In [1]:
```
1.Download the Dataset
```

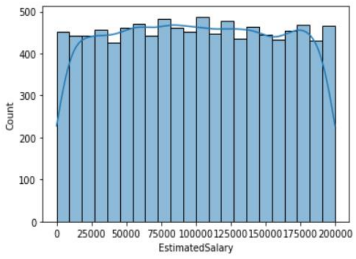
In [ ]:
```
2.Load the dataset
```

In [6]:
```
import numpy as np
import pandas as pd
df = pd.read_csv("Churn_Modelling.csv")
```

In [ ]:
```
3.Univariate Analysis
```

In [7]:
```
import seaborn as sns
sns.histplot(df.EstimatedSalary,kde=True)
```

Out[7]:

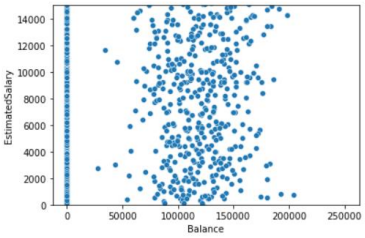


In [ ]:
```
b. Bivariate Analysis
```

In [8]:
```
import seaborn as sns
import matplotlib.pyplot as plt
sns.scatterplot(df.Balance,df.EstimatedSalary)
plt.ylim(0,15000)
```

```
C:\anaconda\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variables as keyword args: x, y. From version 0.12, the only
valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

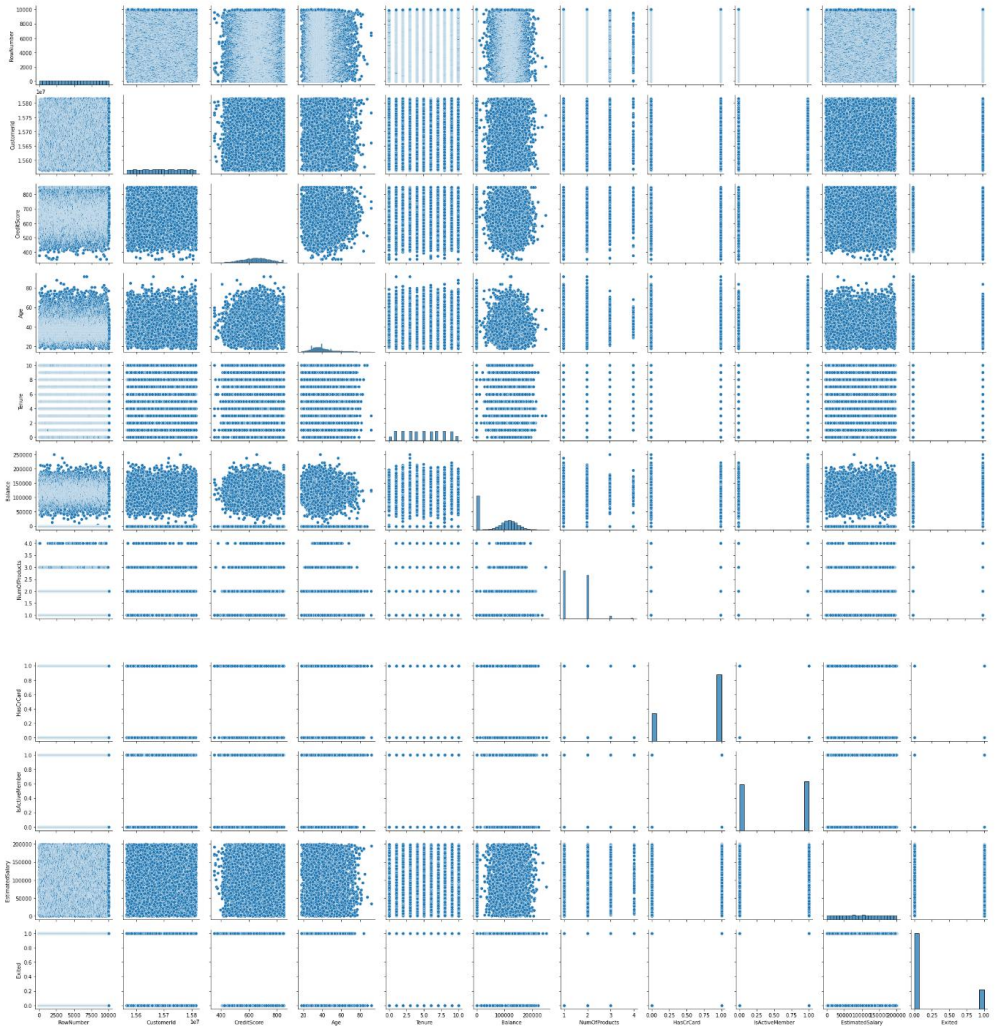
Out[8]:
```
(0.0, 15000.0)
```



In [ ]:
```
c. Multivariate Analysis
```

In [9]:
```
import seaborn as sns
sns.pairplot(df)
```

Out[9]:



In [ ]:
```
4. Perform descriptive statistics on the dataset
```

In [10]:
```
df.describe(include='all')
```

Out[10]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	Estim
count	10000.00000	1.000000e+04	10000	10000.000000	10000	10000	10000.000000	10000.000000	10000.000000	10000.000000	10000.00000	10000.000000	10(
unique	NaN	NaN	2932	NaN	3	2	NaN	NaN	NaN	NaN	NaN	NaN	
top	NaN	NaN	Smith	NaN	France	Male	NaN	NaN	NaN	NaN	NaN	NaN	
freq	NaN	NaN	32	NaN	5014	5457	NaN	NaN	NaN	NaN	NaN	NaN	
mean	5000.50000	1.569094e+07	NaN	650.528800	NaN	NaN	38.921800	5.012800	76485.889288	1.530200	0.70550	0.515100	100
std	2886.89568	7.193619e+04	NaN	96.653299	NaN	NaN	10.487806	2.892174	62397.405202	0.581654	0.45584	0.499797	57
min	1.00000	1.556570e+07	NaN	350.000000	NaN	NaN	18.000000	0.000000	0.000000	1.000000	0.00000	0.000000	
25%	2500.75000	1.562853e+07	NaN	584.000000	NaN	NaN	32.000000	3.000000	0.000000	1.000000	0.00000	0.000000	51
50%	5000.50000	1.569074e+07	NaN	652.000000	NaN	NaN	37.000000	5.000000	97198.540000	1.000000	1.00000	1.000000	100
75%	7500.25000	1.575323e+07	NaN	718.000000	NaN	NaN	44.000000	7.000000	127644.240000	2.000000	1.00000	1.000000	149
max	10000.00000	1.581569e+07	NaN	850.000000	NaN	NaN	92.000000	10.000000	250898.090000	4.000000	1.00000	1.000000	199

In [ ]:
```
5.Handle the missing values
```

In [11]:
```
from ast import increment_lineno
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(color_codes=True)
df.head()
```

Out[11]:

	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

In [ ]:
```
6. Find the outliers and replace the outliers
```

In [12]:
```
import pandas as pd
import matplotlib
from matplotlib import pyplot as pyplot
%matplotlib inline
matplotlib.rcParams['figure.figsize']=(10,6)
df.sample(5)
```

Out[12]:

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
5211	5212	15662263	Castillo	749	Germany	Male	22	4	94762.16	2	1	1	42241.54	0
595	596	15680970	Lombardi	611	Germany	Female	41	2	114206.84	1	1	0	164061.60	0
136	137	15802381	Li	461	Germany	Female	34	5	63663.93	1	0	1	167784.28	0
2918	2919	15649487	Sal	578	Germany	Female	38	4	113150.44	2	1	0	176712.59	1
6294	6295	15742824	Isayeva	696	Germany	Male	42	7	162318.61	1	1	0	121061.89	0

In [ ]:
```
7. Check for Categorical columns and perform encoding
```

In [13]:
```
headers=['RowNumber','CustomerID','Surname','CreditScore','Geography',
 'Gender','Age','Tenure','Balance','NumofProducts','HasCard'
 'IsActiveMember','EstimatedSalary','Exited']
df.head()
```

Out[13]:

	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

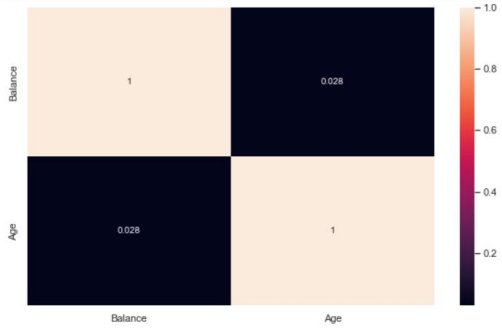
In [ ]:
```
8. Split the data into dependent and independent variables
```

In [14]:
```
x=df.iloc[:,:-1].values
print(x)
y=df.iloc[:,-1]._values
print(y)
```

```
[[1 15634602 'Hargrave' ... 1 1 101348.88]
 [2 15647311 'Hill' ... 0 1 112542.58]
 [3 15619304 'Onio' ... 1 0 113931.57]
 ...
 [9998 15584532 'Liu' ... 0 1 42085.58]
 [9999 15682355 'Sabbatini' ... 1 0 92888.52]
 [10000 15628319 'Walker' ... 1 0 38190.78]]
[1 0 1 ... 1 1 0]
```

In [ ]:
```
9. Scale the independent variables
```

In [15]:
```
dff=df[['Balance','Age']]
sns.heatmap(dff.corr(), annot=True)
sns.set(rc={'figure.figsize':(40,40)})
```



In [ ]:
```
10. Split the data into training and testing
```

In [17]:
```
from scipy.sparse.construct import random
x=df.iloc[:, 1:2].values
y=df.iloc[:,2].values
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('Row count of x_train table'+'->>'+str(f"{len(x_train):,}"))
print('Row count of y_train table'+'->>'+str(f"{len(y_train):,}"))
print('Row count of x_test table'+'->>'+str(f"{len(x_test):,}"))
print('Row count of y_test table'+'->>'+str(f"{len(y_test):,}"))
```

```
Row count of x_train table->>8,000
Row count of y_train table->>8,000
Row count of x_test table->>2,000
Row count of y_test table->>2,000
```