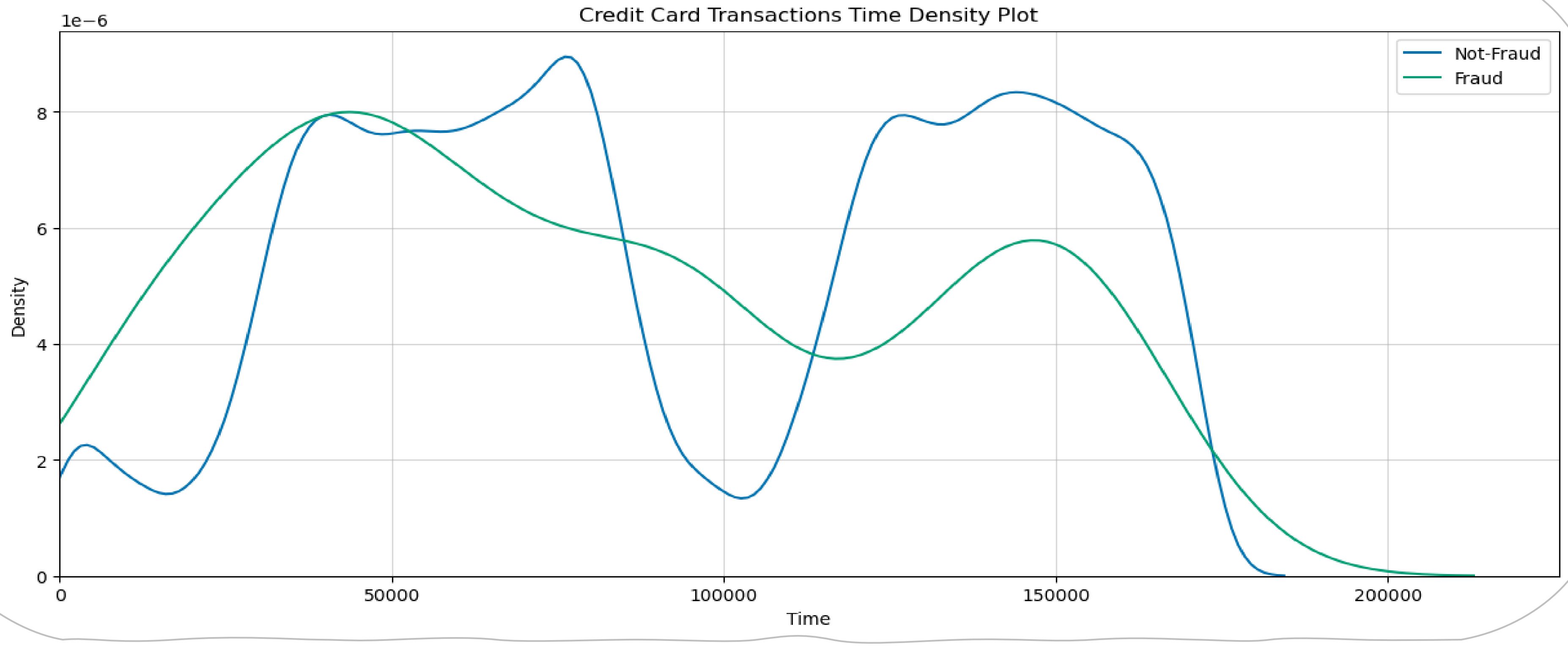
- Number of outliers: 137788
    - Most of the class 1 data are outlier (Fraud)

Explore Data Analysis

In an unbalanced dataset, we have 99.83% legal transactions and only 0.17% fraud transactions

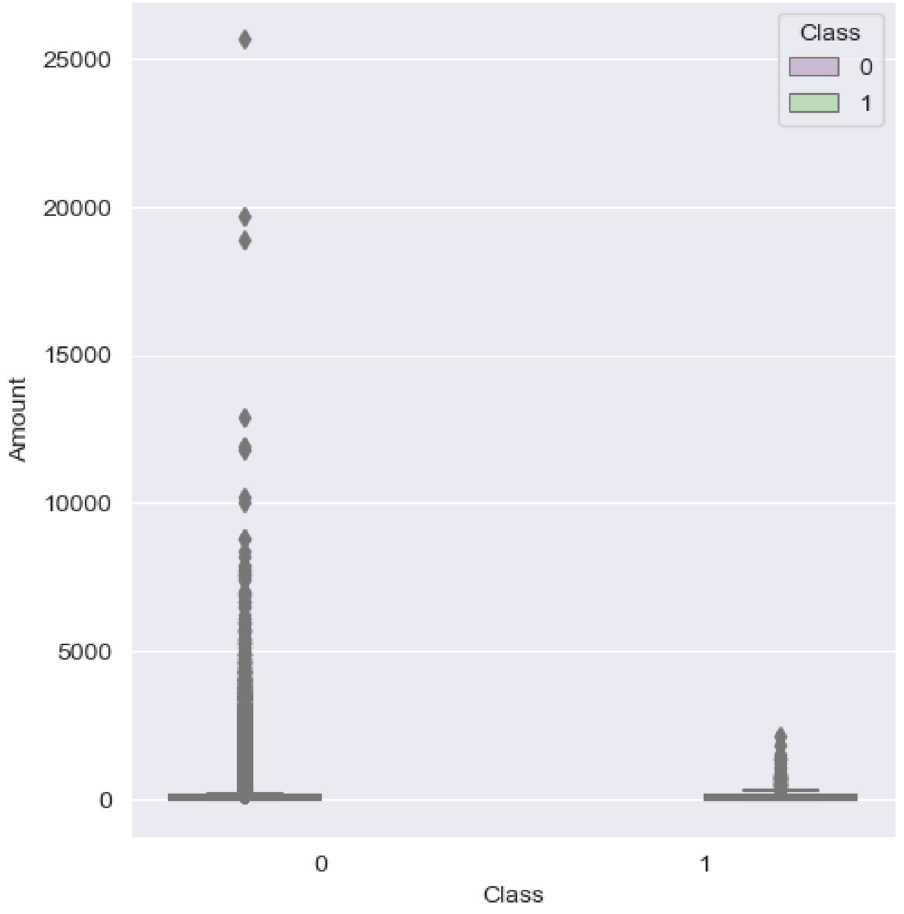


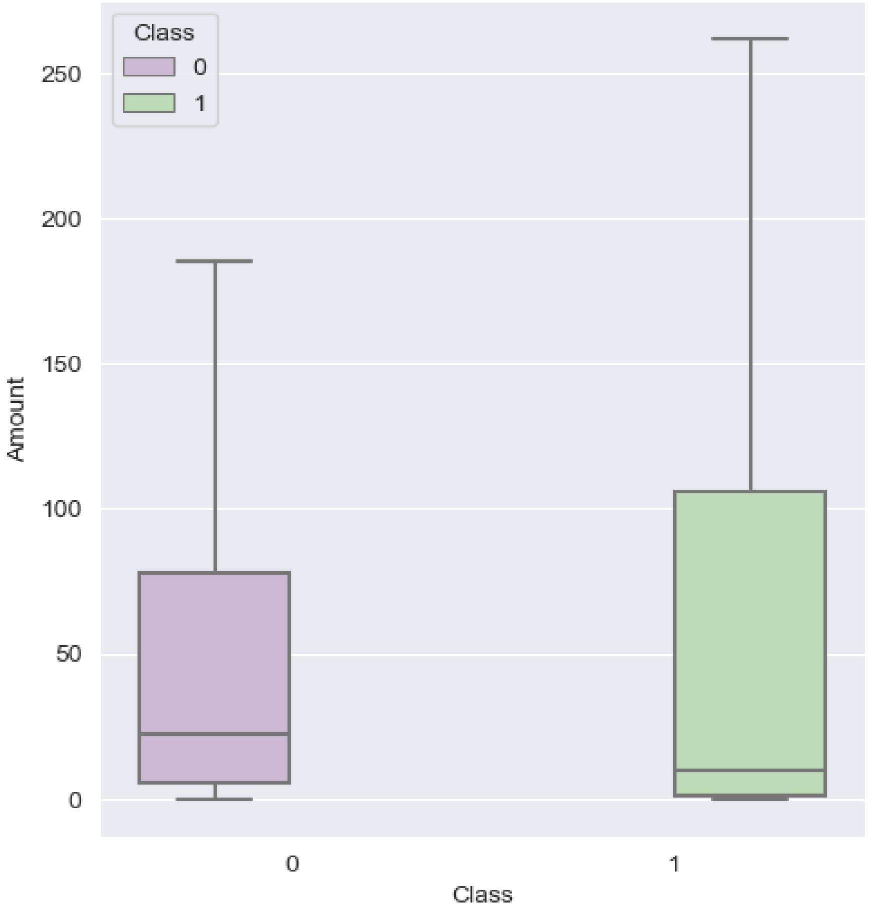
Explore Data Analysis



Fraud transactions have a distribution more even than non fraud transactions

Explore Data Analysis

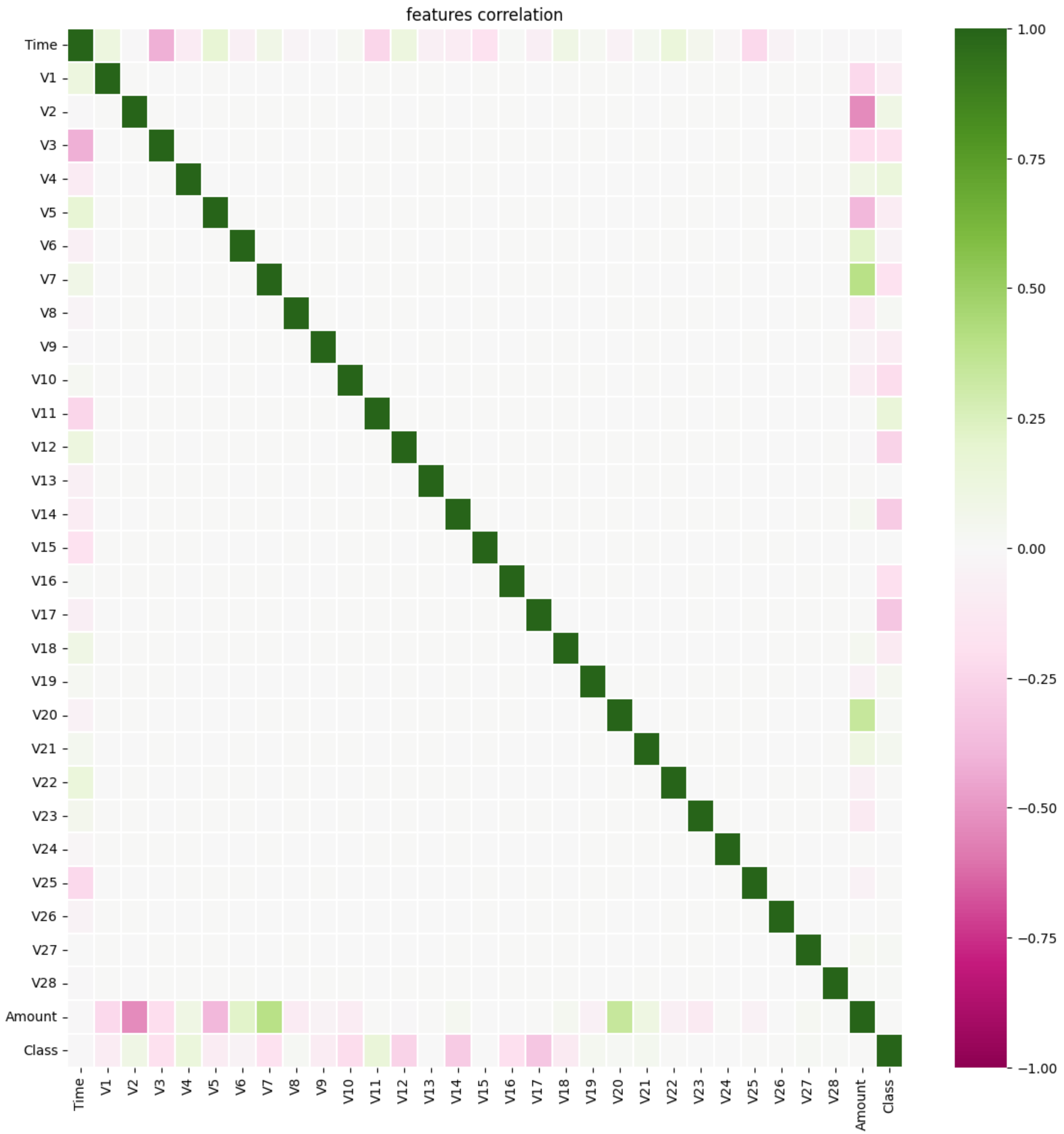




Compared to class 1, the most outlier from class 0, which is legal transactions

Explore Data Analysis

- V7 and V20 have a high positive correlation with the amount
- And V2 and amount have a negative correlation.
- V11 and V4 have a positive correlation with the class.
- And V12, V14, and V17 have a negative correlation with the class.



Predictive Models Building

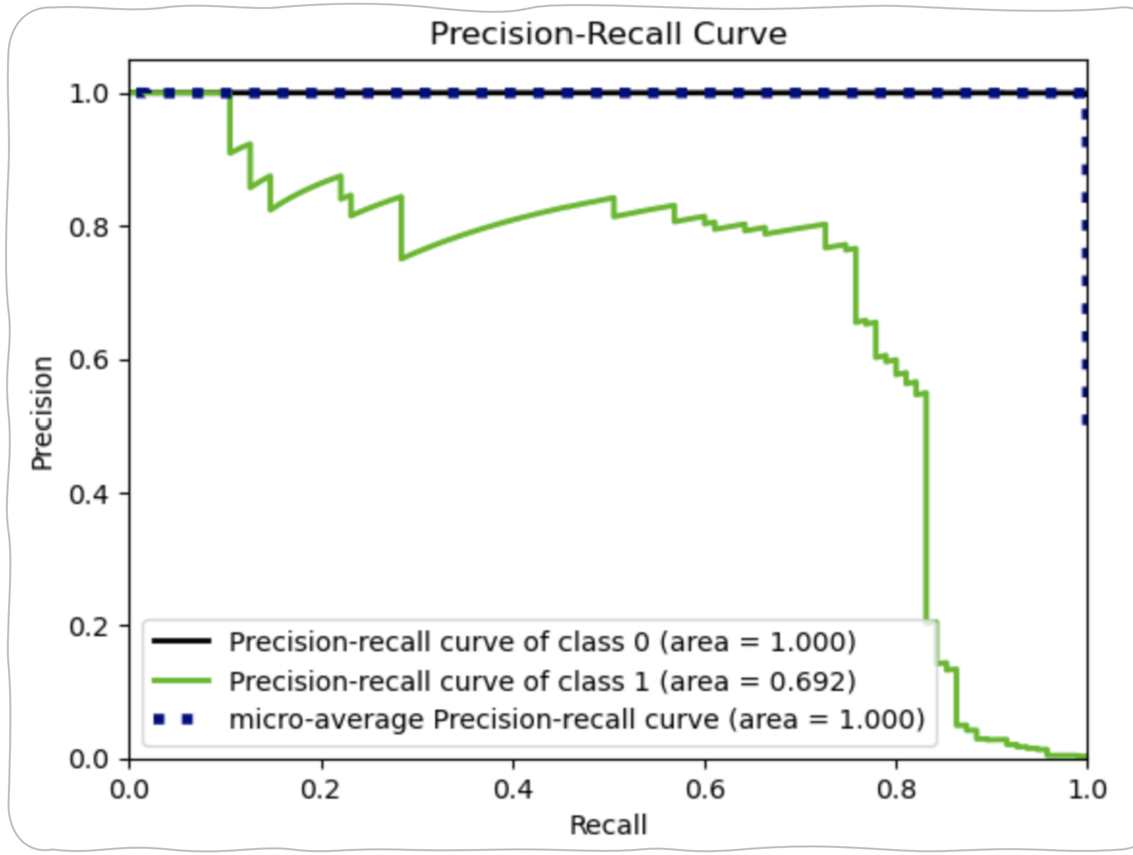
Preprocessing

- We made the Test size = 20% and Train size = 80%
- We Apply Scaling on the dataset with type Standard Scaler
- We decide the Random state = 101

Predictive Models Building

Machin Learning Classification :

Unbalanced data with Best Result for **Logistic Regression** model

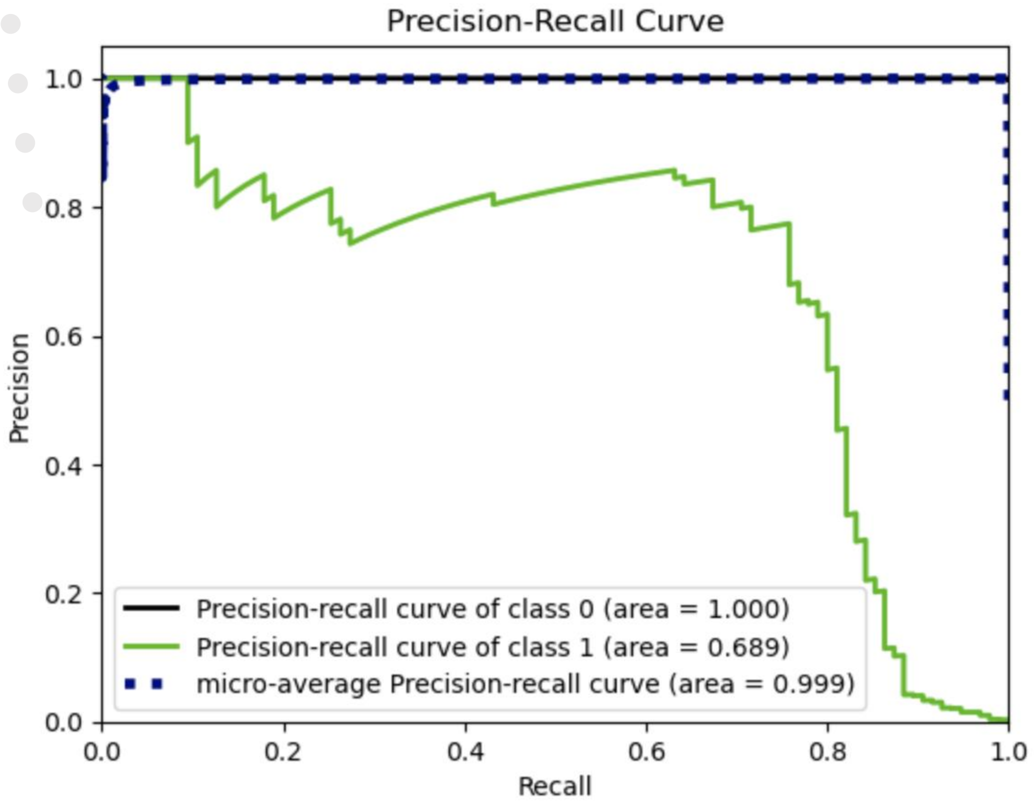



```
Test_Set
[[56640    11]
 [   46    49]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56651
           1       0.82      0.52      0.63        95

    accuracy                           1.00     56746
   macro avg       0.91      0.76      0.82     56746
weighted avg       1.00      1.00      1.00     56746


Train_Set
[[226571     31]
 [   143    235]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    226602
           1       0.88      0.62      0.73       378

    accuracy                           1.00    226980
   macro avg       0.94      0.81      0.86    226980
weighted avg       1.00      1.00      1.00    226980
```

Predictive Models Building

Machin Learning Classification :

balanced data with Best Result for **Logistic Regression** model

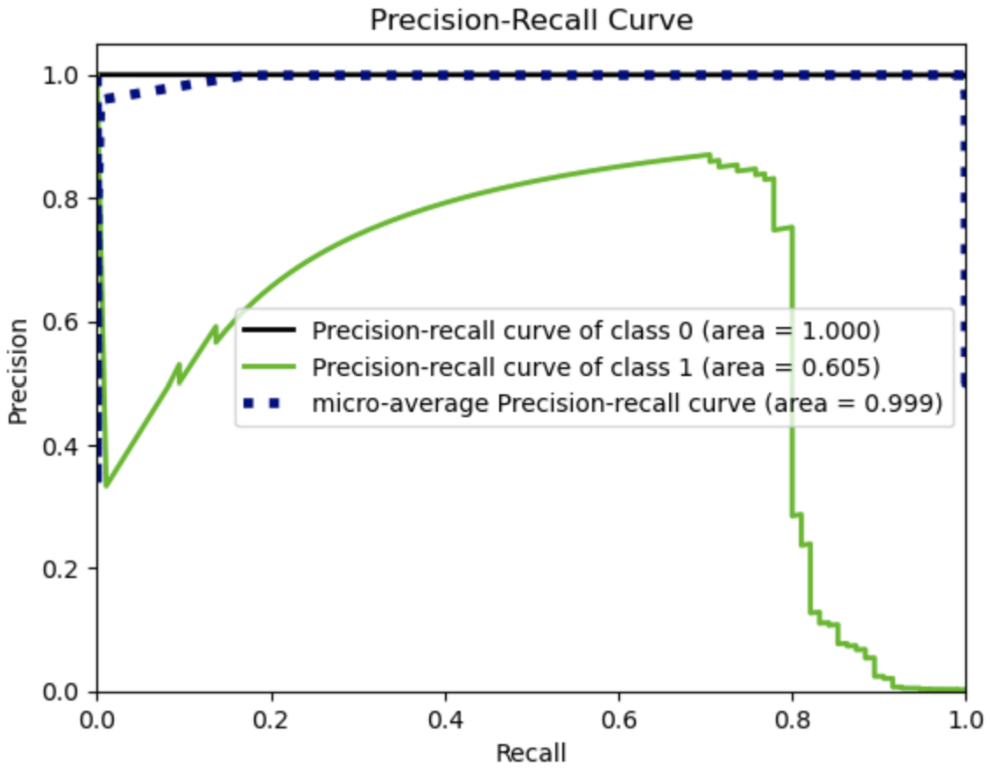



```
Test_Set
[[56597     54]
 [   19     76]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56651
           1       0.58      0.80      0.68        95

    accuracy                           1.00     56746
   macro avg       0.79      0.90      0.84     56746
weighted avg       1.00      1.00      1.00     56746


Train_Set
[[226414    188]
 [    60    318]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    226602
           1       0.63      0.84      0.72       378

    accuracy                           1.00    226980
   macro avg       0.81      0.92      0.86    226980
weighted avg       1.00      1.00      1.00    226980
```

Predictive Models Building

Machin Learning Classification :

balanced data with Best Results for **Random Forest** model

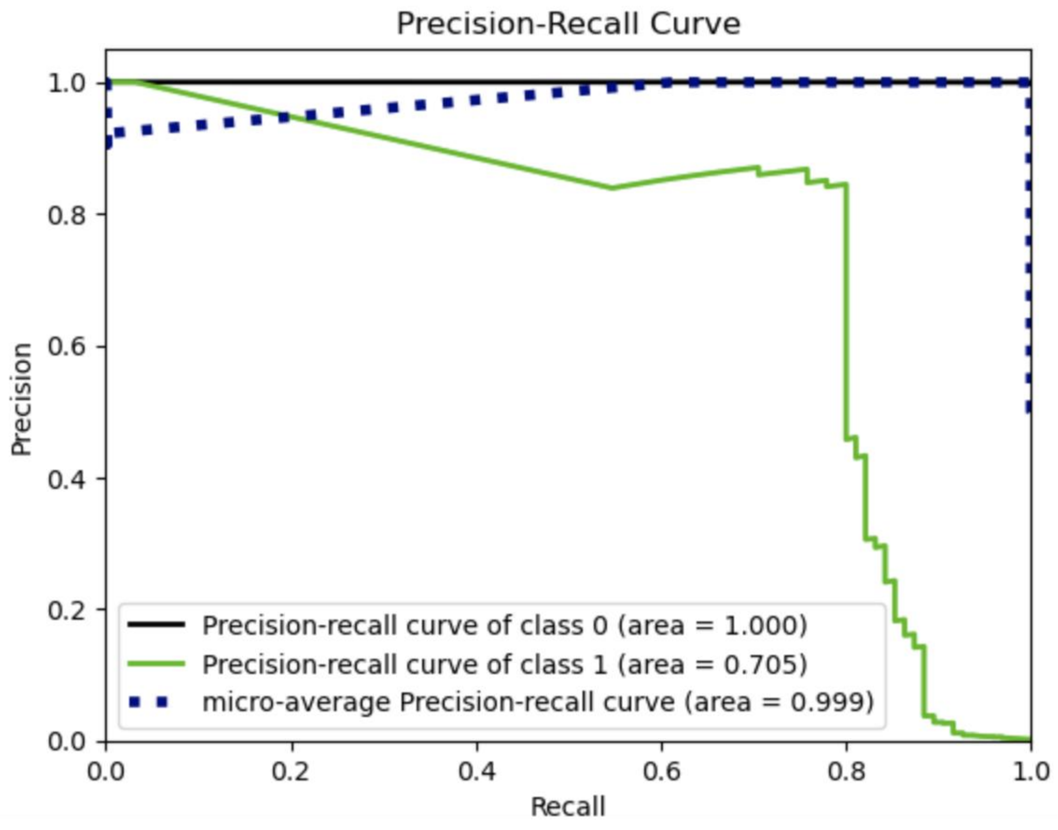



```
Test_Set
[[56623     28]
 [   19     76]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56651
           1       0.73      0.80      0.76        95

    accuracy                           1.00     56746
   macro avg       0.87      0.90      0.88     56746
weighted avg       1.00      1.00      1.00     56746

Train_Set
[[226462    140]
 [    60    318]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    226602
           1       0.69      0.84      0.76       378

    accuracy                           1.00    226980
   macro avg       0.85      0.92      0.88    226980
weighted avg       1.00      1.00      1.00    226980
```

Predictive Models Building

Machin Learning Classification :

Best Results from **XGBoost** Model




```
Test_Set
[[56632    19]
 [   19    76]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56651
           1       0.80      0.80      0.80        95

    accuracy                           1.00     56746
   macro avg       0.90      0.90      0.90     56746
weighted avg       1.00      1.00      1.00     56746


Train_Set
[[226529     73]
 [    56    322]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    226602
           1       0.82      0.85      0.83       378

    accuracy                           1.00    226980
   macro avg       0.91      0.93      0.92    226980
weighted avg       1.00      1.00      1.00    226980
```

Predictive Models Building

Deep Learning Classification :

with class weight

Layers used:

- **First layer** = 128 with Relu
- **Second layer** = 64 with Relu
- **Third layer** = 32 with Relu
- **Output layer** = 1 with sigmoid

Adam optimizer is used

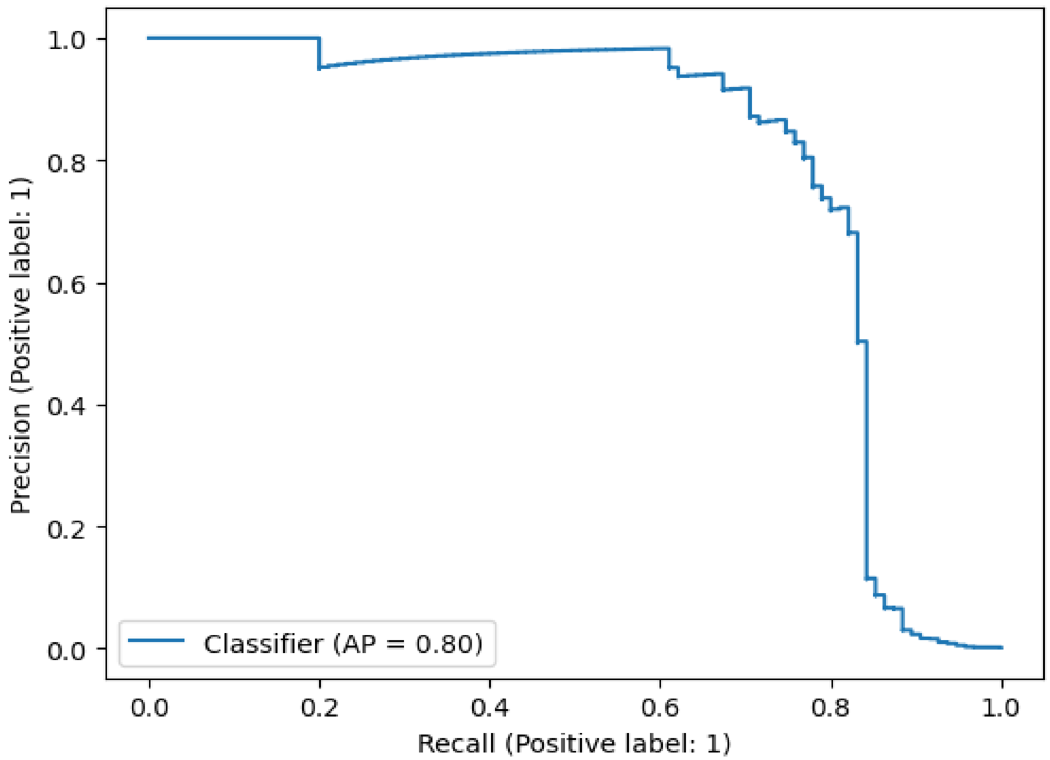
Loss binary cross entropy since it is a binary classification,

Early stopping with monitor on **Max val_recall**
Batch size = 8192 {0: 1, 1: 5}
We add class **weight = "balanced"** ,

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               3968

 dense_1 (Dense)             (None, 64)                8256

 dense_2 (Dense)             (None, 32)                2080

 dense_3 (Dense)             (None, 1)                 33

=================================================================
Total params: 14,337
Trainable params: 14,337
Non-trainable params: 0
_________________________________________________________________
```

Predictive Models Building

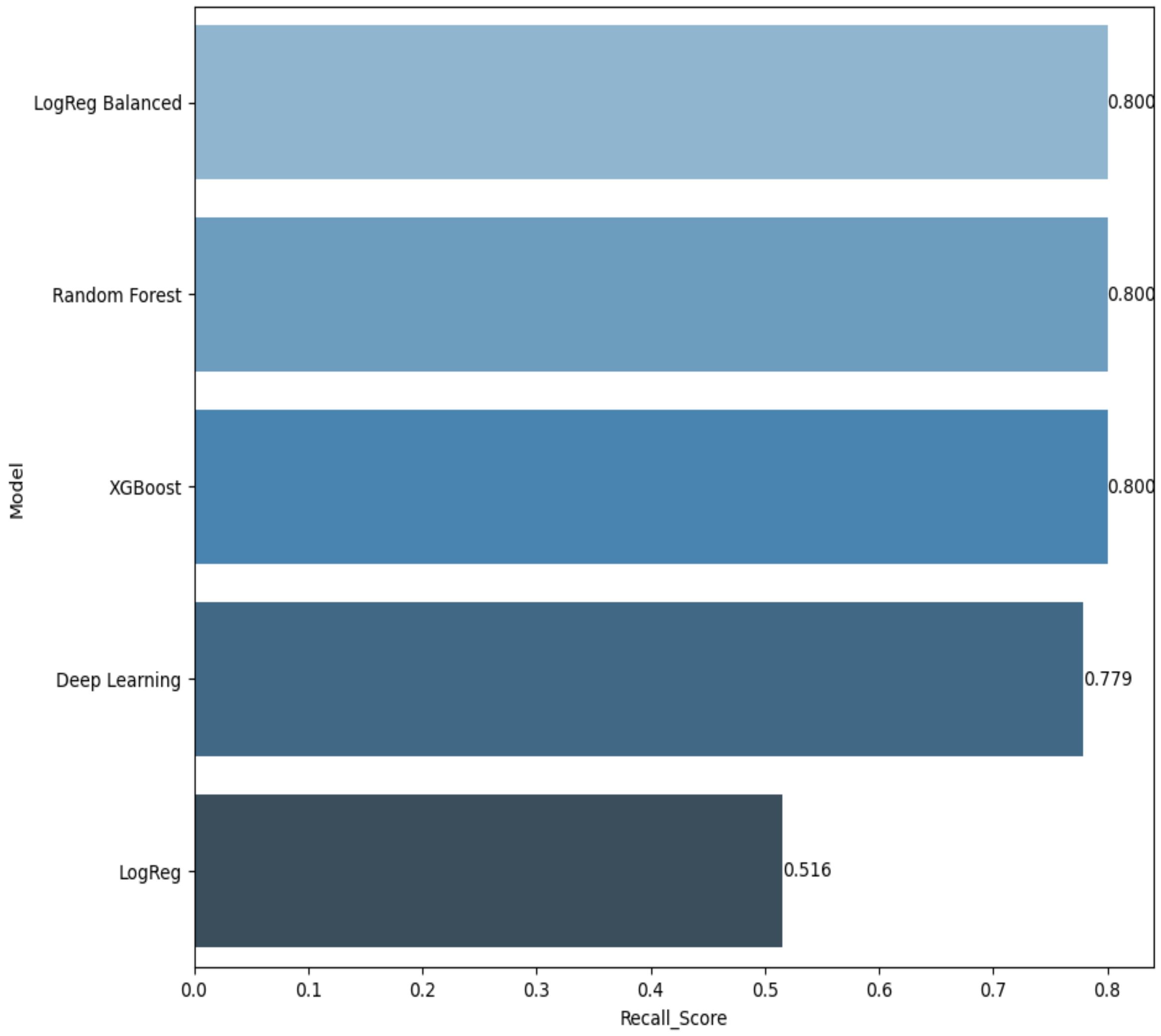
Evolution of Deep Learning




```
Test_Set
[[56629    22]
 [   21    74]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56651
           1       0.77      0.78      0.77        95

    accuracy                           1.00     56746
   macro avg       0.89      0.89      0.89     56746
weighted avg       1.00      1.00      1.00     56746


Train_Set
[[181228     54]
 [    32    270]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    181282
           1       0.83      0.89      0.86       302

    accuracy                           1.00    181584
   macro avg       0.92      0.95      0.93    181584
weighted avg       1.00      1.00      1.00    181584
```

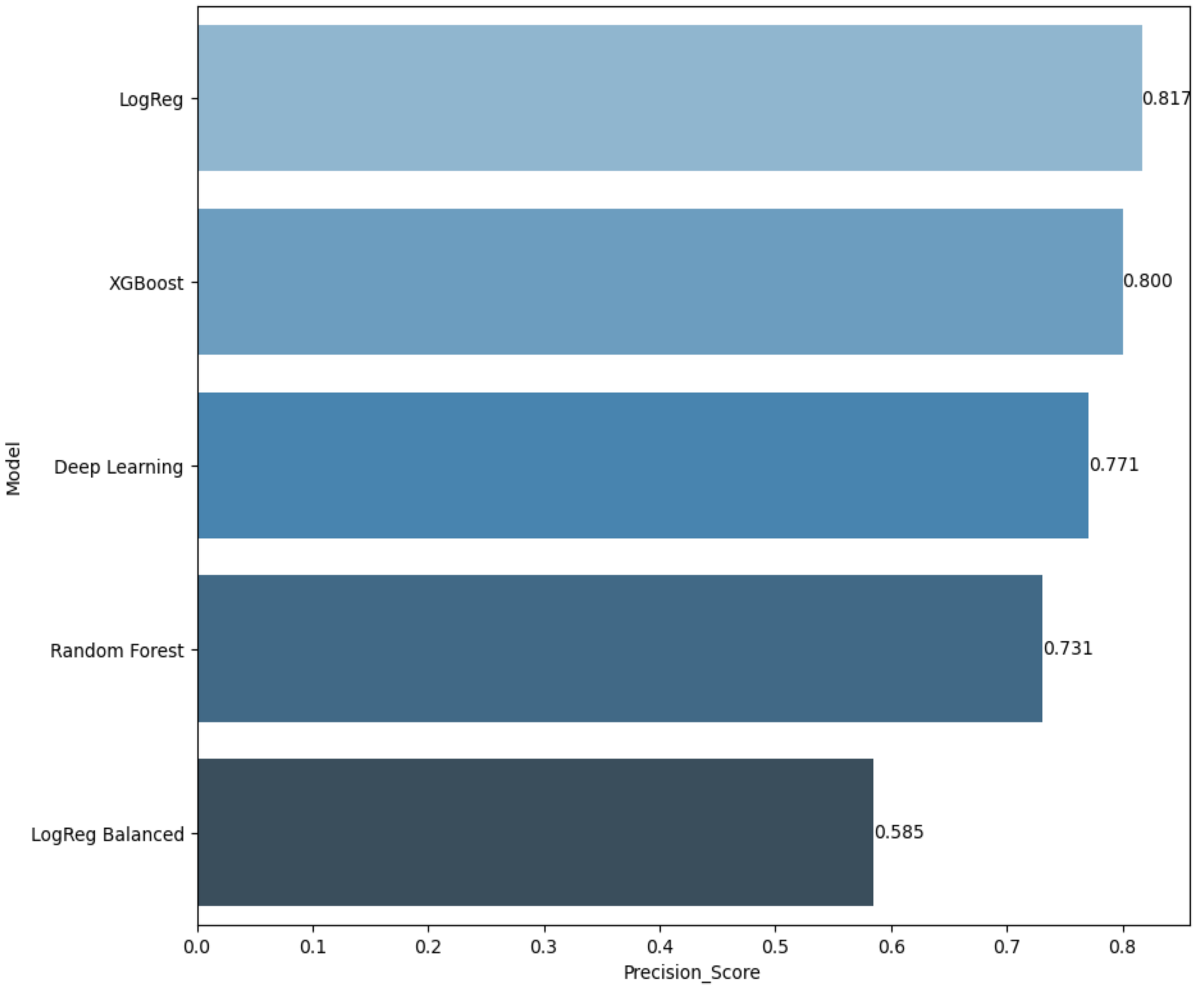
Evaluating Models Performance

We will be comparing model performances according :Recall Score



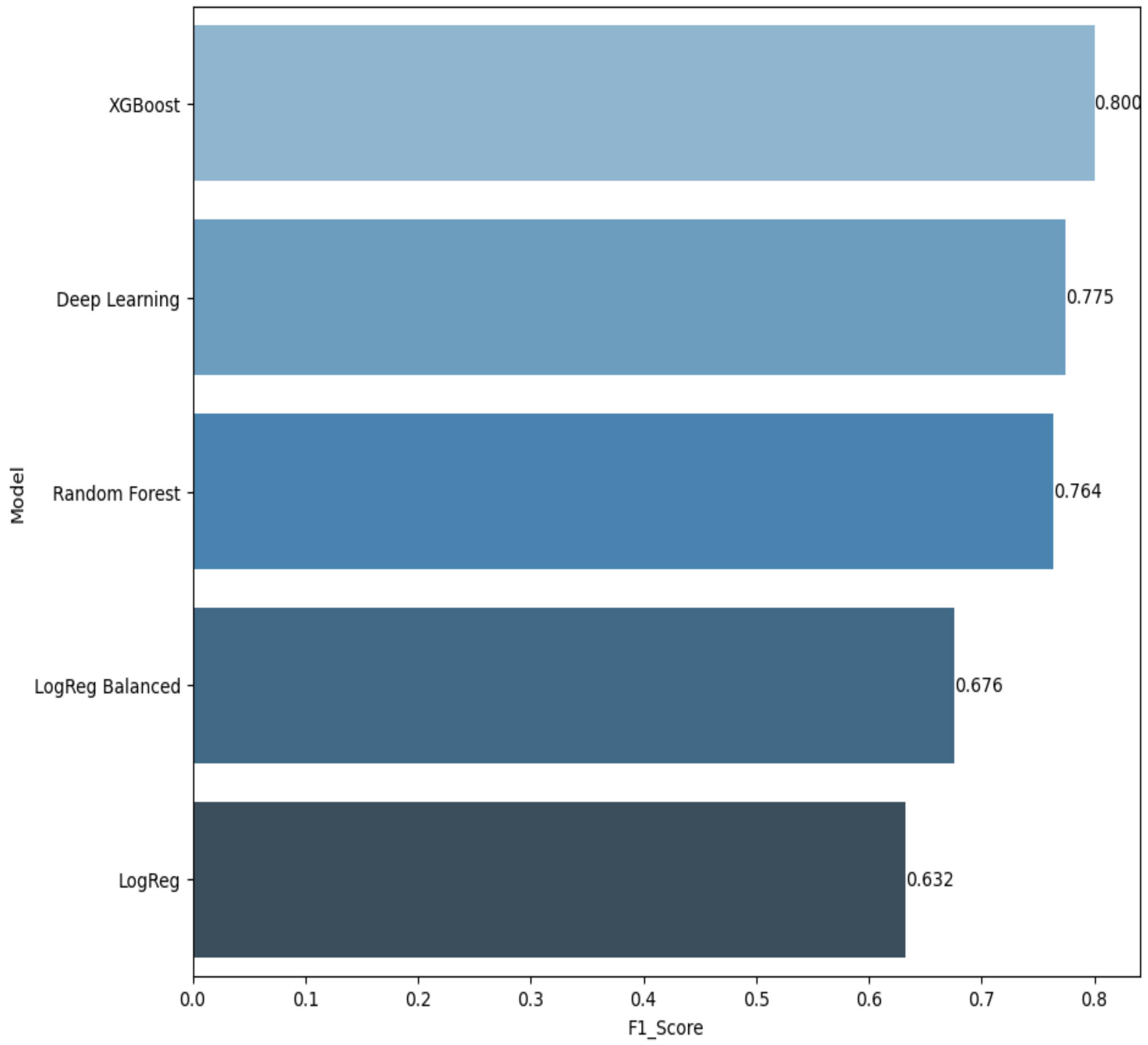
Evaluating Models Performance

We will be comparing model performances according :Precision score



Evaluating Models Performance

We will be comparing model performances according :F1 score



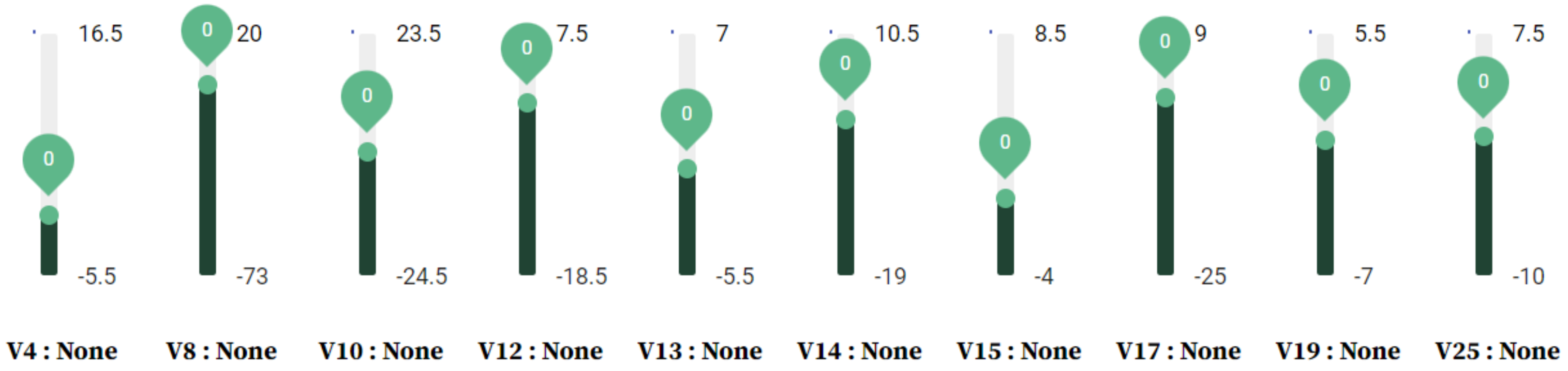
```
[[283148    105]
 [    73    400]]
--------------------------------------------------------
              precision    recall  f1-score   support

           0       1.00      1.00      1.00    283253
           1       0.79      0.85      0.82       473

    accuracy                           1.00    283726
   macro avg       0.90      0.92      0.91    283726
weighted avg       1.00      1.00      1.00    283726
```

Evaluating Models Performance

Based on evaluating graphs we chooses XGBoost

Model Deployment

Credit Card Fraud Detection



Transaction Amount

0 − +

prediction

Conclusion

To sum up, after analysis of the dataset :

- The most important feature that will give better performance is: ['V14', 'V17', 'V8', 'V10', 'V12', 'V4', 'V15', 'Amount', 'V19', 'V25', 'V13’]
- Dealing with a highly unbalanced dataset will effectively model performance, so we apply balanced techniques to our dataset by using class weight.
- The best model for our dataset is **XGBoost**; it gives the best result in f1 score.
- Logistic regression gives the worst result in recall, and when applying the balance technique to logistic regression, it gives the worst result in precision .

Any Questions?
Thank You!