DATE	28/09/2022
TEAM ID	PNT2022TMID27851
STUDENT NAME	SWAATHI CM
DOMAIN NAME	HEALTH CARE
PROJECT NAME	EARLY DETECTION OF CHRONIC KIDNEY DISEASE USING MACHINE LEARNING
MAXIMUM MARKS	2 MARKS

LOAD THE DATASET

IMPORT THE REQUIRED LIBRARIES

```
[1] import pandas as pd
    import numpy as np
    import matplotlib.pyplot as plt
    import seaborn as sns
```

LOAD THE DATASET

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

```
df.head()
```

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

PERFORM VISUALIZATIONS

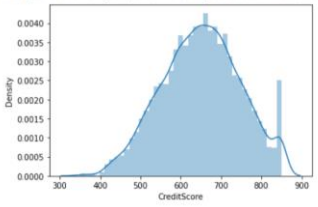
Removing Unwanted Values

```
[4] df = df.drop(columns=['RowNumber','CustomerId','Surname'])
```

- UNIVARIATE

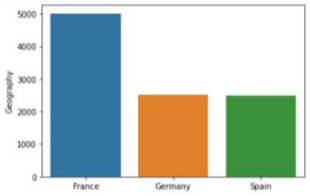
```
[5] ## FOR NUMERICAL COLUMN
    sns.distplot(df.CreditScore)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52ac13d190>
```




```
[6] ## FOR CATEGORICAL COLUMN
    sns.barplot(df.Geography.value_counts().index,df.Geography.value_counts())
```

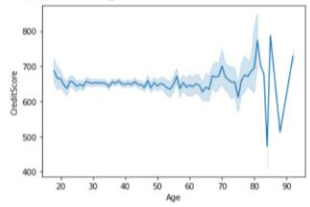
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argume
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f52ac0070d0>
```



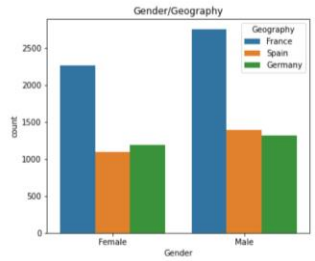
- BIVARIATE

```
[7]  ## FOR NUMERICAL COLUMN
     sns.lineplot(df.Age,df.CreditScore)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only valid positional argume
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f52ac007810>
```



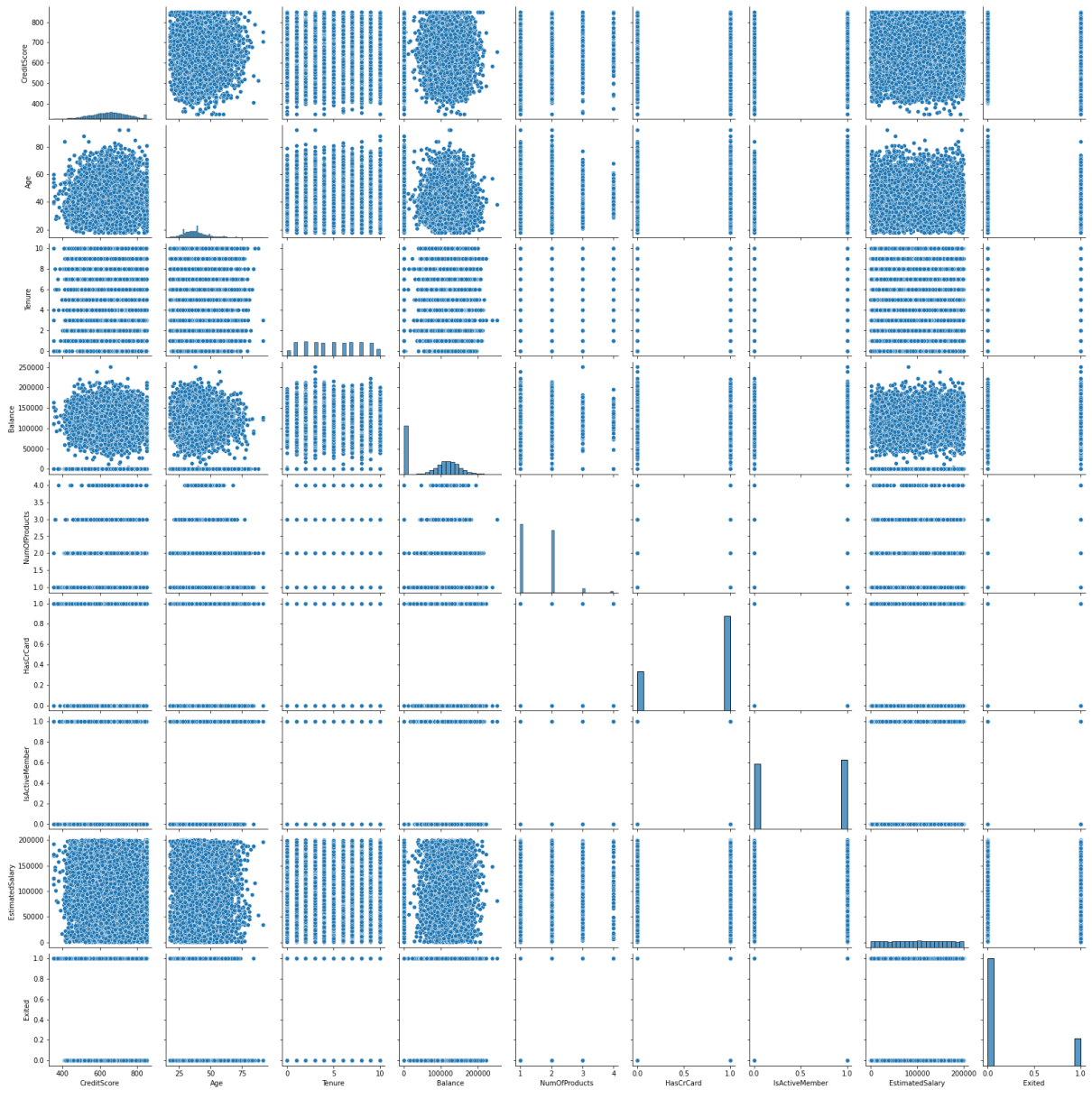

```
[8]  ## CATEGORICAL COLUMN
     def countplot_of_2(x,hue,title=None,figsize=(6,5)):
       plt.figure(figsize = figsize)
       sns.countplot(data = df[[x,hue]],x=x,hue=hue)
       plt.title(title)
       plt.show()
     countplot_of_2('Gender','Geography','Gender/Geography')
```



- MULTIVARIATE

MULTIVARIATE ANALYSIS

```
## CATEGORICAL COLUMN AND NUMERICAL COLUMN
sns.pairplot(df)
```



PERFORM DESCRIPTIVE ANALYSIS ON THE DATASET

```
[10] df.describe()
```

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

```
[11] df.shape

(10000, 11)
```

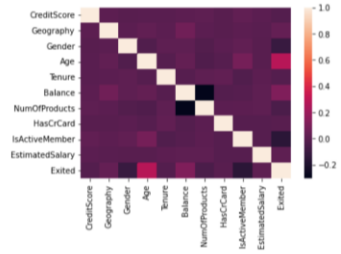
```
[12] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   CreditScore      10000 non-null  int64
 1   Geography        10000 non-null  object
 2   Gender           10000 non-null  object
 3   Age              10000 non-null  int64
 4   Tenure           10000 non-null  int64
 5   Balance          10000 non-null  float64
 6   NumOfProducts    10000 non-null  int64
 7   HasCrCard        10000 non-null  int64
 8   IsActiveMember   10000 non-null  int64
 9   EstimatedSalary  10000 non-null  float64
 10  Exited           10000 non-null  int64
dtypes: float64(2), int64(7), object(2)
memory usage: 859.5+ KB
```

```
[13] df.isnull().any()

CreditScore        False
Geography          False
Gender             False
Age                False
Tenure             False
Balance            False
NumOfProducts      False
HasCrCard          False
IsActiveMember     False
EstimatedSalary    False
Exited             False
dtype: bool
```

```
[46] df.corr()
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
CreditScore	1.000000	0.008939	-0.003613	-0.001992	-0.000650	0.007074	0.012293	-0.003942	0.023596	0.001619	-0.018298
Geography	0.008939	1.000000	0.004719	0.022812	0.003739	0.069408	0.003972	-0.008523	0.006724	-0.001369	0.035943
Gender	-0.003613	0.004719	1.000000	-0.027544	0.014733	0.012087	-0.021859	0.005766	0.022544	-0.008112	-0.106512
Age	-0.001992	0.022812	-0.027544	1.000000	-0.009997	0.028308	-0.030680	-0.011721	0.085472	-0.007201	0.285323
Tenure	-0.000650	0.003739	0.014733	-0.009997	1.000000	-0.012254	0.013444	0.022583	-0.028362	0.007784	-0.014001
Balance	0.007074	0.069408	0.012087	0.028308	-0.012254	1.000000	-0.304180	-0.014858	-0.010084	0.012797	0.118533
NumOfProducts	0.012293	0.003972	-0.021859	-0.030680	0.013444	-0.304180	1.000000	0.003183	0.009612	0.014204	-0.047820
HasCrCard	-0.003942	-0.008523	0.005766	-0.011721	0.022583	-0.014858	0.003183	1.000000	-0.011866	-0.009933	-0.007138
IsActiveMember	0.023596	0.006724	0.022544	0.085472	-0.028362	-0.010084	0.009612	-0.011866	1.000000	-0.011421	-0.156128
EstimatedSalary	0.001619	-0.001369	-0.008112	-0.007201	0.007784	0.012797	0.014204	-0.009933	-0.011421	1.000000	0.012097
Exited	-0.018298	0.035943	-0.106512	0.285323	-0.014001	0.118533	-0.047820	-0.007138	-0.156128	0.012097	1.000000

```
sns.heatmap(df.corr())

<matplotlib.axes._subplots.AxesSubplot at 0x7f52a3907590>
```

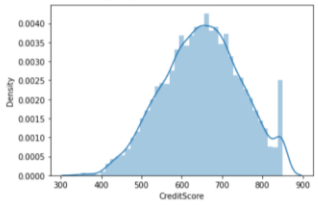


HANDLE THE MISSING VALUES

```
[14] ## NUMERICAL COLUMN
```

```
[15] sns.distplot(df.CreditScore)
```

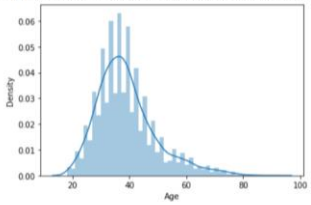
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a6fe1910>
```




```
[16] df['CreditScore'].fillna(df['CreditScore'].mean(),inplace=True)
```

```
[17] sns.distplot(df.Age)
```

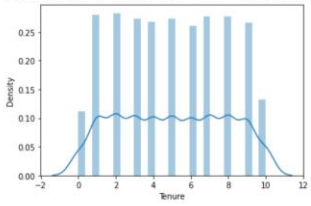
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a5693d90>
```




```
[18] df['Age'].fillna(df['Age'].median(),inplace=True)
```

```
[19] sns.distplot(df.Tenure)
```

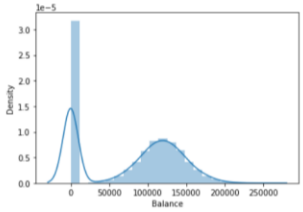
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a55978d0>
```




```
[20] df['Tenure'].fillna(df['Tenure'].median(),inplace = True)
```

```
[21] sns.distplot(df.Balance)
```

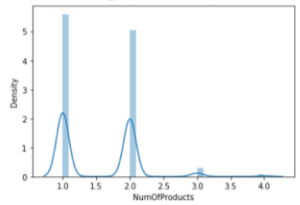
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a559ef90>
```




```
[22] df['Balance'].fillna(df['Balance'].median(),inplace = True)
```

```
[23] sns.distplot(df.NumOfProducts)
```

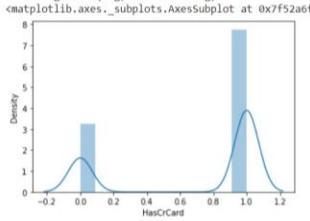
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a544e690>
```




```
[24] df['NumOfProducts'].fillna(df['NumOfProducts'].median(),inplace = True)
```

```
[25] sns.distplot(df.HasCrCard)
```

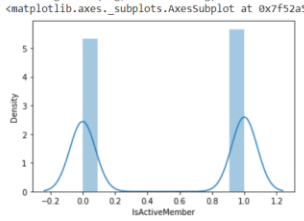
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a6fbc810>
```




```
[26] df['HasCrCard'].fillna(df['HasCrCard'].median(),inplace = True)
```

```
[27] sns.distplot(df.IsActiveMember)
```

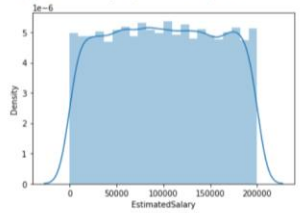
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a531b450>
```




```
[28] df['IsActiveMember'].fillna(df['IsActiveMember'].median(),inplace = True)
```

```
[29] sns.distplot(df.EstimatedSalary)
```

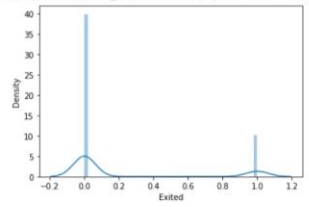
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a5291e90>
```




```
[30] df['EstimatedSalary'].fillna(df['EstimatedSalary'].median(),inplace = True)
```

```
[31] sns.distplot(df.Exited)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a521cd50>
```




```
[32] df['Exited'].fillna(df['Exited'].median(),inplace = True)
```

```
## for categorical columns
```

```
[33] df['Geography'].fillna(df['Geography'].mode(),inplace = True)
```

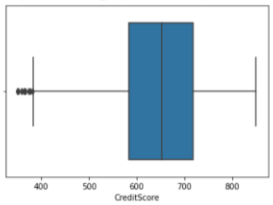
```
[34] df['Gender'].fillna(df['Gender'].mode(),inplace = True)
```

FIND THE OUTLIERS AND REPLACE THE OUTLIERS

FINDING AND REPLACING OUTLIERS

```
[35] sns.boxplot(df.CreditScore)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f52a51b9ed0>
```

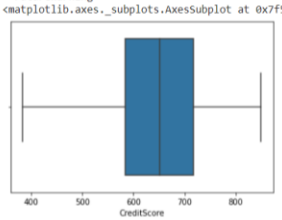



```
[36] df.median()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a futur
  """Entry point for launching an IPython kernel.
CreditScore           652.000
Age                    37.000
Tenure                  5.000
Balance             97198.540
NumOfProducts           1.000
HasCrCard               1.000
IsActiveMember          1.000
EstimatedSalary    100193.915
Exited                  0.000
dtype: float64
```

```
[37] Q1 = df.CreditScore.quantile(0.25)
     Q3 = df.CreditScore.quantile(0.75)
     IQR = Q3 - Q1
     upper_limit = Q3 + (1.5 * IQR)
     lower_limit = Q1 - (1.5 * IQR)
     df['CreditScore'] = np.where(df['CreditScore']<lower_limit,652,df['CreditScore'])
     sns.boxplot(df.CreditScore)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f52aba7a5d0>
```



CHECK FOR CATEGORICAL COLUMNS AND PERFORM ENCODING

CHECK FOR CATEGORICAL COLUMN AND PERFORM ENCODING

```
[38] from sklearn.preprocessing import LabelEncoder
     le = LabelEncoder()
     df.Gender = le.fit_transform(df.Gender)
     df.head()
```

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

```
[39] df.Geography = le.fit_transform(df.Geography)
     df.head()
```

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

SPLIT THE DATA INTO DEPENDENT AND INDEPENDENT VARIABLES

SPLIT THE DATA SET INTO DEPENDENT AND INDEPENDENT VARIABLES

```
[40] X = df.drop(columns = ['Exited'], axis = 1)
     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

```
[41] y = df['Exited']
     y.head()

     0    1
     1    0
     2    1
     3    0
     4    0
     Name: Exited, dtype: int64
```

SCALE THE INDEPENDENT VARIABLES

SCALE THE INDEPENDENT COLUMNS

```
[42] from sklearn.preprocessing import MinMaxScaler
     scale = MinMaxScaler()
     X_scaled = pd.DataFrame(scale.fit_transform(X),columns = X.columns)
     X_scaled.head()
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
0	0.505353	0.0	0.0	0.324324	0.2	0.000000	0.000000	1.0	1.0	0.506735
1	0.481799	1.0	0.0	0.310811	0.1	0.334031	0.000000	0.0	1.0	0.562709
2	0.254818	0.0	0.0	0.324324	0.8	0.636357	0.666667	1.0	0.0	0.569654
3	0.676660	0.0	0.0	0.283784	0.1	0.000000	0.333333	0.0	0.0	0.469120
4	1.000000	1.0	0.0	0.337838	0.2	0.500246	0.000000	1.0	1.0	0.395400

SPLIT THE DATA INTO TRAINING AND TESTING