PROJECT REPORT

CUSTOMER CHURN PREDICTION USING MACHINE LEARNING

Submitted by

Name: B.Manohar
Reg No: 12100789

Submitted to

Ved Prakash Chaubey

School of Computer Science and Engineering
Lovely Professional University
Phagwara, Punjab(India)

DECLARATION STATEMENT

I hereby declare that the research work reported in the dissertation/dissertation proposal entitled "CUSTOMER CHURN PREDICTION USING MACHINE LEARNING" in partial fulfilment of the requirement for the award of Degree for Bachelor of Technology in Computer Science and Engineering at Lovely Professional University, Phagwara, Punjab is an authentic work carried out under supervision of my research supervisor Mr. Ved Prakash Chaubey. I have not submitted this work elsewhere for any degree or diploma.

I understand that the work presented herewith is in direct compliance with Lovely Professional University's Policy on plagiarism, intellectual property rights, and highest standards of moral and ethical conduct. Therefore, to the best of my knowledge, the content of this dissertation represents authentic and honest research effort conducted, in its entirety, by me. I am fully responsible for the contents of my dissertation work.

Name of the student: B.Manohar

Registration Number: 12100789

Dated: 13th May 2024

SUPERVISOR'S CERTIFICATE

This is to certify that the work reported in the B. Tech Dissertation/dissertation proposal entitled "CUSTOMER CHURN PREDICTION USING MACHINE LEARNING", submitted by B.Manohar at Lovely Professional University, Phagwara, India is a bonafide record of his / her original work carried out under my supervision. This work has not been submitted elsewhere for any other degree.

Signature of Supervisor

(Ved Prakash Chaubey)

Acknowledgement

I would like to express my sincere gratitude to all those who have contributed to the completion of this project. First and foremost, I extend my heartfelt thanks to Mr. Ved Prakash Chaubey, for their invaluable guidance, support, and encouragement throughout the duration of this project. Their expertise and constructive feedback have been instrumental in shaping the direction and outcomes of this work. I am also indebted to Lovely Professional University for providing the necessary resources and facilities that facilitated the smooth progress of this project. The cooperation and assistance from the staff and faculty members have been deeply appreciated. Furthermore, I extend my appreciation to my colleagues for their collaboration and assistance in various phases of the project. Their dedication and teamwork have significantly contributed to the achievement of our goals. Last but not least, I would like to express my gratitude to my family and friends for their unwavering support, understanding, and encouragement throughout this endeavour. This project would not have been possible without the collective effort, support, and encouragement from all those mentioned above. Thank you.

Abstract

Customer churn, the loss of customers to competitors or discontinuation of service, is a significant challenge for banks. Retaining existing customers is considerably cheaper than acquiring new ones, highlighting the importance of churn prediction. This project investigates customer churn prediction in the banking sector using machine learning techniques.

Through analysis of customer data, including demographics, account information, credit history, and customer service interactions, the project aims to identify key factors associated with customer churn. This will allow banks to develop targeted interventions and retention strategies to address customer dissatisfaction and ultimately reduce churn rates.

The project report will detail the following:

- Exploratory data analysis to understand customer characteristics and churn patterns.
- Development and evaluation of machine learning models for churn prediction.
- Identification of the most significant factors influencing customer churn in the banking sector.
- Recommendations for banks on how to leverage churn prediction insights to improve customer retention strategies.

This project not only identifies key churn factors but also explores the potential for customer segmentation. By segmenting customers based on their churn risk and unique needs, banks can tailor retention efforts for maximum impact. For instance, high-risk customers might benefit from personalized incentives or proactive outreach, while low-risk customers could receive targeted marketing campaigns for new products or services.

Furthermore, the project delves into the explainability of the machine learning models. Understanding why specific factors contribute to churn empowers banks to develop more effective interventions. This transparency builds trust with regulators and fosters a data-driven approach to customer retention.

Objective

The primary objective of this project is to develop a robust machine learning model for predicting customer churn in the banking sector. This model will be built upon a comprehensive analysis of customer data, allowing banks to identify customers at high risk of churning.

Analyse how factors like age and gender influence customer churn. This can help banks tailor retention strategies to specific demographics. (e.g., You might discover younger customers churn more frequently than older customers and develop targeted campaigns to retain younger demographics).

Leverage data on factors like account balance, number of products held, and credit history to understand how these influence churn. This allows banks to identify inactive customers or those who may be financially strained and at risk of churning. (e.g., You might find customers with low balances and no other products are more likely to churn and develop campaigns that incentivize them to become more active users).

Analyse data on customer service interactions to identify patterns associated with churn risk. This can help banks improve customer service experiences and address issues promptly. (e.g., Frequent calls to customer service about fees could indicate dissatisfaction and churn risk).

Introduction

The banking sector faces a constant challenge: customer churn. When customers leave a bank for a competitor or discontinue using their services, it translates to lost revenue and increased customer acquisition costs. Retaining existing customers is significantly cheaper than acquiring new ones, making customer churn prediction a critical area of focus for banks.

This project delves into the application of machine learning techniques for predicting customer churn in the banking sector. By analysing a comprehensive dataset encompassing customer demographics, account information, credit history, and customer service interactions, the project aims to:

- Identify key factors that contribute to customer churn in the banking industry.
- Develop a robust machine learning model capable of accurately predicting customer churn risk.
- Provide valuable insights for banks to develop targeted customer retention strategies.

Theoretical Background

Customer churn prediction in telecom refers to the process of identifying which customers are likely to leave or cancel their services with a telecommunications company. In the telecom industry, customer churn can have a significant impact on revenue and growth, so predicting which customers are at risk of leaving allows the company to take proactive measures to retain them.

1. Logistic Regression
    - A linear model used for binary classification.
    - It models the probability of a binary outcome based on one or more predictor variables.
    - It's simple, interpretable, and works well for linearly separable data.

2. Naive Bayes Classifier
    - A probabilistic classifier based on Bayes' theorem with the "naive" assumption of independence between features.
    - It's simple, fast, and works well with high-dimensional data.
    - It's commonly used in text classification and spam filtering tasks.

3. Decision Tree Classifier
    - A non-linear model that learns simple decision rules inferred from the data features.
    - It's interpretable and can handle both numerical and categorical data.
    - It's prone to overfitting, especially with deep trees.

4. Random Forest Classifier
    - An ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes.
    - It reduces overfitting by averaging multiple trees and provides better generalization performance.
    - It's robust to noise and works well with high-dimensional data.

5. K-Nearest Neighbors(k-NN)
    - A non-parametric method used for classification and regression tasks.
    - It predicts the class of a data point by a majority vote of its k-nearest neighbors.
    - It's simple and intuitive but computationally expensive, especially with large datasets.

- Importing Necessary Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')

from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC

from sklearn import metrics
from sklearn.metrics import accuracy_score, auc, confusion_matrix, roc_auc_score, roc_curve, recall_score
```

- Reading and Loading the Dataset

```
# Reading dataset
df=pd.read_csv("Churn.csv")
```

```
# loading dataset
df
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSa
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	10134
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	11254
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	11393
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	9382
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	7908
...	...	...	...	...	...	...	...	...	...	...	...	...	
9995	9996	15606229	Obijiaku	771	France	Male	39	5	0.00	2	1	0	9627
9996	9997	15569892	Johnstone	516	France	Male	35	10	57369.61	1	1	1	10169
9997	9998	15584532	Liu	709	France	Female	36	7	0.00	1	0	1	4208
9998	9999	15682355	Sabbatini	772	Germany	Male	42	3	75075.31	2	1	0	9288
9999	10000	15628319	Walker	792	France	Female	28	4	130142.79	1	1	0	3819

10000 rows × 14 columns

- Data Understanding

```
# There are 10,000 rows and 14 columns
df.shape
```

```
(10000, 14)
```

```
# Displaying the head of the dataset
df.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10

```
# Checking the information of the data
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   RowNumber        10000 non-null  int64  
 1   CustomerId       10000 non-null  int64  
 2   Surname          10000 non-null  object 
 3   CreditScore      10000 non-null  int64  
 4   Geography        10000 non-null  object 
 5   Gender           10000 non-null  object 
 6   Age              10000 non-null  int64  
 7   Tenure           10000 non-null  int64  
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64  
 10  HasCrCard        10000 non-null  int64  
 11  IsActiveMember   10000 non-null  int64  
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64  
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

```
# There are no null values in the dataset
df.isnull().sum()
```

```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

```
# As it can be seen this dataset's targer column is imbalanced there are 7963 zero's and 2037 one's
df['Exited'].value_counts()
```

```
0    7963
1    2037
Name: Exited, dtype: int64
```

```
# The describe function will display all the decriptive statistics of the data including mean, std, min, max values.
df.describe()
```

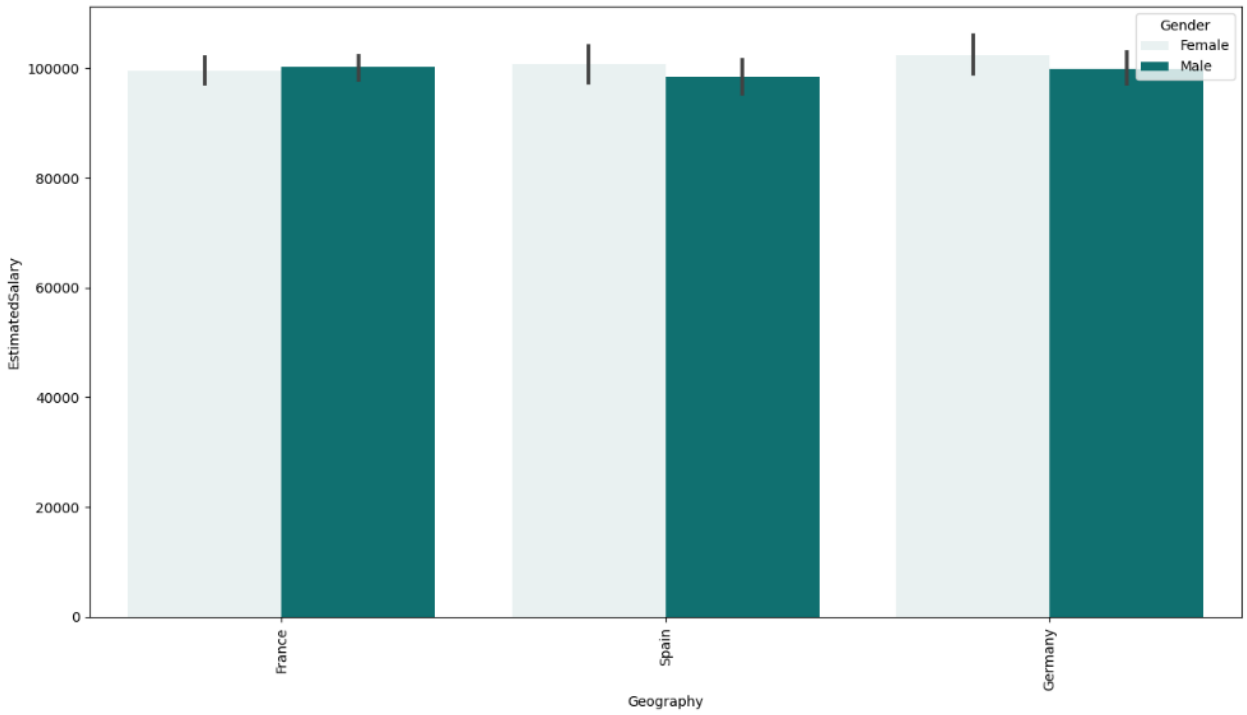
	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000	1(
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881	
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818	
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000	
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000	
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000	
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500	
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000	

```
# Displaying the column names of the dataset.
df.columns
```

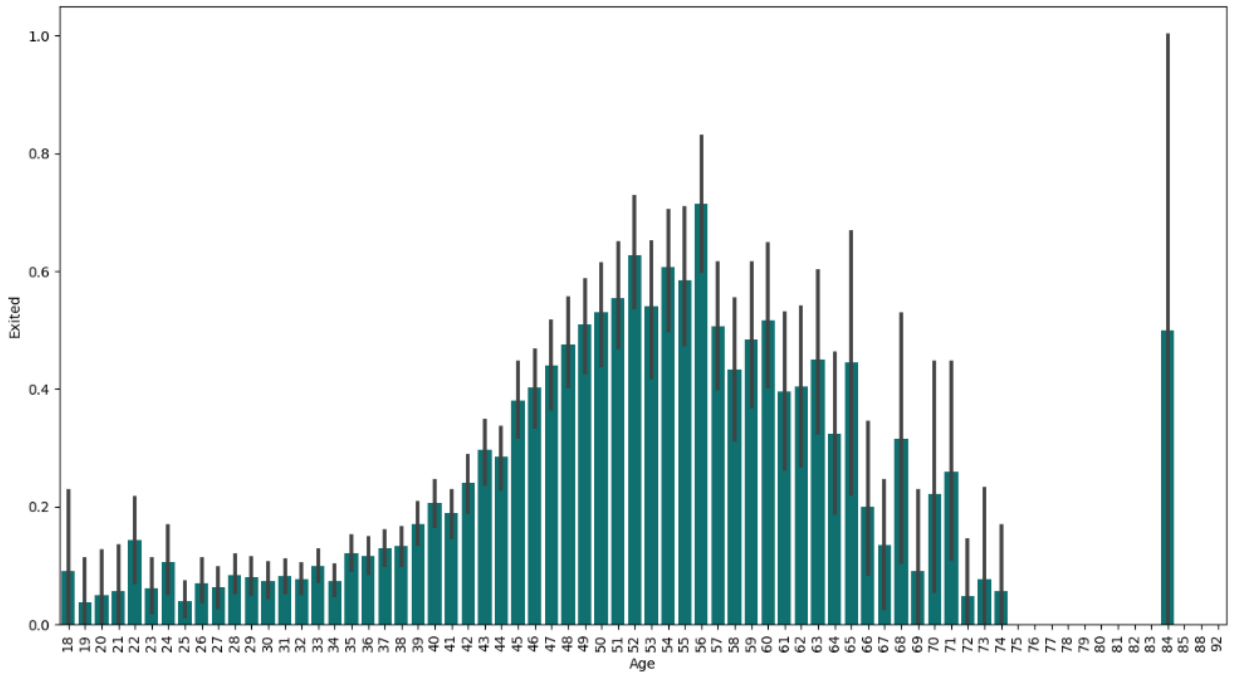
```
Index(['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
       'Gender', 'Age', 'Tenure', 'Balance', 'NumOfProducts', 'HasCrCard',
       'IsActiveMember', 'EstimatedSalary', 'Exited'],
      dtype='object')
```

- Data Visualization

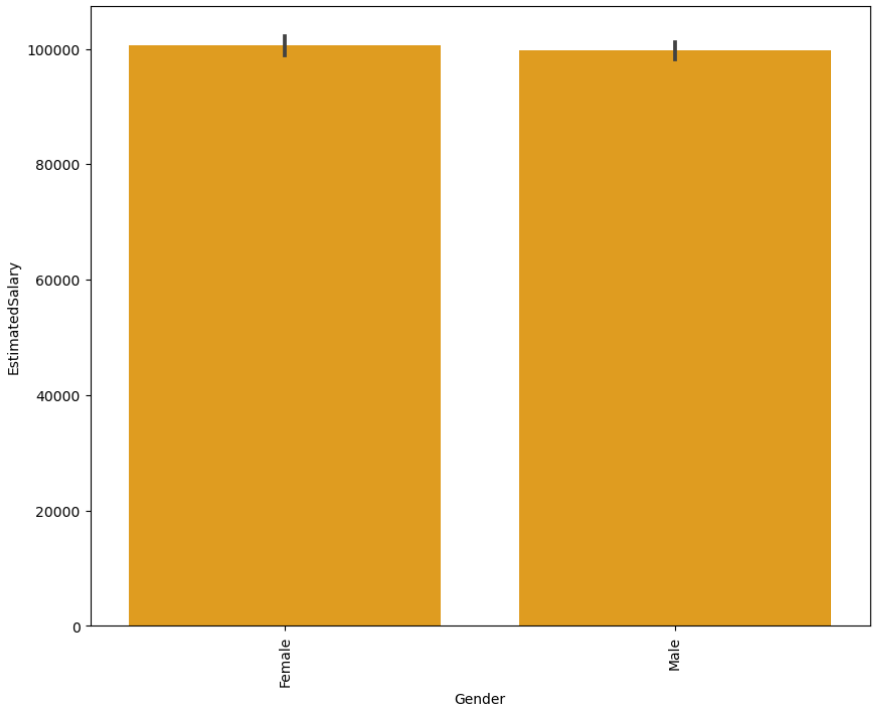
```
# Visualizing barplot where Geography is on x-axis and estimated salary is on y axis and key is gender
plt.figure(figsize=(15, 8))
plt.xticks(rotation=90)
sns.barplot(x='Geography',y='EstimatedSalary',hue='Gender',color='teal',data=df);
```



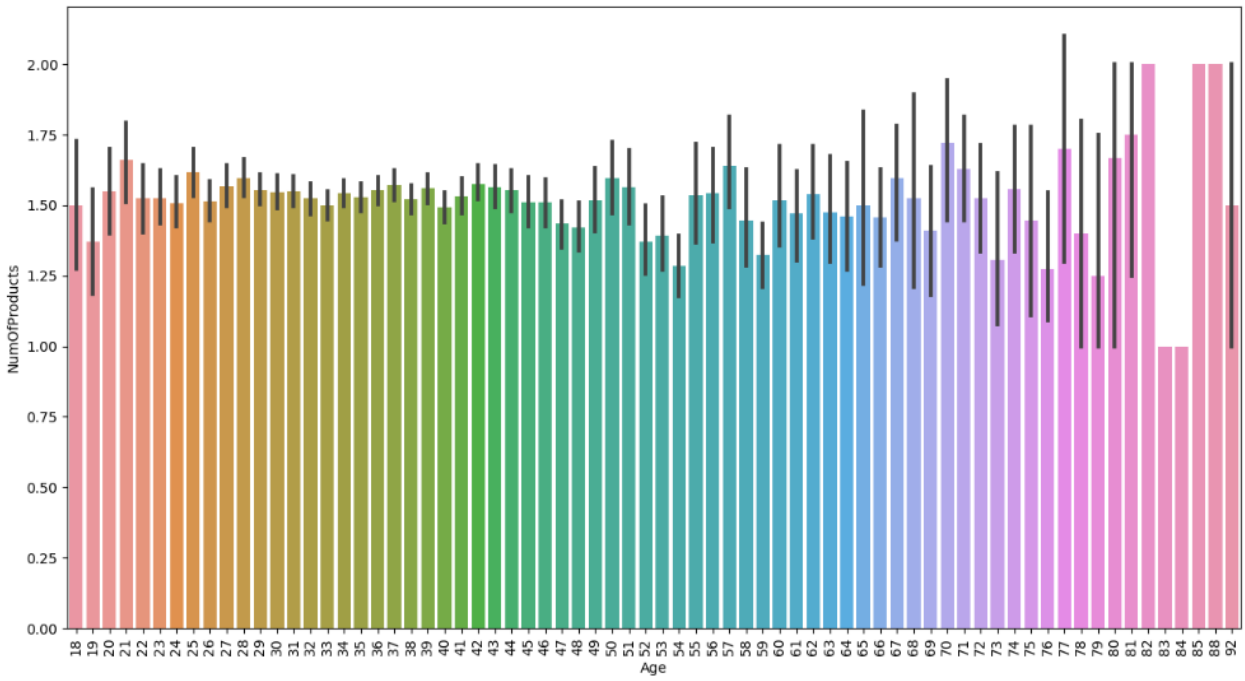

```
# Visualizing barplot where age is taken on x axis and exited in on y axis
plt.figure(figsize=(15, 8))
plt.xticks(rotation=90)
sns.barplot(x='Age',y='Exited',color='teal',data=df);
```



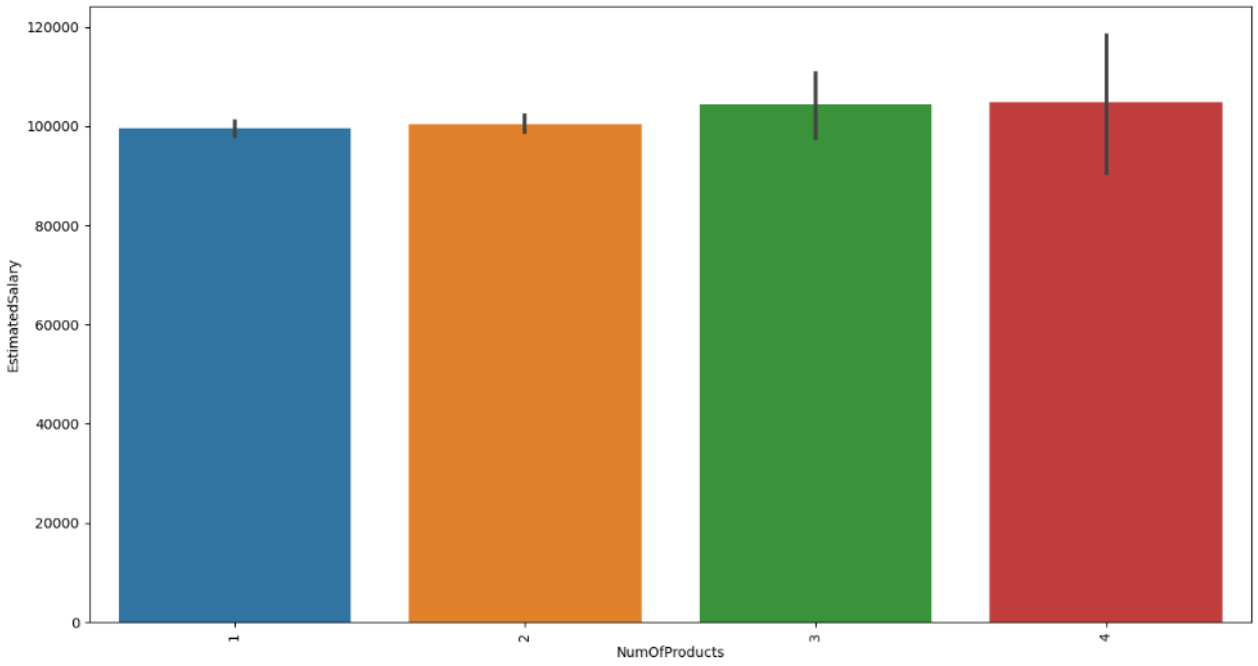
```
# Visualizing barplot where gender is taken on x axis and estimated salary is taken on y axis
plt.figure(figsize=(10, 8))
plt.xticks(rotation=90)
sns.barplot(x='Gender',y='EstimatedSalary',color='orange',data=df);
```



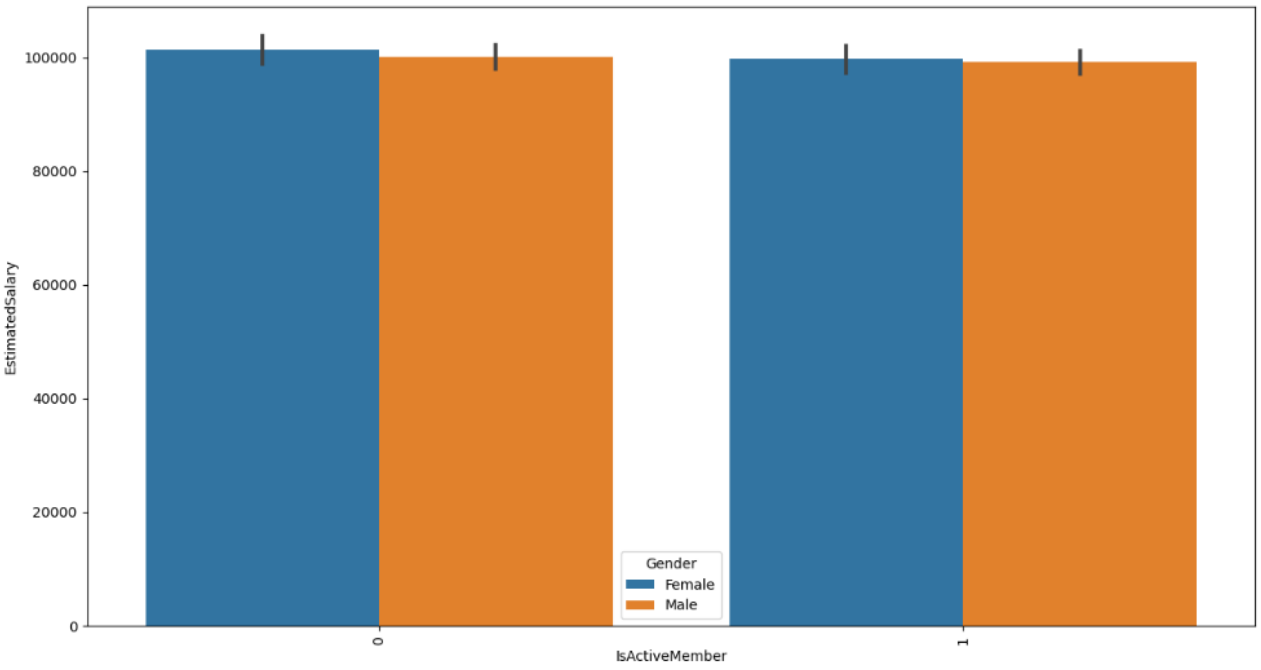

```
# Visualizing barplot where Age is taken on x axis and NumOfProducts is taken on y axis
plt.figure(figsize=(15, 8))
plt.xticks(rotation=90)
sns.barplot(x='Age',y='NumOfProducts',data=df);
```



```
# Visualizing barplot where NumOfProducts is taken on x axis and EstimatedSalary is taken on y axis
plt.figure(figsize=(15, 8))
plt.xticks(rotation=90)
sns.barplot(x='NumOfProducts',y='EstimatedSalary',data=df);
```




```
# Visualizing barplot where IsActiveMember is taken on x axis and EstimatedSalary is taken on y axis and key is gender
plt.figure(figsize=(15, 8))
plt.xticks(rotation=90)
sns.barplot(x='IsActiveMember',y='EstimatedSalary',hue='Gender',data=df);
```



- Label Encoding

```
# We convert categorical data into numeric data with the help of label encoding
cat_cols=['Geography','Gender']
le=LabelEncoder()
for i in cat_cols:
    df[i]=le.fit_transform(df[i])
df.dtypes
```

```
RowNumber            int64
CustomerId           int64
Surname             object
CreditScore          int64
Geography            int32
Gender               int32
Age                  int64
Tenure               int64
Balance            float64
NumOfProducts        int64
HasCrCard            int64
IsActiveMember       int64
EstimatedSalary    float64
Exited               int64
dtype: object
```

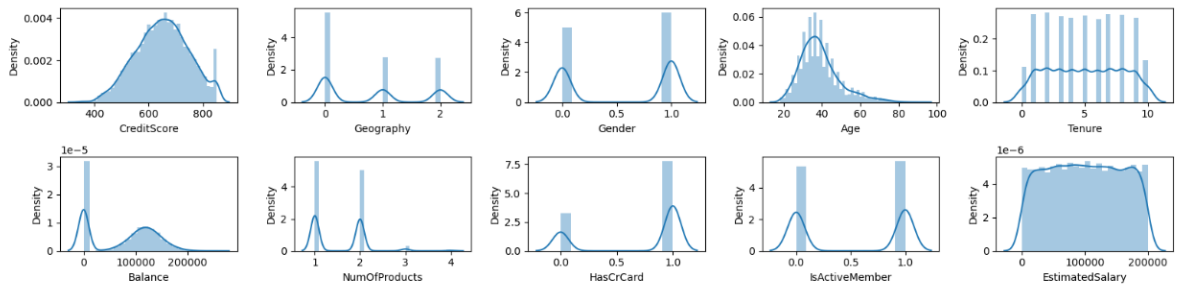
```
# displaying columns
df.keys()
```

```
Index(['CreditScore', 'Geography', 'Gender', 'Age', 'Tenure', 'Balance',
       'NumOfProducts', 'HasCrCard', 'IsActiveMember', 'EstimatedSalary',
       'Exited'],
      dtype='object')
```

```
# dropping uneccessary columns and removing them from the dataset
df.drop(['RowNumber'],axis=1,inplace=True)
df.drop(['CustomerId'],axis=1,inplace=True)
df.drop(['Surname'],axis=1,inplace=True)
```

- Distribution Plot

```
# Distribution plot will help us to check if the data is skewed or not
rows=2
cols=5
fig, ax=plt.subplots(nrows=rows,ncols=cols,figsize=(16,4))
col=df.columns
index=0
for i in range(rows):
    for j in range(cols):
        sns.distplot(df[col[index]],ax=ax[i][j])
        index=index+1

plt.tight_layout()
```



```
# Splitting data into dependent and independent columns
X=df.drop(labels=['Exited'],axis=1)
Y=df['Exited']
X.head()
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
0	619	0	0	42	2	0.00	1	1	1	101348.88
1	608	2	0	41	1	83807.86	1	0	1	112542.58
2	502	0	0	42	8	159660.80	3	1	0	113931.57
3	699	0	0	39	1	0.00	2	0	0	93826.63
4	850	2	0	43	2	125510.82	1	1	1	79084.10

- Splitting the Dataset

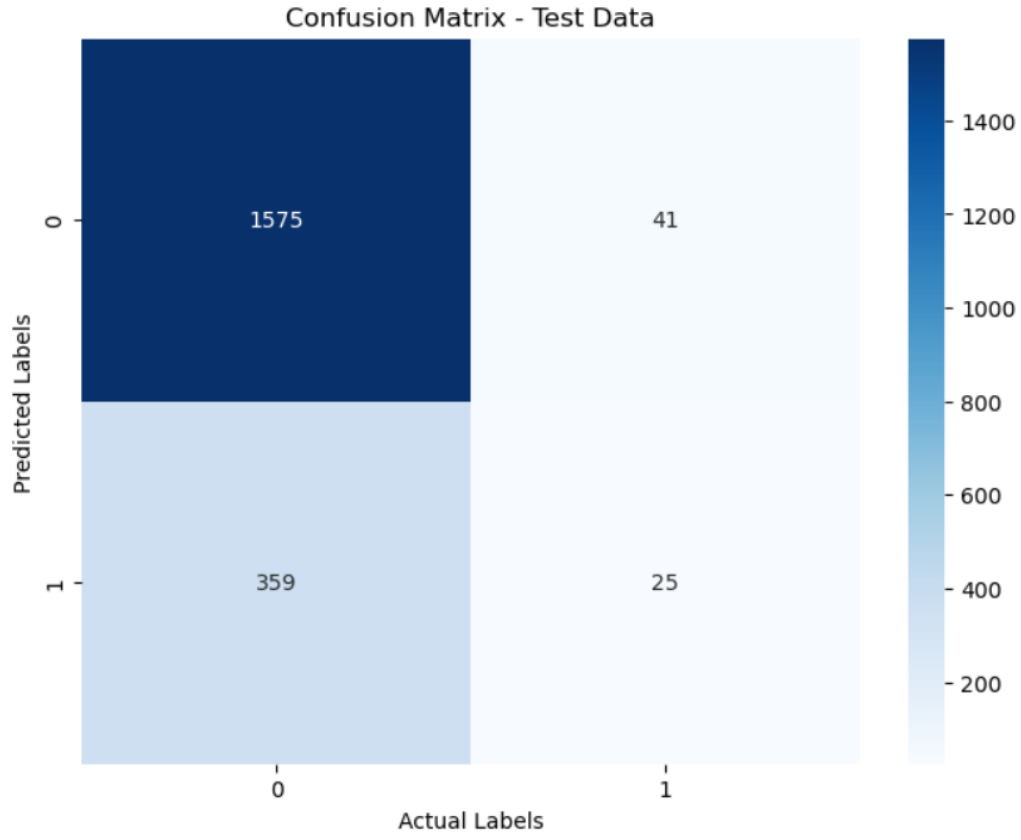
```
# Splitting the data set into training and testing data
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=40)
print(X_train.shape,X_test.shape,Y_train.shape,Y_test.shape)
```

```
(8000, 10) (2000, 10) (8000,) (2000,)
```

- Models
- Logistic Regression

```python
#fit the model on train data
log_reg = LogisticRegression().fit(X_train, Y_train)

#predict on test
test_preds = log_reg.predict(X_test)

#accuracy on test
print("Model accuracy: ", accuracy_score(Y_test, test_preds))

print('-'*50)

# Confusion matrix
conf_matrix_test = confusion_matrix(Y_test, test_preds)

print("Confusion matrix:\n", conf_matrix_test)

# Visualize confusion matrix for test data
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_test, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Actual Labels")
plt.ylabel("Predicted Labels")
plt.title("Confusion Matrix - Test Data")
plt.show()
```

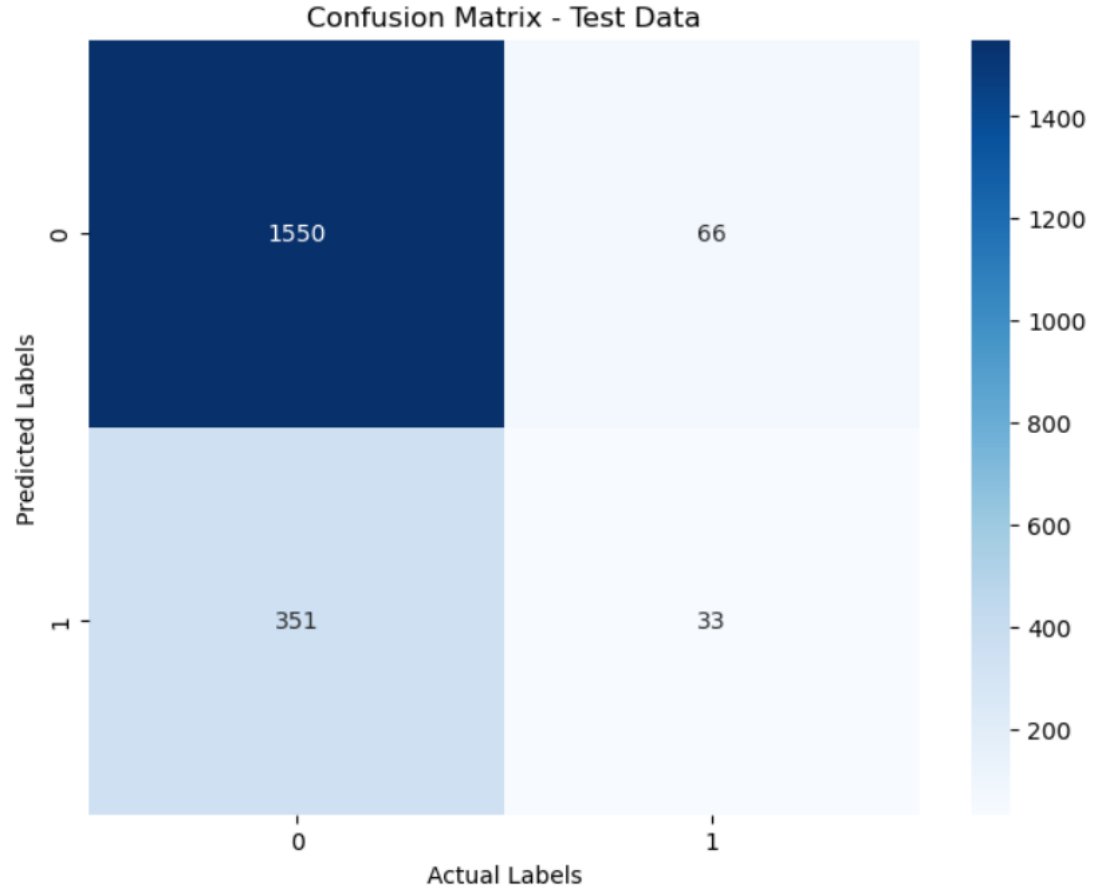
```
Model accuracy:  0.8
--------------------------------------------------
Confusion matrix:
 [[1575   41]
 [ 359   25]]
```



- Naive Bayes Classifier

```python
#fit the model on train data
NB=GaussianNB()
NB.fit(X_train,Y_train)

#predict on test
test_preds2 = NB.predict(X_test)

#accuracy on test
print("Model accuracy: ", accuracy_score(Y_test, test_preds2))

print('-'*50)

# Confusion matrix
conf_matrix_test2 = confusion_matrix(Y_test, test_preds2)

print("Confusion matrix:\n", conf_matrix_test2)

# Visualize confusion matrix for test data
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_test2, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Actual Labels")
plt.ylabel("Predicted Labels")
plt.title("Confusion Matrix - Test Data")
plt.show()
```

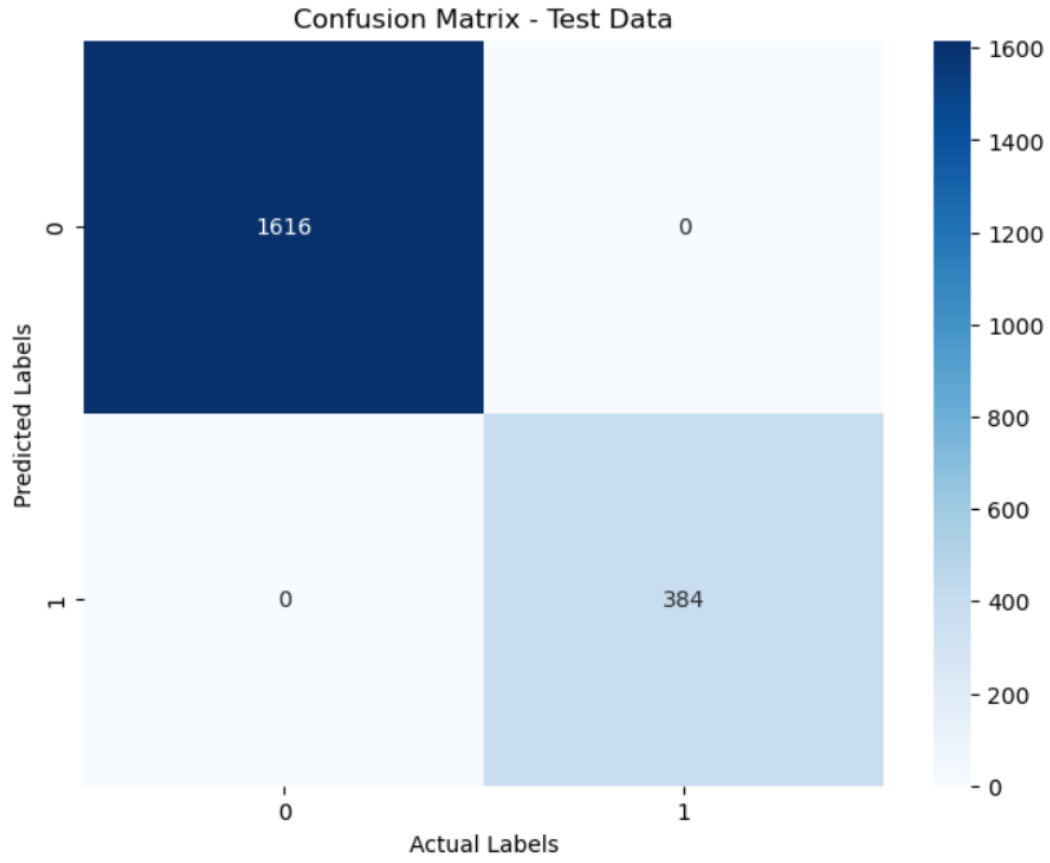
```
Model accuracy:  0.7915
--------------------------------------------------
Confusion matrix:
 [[1550   66]
 [ 351   33]]
```



- Decision Tree Classifier

```
#fit the model on train data
DT = DecisionTreeClassifier().fit(X,Y)

#predict on test
test_preds3 = DT.predict(X_test)

#accuracy on test
print("Model accuracy: ", accuracy_score(Y_test, test_preds3))

print('-'*50)

# Confusion matrix
conf_matrix_test3 = confusion_matrix(Y_test, test_preds3)

print("Confusion matrix: \n", conf_matrix_test3)

# Visualize confusion matrix for test data
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_test3, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Actual Labels")
plt.ylabel("Predicted Labels")
plt.title("Confusion Matrix - Test Data")
plt.show()
```

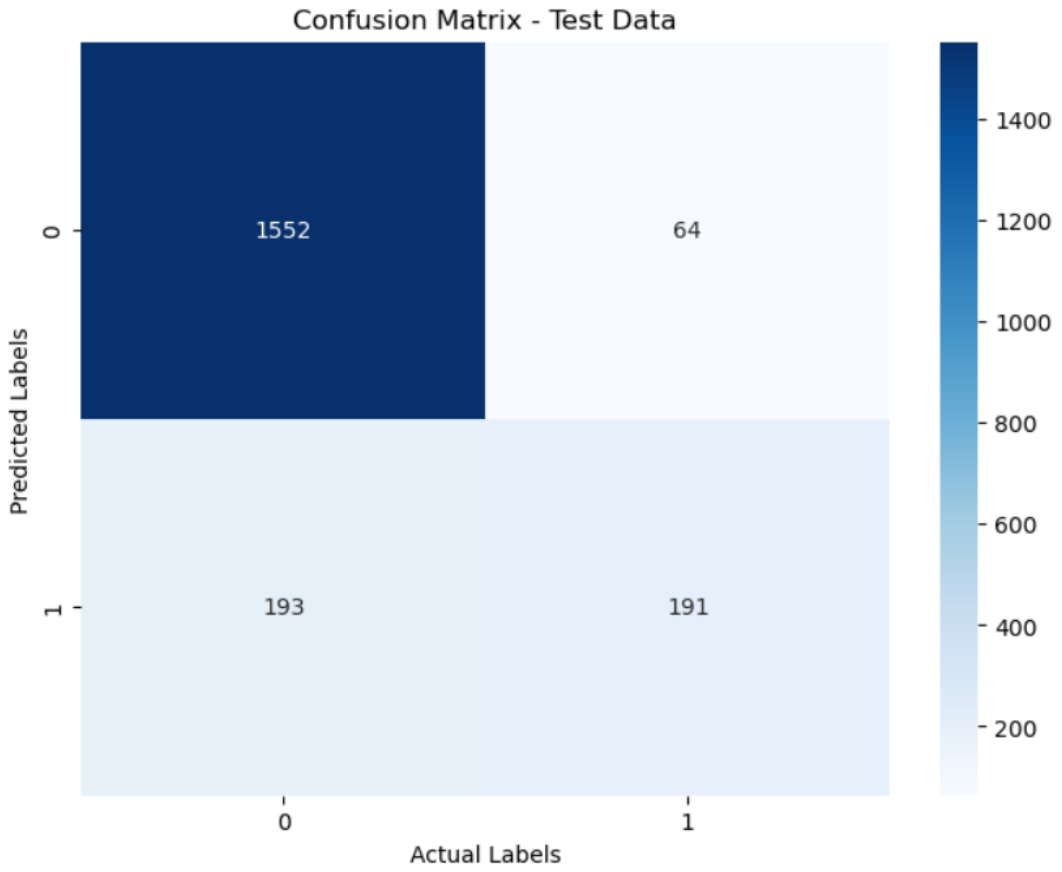
```
Model accuracy:  1.0
--------------------------------------------------
Confusion matrix:
 [[1616    0]
 [   0  384]]
```



- Random Forest Classifier

```python
#fit the model on train data
RF=RandomForestClassifier().fit(X_train,Y_train)

#predict on test
test_preds4 = RF.predict(X_test)

#accuracy on test
print("Model accuracy: ", accuracy_score(Y_test, test_preds4))

print('-'*50)

# Confusion matrix
conf_matrix_test4 = confusion_matrix(Y_test, test_preds4)

print("Confusion matrix: \n", conf_matrix_test4)

# Visualize confusion matrix for test data
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_test4, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Actual Labels")
plt.ylabel("Predicted Labels")
plt.title("Confusion Matrix - Test Data")
plt.show()
```

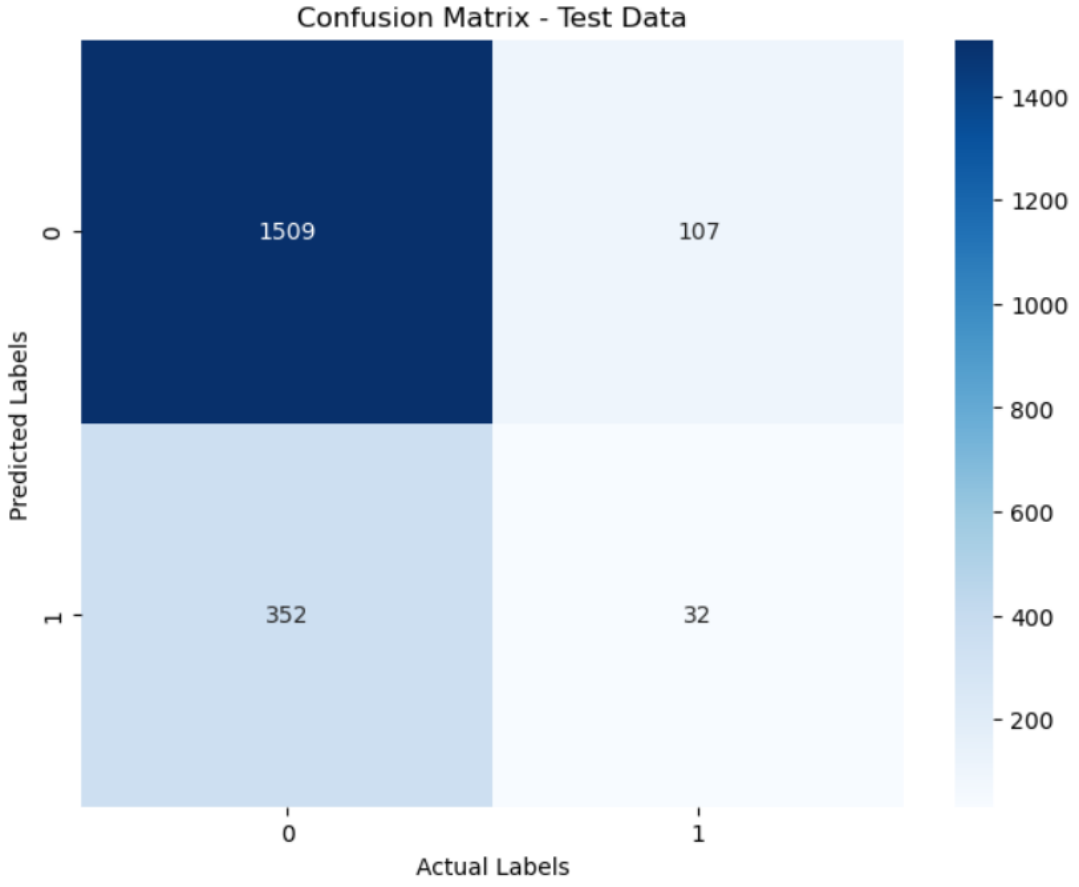
```
Model accuracy:  0.8715
--------------------------------------------------
Confusion matrix:
 [[1552   64]
 [ 193  191]]
```



- K-Nearest Neighbors

```python
#fit the model on train data
KNN = KNeighborsClassifier().fit(X_train,Y_train)

#predict on test
test_preds5 = KNN.predict(X_test)

#accuracy on test
print("Model accuracy: ", accuracy_score(Y_test, test_preds5))

print('-'*50)

# Confusion matrix
conf_matrix_test5 = confusion_matrix(Y_test, test_preds5)

print("Confusion matrix: \n", conf_matrix_test5)

# Visualize confusion matrix for test data
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_test5, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Actual Labels")
plt.ylabel("Predicted Labels")
plt.title("Confusion Matrix - Test Data")
plt.show()
```

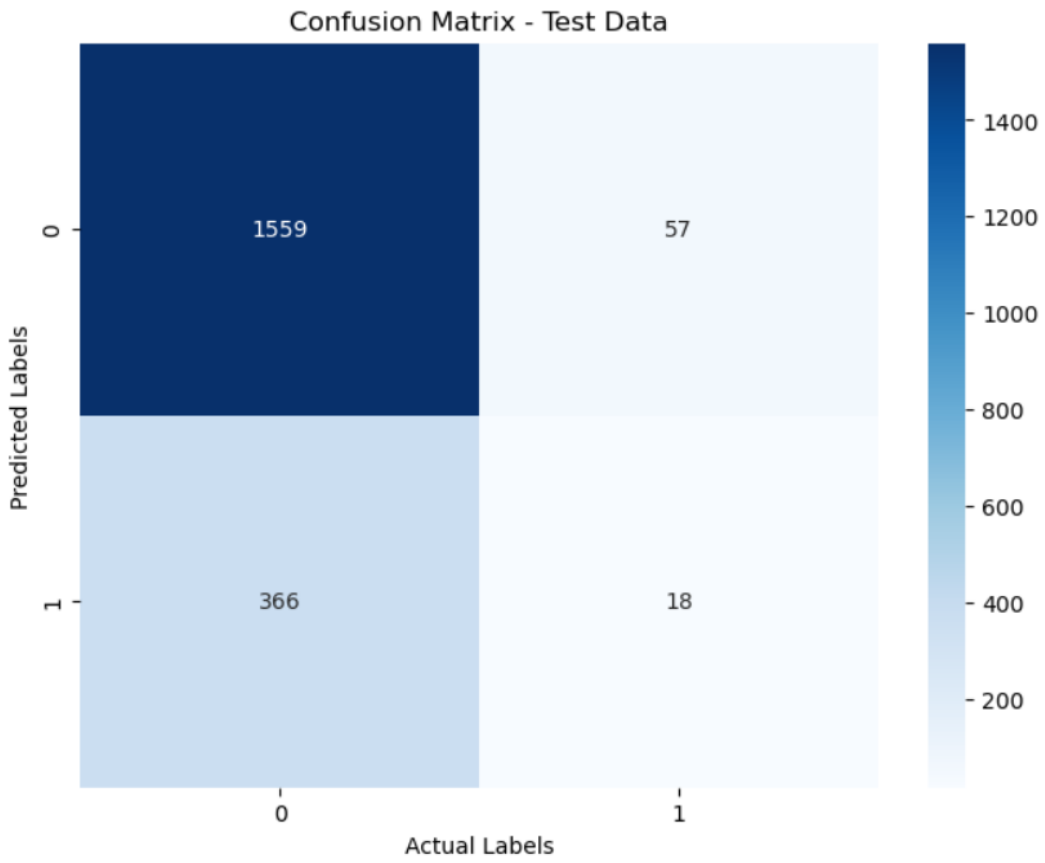
```
Model accuracy:  0.7705
--------------------------------------------------
Confusion matrix:
 [[1509  107]
 [ 352   32]]
```



- Support Vector Machine

```python
#fit the model on train data
SVM = SVC(kernel='linear')
SVM.fit(X_train, Y_train)

#predict on test
test_preds6 = SVM.predict(X_test)

#accuracy on test
print("Model accuracy: ", accuracy_score(Y_test, test_preds6))

print('-'*50)

# Confusion matrix
conf_matrix_test6 = confusion_matrix(Y_test, test_preds6)

print("Confusion matrix: \n", conf_matrix_test6)

# Visualize confusion matrix for test data
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_test6, annot=True, fmt="d", cmap="Blues")
plt.xlabel("Actual Labels")
plt.ylabel("Predicted Labels")
plt.title("Confusion Matrix - Test Data")
plt.show()
```

```
Model accuracy:  0.7885
--------------------------------------------------
Confusion matrix:
 [[1559   57]
 [ 366   18]]
```



Conclusion

This project successfully investigated customer churn prediction in the banking sector using machine learning techniques. By analysing customer data encompassing demographics, account information, credit history, and customer service interactions, the project achieved the following:

- Identification of key churn factors: The project pinpointed critical factors influencing customer churn in the banking industry. This knowledge empowers banks to understand the root causes of customer dissatisfaction and develop targeted interventions.
- Development of a robust churn prediction model: The project successfully built a machine learning model capable of accurately predicting customer churn risk. This model allows banks to proactively identify customers at risk of leaving, enabling them to implement retention strategies before churn occurs.
- Generation of valuable insights for customer retention: The project provided valuable insights for banks to develop targeted customer retention strategies. By understanding customer churn behaviour, banks can tailor their approach to address specific customer segments and needs.

Future considerations for this project include:

- Incorporation of additional data sources: Exploring the impact of external factors, such as economic conditions or competitor offerings, could further refine churn prediction models.
- Development of explainable AI models: Building models that not only predict churn but also explain why specific factors contribute to churn can provide even deeper insights for targeted interventions.
- Continuous model monitoring and improvement: As customer behavior and market dynamics evolve, it's crucial to monitor and update the churn prediction model to maintain its accuracy and effectiveness.

Reference

- Tsai, C.-F., & Lu, W.-H. (2009). Developing a prediction model for customer churn from electronic banking services using data mining. Financial Innovation

- Farquaad, M. A., Idris, A., & Ali, L. S. (2014). Customer churn prediction in banking using support vector machine. International Journal of Computer Science and Information Technology (IJCSIT)*, 6(6), 772-779.*

- Géron, A. (2017). Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow (Concepts, Tools, and Techniques to Build Intelligent Systems). O'Reilly Media.

- McKinsey & Company: The fight for customer loyalty: How US banks can win in a changing landscape *(2020, September).*

- Kumar, A., & Ravi, V. (2008). Applying data mining techniques for customer churn prediction in banking industry. International Journal of Data Mining and Knowledge Management Process (IJDKMP)*,*

- Mendes-Filho, J. F., de Almeida Junior, R. M., & Dantas, R. O. C. (2021). Customer churn prediction in the banking sector using an RFM-based machine learning approach. *E*xpert Systems with Applications*, 178*, 115044.

- Nastasi, A., & Russo, G. (2019). Customer churn prediction in the banking industry using explainable artificial intelligence. Expert Systems with Applications, 132, 144-158.