Sanchez Sara

Individual Report

For this project, we choose to develop a scoring problem. I found that building a scoring model will be helpful for us because in the real world is something valuable and sometimes a little difficult to develop. In my case, I have worked in a risk area from a financial institution, and in here where I discovered the importance of this type of tool, and to dig deeper, I proposed to my group to make a scoring model.

After finding the data, I began to analyze it to understand it better and thus generate efficient models. Therefore, I built a code in which we used certain parts, and my colleagues' optimized others. Also, use the scorecardpy to make graphs like the weight of evidence vs. the default.

2. Description of your individual work. Provide some background information on the development of the algorithm and include necessary equations and figures.

```
#using the feature selecction to have the most inportant variables
from sklearn.feature_selection import RFE
from sklearn.ensemble import RandomForestClassifier

estimator = RandomForestClassifier(n_estimators=50,random_state=123,n_jobs=-
1)

selector = RFE(estimator, n_features_to_select=25,step=0.05, verbose=1)
selector = selector.fit(X_train, y_train)

list_rf=selector.support_

X_train.iloc[:,list_rf].info()

from sklearn.linear_model import LogisticRegression

# creating the classifier object
lr =
LogisticRegression(penalty='none',fit_intercept=True,random_state=123,class_w
eight='balanced')

# calculate parameters for logistic Model on Training
lr.fit(X_train.iloc[:,list_rf], y_train)

df_training_pred_lr = pd.DataFrame({'actual':y_train,'predicted':
lr.predict(X_train.iloc[:,list_rf]),

'Non_Target':lr.predict_proba(X_train.iloc[:,list_rf])[:,0],

'Target':lr.predict_proba(X_train.iloc[:,list_rf])[:,1],
                                })

df_testing_pred_lr = pd.DataFrame({'actual':y_test,'predicted':
lr.predict(X_test.iloc[:,list_rf]),
'Non_Target':lr.predict_proba(X_test.iloc[:,list_rf])[:,0],

'Target':lr.predict_proba(X_test.iloc[:,list_rf])[:,1],
                                })

from sklearn import metrics
import matplotlib.pyplot as plt

accuracy_training=metrics.accuracy_score(df_training_pred_lr.actual,df_traini
ng_pred_lr.predicted)
accuracy_testing=metrics.accuracy_score(df_testing_pred_lr.actual,df_testing_
pred_lr.predicted)

f1_score_training=metrics.f1_score(df_training_pred_lr.actual,df_training_pre
d_lr.predicted)
f1_score_testing=metrics.f1_score(df_testing_pred_lr.actual,df_testing_pred_l
r.predicted)

auc_score_training = metrics.roc_auc_score(df_training_pred_lr.actual,
df_training_pred_lr.predicted)
auc_score_testing = metrics.roc_auc_score(df_testing_pred_lr.actual,
df_testing_pred_lr.predicted)
```

Hyperparameters for each model:

```
lr =
LogisticRegression(penalty='none',fit_intercept=True,random_state=123,class_w
eight='balanced')
rf =
RandomForestClassifier(n_estimators=300,max_depth=10,min_samples_leaf=0.01,cl
ass_weight='balanced',random_state=123)
dt =
DecisionTreeClassifier(max_depth=5,min_samples_leaf=0.01,criterion='gini',cla
ss_weight='balanced',random_state=123)
```

3. Describe the portion of the work that you did on the project in detail. It can be figures, codes, explanation, pre-processing, training, etc.

In general, I built some parts of the preprocessing, as it is possible to observe from the code attached. I also created figures like the weight of evidence, other bar plots and density plots, and the correlation matrix. For the training, I built several versions of Decision Tree, Random Forest, and Logistic Regression that also were tested.

4. Results. Describe the results of your experiments, using figures and tables wherever possible. Include all results (including all figures and tables) in the main body of the report, not in appendices. Provide an explanation of each figure and table that you include. Your discussions in this section will be the most important part of the report.

Unfortunately, in one of the first versions, I found the overfitting problem, which means that the training indicators (like accuracy) were almost 100% and for the testing was like 29%, this means that it was not stable. So first, to solve this problem, I applied some codes to normalize the data,

and using hyperparameters inside of each model can overcome this problem a little. Also, I built buckets for some variables, but considering the default rate, each bucket has a different default rate so that the variable can discriminate between profiles. The variables selected were the ones that have a lot of categories like supplier_id, branch_id, manufacturer_id, and state_id.
For the final report, I produce all the introduction, the EDA analysis, the analysis of the outputs for each model, a little explanation related to the confusion matrix, an essential concept inside this project, and the ROC curve.

5. Summary and conclusions. Summarize the results you obtained, explain what you have learned, and suggest improvements that could be made in the future.

From my part of the code, I use almost a 70% for the training part from internet. For the EDA and data exploratory I used a 50% also from internet.

6. Summary and conclusions. Summarize the results you obtained, explain what you have learned, and suggest improvements that could be made in the future.

In the future, I will work in a more organized way, making use of Github; it was my first time using this, and still being a little confusing for me, but practice makes perfect. Also, I learned a lot of python and concepts that are useful when building a model. In my case, it was the first time I made a model in python, It was challenging, but being creative, everything is possible.

As follow this are the main conclusions from my analysis:

INTRODUCTION

Financial institutions incur significant losses due to the default of Vehicle Loans. This situation has led to the constricting of vehicle loan underwriting and increased vehicle loan rejection rates. These institutions also raise the need for a better credit risk scoring model. By doing this, the institutions are trying to accurately predict the Probability of loanee defaulting on a vehicle loan on the due date. In this sense, the credit scores are significant as a tool that can decide which clients can take or not a credit, considering their unique characteristics. The scoring is also a helpful tool for the clients because this can avoid accepting a loan that will not be able to be paid in the future and, consequently, prevent having problems with financial institutions.

Furthermore, the scoring tool is helpful not only for cash loans but also for mortgages and car loans, so this is the main reason that motivated us to choose this topic because it is helpful for many kinds of businesses.

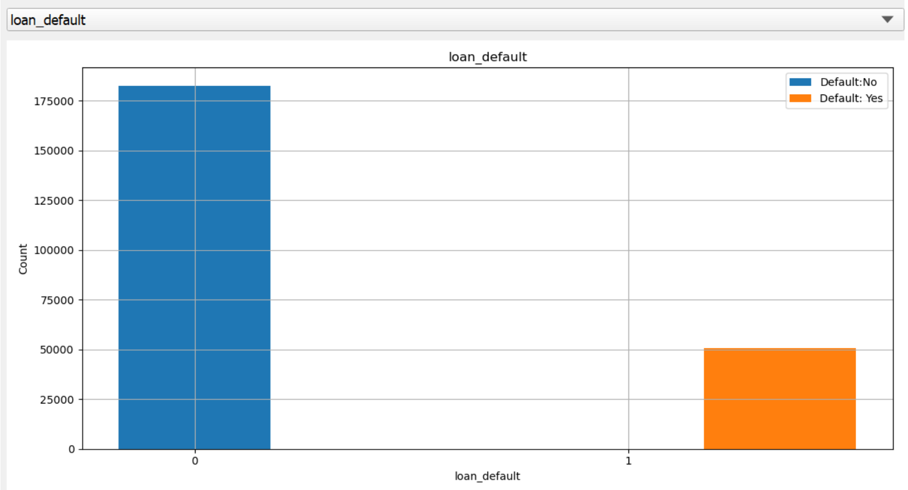
To develop this scoring project, we will analyze an L&T car loan company database from Kaggle. Then we are doing some preprocessing and cleaning of the data to apply the classification models like Decision tree, Random Forest, logistic, and Gradient Boosting. Finally, we will show the results by using the Pyqt5.

The following report will be organized: first, we will describe the data set, then the methodology to be used, and finally, the results and main conclusions.

EXPLORATORY DATA ANALYSIS (EDA)

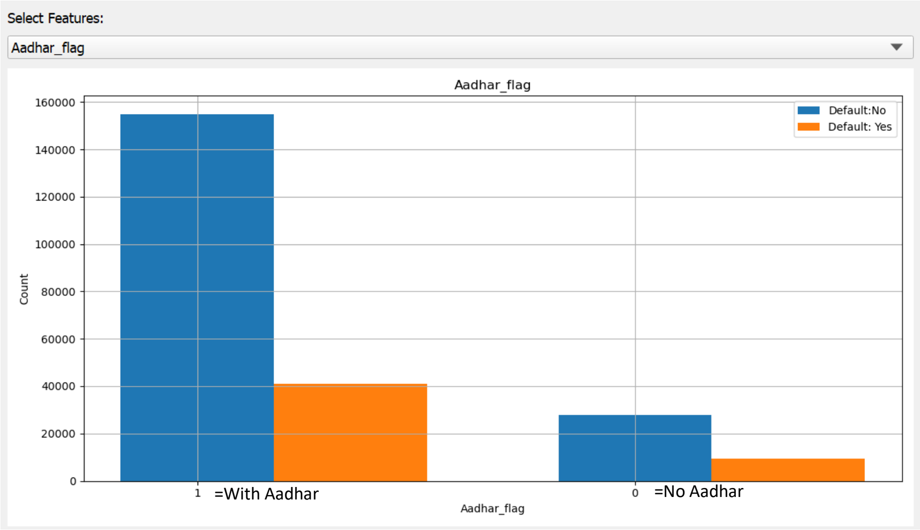
From the car loan database, we will develop the Exploratory data analysis (EDA). This kind of analysis permits a better understanding of the data and builds a better model. The target variable is the “loan default,” this gives us information related to the number of persons that defaulted, and for this database, the default ratio is around 27.7%.

Figure 1 – Loan_deafult



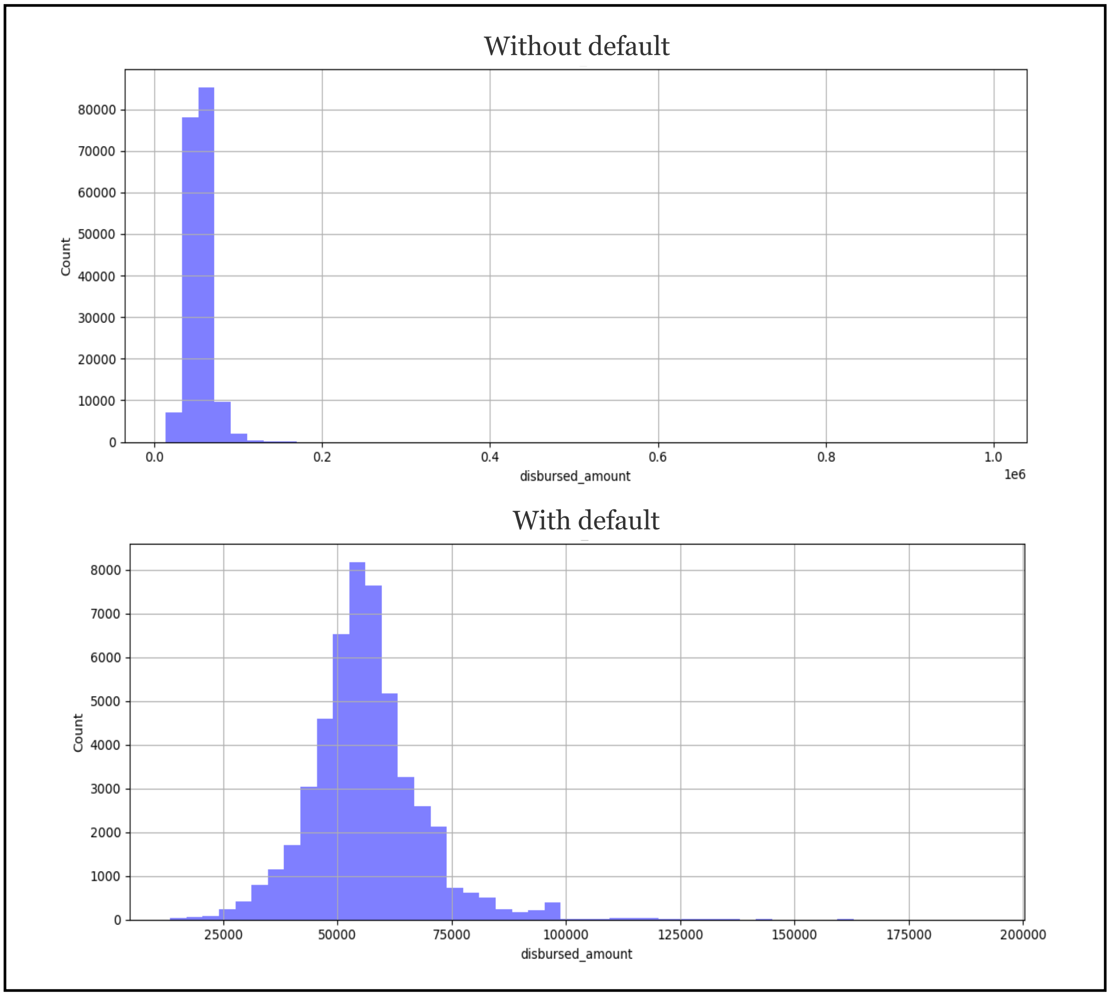
Another essential variable is the Aadhar flag. As this is a database from India, it is crucial to highlight that Aadhar is given to all the citizens from this country. A person who doesn’t have a citizen status will not have an Aadhar. The graph shows that people who have an Aadhar are more likely to be on default than others.

Figure 2 – Aadhar flag



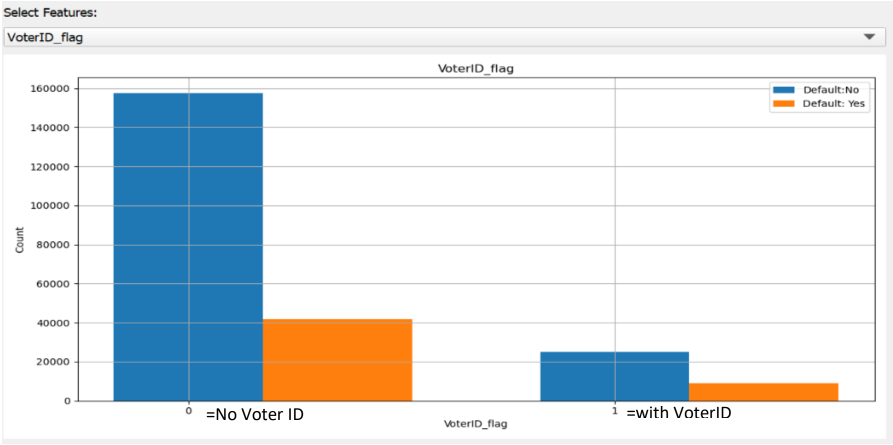
The disbursed amount is the financial institution's quantity to the borrower. Figure number three shows a significant difference in the amount between those without and those with default. Again, there is a concentration around shorter amounts. This is different from the former who have a normal distribution for the latter.

Figure 3 – Disbursed amount



From figure number four, we can analyze the voter ID flag. This variable is essential because this makes differences between profiles with and without default. As is possible to observe, most people from the database do not have a Voter ID, and, this is the group with the biggest default. Therefore, it is possible to conclude that people who cannot have the biggest default could be an indicator or informality.

Figure 4 – Voter ID

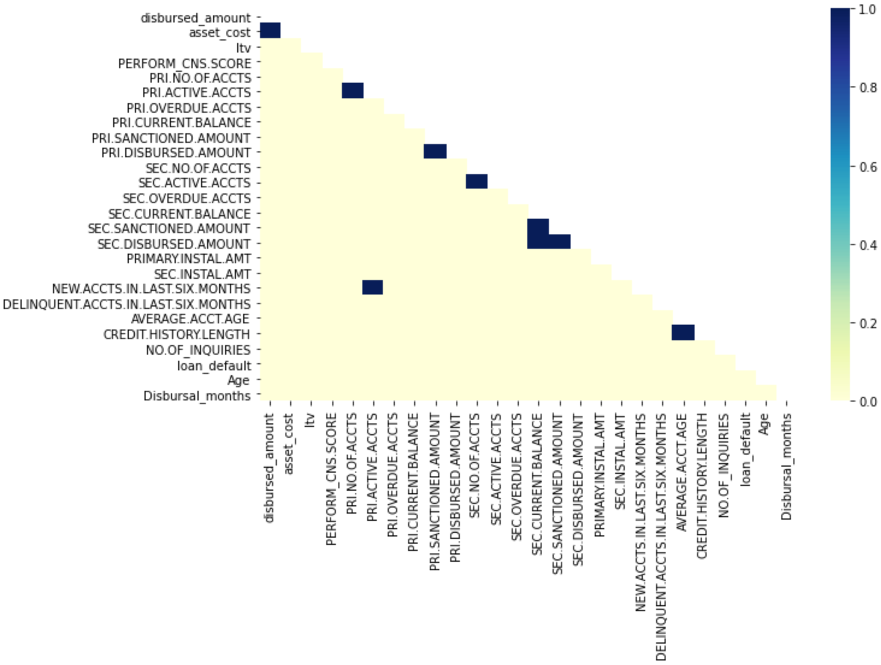


Select Features:

VoterID_flag

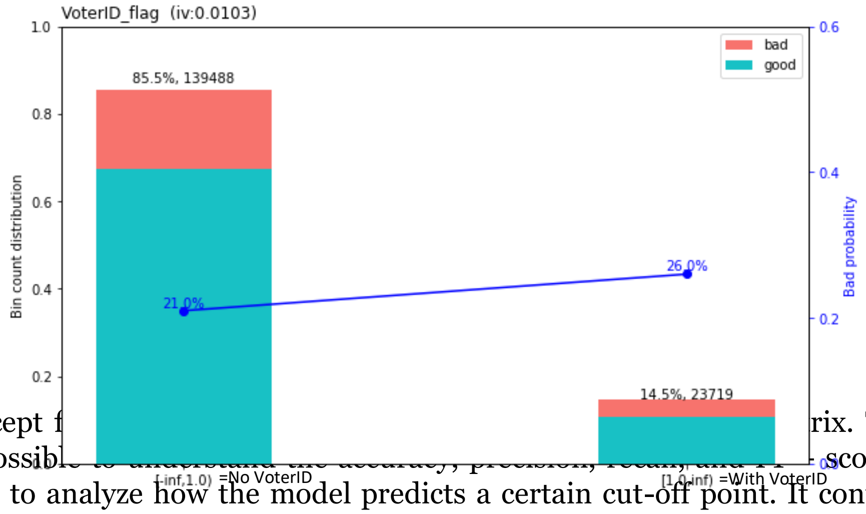
From this plot, we can observe a positive and strong correlation. Therefore, we choose to select those points with a correlation more significant than 70%. The variables that fulfill this requirement are asset cost, disbursed amount, sanctioned amount, credit history length, and average account age.

Figure 5 – Correlation Plot



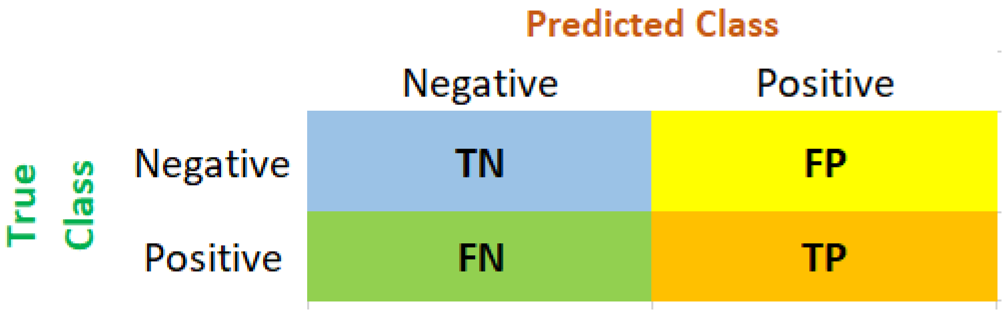
The weight of the evidence plot is well known among the risk areas from financial institutions. Therefore, we can interpret this plot as 21 over 100 will default from those with a Voter ID. On the other hand, from those who do not have a Voter ID, 26 over 100 will have a default. This is a crucial plot given that confirms the importance of the variable as a predictor because those who contribute more to the model are the ones that can differentiate the profiles better.

Figure 6 – Weight of evidence from Voter ID



A critical concept f [illegible] rix. Thanks to this matrix, it is possible to understand the accuracy, precision, recall, and F1 score values. This matrix is used to analyze how the model predicts a certain cut-off point. It contains two rows corresponding to the subjects that the model has alerted as non-default and those it has warned as default. The two columns represent the actual value that said subject or operation takes over time. Then the model's prediction results are compared.

Figure 7 –Confusion Matrix concept



True Class	Predicted Class: Negative	Predicted Class: Positive
Negative	TN	FP
Positive	FN	TP

Therefore, the interpretation of each of the elements of the matrix is as follows:

- Cell TN: They are called true negatives and represent the number of operations that the model alerted as default, and they were default.

Cell FN: They are called false negatives and represent the number of operations the model alerted as default and did not default.

- Cell FP: They are called false positives and represent the number of operations that the model alerted as not default, and they were default.

- Cell TP: They are called true positives and represent the number of operations that the model alerted as no default, and they were no default

Giving this concept, the formulas for the indicators are:

$$Accuracy = \frac{TP + TN}{P + N}$$

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

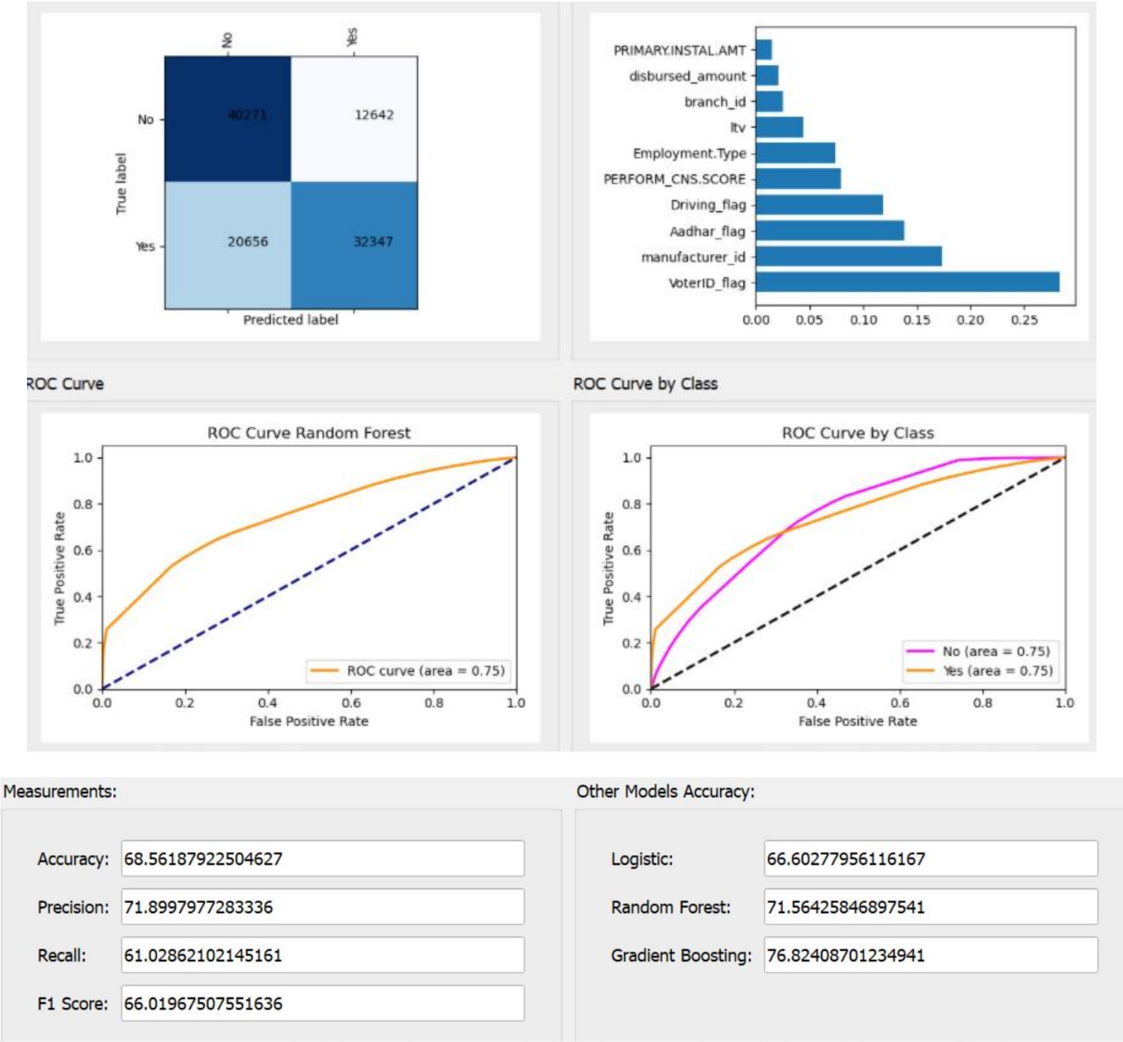
$$Specificity \ = \frac{TN}{TN + FP}$$

$$F1 - Score \ = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Another important concept is the ROC in which the sensitivity is presented as a function of false positives. If the ROC curve is further from the origin, it will be better.

Decision tree:

Figure 8 –Decision Tree (Outputs)



Measurements:

Accuracy:	68.56187922504627
Precision:	71.8997977283336
Recall:	61.02862102145161
F1 Score:	66.01967507551636

Other Models Accuracy:

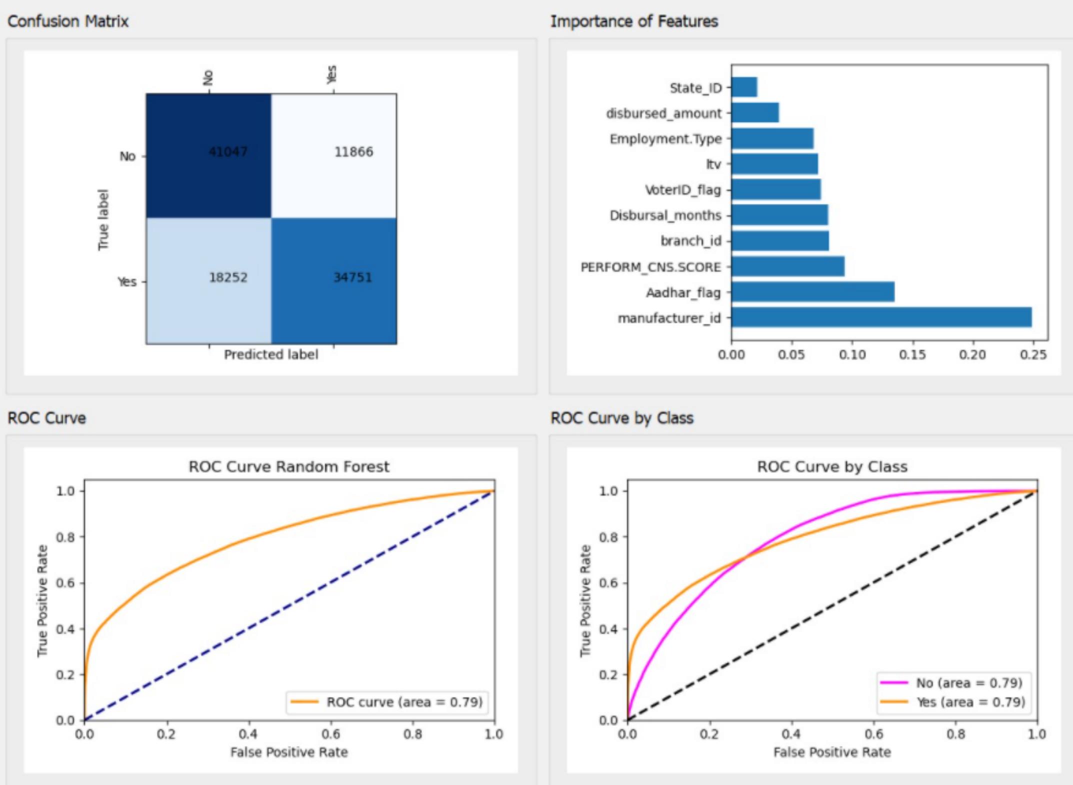
Logistic:	66.60277956116167
Random Forest:	71.56425846897541
Gradient Boosting:	76.82408701234941

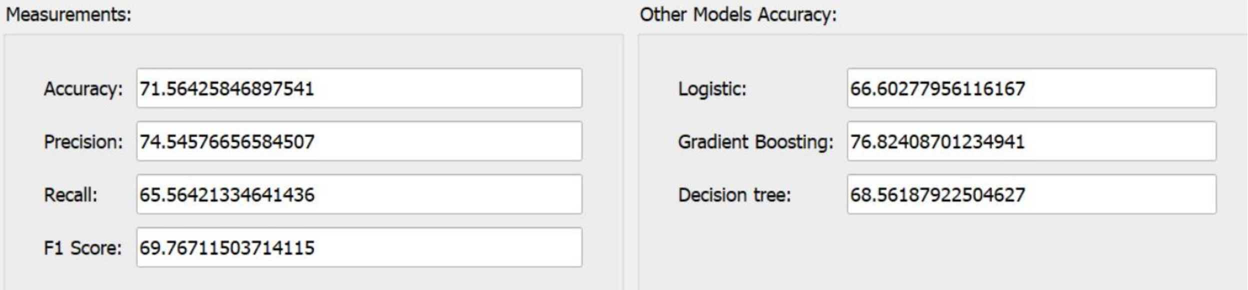
First, the decision tree has an accuracy of 68%, which means that 68 over 100 predictions the model is going to be right; second, the precision gives us 71%, which means that 71 over the 100 that the model predicted that they would default are correct. Third, the recall is 61%, which means that the model predicts 61 over all the default cases. Finally, the F1-score is 66%, which is the harmonic mean of precision and recall. The ROC curve has a regular performance for this model, given that it is not so close to the dotted line and its area is 0.75.

As part of our analysis, we also developed the tree that is attached to the presentation file. This tree gives us an essential variable, the Aadhar_flag, as it was found this is an important variable that discriminates between good and bad profiles.

Random Forest:

Figure 9 –Random Forest (Outputs)





Measurements:

Accuracy: 71.56425846897541

Precision: 74.54576656584507

Recall: 65.56421334641436

F1 Score: 69.76711503714115

Other Models Accuracy:

Logistic: 66.60277956116167

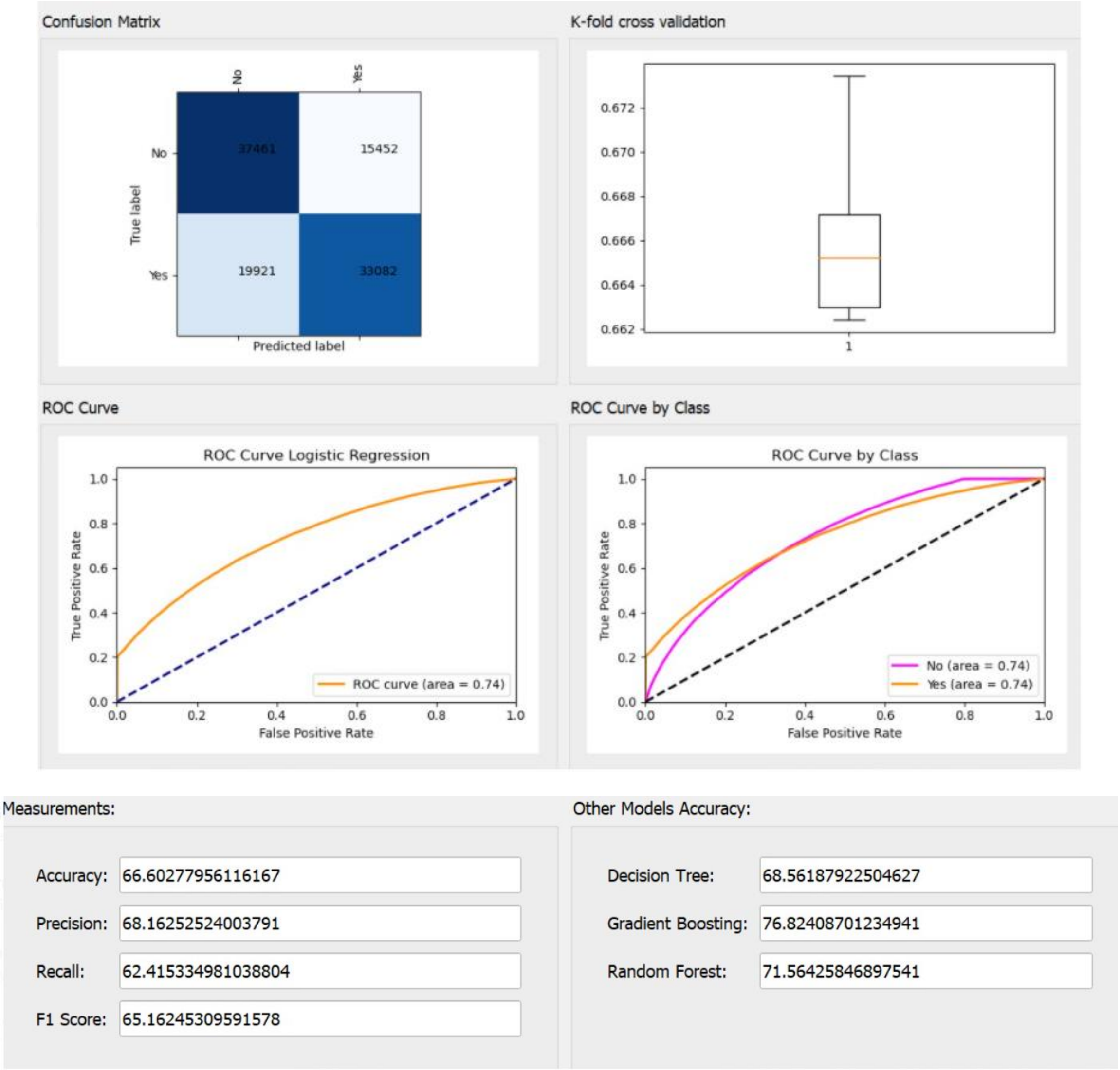
Gradient Boosting: 76.82408701234941

Decision tree: 68.56187922504627

First, the random forest has an accuracy of 71%, which means that 71 over 100 predictions the model is going to be right; second, the precision gives us 74%, which means that 74 over the 100 that the model predicted that they would default are correct. Third, the recall is 65%, which means that the model predicts 65 over all the default cases. Finally, the F1-score is 69.7%, which is the harmonic mean of precision and recall. The ROC curve has a better performance than in the decision tree, given that it is a little farther from the dotted line and its area is 0.79.

Logistic Regression:

Figure 10 –Logistic Regression (Outputs)



Measurements:

Accuracy: 66.60277956116167

Precision: 68.16252524003791

Recall: 62.415334981038804

F1 Score: 65.16245309591578

Other Models Accuracy:

Decision Tree: 68.56187922504627

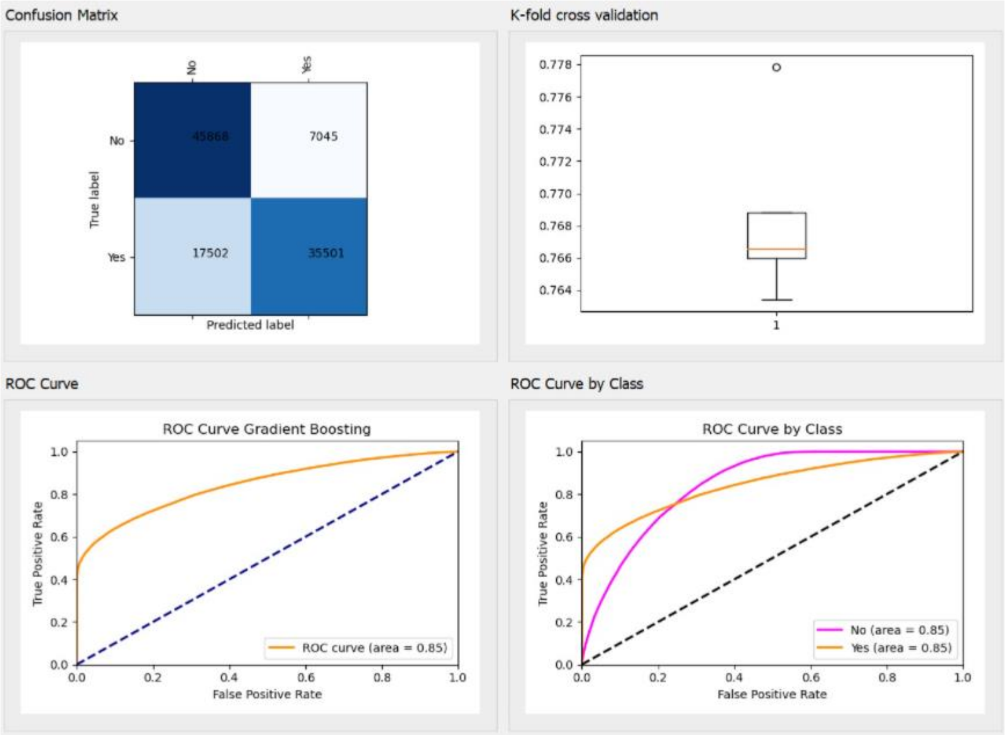
Gradient Boosting: 76.82408701234941

Random Forest: 71.56425846897541

First, the logistic regression has an accuracy of 66%, which means that 66 over 100 predictions the model is going to be right; second, the precision gives us 68%, which means that 68 over the 100 that the model predicted that they would default are correct. Third, the recall is 62%, which means that the model predicts 62 over all the default cases. Finally, the F1-score is 65%, which is the harmonic mean of precision and recall. The ROC curve does not have a better performance than in the decision tree and random forest, given that it is not so far from the dotted line, like the other models, and its area is 0.74.

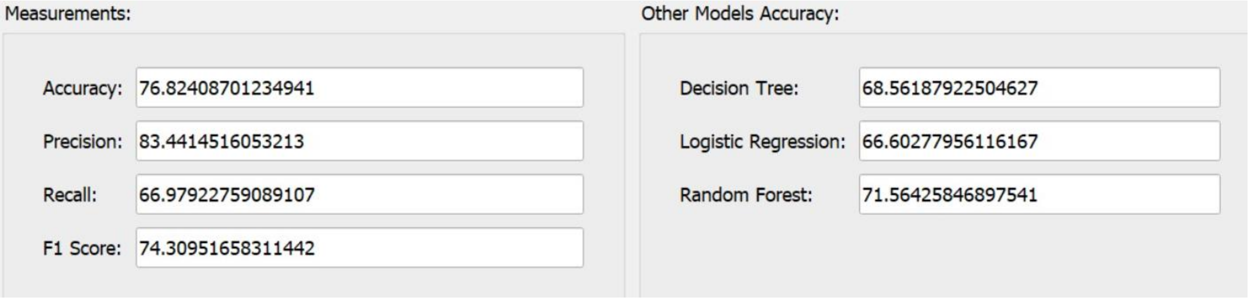
Gradient Boosting:

Figure 11 – Gradient Boosting (Outputs)



	No	Yes
No	45868	7045
Yes	17502	35501

True label / Predicted label



Measurements:

Accuracy: 76.82408701234941

Precision: 83.4414516053213

Recall: 66.97922759089107

F1 Score: 74.30951658311442

Other Models Accuracy:

Decision Tree: 68.56187922504627

Logistic Regression: 66.60277956116167

Random Forest: 71.56425846897541

First, the gradient boosting has an accuracy of 76%, which means that 76 over 100 predictions the model is going to be right; second, the precision gives us 83%, which means that 83 over the 100 that the model predicted that they would default are correct. Third, the recall is 66%, which means that the model predicts 66 over all the default cases. Finally, the F1-score is 74%, which is the harmonic mean of precision and recall. The ROC curve for this model is the best compared to the other three models, given that it is farther from the dotted line and its area is 0.85.