Assignment-2

Exploratory Data Analysis

Assignment Date	25 September 2022
Student Name	Guru Hari Venkat S
Student Roll Number	513419106012
Maximum Marks	2 Marks

Exploratory Data Analysis

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import scale
from sklearn.model_selection import train_test_split
import numpy as np

sns.set_style('darkgrid')
sns.set(font_scale=1.1)
```

1. Download the dataset:

Dataset downloaded and mounted using drive

2. Load the dataset.

```
data = pd.read_csv("drive/MyDrive/Churn_Modelling.csv")

ds = pd.DataFrame(data)
ds.head(10)
```

```
   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
```

```
5          6    15574012       Chu          645     Spain    Male   44

6          7    15592531  Bartlett          822    France    Male   50

7          8    15656148    Obinna          376   Germany  Female   29

8          9    15792365        He          501    France    Male   44

9         10    15592389        H?          684    France    Male   27
```

```
   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
5       8  113755.78              2          1               0
6       7       0.00              2          1               1
7       4  115046.74              4          1               0
8       4  142051.07              2          0               1
9       2  134603.88              1          1               1

   EstimatedSalary  Exited
0        101348.88       1
1        112542.58       0
2        113931.57       1
3         93826.63       0
4         79084.10       0
5        149756.71       1
6         10062.80       0
7        119346.88       1
8         74940.50       0
9         71725.73       0
```

```
ds.shape
```

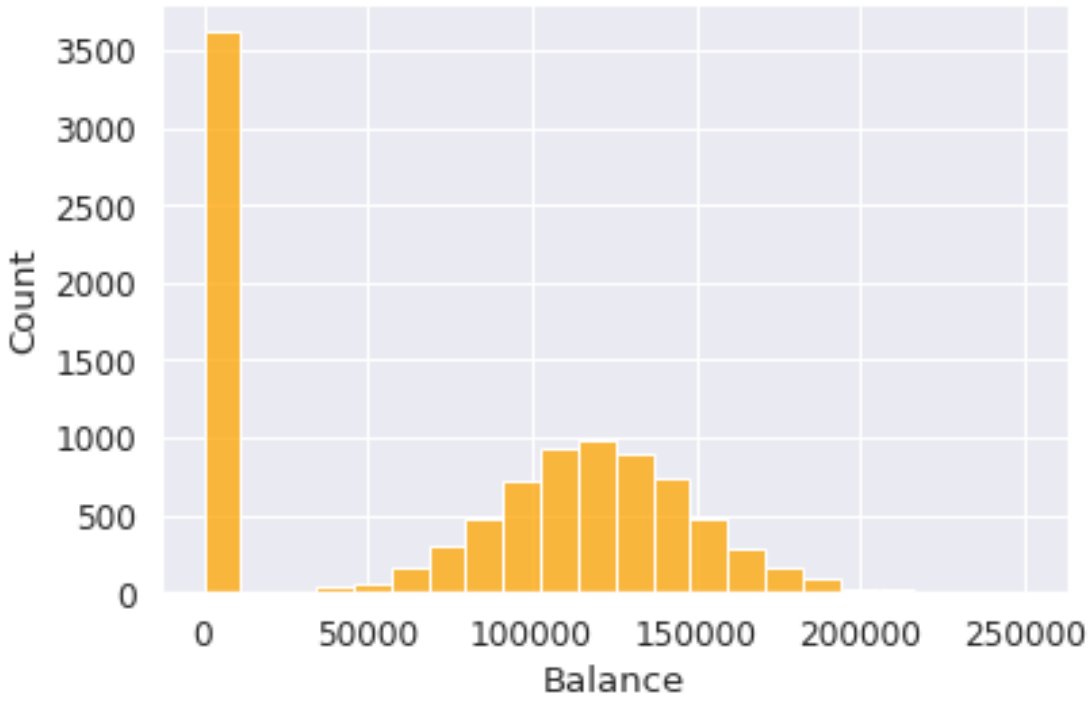
```
(10000, 14)
```

3. Perform Below Visualizations.

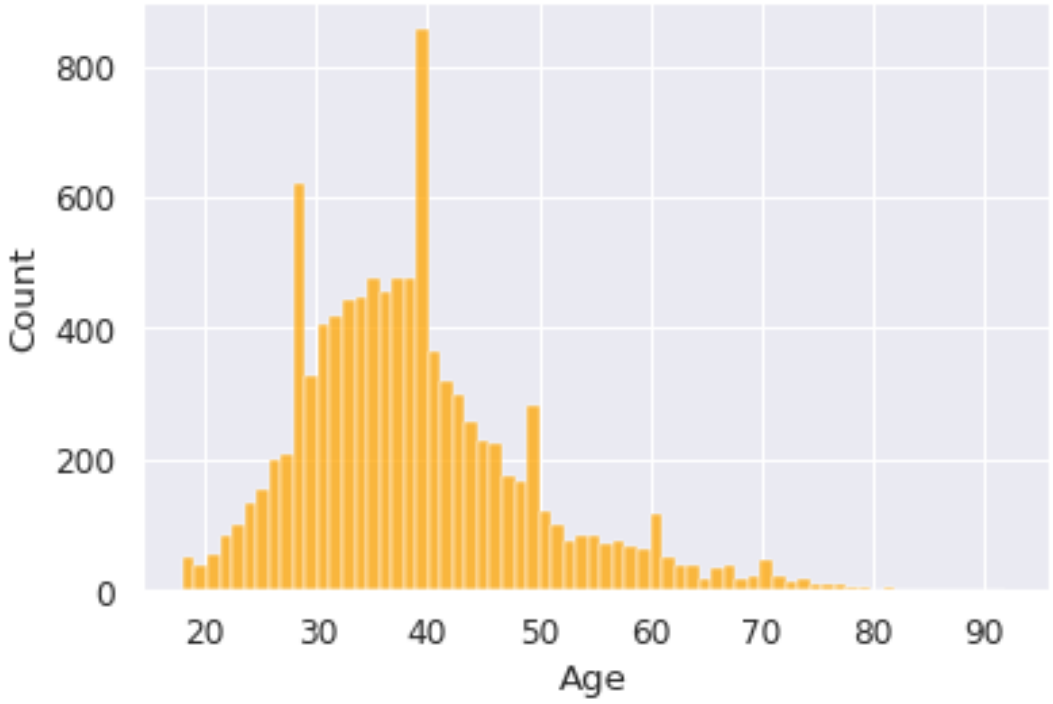
Univariate Analysis

Histogram

```
sns.histplot(ds['Balance'],color='orange');
```



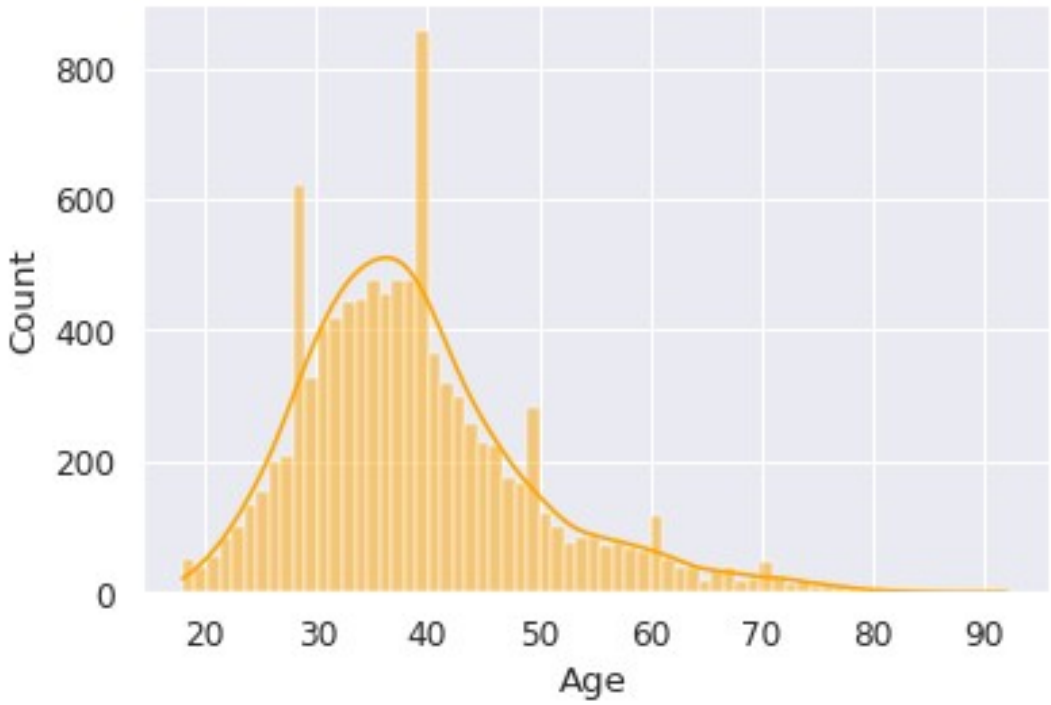

```
sns.histplot(ds['Age'],color='orange');
```



Histogram With KDE

```
sns.histplot(ds['Age'], kde=True, color='orange')
```

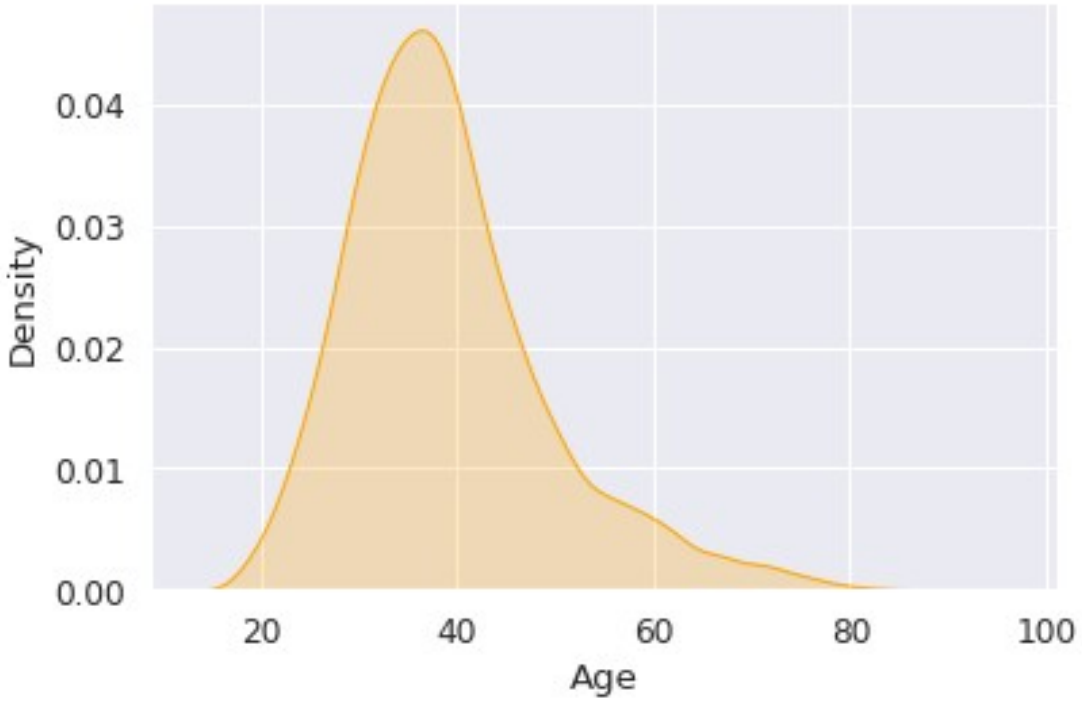
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8184f9d10>
```



KDE Plot

```
sns.kdeplot(ds['Age'],fill='true',color='orange')
```

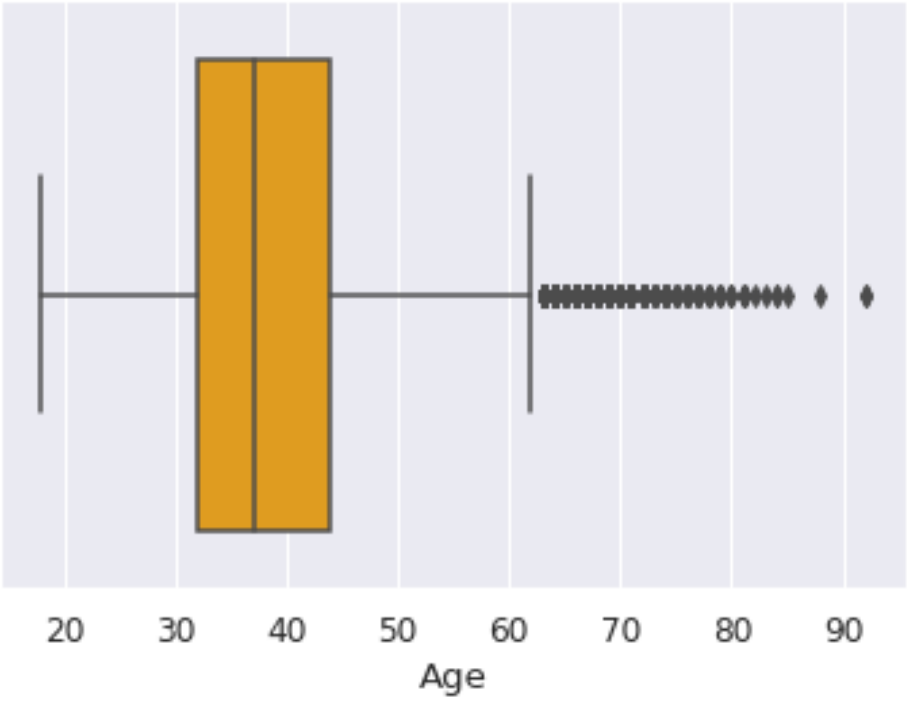
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe81852d490>
```



Boxplot

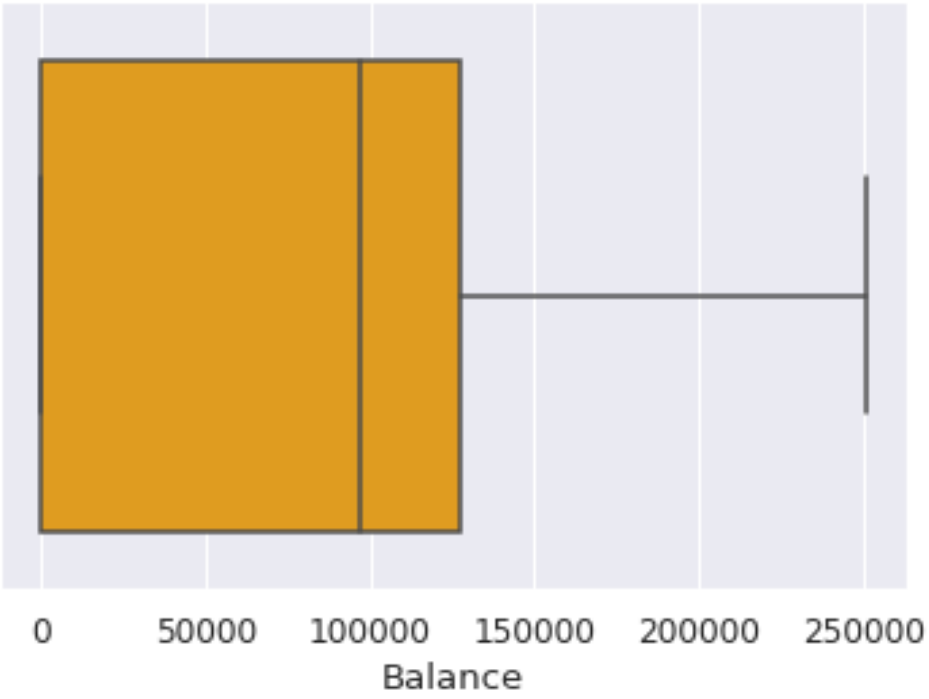
```
sns.boxplot(ds['Age'], color='orange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8185c6dd0>
```




```
sns.boxplot(ds['Balance'], color='orange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe818889b90>
```



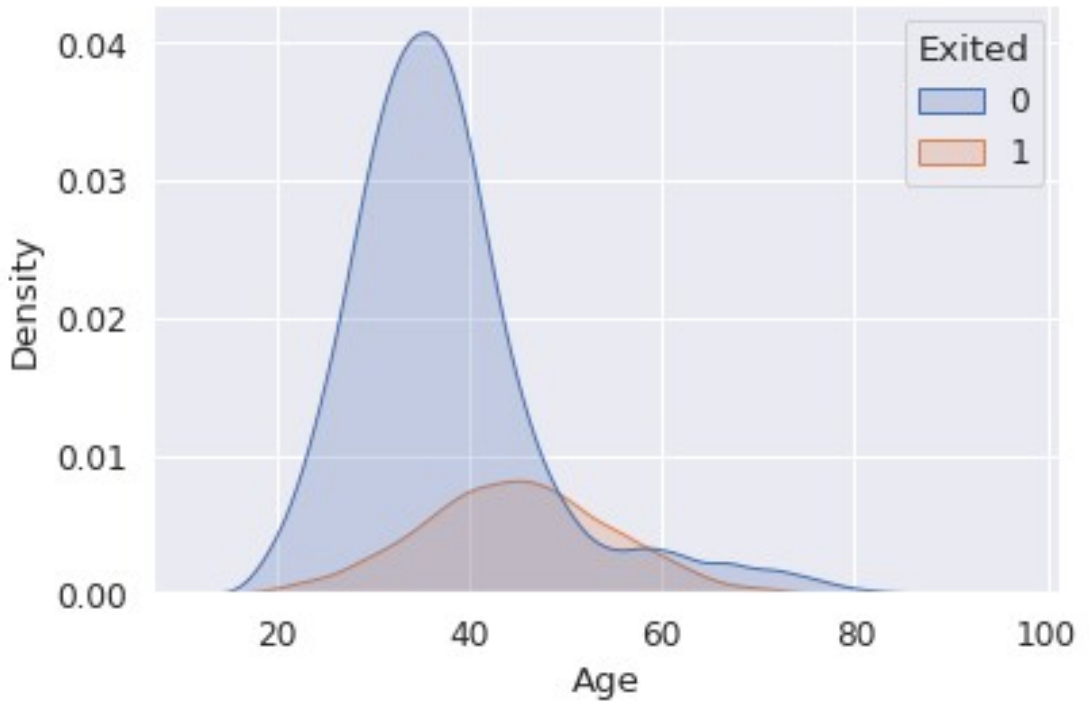
Bi - Variate Analysis

ds[['Age','Balance','Exited','EstimatedSalary','NumOfProducts']].corr()

KDE plot

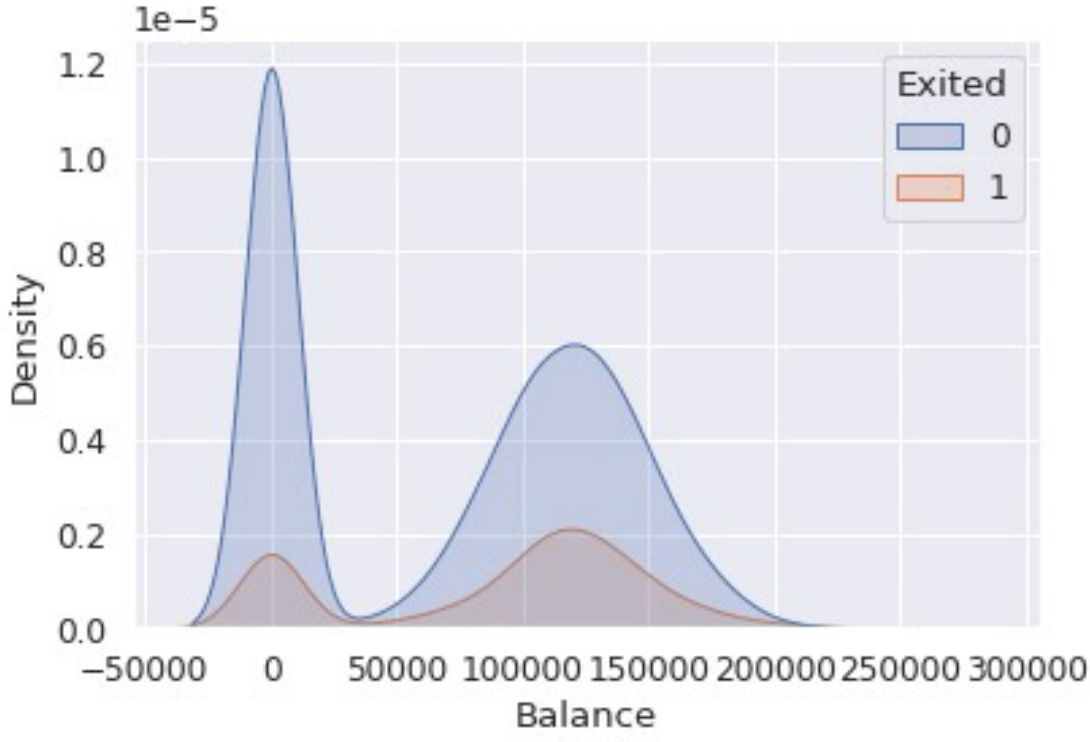
```
sns.kdeplot(ds['Age'], data=ds, hue=ds['Exited'], fill='true')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8182cffd0>
```



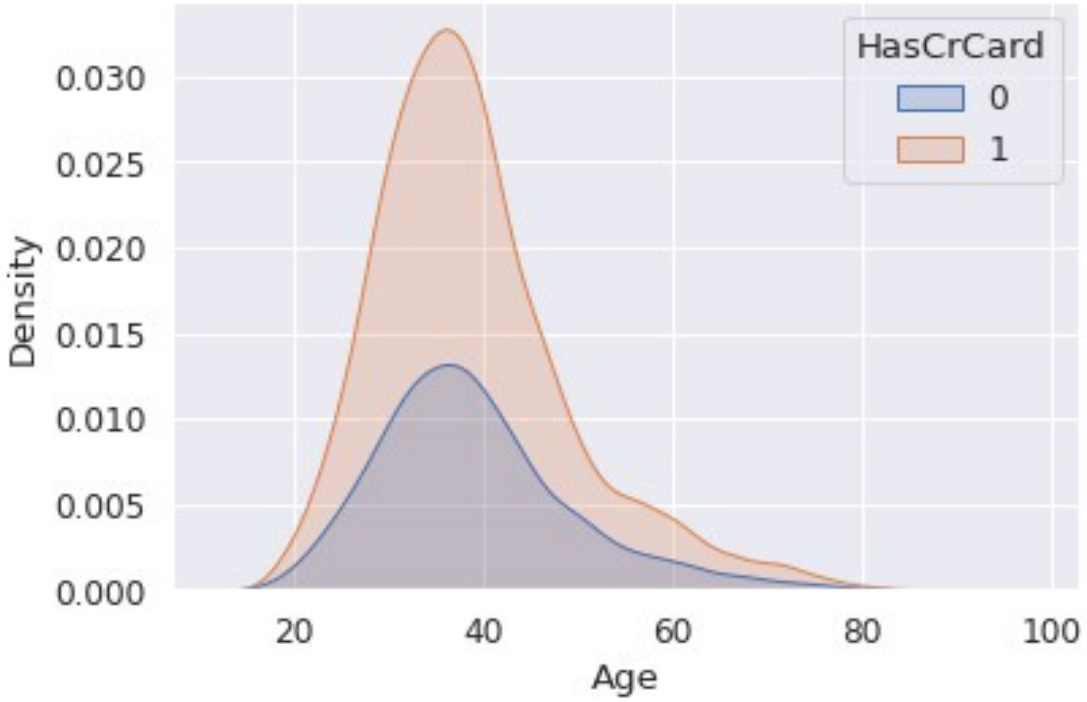

```
sns.kdeplot(ds['Balance'], data=ds, hue=ds['Exited'], fill='true')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe81eb2f050>
```

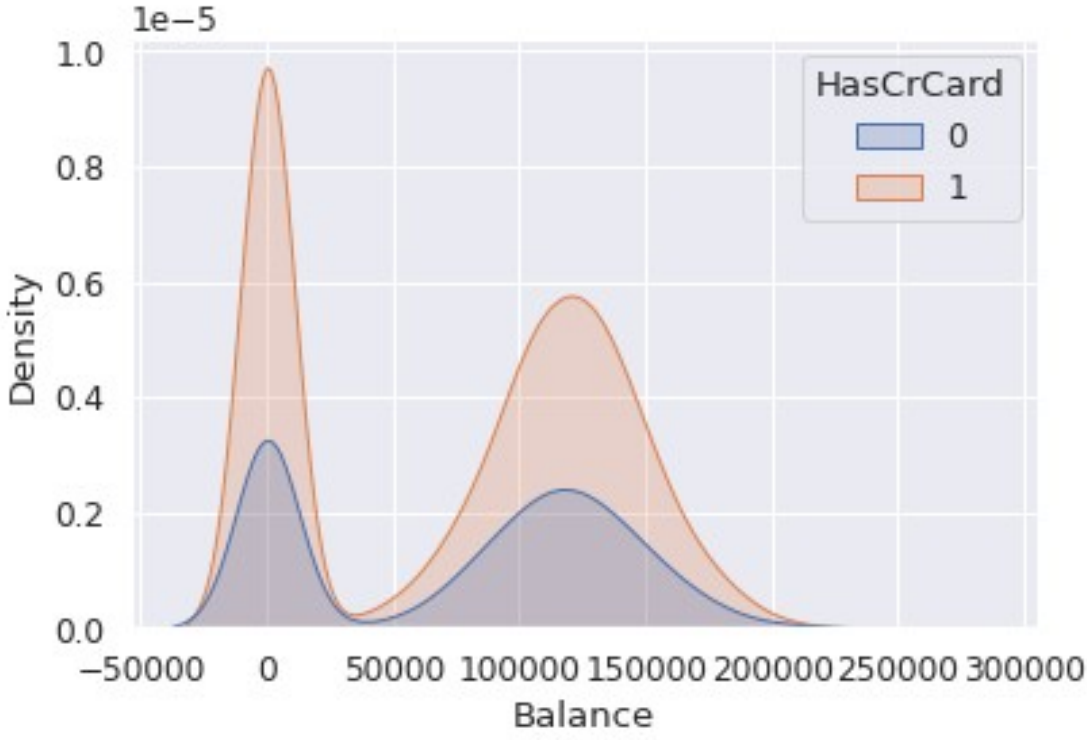



```
sns.kdeplot(ds['Age'], data=ds, hue=ds['HasCrCard'], fill='true')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe81ed35550>
```



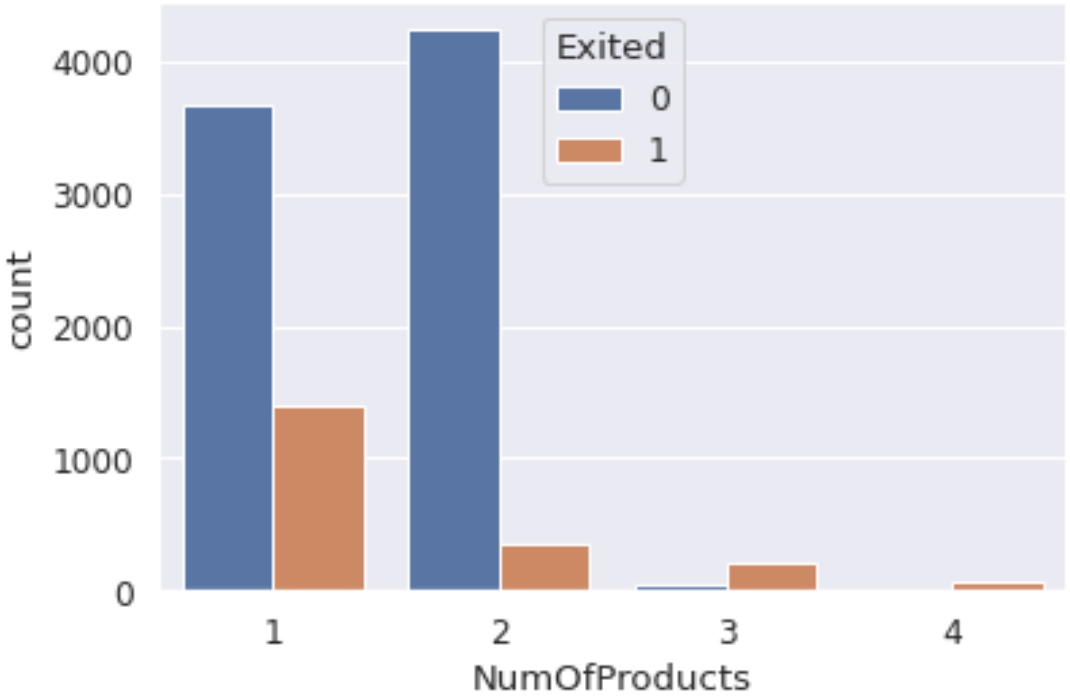

```
sns.kdeplot(ds['Balance'], data=ds, hue=ds['HasCrCard'], fill='true')
<matplotlib.axes._subplots.AxesSubplot at 0x7fe81ee78110>
```



Count Plot with Hue

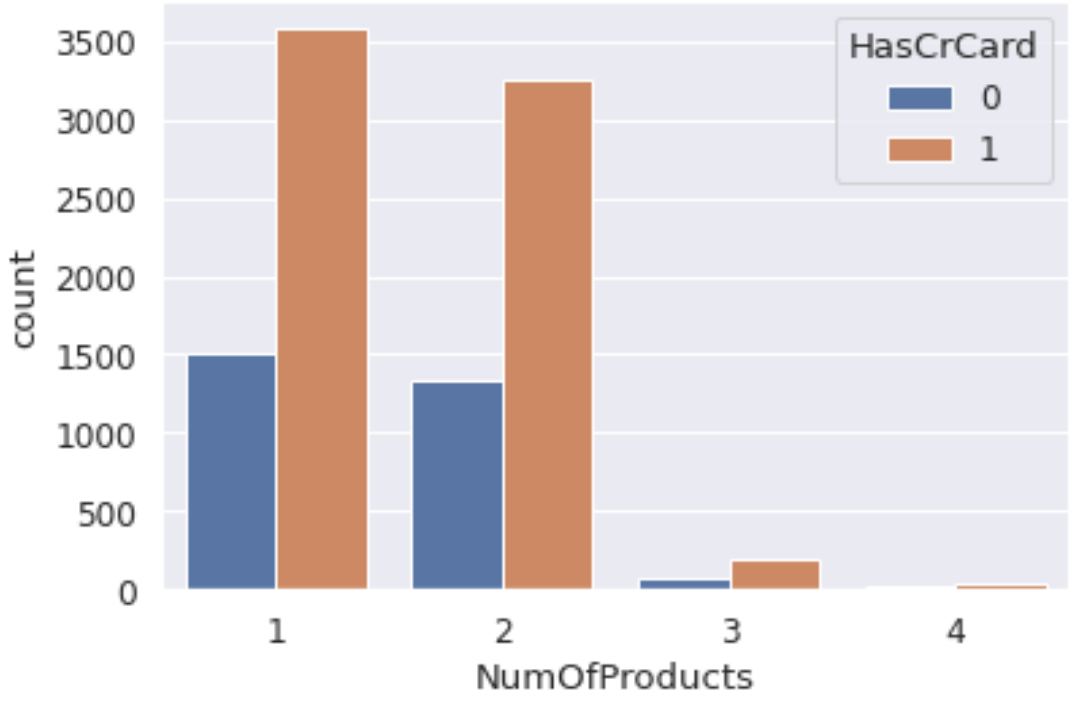
```
sns.countplot(ds['NumOfProducts'],hue=ds['Exited'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe818854610>
```




```
sns.countplot(ds['NumOfProducts'],hue=ds['HasCrCard'])
```

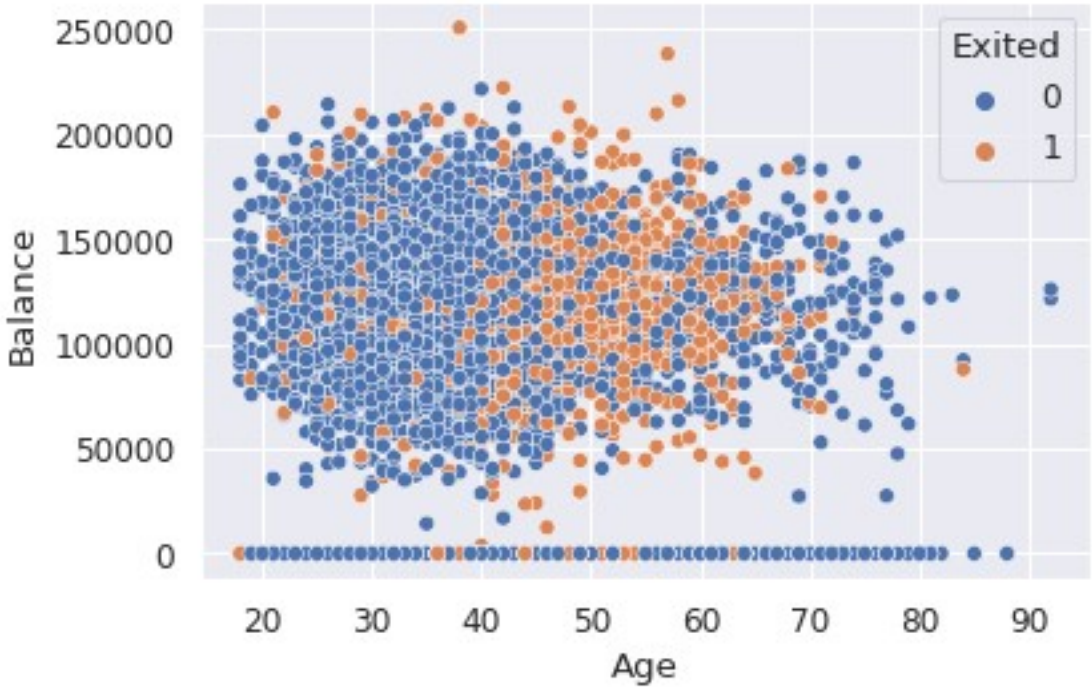
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8187d1d10>
```



Scatter Plot

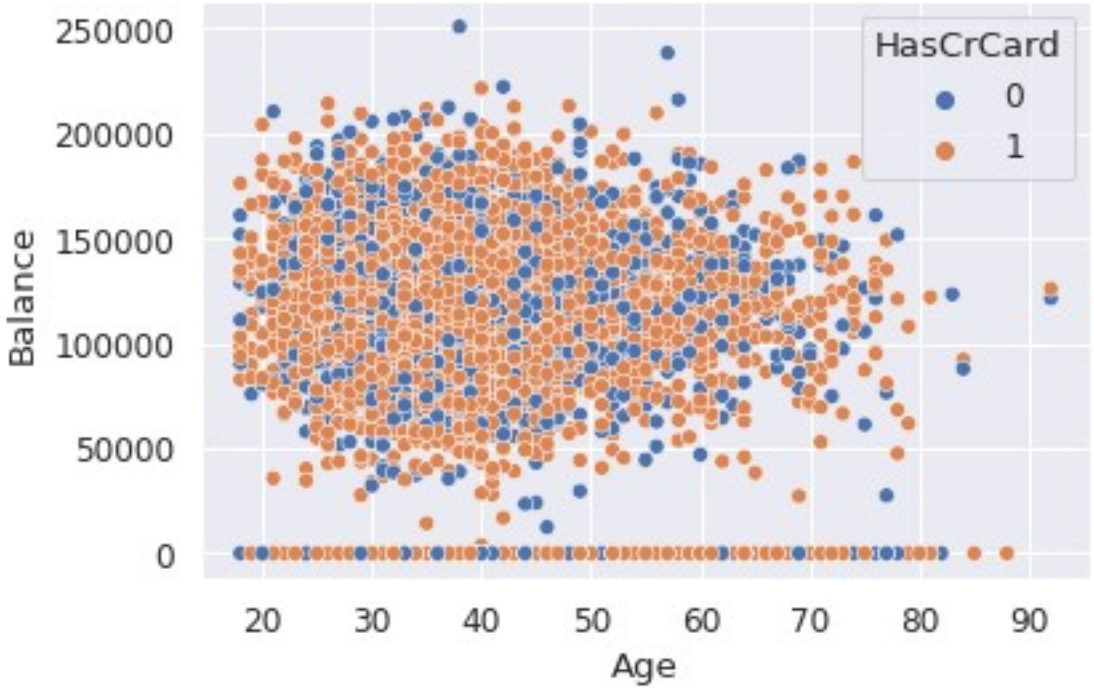
```
sns.scatterplot(x=ds.Age, y=ds.Balance, hue=ds.Exited)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe81eaf9e10>
```




```
sns.scatterplot(x=ds.Age, y=ds.Balance, hue=ds.HasCrCard)
```

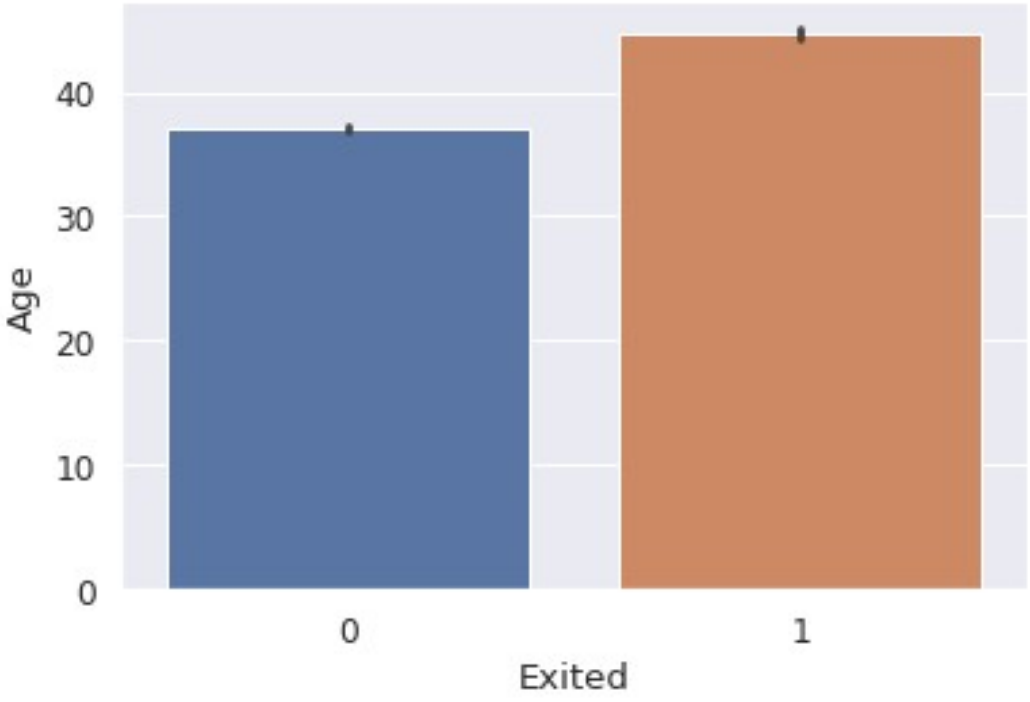
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe81ea14a10>
```



Bar Plot

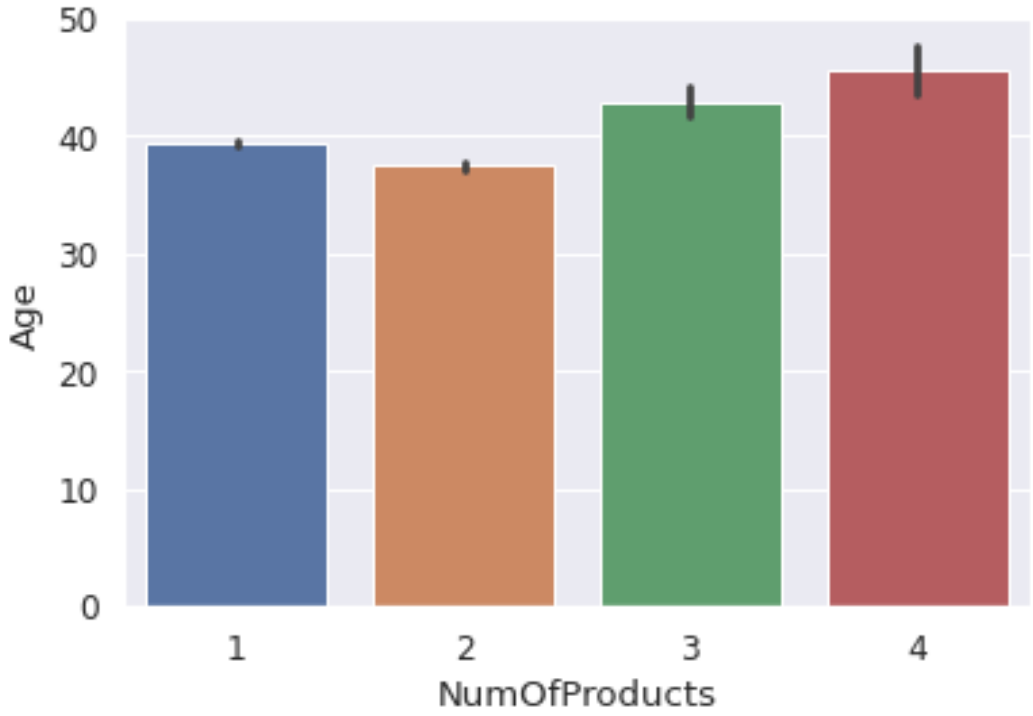
```
sns.barplot(ds["Exited"],ds["Age"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe818745e10>
```



```
sns.barplot(ds["NumOfProducts"],ds["Age"])
```

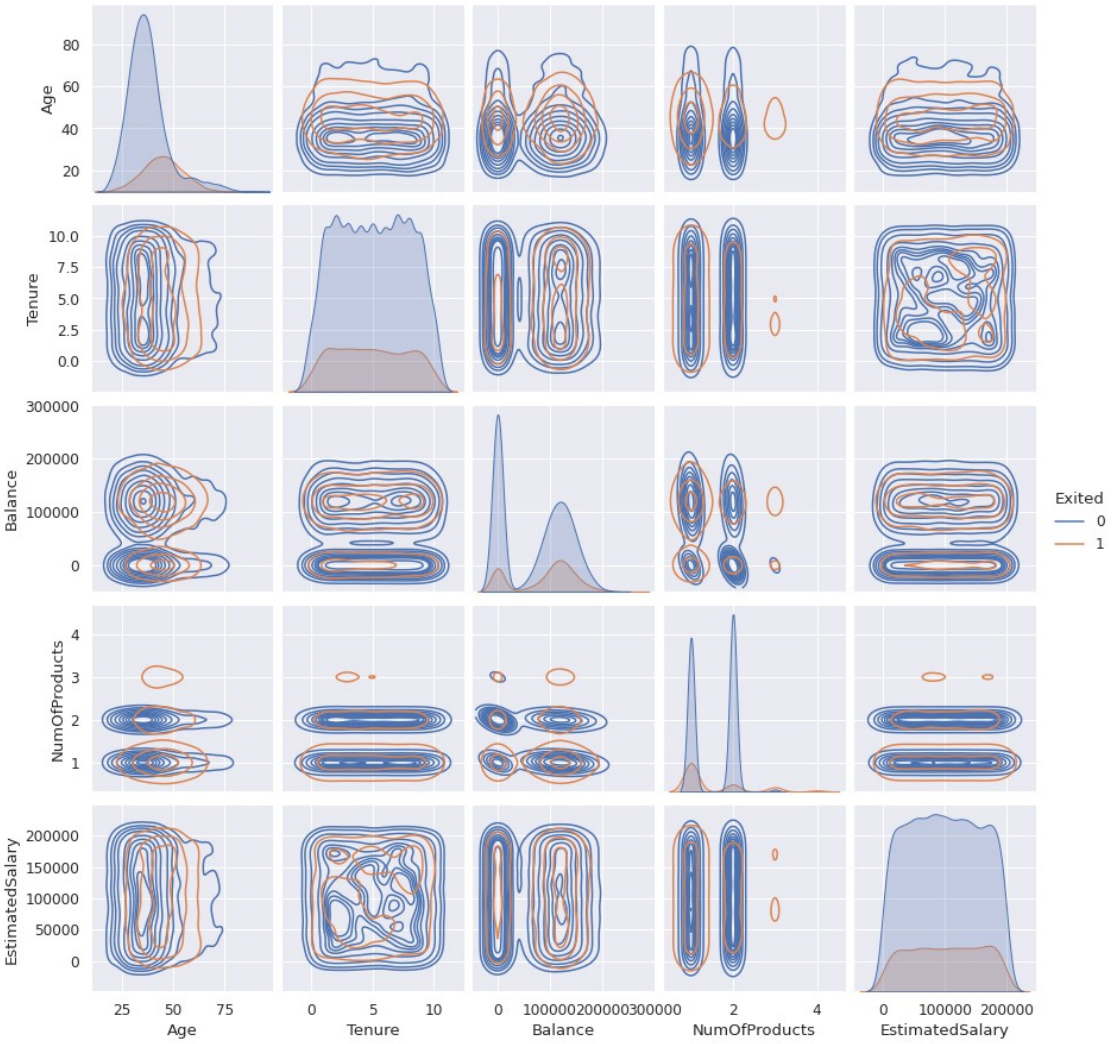
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe818725b10>
```



Multi - Variate Analysis

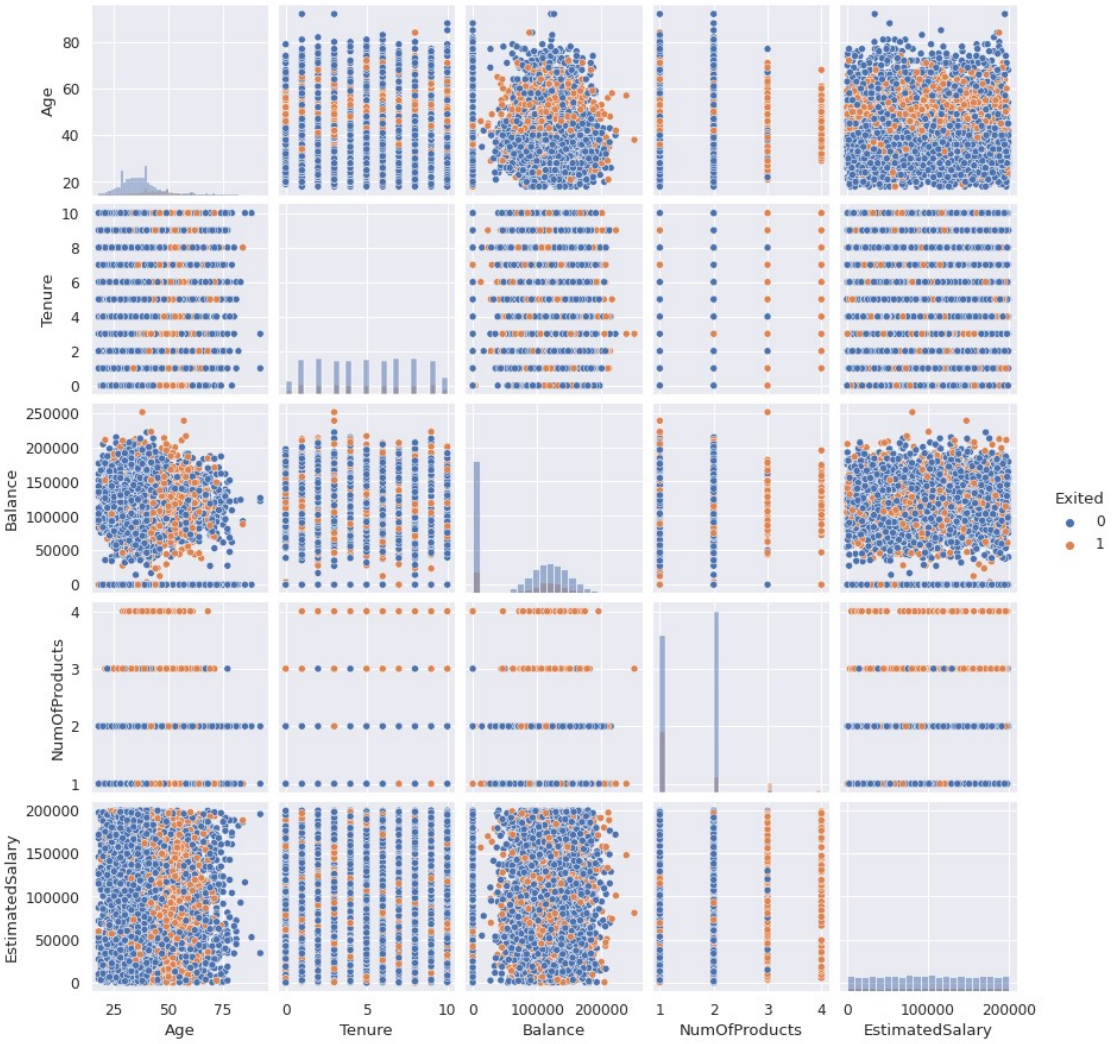
```
sns.pairplot(data=ds[["Age","Tenure","Balance","NumOfProducts","EstimatedSalary","Exited"]], hue="Exited", kind="kde")
```

```
<seaborn.axisgrid.PairGrid at 0x7fe81d0e7350>
```




```
sns.pairplot(data=ds[["Age","Tenure","Balance","NumOfProducts","EstimatedSalary","Exited"]], hue="Exited", diag_kind="hist")
```

```
<seaborn.axisgrid.PairGrid at 0x7fe81cbf7750>
```



4. Perform descriptive statistics on the dataset.

```
ds.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

             Balance  NumOfProducts    HasCrCard  IsActiveMember  \
count   10000.000000   10000.000000  10000.00000    10000.000000
mean    76485.889288       1.530200      0.70550        0.515100
std     62397.405202       0.581654      0.45584        0.499797
min         0.000000       1.000000      0.00000        0.000000
25%         0.000000       1.000000      0.00000        0.000000
50%     97198.540000       1.000000      1.00000        1.000000
75%    127644.240000       2.000000      1.00000        1.000000
max    250898.090000       4.000000      1.00000        1.000000

       EstimatedSalary        Exited
count     10000.000000  10000.000000
mean     100090.239881      0.203700
std       57510.492818      0.402769
min          11.580000      0.000000
25%       51002.110000      0.000000
50%      100193.915000      0.000000
75%      149388.247500      0.000000
max      199992.480000      1.000000
```

5. Handle the Missing values.

```
ds.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
```

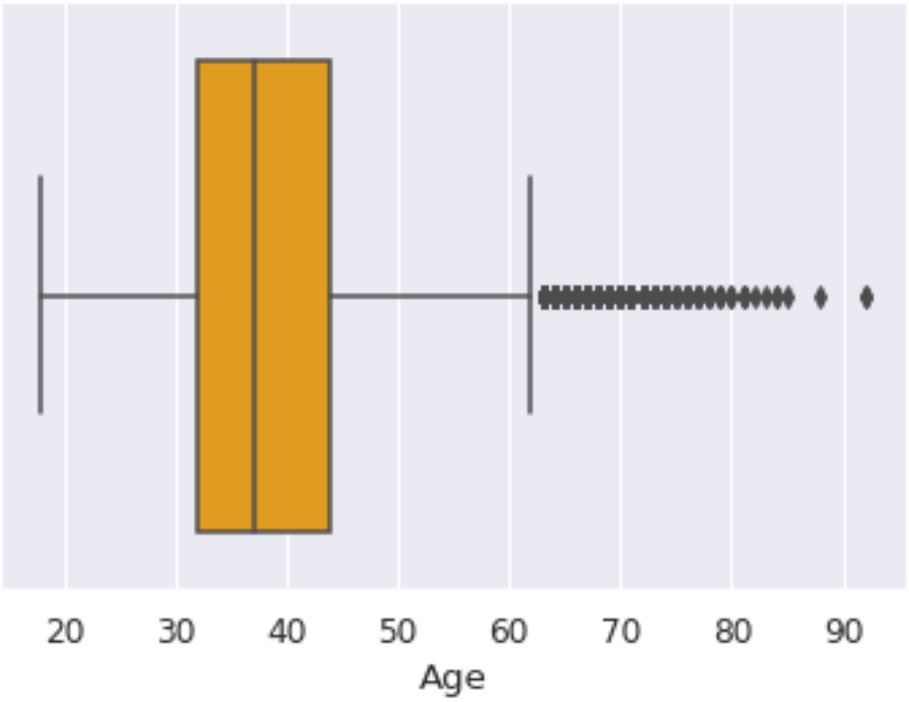
There is no null values in the dataset

6. Find the outliers and replace the outliers

Identifying Outliers

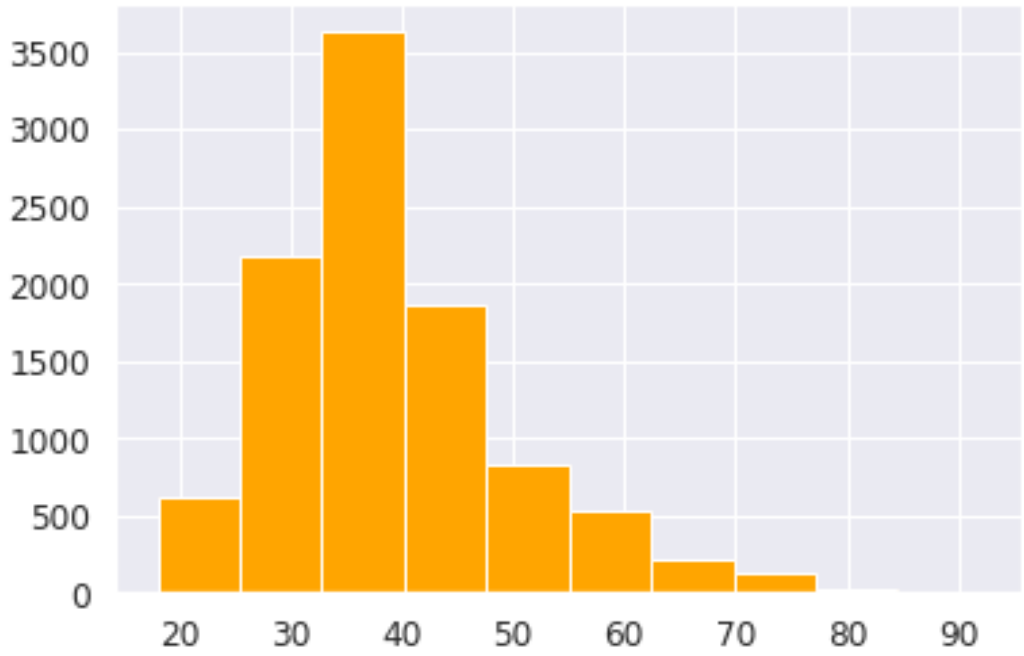
```
sns.boxplot(ds['Age'],data=ds,color='orange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe818248850>
```




```
ds['Age'].hist(color='orange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8181d1550>
```




```
print('skewness value of Age: ',ds['Age'].skew())
```

```
skewness value of Age:  1.0113202630234552
```

```
Q1=ds['Age'].quantile(0.25)
Q3=ds['Age'].quantile(0.75)
IQR=Q3 - Q1
IQR
```

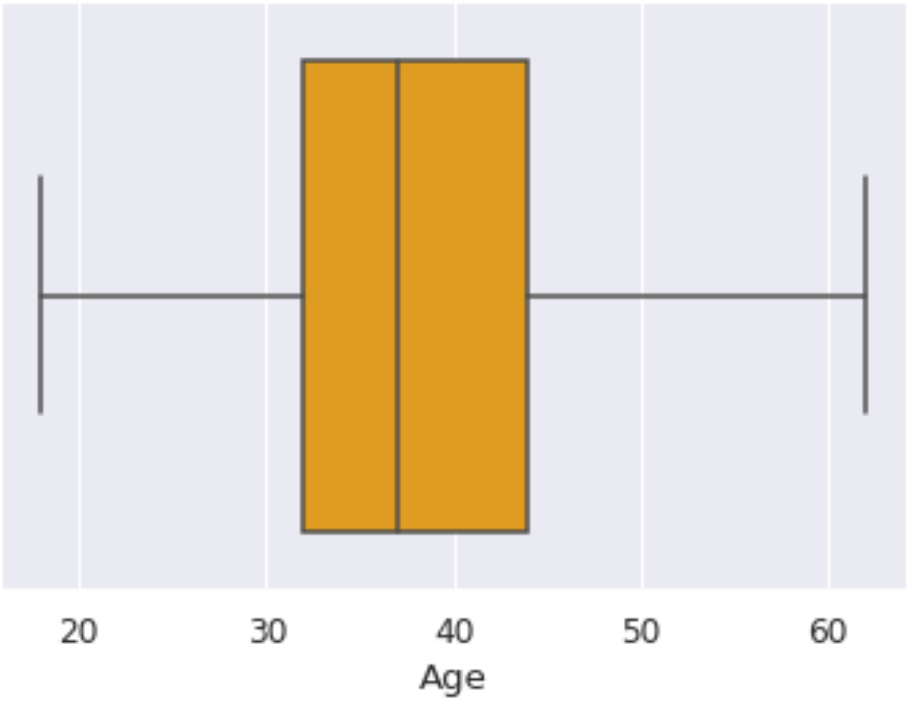
```
12.0
```

```
lower_limit = Q1 - (IQR * 1.5)
upper_limit = Q3 + (IQR * 1.5)
```

```
ds["Age"] = np.where(ds["Age"] > upper_limit, upper_limit,
np.where(ds["Age"] < lower_limit, lower_limit, ds["Age"]))
```

```
sns.boxplot(ds['Age'],data=ds,color='orange')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8181391d0>
```



7. Check for Categorical columns and perform encoding.

Check for Categorical Columns

```
ds.dtypes
```

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

```
ds["Geography"].value_counts()
```

```
France     5014
Germany    2509
Spain      2477
Name: Geography, dtype: int64
```

```
ds["Gender"].value_counts()
```

```
Male      5457
Female    4543
Name: Gender, dtype: int64
```

Replace with encoding

```
ds["Geography"].replace({"France":1,"Spain":2,"Germany":3},inplace =
True)
ds["Gender"].replace({"Female":0,"Male":1},inplace = True)
```

8. Split the data into dependent and independent variables.

```
dependent = ds['Exited']
dependent.head()
```

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

```
in_dependent = ds.drop(columns=["Exited","Surname"],axis=1)
in_dependent.head()
```

```
   RowNumber  CustomerId  CreditScore  Geography  Gender   Age  Tenure
\
0          1    15634602          619          1       0  42.0       2

1          2    15647311          608          2       0  41.0       1

2          3    15619304          502          1       0  42.0       8

3          4    15701354          699          1       0  39.0       1

4          5    15737888          850          2       0  43.0       2


     Balance  NumOfProducts  HasCrCard  IsActiveMember
EstimatedSalary
0       0.00              1          1               1
101348.88
1   83807.86              1          0               1
112542.58
2  159660.80              3          1               0
113931.57
3       0.00              2          0               0
93826.63
```

```
4  125510.82              1          1               1
79084.10
```

9. Scale the independent variables

```
index=in_dependent.columns
index
```

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

```
scaled_values = scale(in_dependent)
scaled_values
```

```
array([[-1.73187761, -0.78321342, -0.32622142, ...,  0.64609167,
         0.97024255,  0.02188649],
       [-1.7315312 , -0.60653412, -0.44003595, ..., -1.54776799,
         0.97024255,  0.21653375],
       [-1.73118479, -0.99588476, -1.53679418, ...,  0.64609167,
        -1.03067011,  0.2406869 ],
       ...,
       [ 1.73118479, -1.47928179,  0.60498839, ..., -1.54776799,
         0.97024255, -1.00864308],
       [ 1.7315312 , -0.11935577,  1.25683526, ...,  0.64609167,
        -1.03067011, -0.12523071],
       [ 1.73187761, -0.87055909,  1.46377078, ...,  0.64609167,
        -1.03067011, -1.07636976]])
```

```
scaled_df = pd.DataFrame(scaled_values,columns=index)
scaled_df
```

```
      RowNumber  CustomerId  CreditScore  Geography    Gender
Age  \
0     -1.731878   -0.783213    -0.326221  -0.902587 -1.095988
0.342615
1     -1.731531   -0.606534    -0.440036   0.301665 -1.095988
0.240011
2     -1.731185   -0.995885    -1.536794  -0.902587 -1.095988
0.342615
3     -1.730838    0.144767     0.501521  -0.902587 -1.095988
0.034803
4     -1.730492    0.652659     2.063884   0.301665 -1.095988
0.445219
...         ...         ...          ...        ...       ...       ..
.
9995   1.730492   -1.177652     1.246488  -0.902587  0.912419
0.034803
```

```
9996   1.730838   -1.682806    -1.391939  -0.902587  0.912419 -
0.375612
9997   1.731185   -1.479282     0.604988  -0.902587 -1.095988 -
0.273008
9998   1.731531   -0.119356     1.256835   1.505917  0.912419
0.342615
9999   1.731878   -0.870559     1.463771  -0.902587 -1.095988 -
1.093840

        Tenure   Balance  NumOfProducts  HasCrCard  IsActiveMember  \
0    -1.041760 -1.225848      -0.911583   0.646092        0.970243
1    -1.387538  0.117350      -0.911583  -1.547768        0.970243
2     1.032908  1.333053       2.527057   0.646092       -1.030670
3    -1.387538 -1.225848       0.807737  -1.547768       -1.030670
4    -1.041760  0.785728      -0.911583   0.646092        0.970243
...        ...       ...            ...        ...             ...
9995 -0.004426 -1.225848       0.807737   0.646092       -1.030670
9996  1.724464 -0.306379      -0.911583   0.646092        0.970243
9997  0.687130 -1.225848      -0.911583  -1.547768        0.970243
9998 -0.695982 -0.022608       0.807737   0.646092       -1.030670
9999 -0.350204  0.859965      -0.911583   0.646092       -1.030670

      EstimatedSalary
0            0.021886
1            0.216534
2            0.240687
3           -0.108918
4           -0.365276
...               ...
9995        -0.066419
9996         0.027988
9997        -1.008643
9998        -0.125231
9999        -1.076370

[10000 rows x 12 columns]
```

10. Split the data into training and testing

```
x = scaled_df
y = dependent

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,rando
m_state=0)

x_train.head()
```

```
      RowNumber  CustomerId  CreditScore  Geography    Gender
Age  \
7389   0.827747   -0.195066     0.170424   0.301665 -1.095988 -
```

```
0.478216
9275   1.481077    0.810821    -2.312802   1.505917  0.912419
0.342615
2995  -0.694379   -1.507642    -1.195351  -0.902587 -1.095988 -
0.991236
5316   0.109639    1.243462     0.035916   0.301665  0.912419
0.137407
356   -1.608556   -1.100775     2.063884   0.301665 -1.095988
1.881674

        Tenure   Balance  NumOfProducts  HasCrCard  IsActiveMember  \
7389 -0.004426 -1.225848       0.807737   0.646092       -1.030670
9275 -1.387538 -0.012892      -0.911583   0.646092        0.970243
2995 -1.041760  0.575076      -0.911583   0.646092       -1.030670
5316 -0.004426  0.467955      -0.911583   0.646092       -1.030670
356   1.032908  0.806010       0.807737   0.646092        0.970243

      EstimatedSalary
7389         1.108382
9275        -0.747592
2995         1.487464
5316         1.278558
356          0.560069

x_train.shape,y_train.shape,x_test.shape,y_test.shape

((8000, 12), (8000,), (2000, 12), (2000,))
```