```
from google.colab import drive
drive.mount('/content/drive')
```

```
import numpy as np
import pandas as pd

path="/content/drive/MyDrive/Churn_Modelling.csv"
df=pd.read_csv(path)
df.describe
```

```
<bound method NDFrame.describe of       RowNumber  CustomerId    Surname  
CreditScore Geography  Gender  Age  \
0             1    15634602   Hargrave          619    France  Female   42
1             2    15647311       Hill          608     Spain  Female   41
2             3    15619304       Onio          502    France  Female   42
3             4    15701354       Boni          699    France  Female   39
4             5    15737888   Mitchell          850     Spain  Female   43
...         ...         ...        ...          ...       ...     ...  ...
9995       9996    15606229   Obijiaku          771    France    Male   39
9996       9997    15569892  Johnstone          516    France    Male   35
9997       9998    15584532        Liu          709    France  Female   36
9998       9999    15682355  Sabbatini          772   Germany    Male   42
9999      10000    15628319     Walker          792    France  Female   28

      Tenure    Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0          2       0.00              1          1               1
1          1   83807.86              1          0               1
2          8  159660.80              3          1               0
3          1       0.00              2          0               0
4          2  125510.82              1          1               1
...      ...        ...            ...        ...             ...
9995       5       0.00              2          1               0
9996      10   57369.61              1          1               1
9997       7       0.00              1          0               1
9998       3   75075.31              2          1               0
9999       4  130142.79              1          1               0

      EstimatedSalary  Exited
0           101348.88       1
1           112542.58       0
2           113931.57       1
3            93826.63       0
4            79084.10       0
...               ...     ...
9995         96270.64       0
9996        101699.77       0
9997         42085.58       1
9998         92888.52       1
9999         38190.78       0

[10000 rows x 14 columns]>
```

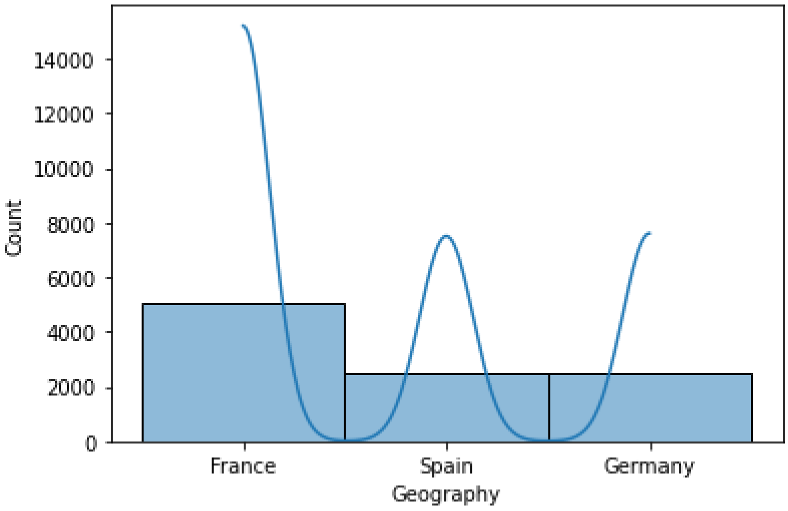
```
import matplotlib.pyplot as plt


import seaborn as sns


%matplotlib inline
```

```
df[['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
       'Gender', 'Age', 'Tenure']].describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000

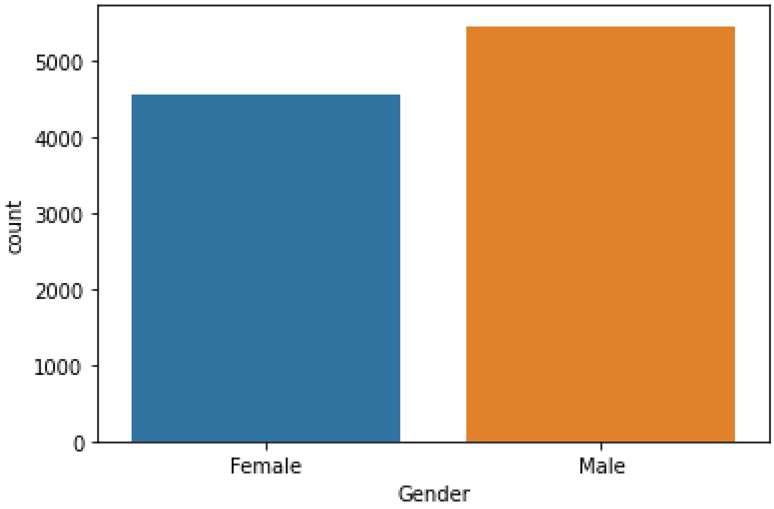
```
sns.histplot(df.Geography,kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f57aa309350>
```



```
sns.countplot(df.Gender)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: P
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f5797ef58d0>
```

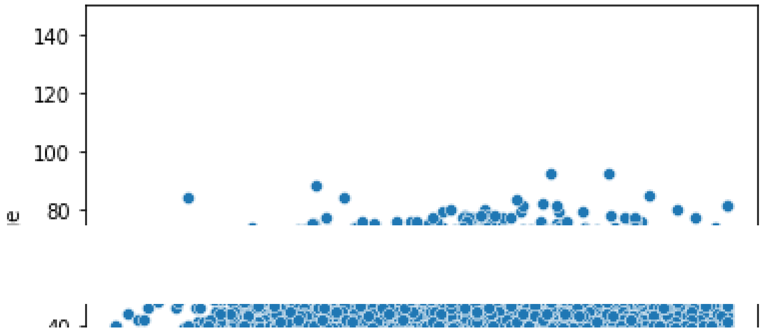


```
df[['RowNumber', 'CustomerId', 'Surname', 'CreditScore', 'Geography',
       'Gender', 'Age', 'Tenure']].corr()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure
RowNumber	1.000000	0.004202	0.005840	0.000783	-0.006495
CustomerId	0.004202	1.000000	0.005308	0.009497	-0.014883
CreditScore	0.005840	0.005308	1.000000	-0.003965	0.000842
Age	0.000783	0.009497	-0.003965	1.000000	-0.009997
Tenure	-0.006495	-0.014883	0.000842	-0.009997	1.000000

```
sns.scatterplot(df.CreditScore,df.Age)
plt.ylim(0,150)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pas
  FutureWarning
(0.0, 150.0)
```



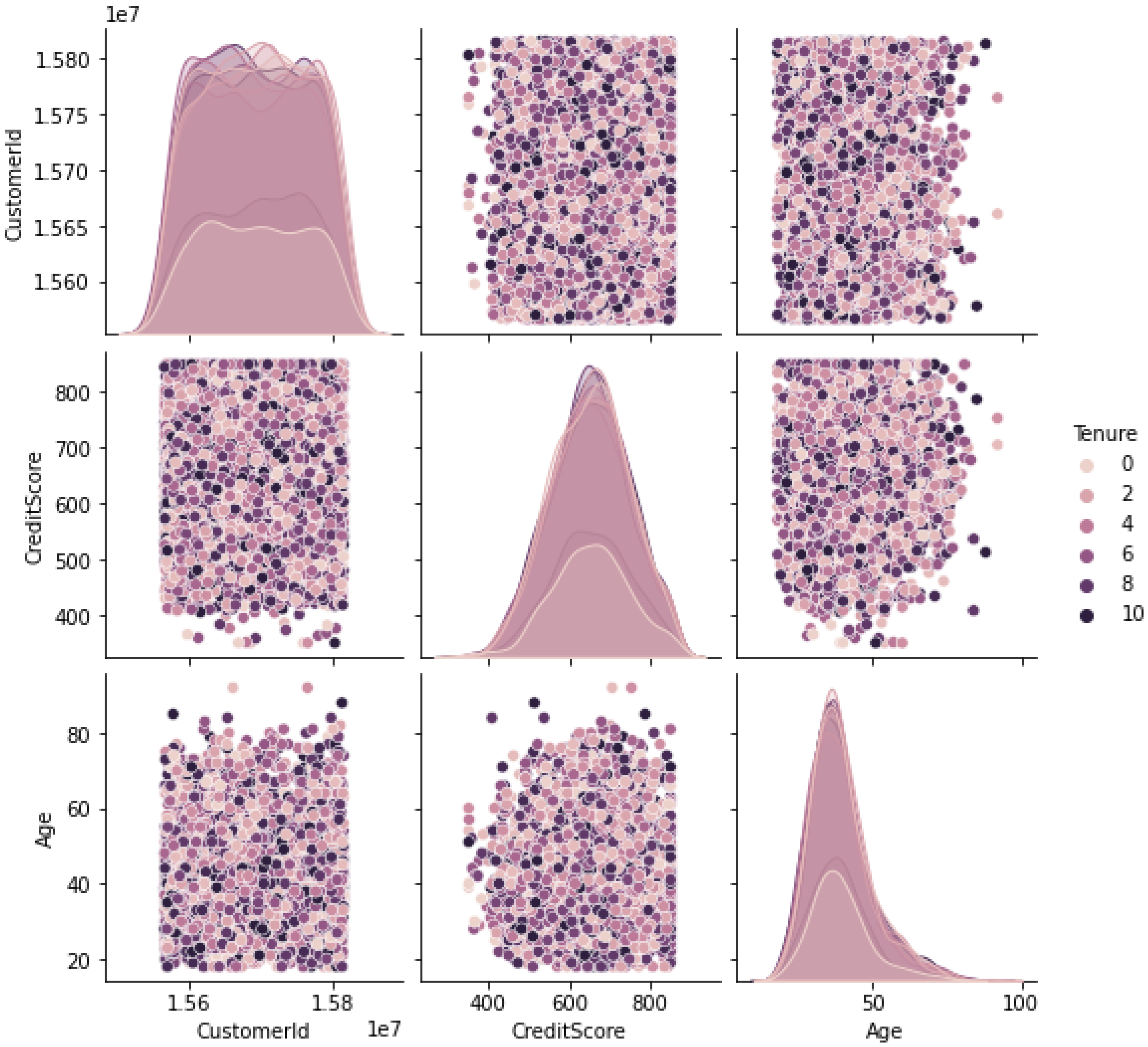
Double-click (or enter) to edit

3 Multi - Variate Analysis

Double-click (or enter) to edit

```
sns.pairplot(data=df[['CustomerId', 'Surname', 'CreditScore', 'Geography','Gender', 'Age',
```

```
<seaborn.axisgrid.PairGrid at 0x7f57979ec4d0>
```



Double-click (or enter) to edit

4 Descriptive statistics

```
df.describe()
```

	RowNumber	CustomerId	CreditScore	Age	Tenure	Bala
count	10000.00000	1.000000e+04	10000.000000	10000.000000	10000.000000	10000.000
mean	5000.50000	1.569094e+07	650.528800	38.921800	5.012800	76485.889
std	2886.89568	7.193619e+04	96.653299	10.487806	2.892174	62397.405
min	1.00000	1.556570e+07	350.000000	18.000000	0.000000	0.000
25%	2500.75000	1.562853e+07	584.000000	32.000000	3.000000	0.000
50%	5000.50000	1.569074e+07	652.000000	37.000000	5.000000	97198.540
75%	7500.25000	1.575323e+07	718.000000	44.000000	7.000000	127644.240
max	10000.00000	1.581569e+07	850.000000	92.000000	10.000000	250898.090

```
df.dtypes
```

```
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
```

```
df['Age'].mode()
```

```
     0    37
     dtype: int64
```

```
df["Age"].mean()
```

```
     38.9218
```

```
round(df["Age"].mean(), 3)
```

```
     38.922
```

```
df["Age"].median()
```

```
     37.0
```

Double-click (or enter) to edit

5 Handling Missing Values

```
df.isna().any()
```

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

```
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
```

```
df.isnull()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenur
0	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	.
9995	False	False	False	False	False	False	False	Fals
9996	False	False	False	False	False	False	False	Fals
9997	False	False	False	False	False	False	False	Fals
9998	False	False	False	False	False	False	False	Fals
9999	False	False	False	False	False	False	False	Fals

10000 rows × 14 columns

```
df.notnull()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure
0	True	True	True	True	True	True	True	True
1	True	True	True	True	True	True	True	True
2	True	True	True	True	True	True	True	True
3	True	True	True	True	True	True	True	True
4	True	True	True	True	True	True	True	True
...	...	...	...	...	...	...	...	...
9995	True	True	True	True	True	True	True	True
9996	True	True	True	True	True	True	True	True
9997	True	True	True	True	True	True	True	True
9998	True	True	True	True	True	True	True	True
9999	True	True	True	True	True	True	True	True

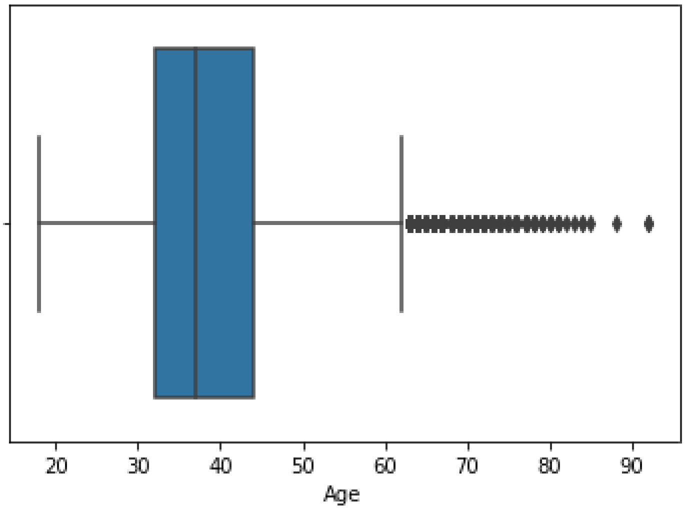
10000 rows × 14 columns

Double-click (or enter) to edit

6 Find the outliers and replace the outliers

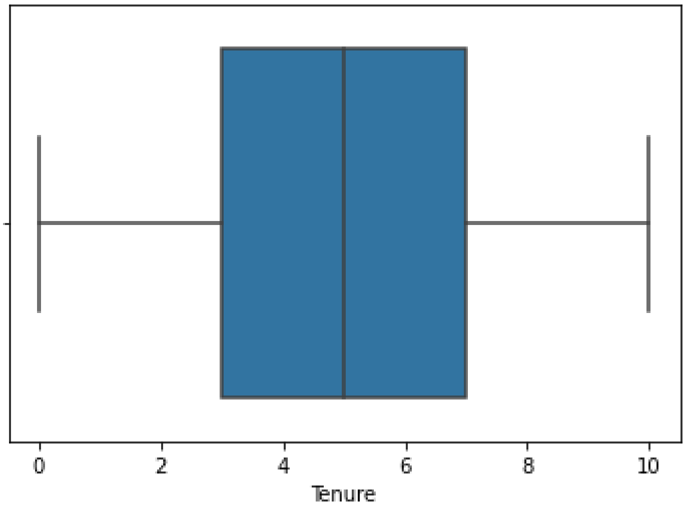
```
sns.boxplot(x=df['Age'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f57978189d0>
```



```
sns.boxplot(x=df['Tenure'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5794c20a90>
```



Double-click (or enter) to edit

Double-click (or enter) to edit

7 Check for Categorical columns and perform encoding.

```
a=df.columns
```

```
b=df._get_numeric_data().columns
```

```
b
```

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

```
list(set(a) - set(b))
```

```
['Gender', 'Geography', 'Surname']
```

Double-click (or enter) to edit

8 Split the data into dependent and independent variables.

Double-click (or enter) to edit

```
# x  -Independent
# y  -Dependent
x =df.drop('Exited',axis=1)
y=df['Exited']
```

```
x.head()
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Ba
0	1	15634602	Hargrave	619	France	Female	42	2	
1	2	15647311	Hill	608	Spain	Female	41	1	83
2	3	15619304	Onio	502	France	Female	42	8	159
3	4	15701354	Boni	699	France	Female	39	1	
4	5	15737888	Mitchell	850	Spain	Female	43	2	125

```
y.head()
```

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

Double-click (or enter) to edit

9 Scale the independent variables

```
from sklearn import linear_model
```

```
from sklearn.preprocessing import StandardScaler
```

```
scale = StandardScaler()
```

```
x=df[['Age','Tenure']]
```

```
scaledx = scale.fit_transform(x)
```

```
print(scaledx)
```

```
[[ 0.29351742 -1.04175968]
 [ 0.19816383 -1.38753759]
 [ 0.29351742  1.03290776]
 ...
 [-0.27860412  0.68712986]
 [ 0.29351742 -0.69598177]
 [-1.04143285 -0.35020386]]
```

Double-click (or enter) to edit

10 Split the data into training and testing

```
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)
```

```
print('X Train shape:{},Y.Train SHape:{}'.format(x_train.shape,y_train.shape))
```

```
X Train shape:(8000, 2),Y.Train SHape:(8000,)
```

```
print('X Test Shape :{},Y Test SHape:{}'.format(x_test.shape,y_test.shape))
```

```
X Test Shape :(2000, 2),Y Test SHape:(2000,)
```

✓ 0s completed at 10:34 AM ● ✕