Predictive Analytics on Customer Churn in Banking Sector

Industry Review

Customer churn is a common problem across businesses in many sectors. If you want to grow as a company, you have to invest in acquiring new clients. Every time a client leaves, it represents a significant investment lost. Both time and effort need to be channeled into replacing them. Being able to predict when a client is likely to leave, and offer them incentives to stay, can offer huge savings to a business.

As a result, understanding what keeps customers engaged is extremely valuable knowledge, as it can help you to develop your retention strategies, and to roll out operational practices aimed at keeping customers from walking out the door.

Current practices and background research

To succeed at retaining customers who are ready to abandon your business, Marketers & Customer Success experts must be able to predict in advance which customers are going to churn and set up a plan of marketing actions that will have the greatest retention impact on each customer. As of now, companies are trying to be reactive to churn situations instead of having a proactive approach of retaining customers.

Collecting comprehensive feedback about the customer's experience periodically is a process that is followed now by banks. Customer feedback surveys are the go-to tool of every business to understand their customer needs and sentiments. The advantages of surveys lie in their simplicity, ease of creation, quick feedback collection

However, these suffer from poor response rates (typically 5%-30%). Hence, they leave the business clueless about its customers. The issue is exacerbated by small surveys that don't provide as much specific insight, and their increased frequency — once per interaction, for instance — can decrease the response rate even further.Survey/feedback can be expensive and infrequent. When you target a random audience to conduct your surveys, you might miss out on your most valuable customers.
Many customers are concerned about giving their personal information like phone number and location via online surveys. Because of this they either end up not completing a survey, skipping questions, and abandoning a survey altogether.

Repeated surveying can become tiresome for your customers to answer. Either they might stop responding to your survey or give half-hearted answers. Unreliable responses do not help you to make good decisions for your business.

Another solution that banks are using currently is to use analytics to study parameters such as the volume of transactions and to predict attrition reactively because such parameters indicate that the process of attrition has already begun.

Customer retention is one of the primary KPI for companies in the banking sector where the Competition is tough since the customer is free to choose from plenty of providers. One bad experience and customer may just move to the competitor resulting in customer churn.

Literature Survey - Publications, Application, past and undergoing research

Churn Prediction for Savings Bank Customers: A Maching Learning Approach || Prashant Verma , Analytics Department, Global IT Centre, State Bank of India, Navi Mumbai, India

Prashant Verma explores churn prediction for savings account customer, based on various statistical & machine learning models and uses under-sampling, to improve the predictive power of these models, considering the imbalance characteristics of customer churn rate in the data. Model Accuracy, Area under the curve (AUC), Gini coefficient, and Receiver Operating Characteristics (ROC) curve have been utilized for model comparison. The results show that out of the various machine learning models, Random Forest which predicts the churn with 78% accuracy, is the most powerful model for the scenario. Customer vintage, customer's age, average balance, occupation code, population type, average debit amount, and an average number of transactions are found to be the variables with high predictive power for the churn prediction model. The article suggests a customized campaign to be initiated by commercial banks to avoid SB customer churn. Hence, by giving better customer satisfaction and experience, the commercial banks can limit the customer churn and maintain their deposits.

Customer churn prediction - a case study in retail banking ||Teemu Mutanen, S. Nousiainen, J. Ahola

This work focuses on one of the central topics in customer relationship management (CRM): transfer of valuable customers to a competitor. Customer retention rate has a strong impact on customer lifetime value, and understanding the true value of a possible customer churn will help the company in its customer relationship management. Customer value analysis along with customer churn predictions will help marketing programs target more specific groups of customers. We predict customer churn with logistic regression techniques and analyze the churning and non churning customers by using data from a consumer retail banking company. The result of the case study shows that using conventional statistical methods to identify possible churners can be successful.

Customer Churn Analysis in Banking Sector Saw Thazin Khine , Win Win Myo University of Information Technology, Faculty of Computing, 11051, Yangon, Myanm

In this research, churn prediction model of classifying bank customer is built by using the hybrid model of k-means and Support Vector Machine data mining methods on bank customer churn dataset to overcome the instability and limitations of single prediction model and predict churn trend of high value users. The model developed will help banks identify clients who are likely to be churners and develop appropriate marketing actions to retain their valuable clients. And this model also supports information about similar customer group to consider which marketing reactions are to be provided. Thus, due to existing customers are retained, it will provide banks with increased profits and revenues. And also proposed combined model K-means-SVM reduces SV

Customer Churn Analysis In Banking Sector Using Data Mining Techniques || A. O. Oyeniyi & A.B. Adeyemo Department of Computer Science, University of Ibadan Ibadan, Nigeria

This paper presents a data mining model that can be used to predict which customers are most likely to churn (or switch banks). The study used real-life customer records provided by a major Nigerian bank. The raw data was cleaned, pre-processed and then analysed using WEKA, a data mining software tool for knowledge analysis. Simple K-Means was used for the clustering phase while a rule-based algorithm, JRip was used for the rule generation phase. The results obtained showed that the methods used can determine patterns in customer behaviours and help banks to identify likely churners and hence develop customer retention modalities.

Data description

The data set comprises a bank having customers based out of Spain, Germany and France. It contains 14 columns. The first 13 columns are the independent variable, while the last column is the dependent variable that contains a binary value of 1 or 0. Here, 1 refers to the case where the customer left the bank after 6 months, and 0 is the case where the customer didn't leave the bank after 6 months.

The data for the independent variables was collected 6 months before the data for the dependent variable, since the task is to develop a machine learning model that can predict whether a customer will leave the bank after 6 months, depending on the current feature values.

Data Dictionary

Name of the Attribute	Data Type	Description
RowNumber	int64	This column corresponds to the record (row) number and has no effect on the output.
CustomerId	int64	This column contains unique identifier for the customer in the bank's database. It has no effect on customer leaving the bank.
Surname	object	The column consists of the surname of a customer
CreditScore	int64	This column showcases the credit score which is an indicator of a person's creditworthiness, or their ability to repay debt.
Geography	object	This column consists of the Customer's location.
Gender	object	This column comprises of the Gender of the customer.
Age	int64	This represents the age of the customer.

Tenure	int64	This column consists of the number of years that the customer has been a client of the bank. Normally, older clients are more loyal and less likely to leave a bank.
Balance	float64	This column consists of the customer's account balance. It is a very good indicator of customer churn, as people with a higher balance in their accounts are less likely to leave the bank compared to those with lower balances.
NumOfProducts	int64	This column consists of the number of products that a customer has purchased through the bank.
HasCrCard	int64	This column denotes whether or not a customer has a credit card. This column is also relevant, since people with a credit card are less likely to leave the bank.
IsActiveMember	int64	This column denotes whether the customer is an active member or not.
EstimatedSalary	float64	This column shows the data of customer's estimated salary. Similar to balance, people with lower salaries are more likely to leave the bank compared to those with higher salaries.
Exited	int64	This column shows whether or not the customer left the bank.

Problem Statement

Customer churn represents a basic problem within the competitive atmosphere of the banking industry. Given the importance of customers as the most valuable assets of organizations, customer retention seems to be an essential, basic requirement for any organization. Banks are no exception to this rule. The competitive atmosphere within which banking services are provided by different banks increases the necessity of customer retention. According to surveys, a bank can increase its profits by up to 85 % by improving the retention rate by up to 5 %. In addition, customer retention is seen as more important than in the past.

Project Outcome

Across industries, data has become an extremely valuable resource. This is especially true in the financial services sector, where big data has opened up new opportunities, delivering benefits to customers and employees alike. Understanding how banking and big data work in practice requires familiarity with the technologies used to collect, clean and analyze the data sets of information gathered from a variety of channels.

The banking market and consumers who utilize finance products generate an enormous amount of data on a daily basis. Analytics and modelling has changed the way this information is processed, making it possible to identify trends and patterns which can then be used to inform business decisions at scale. While one piece of data is a single data point, multiple pieces of information can create a larger picture that can be used to recognize patterns in customer behavior, purchasing choices and other key insights.

Predictive analytics allows for faster movement and long-term planning to decide what types of products to offer customers and when to offer them. AI, specifically, helps drive this proactive strategy, prevent banking customer churn and promote best actions when it comes to retail.

The key to solve the above limitations lies in extracting early warning signs from the already existing data. Advanced machine learning (ML) and data science (DS) techniques can learn from past customer behavior and external triggers that led to churn and use this learning to predict the future occurrence of a churn -like event.

Using predictive analytics, companies can effectively manage their risks, especially since it can monitor so many diverse forms of data sets at once, be it raw or structured. So it can assess the potential risks involved in any field, be it marketing or be it workforce-related. Most importantly, it can be used to detect the root of past mistakes or bad fiscal phases, and to determine ways to fix loopholes.

With analytics, certain calculated parameters can be put in place to assess high risk decisions better, since there’s always an element of unpredictability when it comes to the market.

With the advent of advanced data science and machine learning techniques, it's now possible for companies to identify potential customers who may cease doing business with them in the near future. In this project you'll see how a bank can predict customer churn based on different customer attributes such as age, gender, geography, and more

Data Preprocessing

In [2]:
```
df = pd.read_csv('Churn_Modelling.csv')
df.head()
```

Out[2]:

	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

In [3]:
```
df.shape
```

Out[3]:
```
(10000, 14)
```

In [6]:
```
df.dtypes
```

Out[6]:
```
RowNumber            int64
CustomerId           int64
Surname             object
CreditScore          int64
Geography           object
Gender              object
Age                  int64
Tenure               int64
Balance            float64
NumOfProducts        int64
HasCrCard            int64
IsActiveMember       int64
EstimatedSalary    float64
Exited               int64
dtype: object
```

Differentiating Categorical and Numerical Variables

In [21]:
```
df_numeric = df.select_dtypes(include = np.number)
df_numeric.columns.to_list()
```

Out[21]:
```
['RowNumber',
 'CustomerId',
 'CreditScore',
 'Age',
 'Tenure',
 'Balance',
 'NumOfProducts',
 'HasCrCard',
 'IsActiveMember',
 'EstimatedSalary',
 'Exited']
```

In [22]:
```
df_cat = df.select_dtypes(include = 'object')
df_cat.columns.to_list()
```

Out[22]:
```
['Surname', 'Geography', 'Gender']
```

Counting Null and Unique values

```
In [23]: df.isnull().sum()
```

```
Out[23]: 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
```

```
In [24]: df.nunique()
```

```
Out[24]: RowNumber          10000
         CustomerId         10000
         Surname             2932
         CreditScore          460
         Geography              3
         Gender                 2
         Age                   70
         Tenure                11
         Balance             6382
         NumOfProducts          4
         HasCrCard              2
         IsActiveMember         2
         EstimatedSalary     9999
         Exited                 2
         dtype: int64
```

Describing Numerical values

```
In [25]: df.describe()
```

Out[25]:

	RowNumber	CustomerId	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	1
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000	
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	

Insights driven from the above analytics

- We have the details of exactly 10000 customers with us
- The minimum age of a customer is 18 and maximum age is 92
- The credit score of the customer ranges from 350 to 850
- Most of the customers are using at least one of the bank's product and a maximum 4 product
- The maximum tenure of any customer with the bank is 10 years

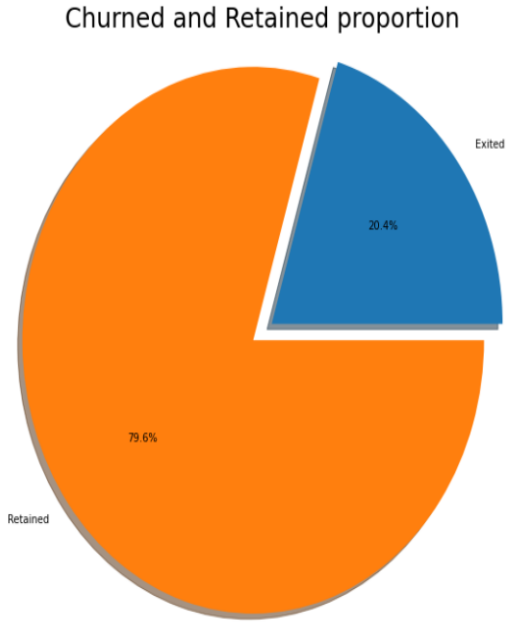
Exploratory Data Analysis

Exploratory Data Analysis (EDA) is the crucial process of using summary statistics and graphical representations to perform preliminary investigations on data in order to uncover patterns, detect anomalies, test hypotheses, and verify assumptions.

This process is a thorough examination meant to uncover the underlying structure of a data set and is important for a company because it exposes trends, patterns, and relationships that are not readily apparent.

Visualization of Percentage of customers Churned vs Retained

```
In [11]: fig, axs = plt.subplots(figsize=(20, 10))
         sizes = [df.Exited[df['Exited']==1].count(), df.Exited[df['Exited']==0].count()]
         axs.pie(sizes, explode=(0, 0.1), labels=['Exited', 'Retained'], autopct='%1.1f%%',shadow=True)
         axs.axis('equal')
         plt.title("Churned and Retained proportion", size = 25)
         plt.show()
```



We are making use of the target column to check the number of customers who have churned/retained. The Target column has only Binary values where 1 indicates that the customer has churned/exited and 0 indicates that the customer has been retained.

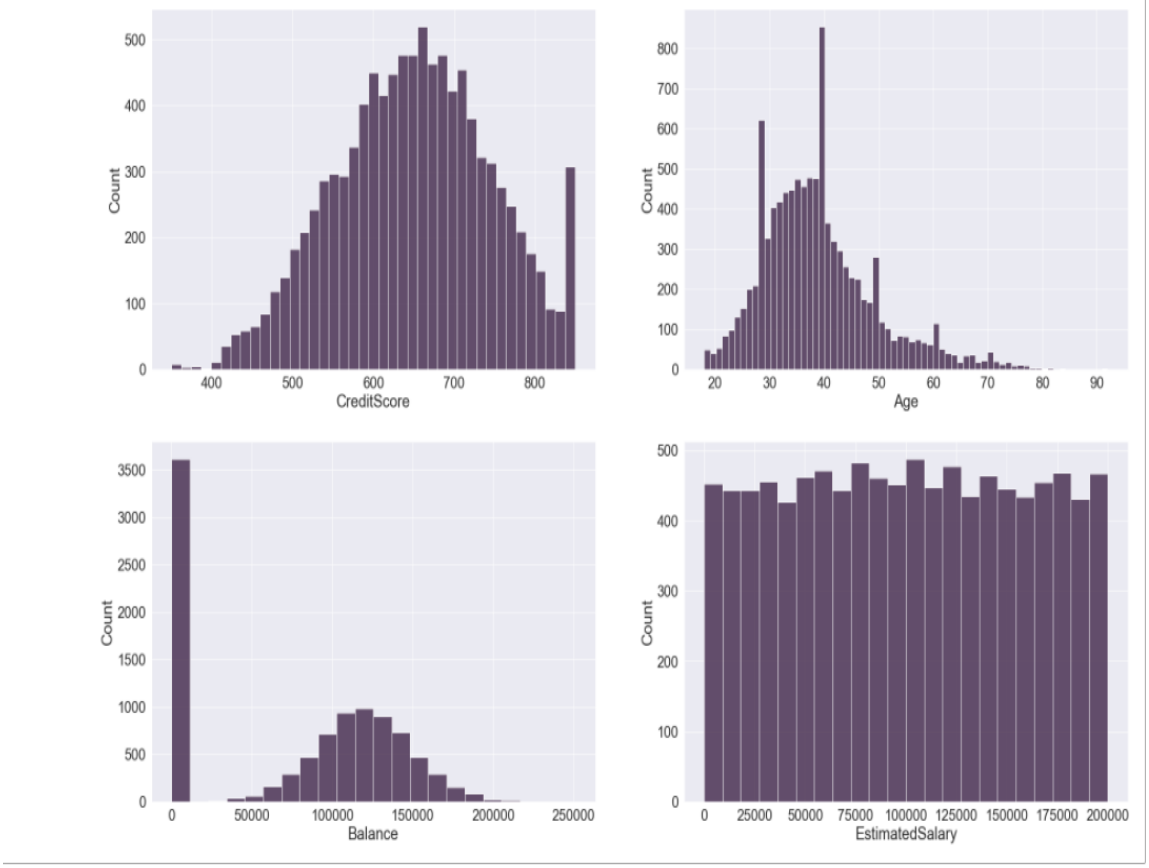
From the above graph we can see that around 20% of customers have churned in the given time period.

Continuous Variable Distribution

In [40]:
```
sns.set(rc={'figure.figsize':(30,20)})
sns.set(font_scale = 2)
fig,axs = plt.subplots(2,2)
sns.set_theme(palette="rocket")
sns.histplot(data = df,x = "CreditScore",ax=axs[0,0])
sns.histplot(data = df,x = "Age",ax=axs[0,1])
sns.histplot(data = df,x = "Balance",ax=axs[1,0])
sns.histplot(data = df,x = "EstimatedSalary",ax=axs[1,1])
```

Out[40]: <AxesSubplot:xlabel='EstimatedSalary', ylabel='Count'>

Out[40]: <AxesSubplot:xlabel='EstimatedSalary', ylabel='Count'>



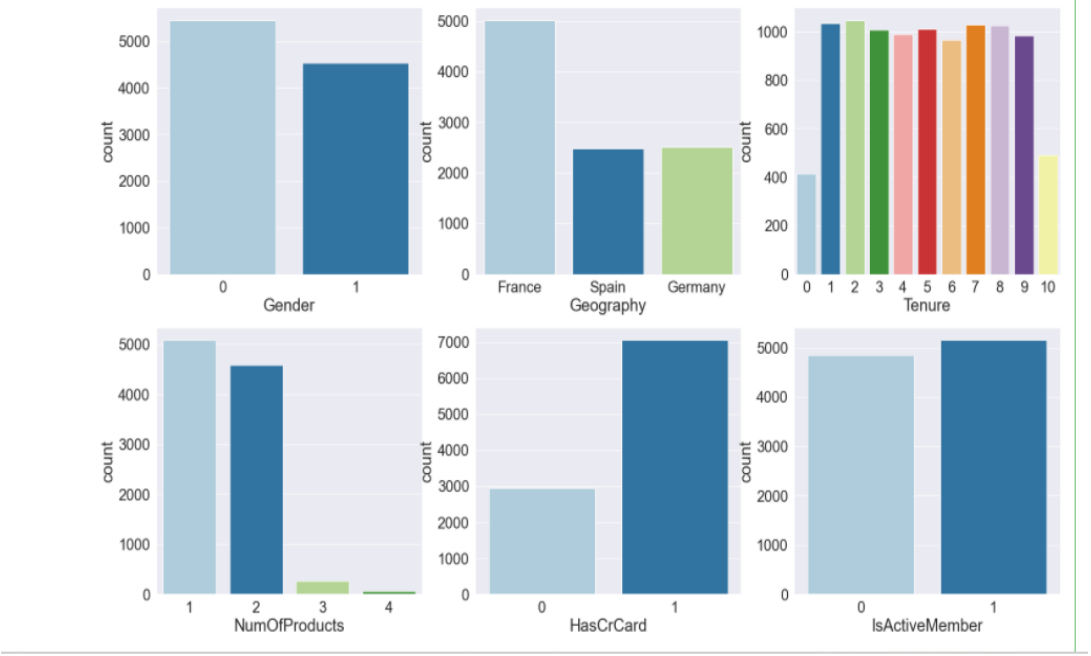
Insights from distribution of continuous columns

- We can see that the majority of our customers are having Credit Score less than 700. However, the ideal credit score is 700 or above. We can interpret that there are more number of non-reliable customers for the bank
- The Age of the customers is skewed towards the right which means that there are few customers whose age is more when compared to mean age of customers
- There is a significant number of younger customers whose age ranges from ranging from 29 to 40 years
- The bank balance of most of the customers are normally distributed. However, approximately 3600 customers have zero bank balance in their account which contributes to one third of the total number of customers.
- The Estimated Salary of the customers have an uniform distribution. We can interpret that there are almost equal number of customers for different estimated salaries.

Categorical Variable Distribution

```
In [48]:
sns.set(rc={'figure.figsize':(25,15)})
sns.set(font_scale = 2)
fig,axs = plt.subplots(2,3)
sns.set_theme(palette="Paired")
sns.countplot(data = df,x = 'Gender',ax=axs[0,0])
sns.countplot(data = df,x = 'Geography',ax=axs[0,1])
sns.countplot(data = df,x = 'Tenure',ax=axs[0,2])
sns.countplot(data = df,x = 'NumOfProducts',ax=axs[1,0])
sns.countplot(data = df,x = 'HasCrCard',ax=axs[1,1])
sns.countplot(data = df,x = 'IsActiveMember',ax=axs[1,2])

Out[48]: <AxesSubplot:xlabel='IsActiveMember', ylabel='count'>
```



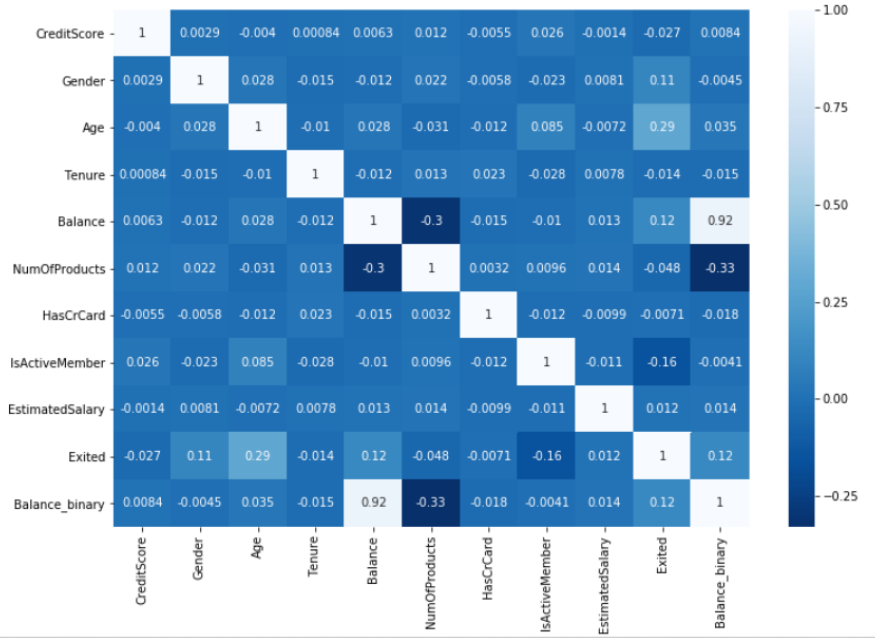
Insights from distribution of categorical columns

- Most of the customers in the bank are Male.
- We have more customers from France and almost the same number of customers from Germany and Spain.
- Majority of our customers have a good tenure with the bank ranging from 1 to 9 years.
- Most of the customers have at least one product purchased from the bank.
- Almost 70% of the customers use credit card.
- The number of customers who are digitally active are almost the same as digitally inactive customers.

Correlation between continuous columns

```
In [30]: df['Gender'].replace({'Male':0, 'Female':1}, inplace=True)
         corr = df.corr()
         plt.figure(figsize=(12,8))
         sns.heatmap(corr, cmap='Blues_r', annot=True)
```

```
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0xdfc71f0>
```

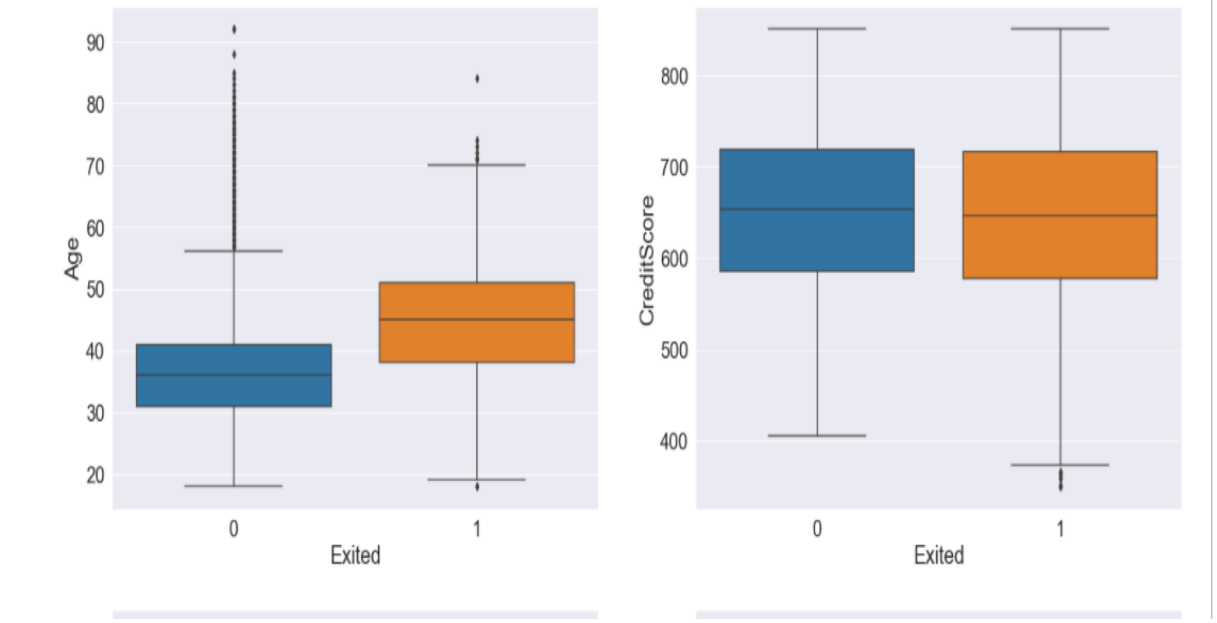


- Age, Balance and “Gender” columns are positively correlated with customer churn (“Exited”)
- There is a negative correlation between being an active member (“IsActiveMember”) and customer churn.
- There is no correlation between independent variables and hence we can conclude that there is no multicollinearity.

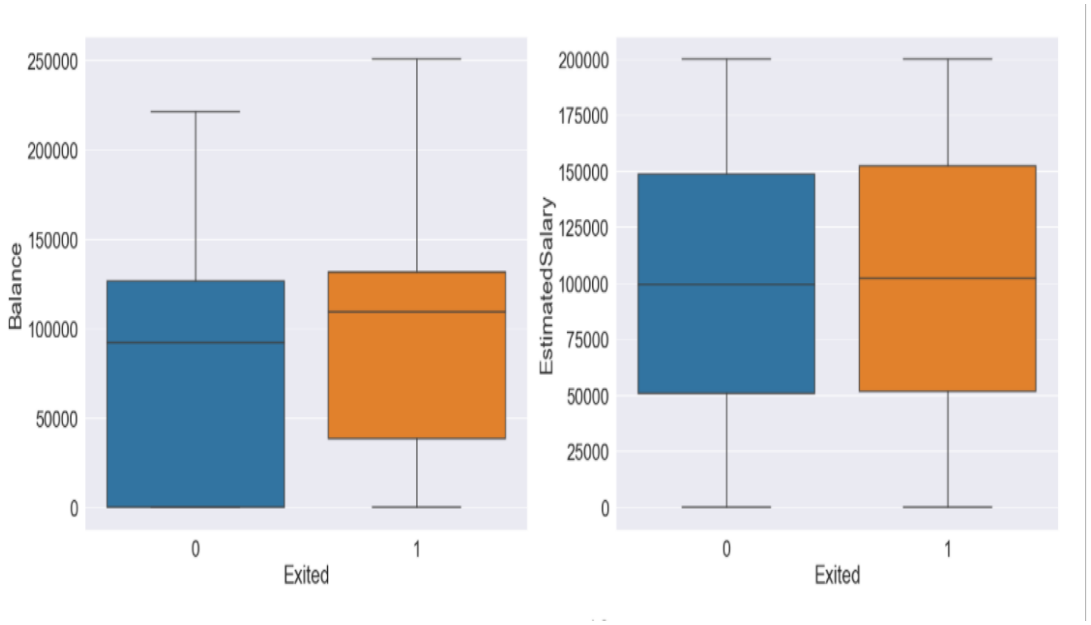
Continuous Variable Exploration with respect to Exit Status

```
In [51]: sns.set(rc={'figure.figsize':(25,30)})
         sns.set(font_scale = 2)
         fig,axs = plt.subplots(3,2)
         sns.set_theme(palette="tab10")
         sns.boxplot(data = df,x = "Exited",y = "Age",ax = axs[0,0])
         sns.boxplot(data = df,x = "Exited",y = "CreditScore",ax = axs[0,1])
         sns.boxplot(data = df,x = "Exited",y = "Balance",ax = axs[1,0])
         sns.boxplot(data = df,x = "Exited",y = "EstimatedSalary",ax = axs[1,1])
         sns.boxplot(data = df,x = "Exited",y = "Tenure",ax=axs[2,0])
```

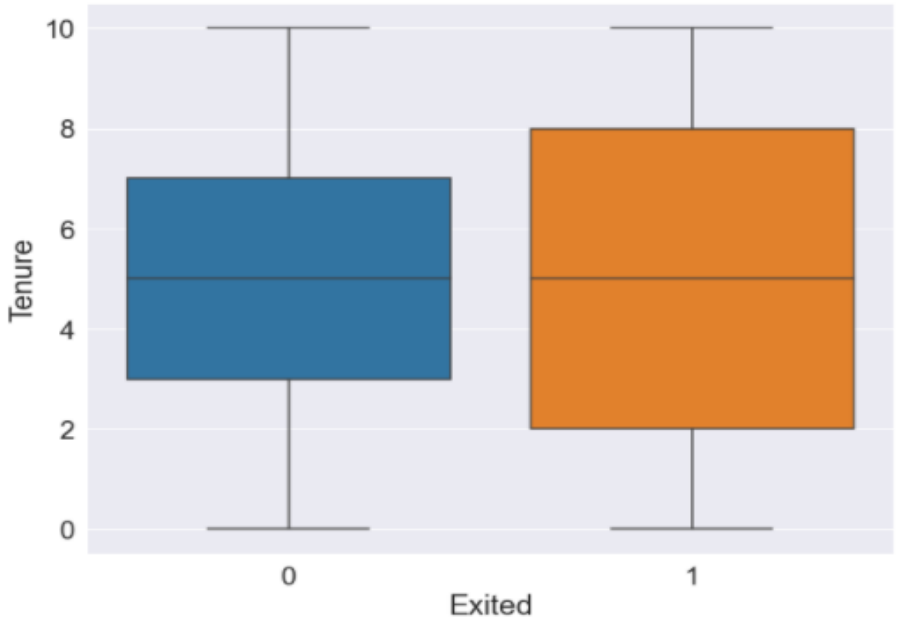
```
: <AxesSubplot:xlabel='Exited', ylabel='Tenure'>
```



- Age has a significant impact on customer churn. Customers with higher age tend to churn more as compared to younger customers.This also confirms the insight that we derived from the correlation plot.
- There is no significant difference in the credit score distribution between retained and churned customers.



- Customers with high balance tend to churn more when compared to customers having lower balance.
- The estimated salary has no significant effect on customer churn.



- The customers on either extreme (spent little time with the bank or a lot of time with the bank) are more likely to churn compared to those that have an average tenure with the bank.

Categorical Variable Exploration with respect to Exit Status

```
In [52]: df[['Geography','Gender','Exited']].groupby(['Geography','Gender']).agg(['mean','count'])
```

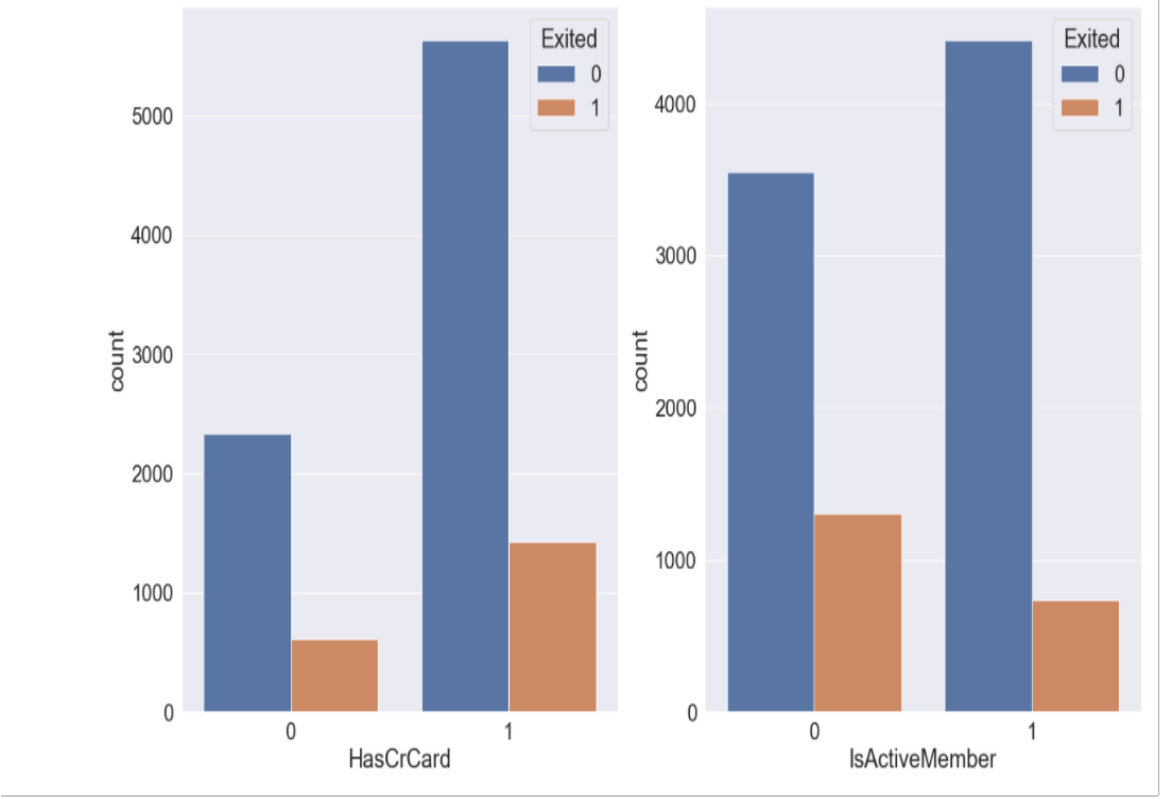
Out[52]:

		Exited	
		mean	count
Geography	Gender		
France	0	0.127134	2753
	1	0.203450	2261
Germany	0	0.278116	1316
	1	0.375524	1193
Spain	0	0.131124	1388
	1	0.212121	1089

- Though there are more number of male populations, females tend to churn more.
- The churn rate in Germany is higher than France and Spain. France and Spain have equal number of customers who have churned.However there are more number of customers from France and hence France has the highest customer retention rate when compared with all three region.

```
In [72]: fig, axarr = plt.subplots(1,2 ,figsize=(20, 12))
         sns.set(font_scale = 2)
         sns.countplot(x='HasCrCard', hue = 'Exited',data = df, ax=axarr[0])
         sns.countplot(x='IsActiveMember', hue = 'Exited',data = df, ax=axarr[1])
```

```
Out[72]: <AxesSubplot:xlabel='IsActiveMember', ylabel='count'>
```



- The customers who own a credit card have churned more in number when compared to customers who don't own a credit card.
- Customers who are not digitally active tend to churn more when compared to digitally active customers.

Feature Engineering

Feature engineering is the process of selecting, manipulating, and transforming raw data into features that can be used in supervised learning. In order to make machine learning work well on new tasks, it might be necessary to design and train better features. Feature engineering, in simple terms, is the act of converting raw observations into desired features using statistical or machine learning approaches.

Feature engineering efforts mainly focus on two goals,

- Preparing the dataset which is compatible with the machine learning algorithm requirements.

- Improving the performance of machine learning models to yield optimal results.

Feature Selection

There are a total of 14 columns in the dataset.After careful observation of the features, we have removed the RowNumber, CustomerId, and Surname columns from our feature set. All the remaining columns contribute or have a relation with customer churn in one way or another.

Feature Engineering:-

In [16]:
```
#Feature Selection :

df_ml = df.copy()
df_ml = df_ml.drop(['RowNumber', 'CustomerId', 'Surname'], axis=1)
df_ml.head()
```

Out[16]:

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

Converting Categorical Columns to Numeric Columns

Machine learning algorithms work best with numerical data. However, in our dataset, we have two categorical columns: Geography and Gender. These two columns need to be converted to numeric columns.

We will be converting such categorical columns to numeric columns using one-hot encoding. Sklearn provides an object/function for one-hot encoding in the preprocessing module.

```
df_ml = df_ml.drop(['Geography','Gender'],axis = 1)
```

```
encoded = pd.get_dummies(df,columns = ['Geography','Gender'])
encoded = encoded.drop(columns= ['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Age', 'Tenure',
       'Balance', 'NumOfProducts', 'HasCrCard', 'IsActiveMember',
       'EstimatedSalary', 'Exited'],axis = 1)
encoded.head()
```

	Geography_France	Geography_Germany	Geography_Spain	Gender_Female	Gender_Male
0	1	0	0	1	0
1	0	0	1	1	0
2	1	0	0	1	0
3	1	0	0	1	0
4	0	0	1	1	0

```
df_ml = pd.concat([df_ml,encoded], axis=1)
df_ml.head()
```

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

Feature Scaling

Feature scaling is a method used to normalize the range of independent variables or features of data. In data processing, it is also known as data normalization and is generally performed during the data preprocessing step.

Standardization scales each input variable separately by subtracting the mean (called centering) and dividing by the standard deviation to shift the distribution to have a mean of zero and a standard deviation of one. Standardization of a dataset is a common requirement for many machine learning estimators

We will be using Standard Scaler from the sklearn library preprocessing module for our data standardization

```
]: df_ml.describe()
```

]:

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited	Geography_Frar
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10000.000000	10000.000000	10000.0000
mean	650.528800	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100090.239881	0.203700	0.5014
std	96.653299	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57510.492818	0.402769	0.5000
min	350.000000	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	11.580000	0.000000	0.0000
25%	584.000000	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51002.110000	0.000000	0.0000
50%	652.000000	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100193.915000	0.000000	1.0000
75%	718.000000	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149388.247500	0.000000	1.0000
max	850.000000	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199992.480000	1.000000	1.0000

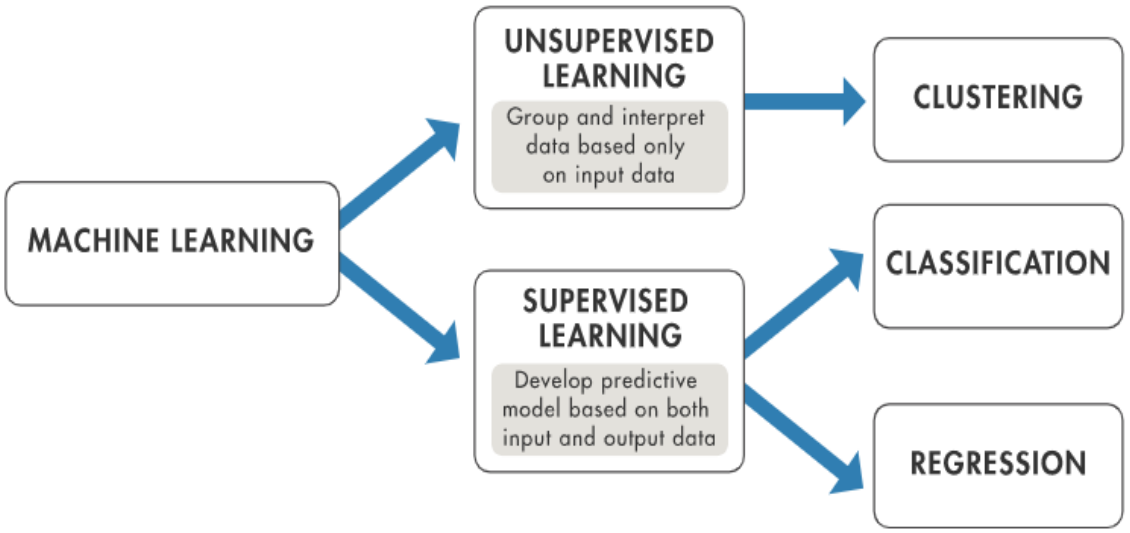
```
]: from sklearn.preprocessing import StandardScaler
sc_x = StandardScaler()
x_train = sc_x.fit_transform(x_train)
x_test = sc_x.transform(x_test)
```

Building a Machine Learning Model

A machine learning model is an expression of an algorithm that combs through mountains of data to find patterns or make predictions. Fueled by data, machine learning (ML) models are the mathematical engines of artificial intelligence.

A machine learning model is a mathematical representation of objects and their relationships to each other. The objects can be anything from “likes” on a social networking post to molecules in a lab experiment.

Machine learning uses two types of techniques: supervised learning, which trains a model on known input and output data so that it can predict future outputs, and unsupervised learning, which finds hidden patterns or intrinsic structures in input data.



Our project deals with Supervised Learning since we have both the input data(features) and the output data (Target column : ‘Exited’)

Since we have completed data preprocessing, The data is now ready to be trained by the machine learning model. As a first step,we need to isolate the target variable from the dataset.

```
:  x = df_scaled.drop(['Exited'], axis=1)
   y = df_scaled['Exited']
```

x contains all the columns except for the target variable and y contains only the target column (‘Exited’)

Now,the data needs to be split into a training and test set. The training set contains the data that will be used to train our machine learning model. The test set will be used to evaluate how good our model is. We'll use 20% of the data for the test set and the remaining 80% for the training set (specified with the test_size argument)

```
: from sklearn.model_selection import train_test_split
  x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2,random_state = 0)
```

Machine Learning Algorithm Training

A machine learning algorithm will identify patterns or trends in the training data. This step is known as algorithm training. Based on the data provided to the model, the algorithm will learn to find associations between the features and outputs. After training the algorithm, we will be able to make predictions.

There are several machine learning algorithms that can be used to make such predictions.
In this project, we are going to apply Logistic Regression and Decision Tree Classifier.
To train this algorithm, we call the fit method and pass in the feature set (x) and the corresponding target column (y). Later, we can use the predict method to make predictions on the test set.

Logistic Regression

Logistic regression, despite its name, is a classification algorithm rather than regression algorithm. Based on a given set of independent variables, it is used to estimate discrete value (0 or 1, yes/no, true/false).

Basically, it measures the relationship between the categorical dependent variable and one or more independent variables by estimating the probability of occurrence of an event using its logistics function.

sklearn.linear_model.LogisticRegression is the module used to implement logistic regression.

```
: from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression()
log_reg.fit(x_train, y_train)
print(log_reg.score(x_train, y_train))
# 80.85%
print(log_reg.score(x_test, y_test))
# 81.1%
# No overfitting detected. Train accuracy ~ Test accuracy.
```

After fitting the model, we could see that the train data received 80.85 % accuracy and the test data received 81.1 % accuracy. Here, we can conclude that there is no overfitting detected and our model is a good model.

Logistic Regression Algorithm Evaluation

```
# Prediction generation and analysis

y_pred = log_reg.predict(x_test)
```

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)

FP = cm[0][1]
FN = cm[1][0]

per_fp = FP / len(y_test) * 100
print(per_fp)# 3.45%

per_fn = FN / len(y_test) * 100
print(per_fn) # 15.45%

# FN is more than 4 times of FP
```

```
3.45
15.45
```

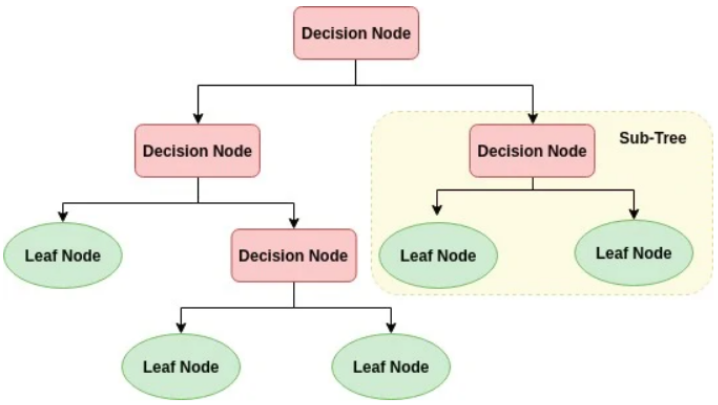
```
from sklearn.metrics import precision_score, recall_score, f1_score
print(precision_score(y_test, y_pred)) # 58.18%
print(recall_score(y_test, y_pred)) # 23.70%
print(f1_score(y_test, y_pred)) # 33.68%
```

```
0.5818181818181818
0.23703703703703705
0.3368421052631579
```

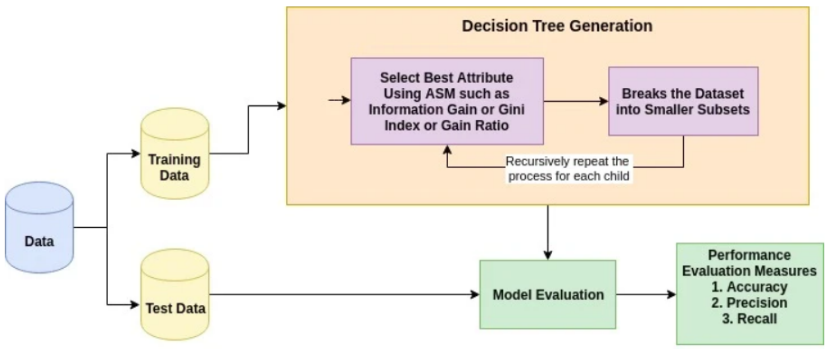
From the confusion matrix, we could see that False negative is more than 4 times of False positive.We shall keep these results aside for now and compare it with Decision Tree Classifier Algorithm.

Decision Tree Algorithm

A decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in a recursive manner and hence is called recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret.



The time complexity of decision trees is a function of the number of records and number of attributes in the given data. The decision tree is a distribution-free or non-parametric method, which does not depend upon probability distribution assumptions. Decision trees can handle high dimensional data with good accuracy.



```
]: from sklearn.tree import DecisionTreeClassifier
   dtf = DecisionTreeClassifier()
   dtf.fit(x_train, y_train)

   print(dtf.score(x_train, y_train)) # 100%
   print(dtf.score(x_test, y_test)) # 81%

   # Severe overfitting detected. # Train accuracy >> Test accuracy
   # Overcome overfitting ----> Apply regularization
```

From the results, we could see that the training data has 100% accuracy and test data has 81% accuracy. This means that severe overfitting is detected in our model**.Overfitting** refers to the condition when the model completely fits the training data but fails to generalize the testing unseen data. Overfit condition arises when the model memorizes the noise of the training data and fails to capture important patterns. A perfectly fit decision tree performs well for training data but performs poorly for unseen test data. To overcome this we need to apply regularization.

In order to apply regularization and overcome overfitting in the model, we introduce the max_depth parameter when calling the decision tree classifier function. max_depth represents the depth of each tree in the forest. The deeper the tree, the more splits it has and it captures more information about the data. We fit each decision tree with depths ranging from 1 to 32 and plot the training and test errors.

```
: # In order to apply regularizatin on tree based algos, we use the following

  dtf = DecisionTreeClassifier(max_depth = 7)
  dtf.fit(x_train, y_train)

  print(dtf.score(x_train, y_train)) #87.36%
  print(dtf.score(x_test, y_test)) #85.7%
```

```
0.873625
0.8575
```

After regularization, we can observe that the training data has an accuracy of 87.36 % and test data has an accuracy of 85.7 % Hence, we can conclude that overfitting is now handled in our model.

```
# Overfitting handled

y_pred_dtf = dtf.predict(x_test)
cm_dtf = confusion_matrix(y_test, y_pred_dtf)

FP = cm_dtf[0][1]
FN = cm_dtf[1][0]

per_fp = FP / len(y_test) * 100
print(per_fp) # 4.85

per_fn = FN / len(y_test) * 100
print(per_fn)# 9.45
```

From the confusion matrix, we can interpret that False Negative is again higher than False positive.

Since the accuracy of Decision Tree classifier(85.7 %) is higher than that of Logistic Regression (81.1 %), we can choose the Decision Tree model to make predictions on customer churn in our given dataset.

Fine Tuning the Selected Model

We had taken max depth as 7 using a trial and error method. However, we are going to estimate which is the best parameter for our selected model.We're going to initialize our classifier and GridSearchCv which is the main component which will help us find the best hyperparameters.We simply create a tuple (kind of non edit list) of hyperparameters we want the machine to test with as save them as params.We then give the classifier and list of params as parameters to gridsearchcv.

```
from sklearn.model_selection import GridSearchCV

params = {'max_depth': [3, 5, 7, 9, 11, 13],
          'criterion': ['gini', 'entropy']}

grid = GridSearchCV(DecisionTreeClassifier(), params)
grid.fit(x_train, y_train)

print(grid.best_params_)
best_dt = grid.best_estimator_
```

```
{'criterion': 'entropy', 'max_depth': 7}
```

After performing fine tuning, we can see that the max depth that we had selected earlier is the best parameter for the Decision Tree model with highest accuracy.

Final Evaluation of Selected Model

A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False. More specifically, True Positives, False Positives, True negatives and False Negatives are used to predict the metrics of a classification report as shown below.

:

```
y_pred_dtc = dtf.predict(x_test)

from sklearn.metrics import classification_report

print(classification_report(y_test, y_pred_dtc))
```

```
              precision    recall  f1-score   support

           0       0.89      0.94      0.91      1595
           1       0.69      0.53      0.60       405

    accuracy                           0.86      2000
   macro avg       0.79      0.74      0.76      2000
weighted avg       0.85      0.86      0.85      2000
```

Precision

Precision is the ability of a classifier not to label an instance positive that is actually negative. For each class it is defined as the ratio of true positives to the sum of true and false positives.
TP – True Positives
FP – False Positives

Precision – Accuracy of positive predictions.
Precision = TP/(TP + FP)

Recall

Recall is the ability of a classifier to find all positive instances. For each class it is defined as the ratio of true positives to the sum of true positives and false negatives. FN – False Negatives. Recall: Fraction of positives that were correctly identified.

Recall = TP/(TP+FN)

F1 Score

The F1 score is a weighted harmonic mean of precision and recall such that the best score is 1.0 and the worst is 0.0. Generally speaking, F1 scores are lower than accuracy measures as they embed precision and recall into their computation. As a rule of thumb, the weighted average of F1 should be used to compare classifier models, not global accuracy.

F1 Score = 2*(Recall * Precision) / (Recall + Precision)

Conclusion

- In this project, we had performed Exploratory Data analysis on the given data set and concluded relevant insights.
- We first used a Logistic Regression Algorithm to train our model which yielded 81.1 % accuracy in the test data set.
- Later, we used Decision Tree classifier to train our model which gave an accuracy of 85.7% on the test data.
- Upon comparison of both the accuracy and confusion matrix, we selected the model trained by Decision Tree classifier Algorithm.
- We further used Grid Search to fine tune the model and select the best parameter for an accurate model.
- The confusion matrix and classification report yields very good model performance.
- We have now successfully built a Machine learning Model that can predict the customer churn with 85.7 % accuracy