2021

CREDIT CARD LEAD PREDICTION

PROJECT DOCUMENTATION

MOUMITA MURMU

PROBLEM STATEMENT

Happy Customer Bank is a mid-sized private bank that deals in all kinds of banking products, like Savings accounts, Current accounts, investment products, credit products, among other offerings.

The bank also cross-sells products to its existing customers and to do so they use different kinds of communication like telecasting, e-mails, recommendations on net banking, mobile banking, etc.

In this case, the Happy Customer Bank wants to cross-sell its credit cards to its existing customers. The bank has identified a set of customers that are eligible for taking these credit cards.

Now, the bank is looking for your help in identifying customers that could show higher intent towards a recommended credit card.

- **ID**: Unique Identifier for a row
- **Gender**: Gender of the Customer (Male/Female).
- **Age**: Age of the Customer (in Years).
- **Region_Code**: Code of the Regions of the customers.
- **Occupation**: Occupation of the customers.
- **Channel_Code**: Acquisition Channel Code for the Customer (Encoded)
- **Vintage**: Vintage for the Customer (In Months) - month or quarter in which account was opened (loan/credit card was granted).
- **Credit_Product**: If the Customer has any active credit product (Home loan, Personal loan, Credit Card etc.)
- **Avg_Account_Balance**: Average Account Balance for the Customer in last 12 Months.
- **Is_Active**: If the Customer is Active in last 3 Months.
- **Is_Lead** (Target Variable): 0 - Customer is NOT INTERESTED, 1 - Customer IS INTERESTED

Class 1: The Customer is interested in Credit Card

Class 0: The Customer is NOT interested in Credit Card.

EXPLORATORY DATA ANALYSIS

For Train Data:

```
In [7]:  cc_data_train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 245725 entries, 0 to 245724
Data columns (total 11 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   ID                  245725 non-null  object
 1   Gender              245725 non-null  object
 2   Age                 245725 non-null  int64
 3   Region_Code         245725 non-null  object
 4   Occupation          245725 non-null  object
 5   Channel_Code        245725 non-null  object
 6   Vintage             245725 non-null  int64
 7   Credit_Product      216400 non-null  object
 8   Avg_Account_Balance 245725 non-null  int64
 9   Is_Active           245725 non-null  object
 10  Is_Lead             245725 non-null  int64
dtypes: int64(4), object(7)
memory usage: 20.6+ MB
```

- Total Observations/Rows: 245745; Total Features = 10 (Independent) + 1 (Dependent)
- Total Features – Numerical: 4; Categorical: 7 (All Nominal)
- Feature – Credit_Product has some missing values (29325 datapoints ~ 11.93%)

NUMERICAL FEATURES:

```
In [12]:  cc_data_train.describe()
Out[12]:
```

	Age	Vintage	Avg_Account_Balance	Is_Lead
count	245725.000000	245725.000000	2.457250e+05	245725.000000
mean	43.856307	46.959141	1.128403e+06	0.237208
std	14.828672	32.353136	8.529364e+05	0.425372
min	23.000000	7.000000	2.079000e+04	0.000000
25%	30.000000	20.000000	6.043100e+05	0.000000
50%	43.000000	32.000000	8.946010e+05	0.000000
75%	54.000000	73.000000	1.366666e+06	0.000000
max	85.000000	135.000000	1.035201e+07	1.000000

MEASURE OF CENTRAL TENDENCY				
Independent Feature	**Mean**	**Median**	**Min**	**Max**
Age	43.85	43	23	85
Vintage	46.96	32	7	135
Avg_Account_Balance	1128403.101	894601	20790	10352009

MEASURE OF DISPERSION & SHAPE								
Independent Feature	**Range**	**Q1**	**Q2**	**Q3**	**Variance**	**Std. Dev**	**Skewness**	**Kurtosis**
Age	62	30	43	54	219.88	14.82	0.61 (Moderate)	-0.44 (Meso)
Vintage	128	20	32	73	1046.72	32.35	0.79 (Moderate)	-0.69 (Meso)
Avg_Account_Balance	10331219	604310	894601	1366666	7.275E+11	852936.35	2.96 (High)	14.30 (leptoKurtic)

Note:

- **Skewness**: (-0.5 to 0.5): Symmetrical, (-1 to -0.5 OR 0.5 to 1): Moderately Skewed, (<-1 Or >1): Highly Skewed Distribution. [Positive Skew = Positive Value = Tail is on right side of distribution, Negative Skew = Negative Values = Tail is on the left side of the distribution]
- **Kurtosis**: (k>3): Leptokurtic, (k=3): Mesokurtic, (k<3): Platykurtic Distribution.

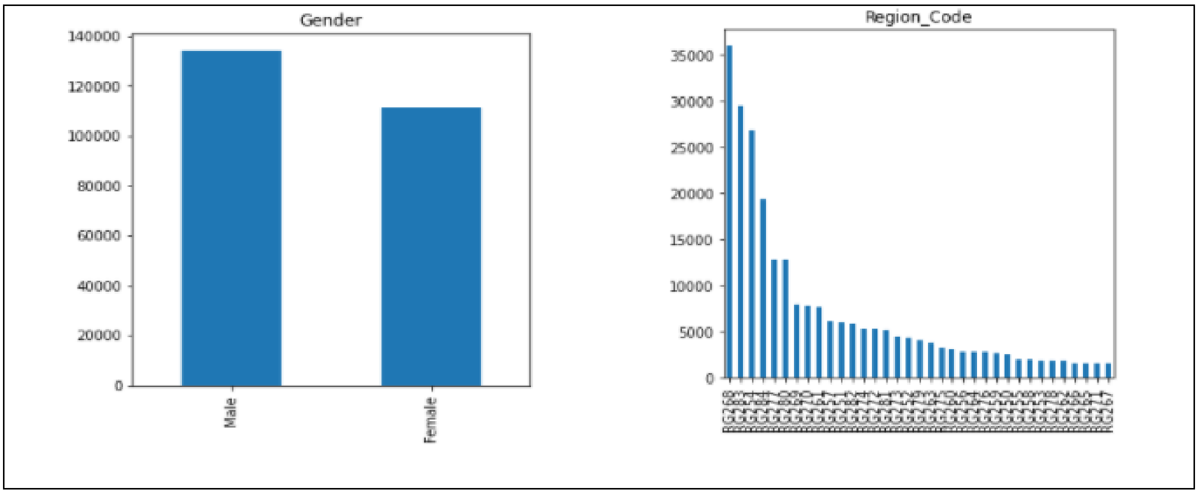
CATEGORICAL FEATURES:

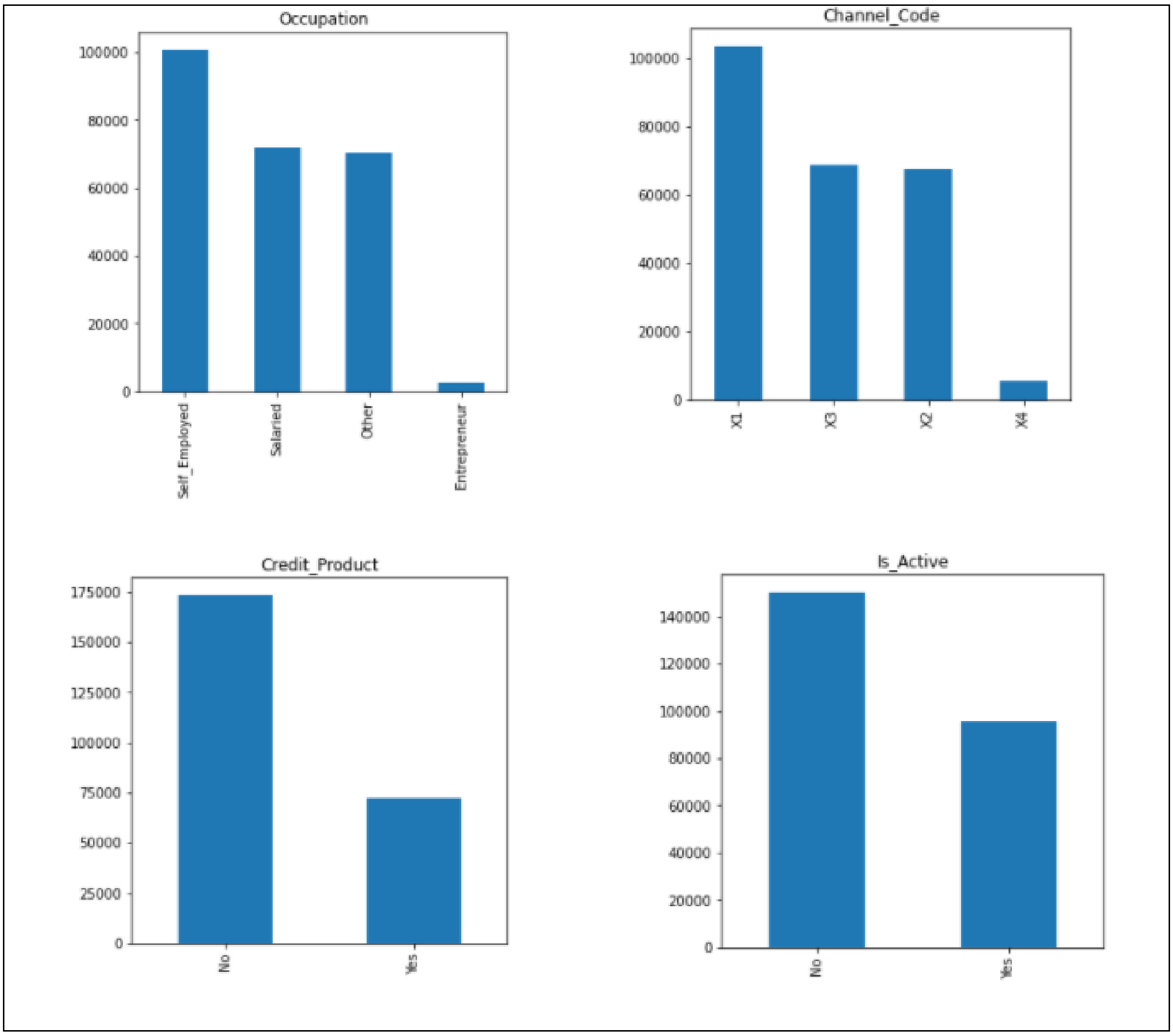
```
In [13]:  1  cc_data_train.describe(include = np.object)
Out[13]:
```

	ID	Gender	Region_Code	Occupation	Channel_Code	Credit_Product	Is_Active
count	245725	245725	245725	245725	245725	216400	245725
unique	245725	2	35	4	4	2	2
top	D8AQWMFF	Male	RG268	Self_Employed	X1	No	No
freq	1	134197	35934	100886	103718	144357	150290

Most common value in category-

- Gender: Male (frequency - 134197)
- Region_Code: RG268 (frequency - 35934)
- Occupation: Self-Employed (frequency - 100886)
- Channel_Code: X1 (frequency – 103718)
- Credit_Product: No (frequency – 144357)
- Is_Active: No (frequency – 150290)




Occupation
100000
80000
60000
40000
20000
0
Self_Employed
Salaried
Other
Entrepreneur
Channel_Code
100000
80000
60000
40000
20000
0
X1
X3
X2
X4
Credit_Product
175000
150000
125000
100000
75000
50000
25000
0
No
Yes
Is_Active
140000
120000
100000
80000
60000
40000
20000
0
No
Yes

DATA PREPROCESSING

1. MISSING VALUES IDENTIFICATION & TREATMENT:

```
In [17]:  cc_data_train.isnull().sum()
Out[17]: ID                        0
         Gender                    0
         Age                       0
         Region_Code               0
         Occupation                0
         Channel_Code              0
         Vintage                   0
         Credit_Product        29325
         Avg_Account_Balance       0
         Is_Active                 0
         Is_Lead                   0
         dtype: int64
```

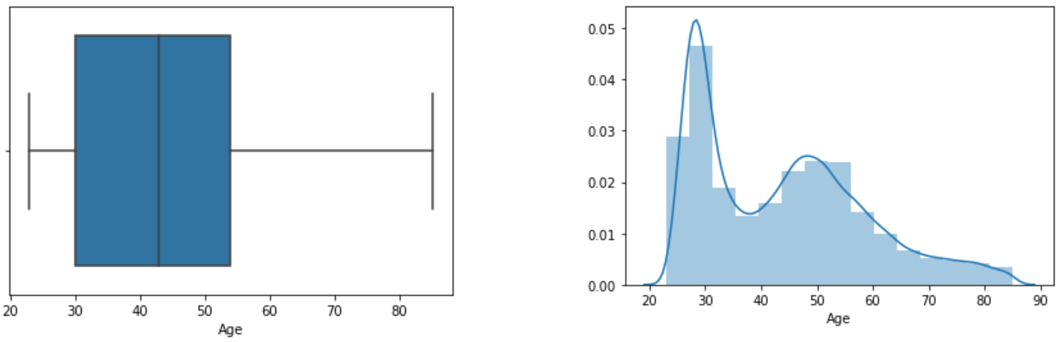
Using MODE IMPUTATION technique (replacing missing value with mode since it's a categorical feature).

```
In [18]:  cc_data_train['Credit_Product'].fillna(cc_data_train['Credit_Product'].mode()[0], inplace=True)
```

```
In [19]:  cc_data_train.isnull().sum()
Out[19]: ID                     0
         Gender                 0
         Age                    0
         Region_Code            0
         Occupation             0
         Channel_Code           0
         Vintage                0
         Credit_Product         0
         Avg_Account_Balance    0
         Is_Active              0
         Is_Lead                0
         dtype: int64
```

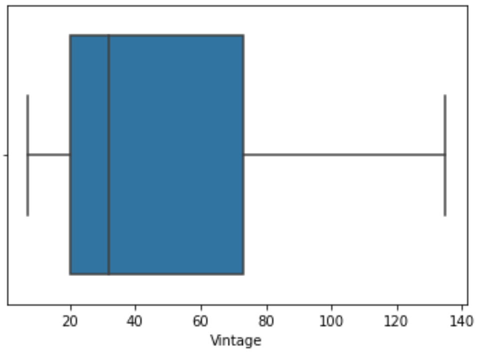
2. OUTLIER DETECTION & TREATMENT:

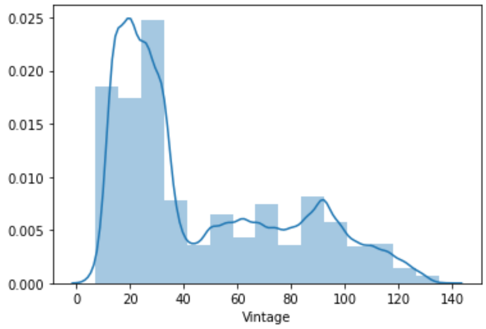
COLUMN: AGE




```
[Q1:  30.0, Q2:  43.0, Q3:  54.0, IQR:  24.0,
Lower Level:  -6.0, Upper Level:  90.0]
```

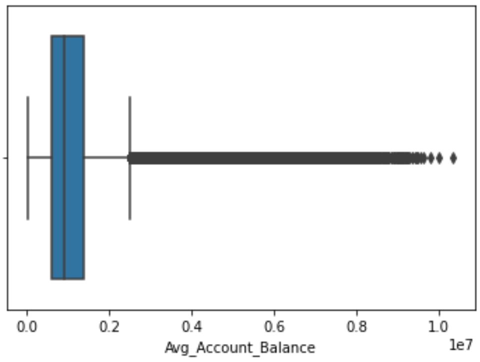
COLUMN: Vintage

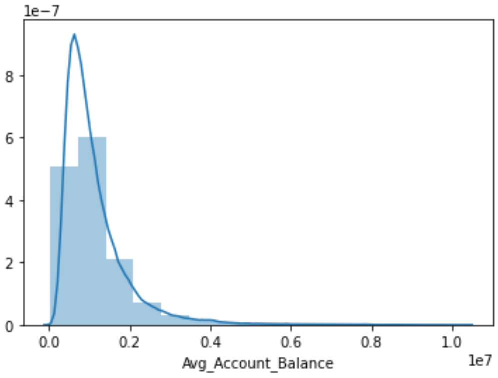





```
Q1:  20.0, Q2:  32.0, Q3:  73.0, IQR:  53.0
Lower Level:  -59.5, Upper Level:  152.5
```

COLUMN: Avg_Account_Balance



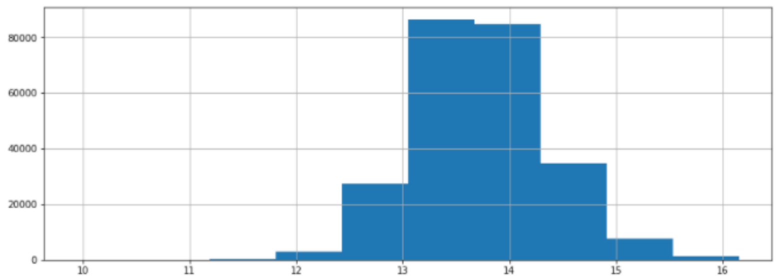



```
Q1:  604310.0, Q2:  894601.0, Q3:  1366666.0, IQR:  762356.0
Lower Level:  -539224.0, Upper Level:  2510200.0
```

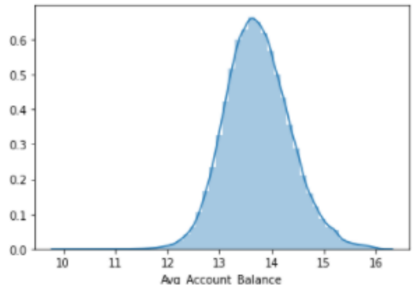
3. FEATURE ENGINEERING & SCALING:

a) VARIABLE TRANSFORMATION – Applying LOG TRANSFORMATION on feature
Avg_Account_Balance.

```
In [36]:  1 train_num_cols['Avg_Account_Balance'] = np.log(train_num_cols['Avg_Account_Balance'])

<ipython-input-36-3ae1ee9a1990>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ve
rsus-a-copy
  train_num_cols['Avg_Account_Balance'] = np.log(train_num_cols['Avg_Account_Balance'])

In [37]:  1 train_num_cols['Avg_Account_Balance'].hist(figsize=(14, 5))

Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x2a407be9400>
```



```
In [34]: sns.distplot(train_num_cols['Avg_Account_Balance'])
Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x1831422c280>
```



b) STANDARDIZATION –

```
In [43]: from sklearn.preprocessing import StandardScaler

         scaler = StandardScaler()

         train_num = scaler.fit_transform(train_num.values)
         train_num = pd.DataFrame(train_num, columns = cols_train)
```

```
In [44]: train_num.head()
Out[44]:
```

	Age	Vintage	Avg_Account_Balance
0	1.965365	-0.122373	0.204552
1	-0.934429	-0.462372	-0.740962
2	0.818935	-0.647825	0.769727
3	-0.664680	-0.864188	-1.084245
4	-0.934429	-0.431463	-0.061413

DATA PARTITION

a. Train-Test Split with 70:30 ratio
b. K-fold Cross Validation (K = 10 Folds)

```
In [67]: X = train_cc_data.drop(['Is_Lead'], axis=1)
         Y = train_cc_data[['Is_Lead']]

In [68]: X.columns
Out[68]: Index(['Age', 'Vintage', 'Avg_Account_Balance', 'Gender', 'Region_Code',
                'Occupation', 'Channel_Code', 'Credit_Product', 'Is_Active'],
               dtype='object')

In [69]: Y.columns
Out[69]: Index(['Is_Lead'], dtype='object')

In [70]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.30, random_state=0)

In [71]: train = pd.concat([X_train, Y_train], axis=1)
         test = pd.concat([X_test, Y_test], axis=1)

In [72]: X_test.columns
Out[72]: Index(['Age', 'Vintage', 'Avg_Account_Balance', 'Gender', 'Region_Code',
                'Occupation', 'Channel_Code', 'Credit_Product', 'Is_Active'],
               dtype='object')
```

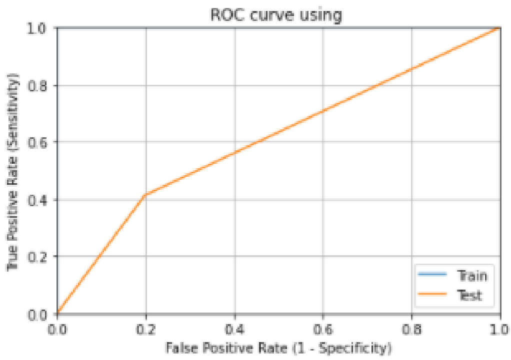
MODEL BUILDING

We will be considering the below classification algorithms and choose the model which provides better predictions for the specific classification problem statement. Below is the metric of classification report of the models considered for evaluation:

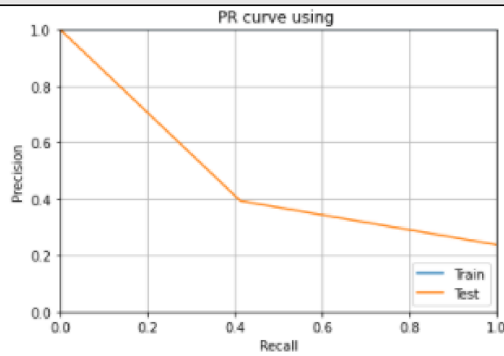
*Train – Marked in Black, *Test – Marked in Blue

MODEL 1 - DECISION TREE CLASSIFIER									
PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
1.00	1.00	1.00	1.00	0.7096	131189	0	0	40818	1.0
0.39	0.41	0.40	0.71		45117	11131	10282	7188	0.6074

ROC CURVE



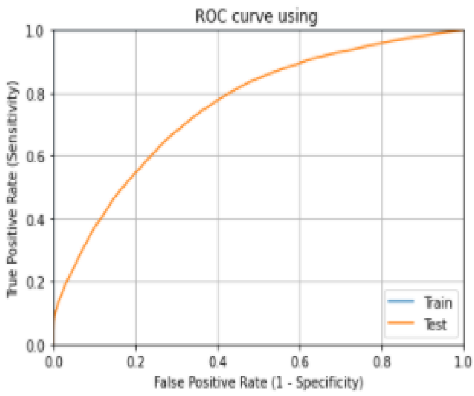
PRECISION – RECALL CURVE



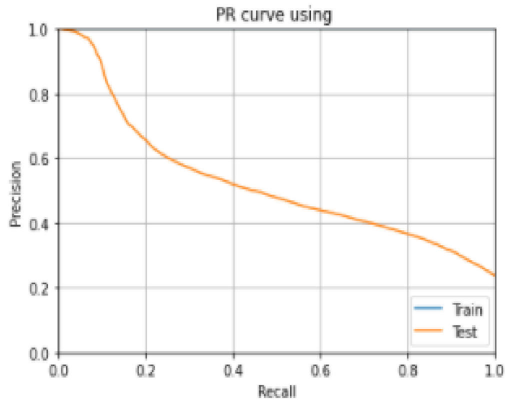
- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **39%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **41%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **80.21%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **19.73%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 52305
- **Incorrect Predictions (FN+FP)**: 21413
- **Misclassification Rate** = 0.29

MODEL 2 - RANDOM FOREST CLASSIFIER									
PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
1.00	0.99	0.99	1.00	0.7786	131189	0	3	40815	0.9999
0.56	0.31	0.40	0.78		51908	4340	12010	5460	0.7556

ROC CURVE



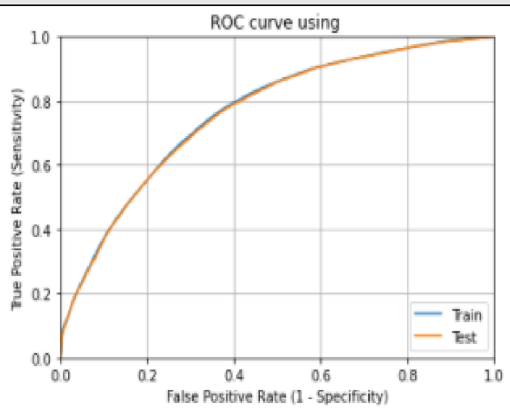
PRECISION – RECALL CURVE



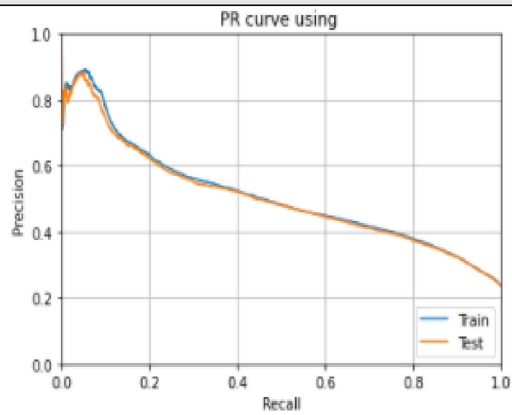
- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **56%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **31%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **92.28%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **7.72%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 57368
- **Incorrect Predictions (FN+FP)**: 16350
- **Misclassification Rate** = 0.22

MODEL 3 - ADABOOST CLASSIFIER									
PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
0.64	0.19	0.30	0.78	0.7816	126587	4602	32776	8042	0.7634
0.63	0.19	0.29	0.78		54254	1994	14079	3391	0.7603

ROC CURVE



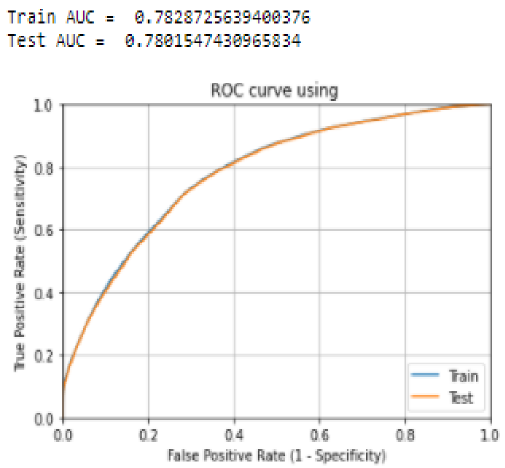
PRECISION – RECALL CURVE

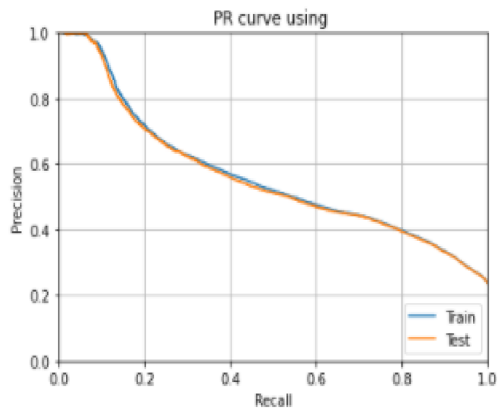


- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **63%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **19%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **96.45%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **3.55%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 57654
- **Incorrect Predictions (FN+FP)**: 16073
- **Misclassification Rate** = 0.21

MODEL 4 - GRADIENT BOOST CLASSIFIER									
PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
0.73	0.19	0.30	0.79	0.7917	128208	2981	32910	7908	0.7828
0.72	0.19	0.30	0.79		54963	1285	14169	3301	0.7801

ROC CURVE	PRECISION – RECALL CURVE

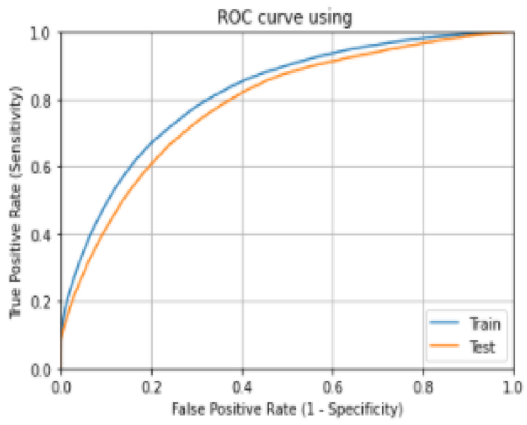




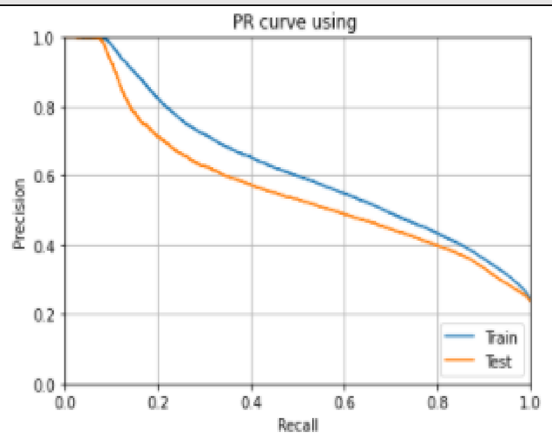
- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **72%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **19%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **97.72%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **2.28%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 58264
- **Incorrect Predictions (FN+FP)**: 15454
- **Misclassification Rate** = 0.21

MODEL 5 - XGBOOST CLASSIFIER									
PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
0.71	0.30	0.43	0.81	0.7924	126214	4975	28321	12497	0.8174
0.64	0.27	0.38	0.79		53602	2646	12699	4771	0.7846

ROC CURVE



PRECISION – RECALL CURVE

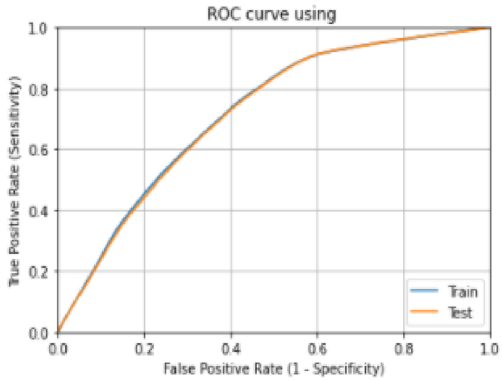


- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **64%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **27%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **95.30%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **4.70%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 58373
- **Incorrect Predictions (FN+FP)**: 15345
- **Misclassification Rate =** 0.21

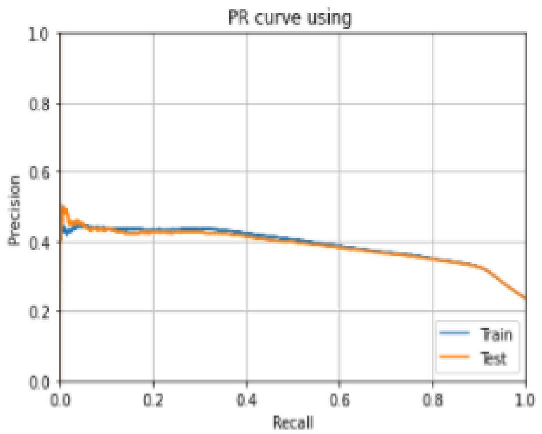
MODEL 6 - LOGISTIC REGRESSION									
PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
0.43	0.10	0.16	0.76	0.7561	125895	5294	36705	4113	0.7184
0.43	0.09	0.16	0.76		53991	2257	15745	1725	0.7153

ROC CURVE

```
Train AUC =  0.7184378331546346
Test AUC =  0.7153908132086889
```



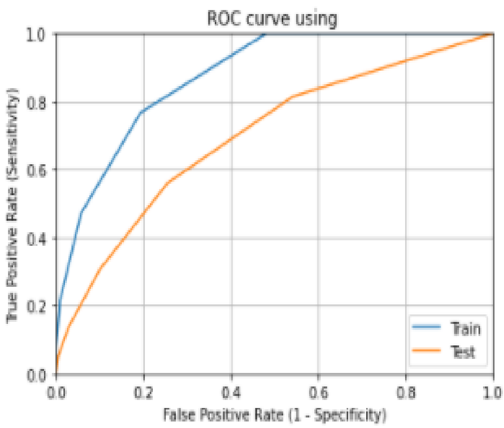
PRECISION - RECALL CURVE



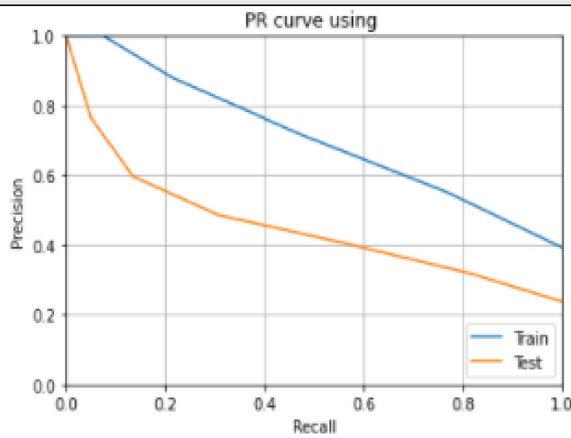
- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **43%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **9%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **95.99%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **4.01%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 55716
- **Incorrect Predictions (FN+FP)**: 18002
- **Misclassification Rate =** 0.24

MODEL 7 - KNN CLASSIFIER									
PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
0.71	0.47	0.56	0.83	0.7615	123681	7508	21611	19207	0.8750
0.48	0.30	0.37	0.76		50571	5677	12105	5365	0.6977

ROC CURVE



PRECISION - RECALL CURVE

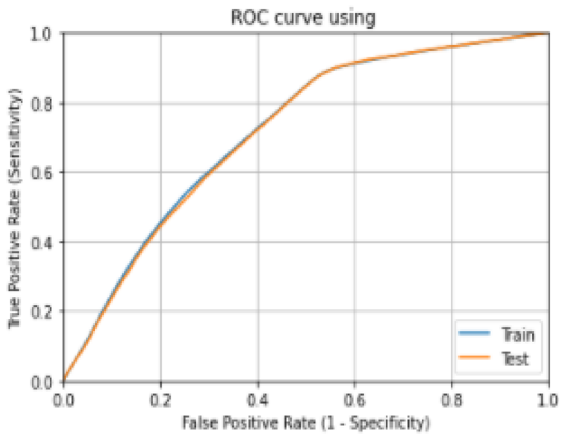


- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **48%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **30%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **89.91%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **10.09%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 55936
- **Incorrect Predictions (FN+FP)**: 17782
- **Misclassification Rate =** 0.24

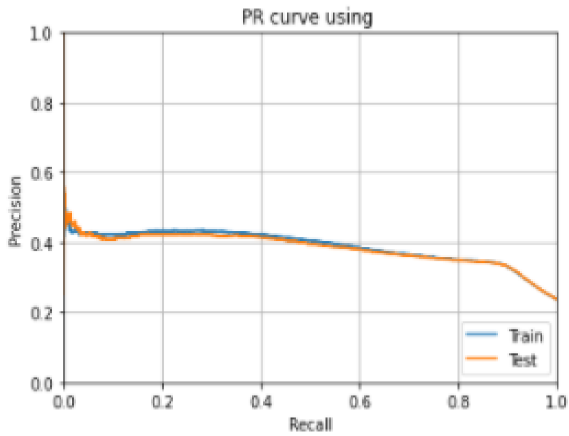
MODEL 8 - NAÏVE BAYES CLASSIFIER

PRECISION	RECALL	F1-SCORE	ACCURACY	CV	TN	FP	FN	TP	AUC
0.41	0.45	0.43	0.72	0.7164	104668	26521	22203	18615	0.7175
0.40	0.44	0.42	0.71		44906	11342	9691	7779	0.7144

ROC CURVE



PRECISION - RECALL CURVE



- **Precision** – When the classifier predicts that the customer will be interested in the credit card, it is correct around **40%** of the time.
- **Recall/ Sensitivity** – Out of the actual positive values (customers interested in credit card), **44%** are correctly predicted (of being interested in credit card).
- **Specificity** – Out of the actual negative values (customers not interested in credit card), **79.84%** are correctly predicted of NOT being interested in credit card.
- **FPR** – Out of the actual negative values (customers not interested in credit card), **20.16%** are incorrectly predicted to be interested.
- **Correct Predictions (TN+TP)**: 52685
- **Incorrect Predictions (FN+FP)**: 21033
- **Misclassification Rate =** 0.28

MODEL EVALUATION

As per the problem statement - bank is looking for its customers who would show higher intent towards their credit cards.

1 – Interested in Credit Card, 0 – Not interested in Credit Card

With correct predictions, it's specific in classifying these customers. What we need to look for are - customers falling under Type-I & Type-II errors category.

In **Type-I/ False Positive** metric – we have customers who are not really interested in credit cards but as per the classifier's prediction, they are interested. If offered, there is higher chances of rejection from customers and additionally time and efforts will go in vain.

In **Type-II/ False Negative** metric - we have those customers who are genuinely interested in credit card but as per the classifier's prediction, they are not interested. If not approached, we will lose out on prospects resulting in impacting revenue. **In our scenario, False Negatives are more critical than False Positives.**

Therefore, the choice of metric, as per the business objective, we should focus on optimizing – R**ecall/ Sensitivity**.

ML Models	TN	FP	FN	TP	Accuracy	Precision	Recall	F1-Score	Specificity	FPR	CV Score	AUC
KNN Classifier	50571	5677	12105	5365	0.7588	0.4859	0.3071	0.3763	0.8991	0.1009	0.7615	0.6977
Naïve Bayes Classifier	44906	11342	9691	7779	0.7147	0.4068	0.4453	0.4252	0.7984	0.2016	0.7164	0.7144
Logistic Regression	53991	2257	15745	1725	0.7558	0.4332	0.0987	0.1608	0.9599	0.0401	0.7561	0.7153
Decision Tree	45117	11131	10282	7188	0.7095	0.3924	0.4114	0.4017	0.8021	0.1979	0.7096	0.6074
Random Forest Classifier	51908	4340	12010	5460	0.7782	0.5571	0.3125	0.4004	0.9228	0.0772	0.7786	0.7553
Adaboost Classifier	54254	1994	14079	3391	0.7820	0.6297	0.1941	0.2967	0.9645	0.0355	0.7816	0.7603
GradientBoost Classifier	54963	1285	14169	3301	0.7904	0.7198	0.1890	0.2993	0.9772	0.0228	0.7917	0.7801
XGBoost Classifier	53602	2646	12699	4771	0.7918	0.6433	0.2731	0.3834	0.9530	0.0470	0.7924	0.7846

Based on following criteria, we will select the best model to solve this classification problem statement -

1. The best model with high sensitivity and high specificity (low FPR).
2. Low FN (as per our business objective).
3. Precision/Recall Trade-off (Good recall score without compromising on precision score)
4. ROC-AUC Curve (High AUC = better model)
5. In ROC curve, TPR is highest at FPR=0.

XGBoost Classifier: The classification threshold is 0.5 (default). As per the model performance on the test data:

- This model is highly specific as out of the actual negative values 95.30% are correctly predicted to be negative. Out of actual negative values, only 4.7% of the observations are incorrectly predicted.
- Sensitivity (27.31%) may not be the highest as compared to other classification models (lie – Naïve Bayes, Decision Tree, Random Forest, etc.) because precision

score for these models below 50%. We want more of predicted positives to be actual positives.

- Precision score of XGBoost Classifier is 64.33% with highest AUC score of 78.46% amongst all the classification models).
- As per the precision-recall trade-off graph, we can bring the recall around 0.40 with precision between 0.55 and 0.60.
- ROC Graph of a best model has high sensitivity and low FPR (or high specificity) value. Better trade-off between TPR & FPR would be at FPR ~ (0.3 to 0.4) which implies., specificity ~ (0.6 to 0.7) and TRP ranging between 0.7 to 0.8.

MODEL IMPROVISATION

- Adjusting the classification threshold to lower end, to increase the FP results in decreasing the Precision and increasing Recall.
- A good mix of observations from both the classes.
- Hyperparameter tuning of the model to get reduced False Negative values.
- Decreasing the classification threshold (below the default) for the Recall score to increase. We need business intervention to decide the threshold value which would further produce more balanced precision score and recall score.

DEPLOYMENT

Deployed the Python Flask web framework App on Heroku (container-based Cloud Platform as a Service.)

Website Link - https://credit-card-lead-predict.herokuapp.com/

VISUALIZATION (using MS POWER BI tool)

credit-card-lead-ge
neration-dashboard