



MICRO-CREDIT DEFAULTER PROJECT

Submitted by:
Kritanjay Singh

ACKNOWLEDGMENT

In this project the dataset is given by the company and the project is done on the Jupyter Notebook in Anaconda.

INTRODUCTION

- Predictions for Loan transaction

Prediction in terms of a probability for each loan transaction, whether the customer will be paying back the loaned amount within 5 days of insurance of loan. With the help of this in real world the person who actually needed the loan can have the loan.

- Conceptual Background of the Domain Problem

A Microfinance Institution (MFI) is an organization that offers financial services to low income populations. Microfinance is widely accepted as a poverty-reduction tool, representing \$70 billion in outstanding loans and a global outreach of 200 million clients. They are collaborating with the Telecom Industry to provide micro-credit on mobile balances to be paid back in 5 days.

- Review of Literature

The Microfinance services (MFS) provided by MFI are Group Loans, Agricultural Loans, Individual Business Loans and so on. Telecom Industry are collaborating with a Microfinance Institution (MFI) to provide micro-credit on mobile balances to be paid back in 5 days. This would help to the low income populations of real world and useful for the especially the unbanked poor families living in remote areas with not much sources of income. Telecom Industry are a fixed wireless telecommunications network provider. They focusing on providing their services and products to low income families and poor customer. With the help of MFI they give micro-credit loan, and consumer is defaulter if he not paying back the loaned amount with the time duration of 5 days. For the loan amount of 5 (in

Indonesian Rupiah), payback amount should be 6 (in Indonesian Rupiah), while, for the loan amount of 10 (in Indonesian Rupiah), the payback amount should be 12 (in Indonesian Rupiah).

- **Motivation for the Problem Undertaken**

The sample data is provided by the client database of Telecom communication and the objective of this project is to improve the selection of customers for the credit. The client wants some predictions that could them in further investment and improvement in selection of customers so that they can give loan to the right person.

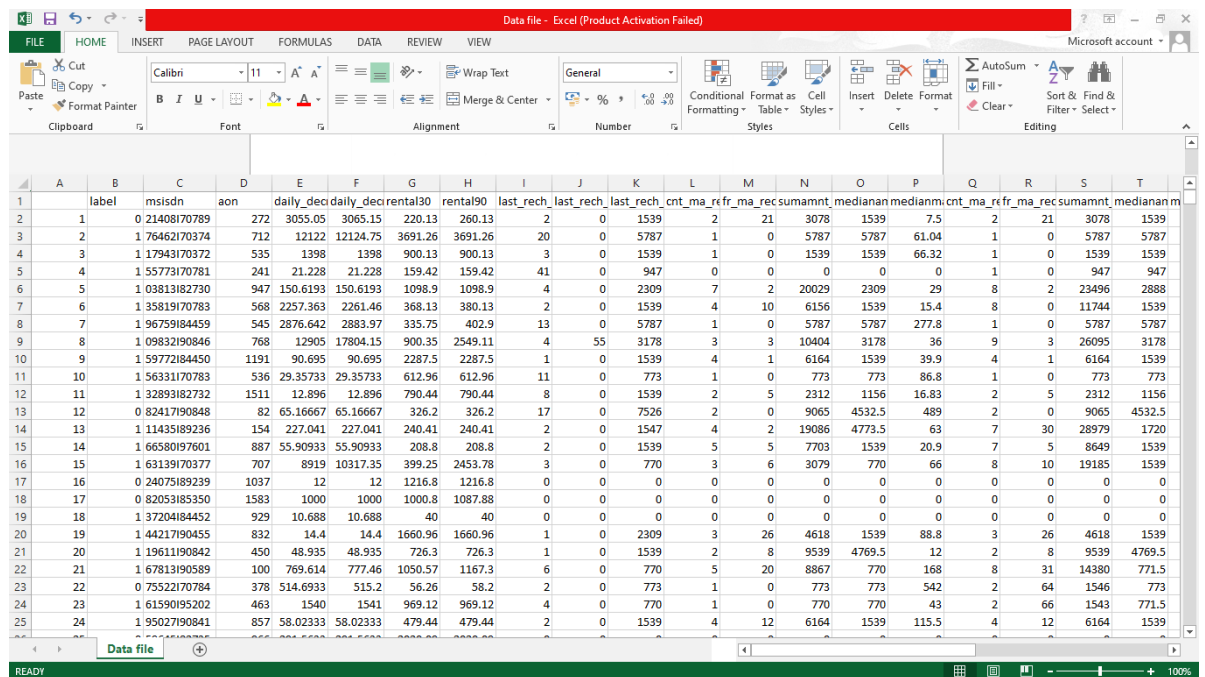
Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

In this project, By the describing the data we calculated mean , median and the standard deviation of the dataset which help to describe the variance in the data and also the outliers and here with correlation method we check the correlation between the features and the target.

- Data Sources and their formats

The sample data is provided is provided by the client database, and the data is in the form of csv i.e. in the excel sheet. Here is the dataset



| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
|----|----|-------|-------------|------|-----------|-----------|----------|----------|-----------|-----------|-----------|--------|-------|--------|---------|----------|----------|--------|-------|--------|
| 1 | | label | msidsn | aon | daily_dec | daily_dec | rental30 | rental90 | last_rech | last_rech | last_rech | cnt_ma | rr_fr | ma_rec | sumamnt | medianan | medianan | cnt_ma | rr_fr | ma_rec |
| 2 | 1 | 0 | 21408170789 | 272 | 3055.05 | 3065.15 | 220.13 | 260.13 | 2 | 0 | 1539 | 2 | 21 | 3078 | 1539 | 7.5 | 2 | 21 | 3078 | 1539 |
| 3 | 2 | 1 | 76462170374 | 712 | 12122 | 12124.75 | 3691.26 | 3691.26 | 20 | 0 | 5787 | 1 | 0 | 5787 | 5787 | 61.04 | 1 | 0 | 5787 | 5787 |
| 4 | 3 | 1 | 17943170372 | 535 | 1398 | 1398 | 900.13 | 900.13 | 3 | 0 | 1539 | 1 | 0 | 1539 | 1539 | 66.32 | 1 | 0 | 1539 | 1539 |
| 5 | 4 | 1 | 55773170781 | 241 | 21.228 | 21.228 | 159.42 | 159.42 | 41 | 0 | 947 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 947 | 947 |
| 6 | 5 | 1 | 03813182730 | 947 | 150.6193 | 150.6193 | 1098.9 | 1098.9 | 4 | 0 | 2309 | 7 | 2 | 20029 | 2309 | 29 | 8 | 2 | 23496 | 2888 |
| 7 | 6 | 1 | 35819170783 | 568 | 2257.363 | 2261.46 | 368.13 | 380.13 | 2 | 0 | 1539 | 4 | 10 | 6156 | 1539 | 15.4 | 8 | 0 | 11744 | 1539 |
| 8 | 7 | 1 | 96759184459 | 545 | 2876.642 | 2883.97 | 335.75 | 402.9 | 13 | 0 | 5787 | 1 | 0 | 5787 | 5787 | 277.8 | 1 | 0 | 5787 | 5787 |
| 9 | 8 | 1 | 09832190846 | 768 | 12905 | 17804.15 | 900.35 | 2549.11 | 4 | 55 | 3178 | 3 | 3 | 10404 | 3178 | 36 | 9 | 3 | 26095 | 3178 |
| 10 | 9 | 1 | 59772184450 | 1191 | 90.695 | 90.695 | 2287.5 | 2287.5 | 1 | 0 | 1539 | 4 | 1 | 6164 | 1539 | 39.9 | 4 | 1 | 6164 | 1539 |
| 11 | 10 | 1 | 56331170783 | 536 | 29.35733 | 29.35733 | 612.96 | 612.96 | 11 | 0 | 773 | 1 | 0 | 773 | 773 | 86.8 | 1 | 0 | 773 | 773 |
| 12 | 11 | 1 | 32893182732 | 1511 | 12.896 | 12.896 | 790.44 | 790.44 | 8 | 0 | 1539 | 2 | 5 | 2312 | 1156 | 16.83 | 2 | 5 | 2312 | 1156 |
| 13 | 12 | 0 | 82417190848 | 82 | 65.16667 | 65.16667 | 326.2 | 326.2 | 17 | 0 | 7526 | 2 | 0 | 9065 | 4532.5 | 489 | 2 | 0 | 9065 | 4532.5 |
| 14 | 13 | 1 | 11435189236 | 154 | 227.041 | 227.041 | 240.41 | 240.41 | 2 | 0 | 1547 | 4 | 2 | 19086 | 4773.5 | 63 | 7 | 30 | 28979 | 1720 |
| 15 | 14 | 1 | 66580197601 | 887 | 55.90933 | 55.90933 | 208.8 | 208.8 | 2 | 0 | 1539 | 5 | 5 | 7703 | 1539 | 20.9 | 7 | 5 | 8649 | 1539 |
| 16 | 15 | 1 | 63139170377 | 707 | 8919 | 10317.35 | 399.25 | 2453.78 | 3 | 0 | 770 | 3 | 6 | 3079 | 770 | 66 | 8 | 10 | 19185 | 1539 |
| 17 | 16 | 0 | 24075189239 | 1037 | 12 | 12 | 1216.8 | 1216.8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 17 | 0 | 82053185350 | 1583 | 1000 | 1000 | 1000.8 | 1087.88 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 19 | 18 | 1 | 37204184452 | 929 | 10.688 | 10.688 | 40 | 40 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 20 | 19 | 1 | 44217190455 | 832 | 14.4 | 14.4 | 1660.96 | 1660.96 | 1 | 0 | 2309 | 3 | 26 | 4618 | 1539 | 88.8 | 3 | 26 | 4618 | 1539 |
| 21 | 20 | 1 | 19611190842 | 450 | 48.935 | 48.935 | 726.3 | 726.3 | 1 | 0 | 1539 | 2 | 8 | 9539 | 4769.5 | 12 | 2 | 8 | 9539 | 4769.5 |
| 22 | 21 | 1 | 67813190589 | 100 | 769.614 | 777.46 | 1050.57 | 1167.3 | 6 | 0 | 770 | 5 | 20 | 8867 | 770 | 168 | 8 | 31 | 14380 | 771.5 |
| 23 | 22 | 0 | 75522170784 | 378 | 514.6933 | 515.2 | 56.26 | 58.2 | 2 | 0 | 773 | 1 | 0 | 773 | 773 | 542 | 2 | 64 | 1546 | 773 |
| 24 | 23 | 1 | 61590195202 | 463 | 1540 | 1541 | 969.12 | 969.12 | 4 | 0 | 770 | 1 | 0 | 770 | 770 | 43 | 2 | 66 | 1543 | 771.5 |
| 25 | 24 | 1 | 95027190841 | 857 | 58.02333 | 58.02333 | 479.44 | 479.44 | 2 | 0 | 1539 | 4 | 12 | 6164 | 1539 | 115.5 | 4 | 12 | 6164 | 1539 |

And the data description is :

| Variable | Definition | Comment |
|----------------------|--|----------------------------|
| label | Flag indicating whether the user paid back the credit amount within 5 days of issuing the loan(1:success, 0:failure) | |
| msisdn | mobile number of user | |
| aon | age on cellular network in days | |
| daily_decr30 | Daily amount spent from main account, averaged over last 30 days (in Indonesian Rupiah) | |
| daily_decr90 | Daily amount spent from main account, averaged over last 90 days (in Indonesian Rupiah) | |
| rental30 | Average main account balance over last 30 days | Unsure of given definition |
| rental90 | Average main account balance over last 90 days | Unsure of given definition |
| last_rech_date_ma | Number of days till last recharge of main account | |
| last_rech_date_da | Number of days till last recharge of data account | |
| last_rech_amt_ma | Amount of last recharge of main account (in Indonesian Rupiah) | |
| cnt_ma_rech30 | Number of times main account got recharged in last 30 days | |
| fr_ma_rech30 | Frequency of main account recharged in last 30 days | Unsure of given definition |
| sumamnt_ma_rech30 | Total amount of recharge in main account over last 30 days (in Indonesian Rupiah) | |
| medianamnt_ma_rech30 | Median of amount of recharges done in main account over last 30 days at user level (in Indonesian Rupiah) | |
| medianmarechprebal30 | Median of main account balance just before recharge in last 30 days at user level (in Indonesian Rupiah) | |
| cnt_ma_rech90 | Number of times main account got recharged in last 90 days | |
| fr_ma_rech90 | Frequency of main account recharged in last 90 days | Unsure of given definition |
| sumamnt_ma_rech90 | Total amount of recharge in main account over last 90 days (in Indonesian Rupiah) | |
| medianamnt_ma_rech90 | Median of amount of recharges done in main account over last 90 days at user level (in Indonesian Rupiah) | |
| medianmarechprebal90 | Median of main account balance just before recharge in last 90 days at user level (in Indonesian Rupiah) | |
| cnt_da_rech30 | Number of times data account got recharged in last 30 days | |
| fr_da_rech30 | Frequency of data account recharged in last 30 days | |
| cnt_da_rech90 | Number of times data account got recharged in last 90 days | |
| fr_da_rech90 | Frequency of data account recharged in last 90 days | |

- Data Preprocessing Done

In the Data Preprocessing, At first we drop the unnecessary variables or the features which are not related to the further processes, after this we remove those feature which are very highly correlated to each other. And then we divide the dataset into dependent and independent sets. After this we scaled our dataset but not the target variable by the Normalizer method and then we select top 25 most important features as the data is very huge and this is done by the sklearn library by importing RFE Recursive feature elimination and based on the ranking of top 25 we select the top 25 columns for the process. And after that outliers were handled with the help of zscore method and then Undersampling is done as dataset is class imbalanced and then splitting the data for Training and Testing.

- Data Inputs- Logic- Output Relationships

By the visualization of the data we see the relationship between the features and also with the correlation tells about the all features related to the Target variable. By the correlation method we can see the relationship and effects of features on the target variable. And also we see the correlation between the features and we deleted those columns which are highly correlated with each other as both of them carry the same type of data.

- Hardware and Software Requirements and Tools Used

The Project is done on the Window 10, here we use the Software Anaconda platform (Python 3.8.5 64 bit) and the project is done on the Jupyter Notebook where we run different python libraries such as

- pandas
- numpy
- scipy
- scikit-learn
- Matplotlib
- Seaborn
- Joblib

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

Here we first check the data is supervised or unsupervised by checking the target column and then when we see the output it shows it's a classification problem. Then we split the data into training and testing and then we trained the data with different model using various classifiers (logistic Regression, Random Forest Classifier, Decision Tree Classifier, Naïve Bayes, Linear SVC) and we check the accuracy score, classification report and confusion matrix and then we check the aucroc curve and score for the better prediction.

- Testing of Identified Approaches (Algorithms)

Here we use the following Algorithms used for training & Testing:

- Logistic Regression
 - Linear SVC
 - Random Forest Classifier
 - Naïve Bayes
 - Decision Tree Classifier
- Run and Evaluate selected models
- Describe all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.

1. Logistic Regression :

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a dependent variable. In **logistic regression**, the dependent variable is a binary variable that contains data coded as 1 (Non defaulter) or 0 (Defaulter) in our case.

LOGISTIC REGRESSION

In [119]:

```
model = LogisticRegression()
model.fit(x_train,y_train)
pred_y = model.predict(x_test)
print("Accuracy Score:",accuracy_score(y_test,pred_y))
# classification report
print(classification_report(y_test, pred_y))
#Confusion matrix
print(confusion_matrix(y_test, pred_y))
```

Accuracy Score: 0.8745666210757339

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.50 | 0.18 | 0.26 | 7908 |
| 1 | 0.89 | 0.97 | 0.93 | 54970 |
| accuracy | | | 0.87 | 62878 |
| macro avg | 0.70 | 0.58 | 0.60 | 62878 |
| weighted avg | 0.84 | 0.87 | 0.85 | 62878 |

```
[[ 1414  6494]
 [ 1393 53577]]
```

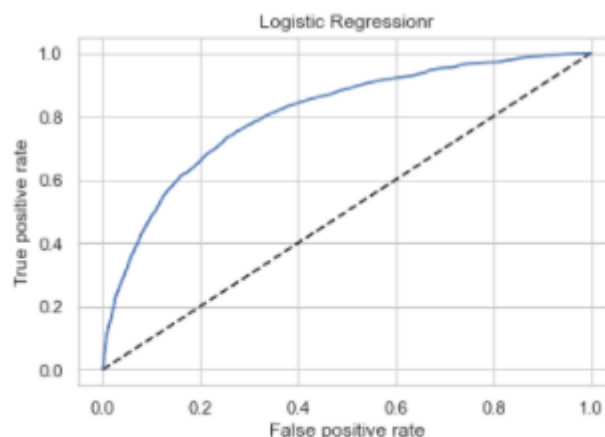
In [120]: *#Now Lets try to do some evaluation for Logistic Regression using cross validation.*

```
model_eval = cross_val_score(estimator = model, X = x_train, y = y_train, cv = 5)
print("The mean of cross_validation score :",model_eval.mean())
```

The mean of cross_validation score : 0.8782196776062434

In [121]: *#Logistic Regression Curve*

```
y_pred_prob = model.predict_proba(x_test)[:,:1]
fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
plt.plot([0,1],[0,1], 'k--')
plt.plot(fpr, tpr, label='Logistic Regression')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('Logistic Regressionr')
plt.show()
auc_score = roc_auc_score(y_test, model.predict(x_test))
auc_score
```



Out[121]: 0.576732588493342

2. Linear SVC :

The objective of a **Linear SVC** (Support Vector Classifier) is to fit to the data we provide, returning a "best fit" hyperplane that divides, or categorizes, our data. From there, after getting the hyperplane, you can then feed some features to our classifier to see what the "predicted" class is.

SUPPORT VECTOR MACHINE

```
In [147]: from sklearn.svm import SVC

# creating the model
clf = LinearSVC()

# feeding the training set into the model
clf.fit(x_train, y_train)

# predicting the results for the test set
y_pred = clf.predict(x_test)

# calculating the accuracy
print("Accuracy Score:", accuracy_score(y_test, y_pred))
# classification report
print(classification_report(y_test, y_pred))
# Confusion matrix
print(confusion_matrix(y_test, y_pred))
```

Accuracy Score: 0.6157002449187315

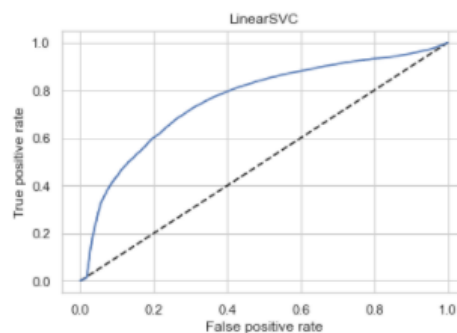
| | | precision | recall | f1-score | support |
|--------------|---|-----------|--------|----------|---------|
| | 0 | 0.50 | 0.18 | 0.26 | 7908 |
| | 1 | 0.89 | 0.97 | 0.93 | 54970 |
| accuracy | | | | 0.87 | 62878 |
| macro avg | | 0.70 | 0.58 | 0.60 | 62878 |
| weighted avg | | 0.84 | 0.87 | 0.85 | 62878 |

```
[[ 2597  5311]
 [18853 36117]]
```

```
In [148]: model_eval = cross_val_score(estimator = clf, X = x_train, y = y_train, cv = 5)
print("The mean of cross_validation score :", model_eval.mean())
```

The mean of cross_validation score : 0.5159390655352214

```
In [133]: #SVC CURVE
predict_proba_dist = clf.decision_function(x_test)
fpr, tpr, thresholds = roc_curve(y_test, predict_proba_dist)
plt.plot([0,1],[0,1], 'k--')
plt.plot(fpr, tpr, label='LinearSVC')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('LinearSVC')
plt.show()
auc_score = roc_auc_score(y_test, model.predict(x_test))
auc_score
```



```
Out[133]: 0.6999800806417701
```

Random Forest Classifier :

A **random forest classifier**. A **random forest** is a meta estimator that fits a number of **decision tree classifiers** on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.

RANDOM FOREST CLASSIFIER

```
In [122]: # creating the model
model = RandomForestClassifier(n_estimators = 100)

# feeding the training set into the model
model.fit(x_train, y_train)

# predicting the results for the test set
y_pred = model.predict(x_test)

# calculating the accuracy
print("Accuracy Score:", accuracy_score(y_test, y_pred))
# classification report
print(classification_report(y_test, y_pred))
# Confusion matrix
print(confusion_matrix(y_test, y_pred))
```

```
Accuracy Score: 0.9110340659690194
      precision    recall  f1-score   support

     0       0.76      0.43      0.55       7908
     1       0.92      0.98      0.95      54970

 accuracy         0.91      62878
 macro avg        0.84      0.71      0.75      62878
 weighted avg     0.90      0.91      0.90      62878

[[ 3405  4503]
 [1091 53879]]
```

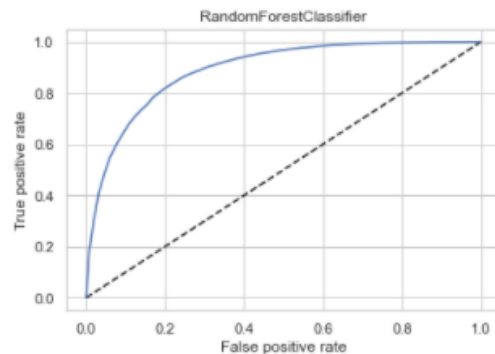
```
In [123]: #Now Lets try to do some evaluation for RandomForestClassifier using cross validation.
```

```
model_eval = cross_val_score(estimator = model, X = x_train, y = y_train, cv = 5)
print("The mean of cross_validation score :", model_eval.mean())
```

```
The mean of cross_validation score : 0.911501891422145
```

```
In [124]: #RandomForest Curve
```

```
y_pred_prob = model.predict_proba(x_test)[:,-1]
fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
plt.plot([0,1],[0,1], 'k--')
plt.plot(fpr, tpr, label='RandomForestClassifier')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('RandomForestClassifier')
plt.show()
auc_score = roc_auc_score(y_test, model.predict(x_test))
auc_score
```



```
Out[124]: 0.7053647209417304
```

3. Naïve Bayes (GaussianNB) :

GAUSSIAN NAIVEBAYES

```
In [128]: # creating the model
model = GaussianNB()

# feeding the training set into the model
model.fit(x_train, y_train)

# predicting the results for the test set
y_pred = model.predict(x_test)

# calculating the accuracy
print("Accuracy score:", accuracy_score(y_test, y_pred))
# classification report
print(classification_report(y_test, y_pred))
# confusion matrix
print(confusion_matrix(y_test, y_pred))
```

Accuracy Score: 0.5661916727631287

| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.50 | 0.18 | 0.26 | 7908 |
| 1 | 0.89 | 0.97 | 0.93 | 54970 |
| accuracy | | | 0.87 | 62878 |
| macro avg | 0.70 | 0.58 | 0.60 | 62878 |
| weighted avg | 0.84 | 0.87 | 0.85 | 62878 |

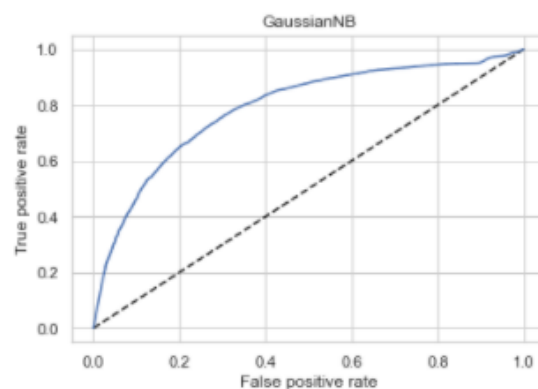
```
[[ 6949  959]
 [26318 28652]]
```

```
In [129]: model_eval = cross_val_score(estimator = model, X = x_train, y = y_train, cv = 5)
print("The mean of cross_validation score :", model_eval.mean())
```

The mean of cross_validation score : 0.5663292778516171

```
In [130]: #GaussianNB CURVE

y_pred_prob = model.predict_proba(x_test)[: ,1]
fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
plt.plot([0,1],[0,1], 'k--')
plt.plot(fpr, tpr, label='GaussianNB')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('GaussianNB')
plt.show()
auc_score = roc_auc_score(y_test, model.predict(x_test))
auc_score
```



```
Out[130]: 0.6999800806417701
```

4. Decision Tree Classifier:

DecisionTreeClassifier

```
In [136]: # creating the model
model = DecisionTreeClassifier(criterion='entropy',random_state=7)

# feeding the training set into the model
model.fit(x_train, y_train)

# predicting the results for the test set
y_pred = model.predict(x_test)

# calculating the accuracy
print("Accuracy Score:",accuracy_score(y_test,y_pred))
# classification report
print(classification_report(y_test, y_pred))
#Confusion_matrix
print(confusion_matrix(y_test, y_pred))
```

```
Accuracy Score: 0.869143420592258
      precision    recall  f1-score   support

     0       0.48      0.49      0.48       7908
     1       0.93      0.92      0.93      54970

 accuracy
macro avg      0.70      0.71      0.70      62878
weighted avg    0.87      0.87      0.87      62878

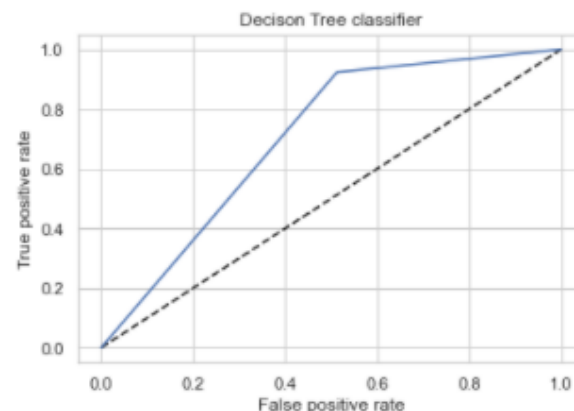
[[ 3851  4057]
 [ 4171 50799]]
```

```
In [137]: model_eval = cross_val_score(estimator = model, X = x_train, y = y_train, cv = 5)
print("The mean of cross_validation score :",model_eval.mean())
```

```
The mean of cross_validation score : 0.8657465153528949
```

```
In [138]: #Decision Tree Curve
```

```
y_pred_prob = model.predict_proba(x_test)[:,:1]
fpr,tpr,thresholds=roc_curve(y_test,y_pred_prob)
plt.plot([0,1],[0,1], 'k--')
plt.plot(fpr,tpr,label='Decision tree classifier')
plt.xlabel('False positive rate')
plt.ylabel('True positive rate')
plt.title('Decison Tree classifier')
plt.show()
auc_score=roc_auc_score(y_test,model.predict(x_test))
auc_score
```



```
Out[138]: 0.7055487317356808
```

- Key Metrics for success in solving problem under consideration

Here we use the Aucroc curve and score for the final prediction.

- Visualizations

Here we use the following graph:

1. **Histogram** for checking the distribution of data of single column.
2. **Scatterplot** to see the distribution of data among two columns.
3. **Count plot** to see the distribution of data in target column.
4. Also the **Stripplot** is used to see the distribution of data along with the target column.
5. **Heatmap** and **Bar** graph is used to see the correlation of columns.
6. **Piechart** is used to see percentage of two output in the target column.

- Interpretation of the Results

By the help of Visualizations ,preprocessing and modelling we see the distribution of data and remove the not so important features and highly correlated features and also remove some outliers and removing some skewness from the data is imbalanced we use the undersampling method and then we do the model training and predicting the outcome and selecting the best model which have the high aucroc score as it tells us how correctly model predicted. And also we see that do the hyperparameter tuning and gets the best model for the prediction so that the person who actually needed the loan can have it.

CONCLUSION

- Key Findings and Conclusions of the Study

Here we see that the people actually needed the loan as they have very less source of income. With the help of different python libraries and machine learning we predict the best model which is suitable for the prediction for the process. Key finding of this case study is to find the best model for the loan transaction. And what I learn through this project is that we need to work hard and continuously for the better the better result as the data is very much so it takes lots of patience to understand the data. Why people actually needed as they have very low income in their main account and their data account also have very amount. There was so much things to look in this project.

- Learning Outcomes of the Study in respect of Data Science

After working on this project I learned about the various pandas libraries and learn how to do real project like when to use which library and with the help of visualization I can clearly see the relation among the columns and also checking some skewness and outliers in the dataset. Data Cleaning helps to reduce the columns which are highly correlated to eachother at first and then selecting top 25 columns which are most important for the prediction and also removing the outliers and skewed data. And then train the model and test the model to give good accuracy score. Here the algorithm which work was decision tree classifier as it has the highest auc roc score which help to understand the distribution of correct data. The most challenging part was to first to understand the actual problem and use different libraries and method to solve the problem as the dataset is very huge and very new to me it also

take very time to understand this. By hit and trial and also the hardwork help me to overcome from this problem.

- **Limitations of this work and Scope for Future Work**

The limitations of this project is that there are very columns which I say irrelevant also some columns are highly correlated to eachother , so I think we can reduce that at the starting and also data was very huge and it takes very time to process in some cases. If the data is more clear and understandable then it will be better to solve and improve the result also.