



# Target Marketing for Canadian Bank

---

# Scenario and Objective

---

## **Scenario:**

In order to increase deposit balances, Bank “A” is looking to execute a marketing campaign. The campaign will target existing clients and will offer them promotional interest rates to attract deposit balance.

## **Objective:**

The task is to predict which customers who are most likely to respond to the campaign.

# Dataset Overview


---

The dataset provided is split into Training data and Testing data.

**Training data** contains 64,000 observations with 37 variables from previous campaigns. The actual responses are in the variable called 'Target'.

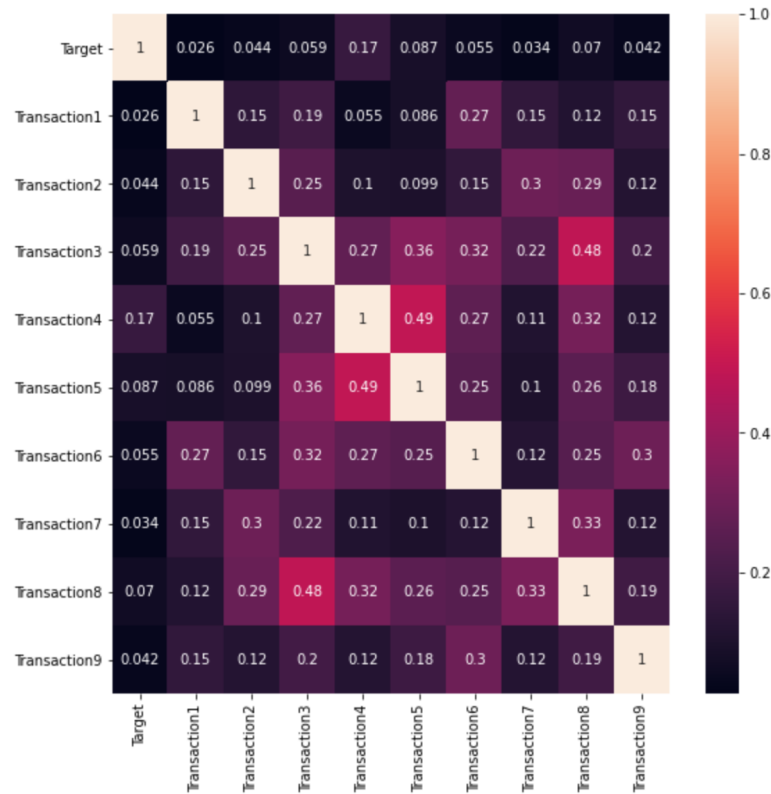
**Testing data** contains 1478 observations with 36 variables and no labels.

# Data Preparation

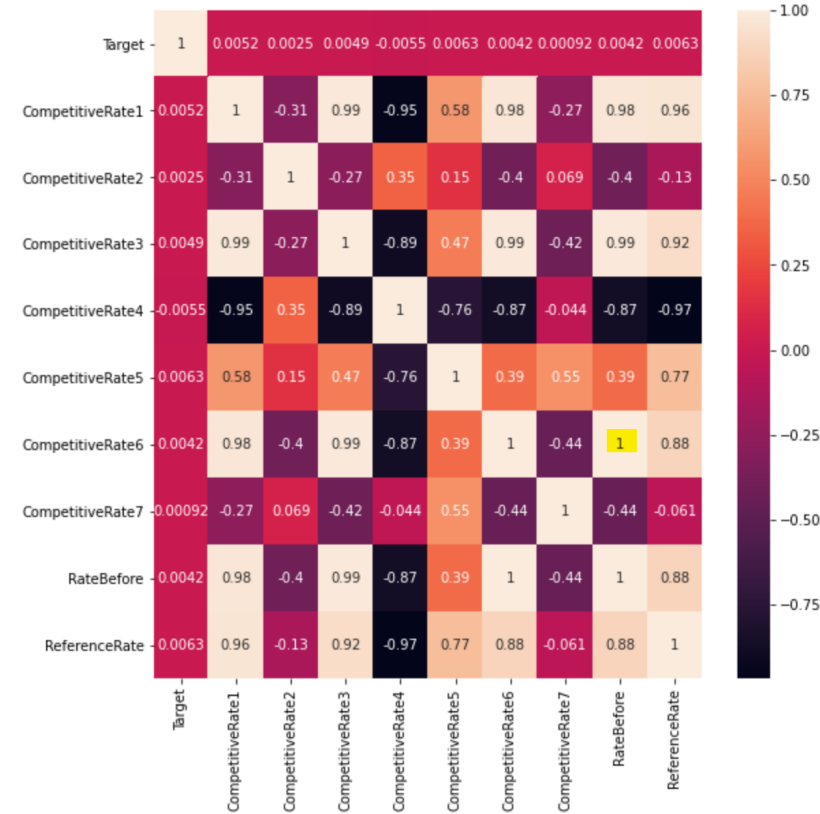
Checking the dataset for:	Results												
Null or N/A values in the dataset	No N/A or Null values were identified. <div>&gt; # of N/As: No N/A</div>												
Customer_id duplicates	No duplicates were identified for 'Customer_id' <div>df.loc[df.Customer_id.duplicated(),:].shape[0]</div> <div>0</div>												
Train 'Target' (potential skewness), where: 0 = no response from the customer 1 = response from the customer	<div></div> <div>&gt; Responding Rate%</div> <table><tr><th></th><th>Target</th><th>%</th></tr><tr><td>Target</td><td></td><td></td></tr><tr><td>0</td><td>32014</td><td>50.022657</td></tr><tr><td>1</td><td>31985</td><td>49.977343</td></tr></table>		Target	%	Target			0	32014	50.022657	1	31985	49.977343
	Target	%											
Target													
0	32014	50.022657											
1	31985	49.977343											

# Heatmaps for Target Correlation

Transactions and Target



Competitive



# Feature Engineering

---

Instead of looking at each column for ExternalAccounts and Products, the sum can be taken to get the total number of External Accounts and Products (e.g., customer A has 4 Products and 2 ExternalAccounts).

ExternalAccount1	ExternalAccount2	ExternalAccount3	ExternalAccount4	ExternalAccount5	ExternalAccount6	ExternalAccount7
0	0	0	0	0	0	0
0	0	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	0	0	1	0
0	1	0	0	0	0	0

Product1	Product2	Product3	Product4	Product5	Product6
0	0	0	0	0	0
0	0	0	0	0	0
1	1	0	0	0	0
0	0	0	0	0	0
1	0	0	0	0	0

# Feature Engineering

---

Taking the sum, mean, median, std, Min, Max, and Count of the Transactions

Transaction1	Transaction2	Transaction3	Transaction4	Transaction5	Transaction6	Transaction7	Transaction8	Transaction9
0.00	0.0	0.0	0.0	0.0	0.00	0.0	0.0	0.0
0.00	0.0	0.0	8525.0	0.0	0.00	0.0	0.0	0.0
13422.35	800.0	0.0	0.0	0.0	13123.28	0.0	0.0	0.0
0.00	0.0	0.0	1000.0	23900.0	0.00	0.0	0.0	0.0
0.00	0.0	0.0	0.0	2000.0	27629.11	0.0	0.0	0.0

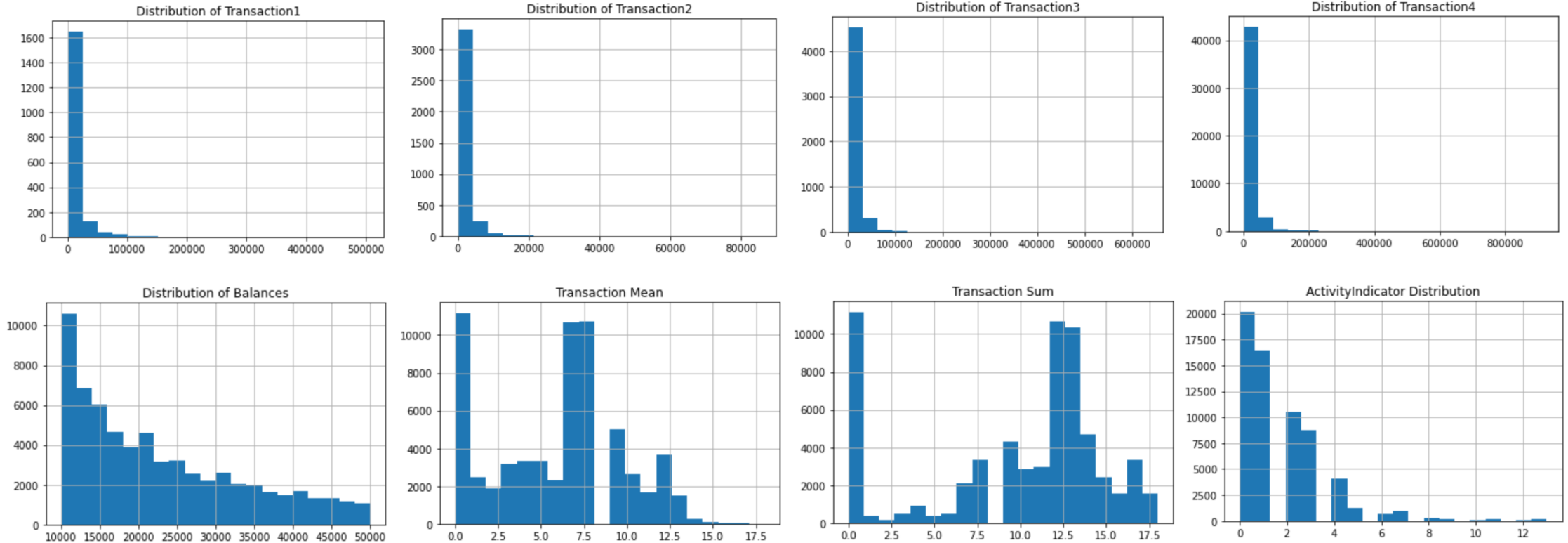
It was observed that 'CompetitiveRate6' has the same data as 'RateBefore' (1-1 Correlation). Therefore, to improve model accuracy, CompetitiveRate6 was dropped from the DF.

Created new feature called 'Rate\_Drop', which takes the difference between 'RateBefore' and 'ReferenceRate'.



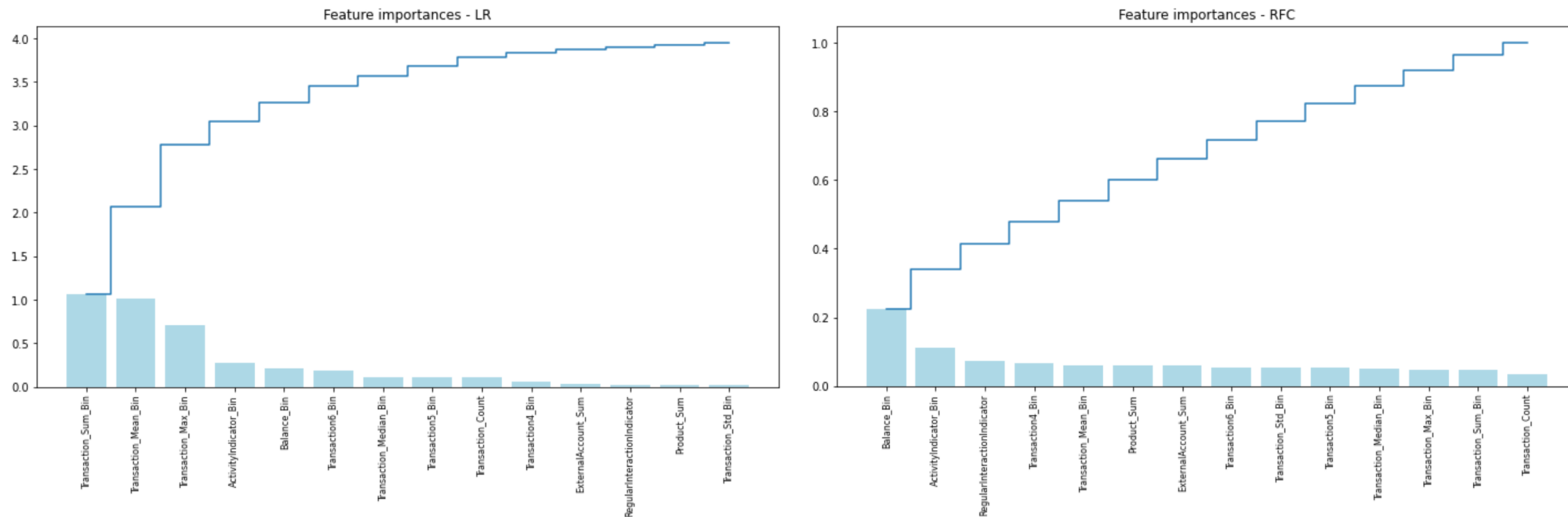
# Distribution of Features

---





# Feature Importance



Top features for both LR and RFC include new features that were engineered from 'Transactions'.

# Models

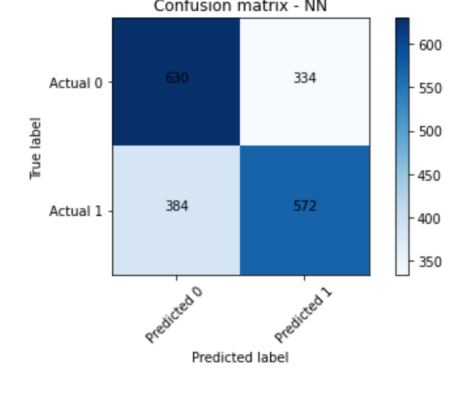
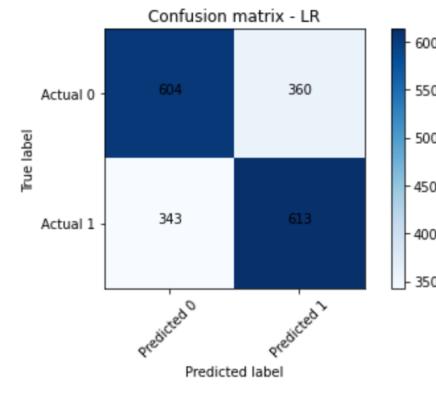
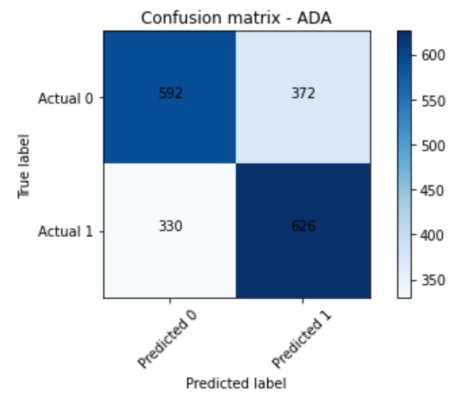
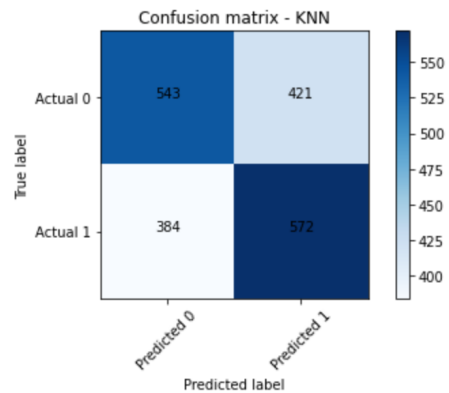
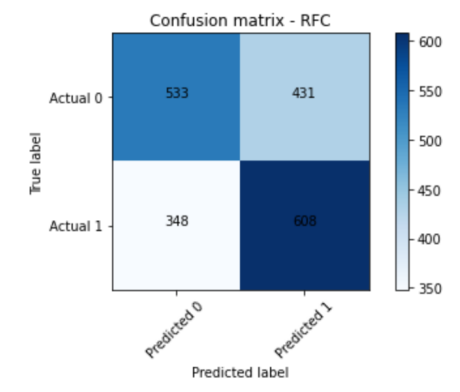
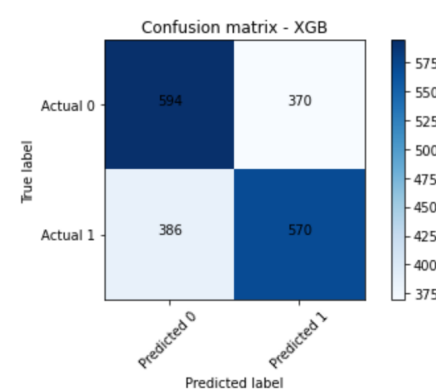
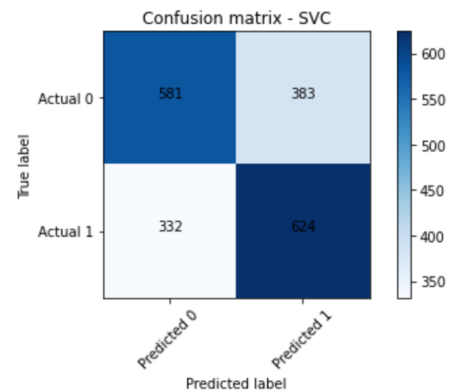
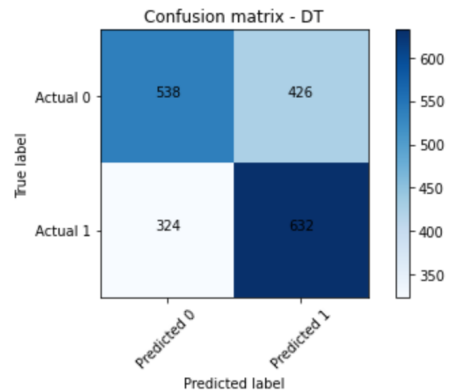
---

```
classifiers = {  
    "DT": tree.DecisionTreeClassifier()  
    , "KNN": KNeighborsClassifier()  
    , "SVC": svm.SVC()  
    , "ADA": AdaBoostClassifier()  
    , "XGB": GradientBoostingClassifier()  
    , "LR": LogisticRegression()  
    , "RFC": RandomForestClassifier()  
    , "NN": keras.wrappers.scikit_learn.KerasClassifier(build_model)  
}
```

Based on the accuracy scores generated by the models above, the following were deemed as the best combination of features:

Balance_Bin	ActivityIndicator_Bin	Transaction4_Bin	Transaction_Max_Bin	Transaction_Median_Bin	Transaction_Mean_Bin	Transaction_Sum_Bin
Transaction_Std_Bin	Transaction6_Bin	RegularInteractionIndicator	Transaction5_Bin	ExternalAccount_Sum	Transaction_Count	Product_Sum

# Confusion Matrix



# Model Performance Summary

---

Precision	Recall	F1-Score	Support	Algorithm/Model
63.62%	63.59%	63.59%	1920	Vote
62.63%	62.60%	62.57%	1920	NN
59.51%	59.43%	59.36%	1920	RFC
63.39%	63.39%	63.38%	1920	LR
60.63%	60.63%	60.62%	1920	XGB
63.47%	63.44%	63.42%	1920	ADA
62.80%	62.76%	62.74%	1920	SVC
58.09%	58.07%	58.06%	1920	KNN
61.08%	60.94%	60.84%	1920	DT

# Model Performance – Kaggle Score

---

Based on the scores obtained from the Training Dataset, the following submissions were made on Kaggle for the Test Dataset. At the end the voting classifier had the best prediction using all the other trained models.

Model	Best Public Score
Voting	66.55%
XGB	64.19%
SVC	62.16%
NN	61.82%

# Challenges

Challenge/Observation	Solution
Models take a long time to run. Encountered incidences where Google-collab timed out.	<p>Select smaller percentage of test_size using StratifiedShuffleSplit</p> <pre>sss = StratifiedShuffleSplit(n_splits=1, test_size=0.3, random_state=random_seed)  for train_index, test_index in sss.split(df_x, df_y):     df_x, df_y = df_x.iloc[train_index,:], df_y.iloc[train_index]  df_x.reset_index(drop=True, inplace=True) df_y.reset_index(drop=True, inplace=True) df_x.shape</pre>
Re-engineer features to improve model performance/ accuracy.	Testing all possible new features and check for model performance/accuracy (e.g., taking the sum, mean, of Transactions). There is no real solution until all possibilities are tried and assessed.
Accuracy variation with Kaggle submission scores (Best performing model from the train data-set did not result in the best score from Kaggle submission).	Given the Kaggle competition environment, it is hard to predict which model would result in the best score. Thus, multiple submissions were made with different models, rather than submitting multiple times for the same model.

# Lessons Learnt/ Recommendations

---

## **Findings:**

Through the modeling process it was discovered that the Voting Classifier and Logistic Regression had the best results in terms of accuracy. The Logistic Regression is more preferable as it is easier to implement and has faster performance. While working with the data it observed that the features were not very informative and this greatly limited the accuracy of the prediction.

## **Recommendation:**

A recommendation to the bank would be to collect more features that have a stronger correlation to the target (e.g. more identity related features that personalize the dataset and give insight into customer inclinations).