

Assignment -2

Python Programming

Assignment Date	19 September 2022
Student Name	S.Ronald Jim Roy
Student Roll Number	210519205041
Maximum Marks	2 Marks

```
▶ #!load the dataset
  from google.colab import drive

[1] Python

drive.mount('/content/drive')

[2] Python
... Mounted at /content/drive

from pandas.io.formats.style import plt
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import rcParams

[3] Python

df=pd.read_csv('/content/drive/MyDrive/IBM/dataset')

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

[5] Python
... Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

df.head()

[6] Python
...

#6 check for categorical column and perform encoding
dummy=pd.get_dummies(df['Gender'])
dummy.head()

[7] Python
...

df2=pd.concat((df,dummy),axis=1)
df2.head()

[8] Python
```

```
[9] df2.drop(['Gender'],axis=1) Python
```

...

```
[10] df2=df2.drop(['Gender'],axis=1)
df2.head() Python
```

...

```
[11] df2=df2.drop(['Male'],axis=1)
df2.head() Python
```

...

```
[12] df2.rename(columns={"Female":"Gender"}) Python
```

...

```
[13] df.shape Python
```

... (10000, 14)

```
[14] df.info() Python
```

```
... <class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   RowNumber        10000 non-null  int64
1   CustomerId       10000 non-null  int64
2   Surname          10000 non-null  object
3   CreditScore      10000 non-null  int64
4   Geography        10000 non-null  object
5   Gender           10000 non-null  object
6   Age              10000 non-null  int64
7   Tenure           10000 non-null  int64
8   Balance          10000 non-null  float64
9   NumOfProducts   10000 non-null  int64
10  HasCrCard        10000 non-null  int64
11  IsActiveMember   10000 non-null  int64
12  EstimatedSalary  10000 non-null  float64
13  Exited           10000 non-null  int64
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

```
[15] #4 Handling Missing Values
df.isnull().any() Python
```

```
... 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()
```

[16]

Python

```
... 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().sum().sum()
```

[17]

Python

```
... 0
```

```
#7 split the data into independent and dependent variable
y=df2['Tenure']
y
```

[18]

Python

```
... 0      2
    1      1
    2      8
    3      1
    4      2
    ..
    ..
```

```
9995      5
9996     10
9997      7
9998      3
9999      4
```

Name: Tenure, Length: 10000, dtype: int64

```
X=df.drop(columns=['Balance'],axis=1)
X.head()
```

[19]

Python

```
...
```

```
X=df.iloc[:,7:10]
X
```

[20]

Python

```
...
```

```
print(X.shape)
```

```
... (10000, 3)
```

```
Y=df.iloc[:,7]  
Y
```

```
[22]
```

```
Python
```

```
...  
0      2  
1      1  
2      8  
3      1  
4      2  
..  
9995    5  
9996   10  
9997    7  
9998    3  
9999    4  
Name: Tenure, Length: 10000, dtype: int64
```

```
print(y.shape)
```

```
... (10000,)
```

```
#8 Scale the independent variable  
from sklearn.preprocessing import scale  
x_scaled=pd.DataFrame(scale(X),columns=X.columns)  
x_scaled.head()
```

```
[24]
```

```
Python
```

```
...
```

```
#9 Train test split  
from sklearn.model_selection import train_test_split
```

```
[25]
```

```
Python
```

```
# Split data (train & test data)  
X_train, X_test, y_train, y_test = train_test_split(X,y)
```

```
[26]
```

```
Python
```

```
#display shape  
print(X_train.shape)  
print(X_test.shape)  
print(y_train.shape)  
print(y_test.shape)
```

```
[27]
```

```
Python
```

```
... (7500, 3)  
(2500, 3)  
(7500,)  
(2500,)
```

```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.30,random_state=0)  
print(X_train.shape)  
print(X_test.shape)  
print(y_train.shape)  
print(y_test.shape)
```

```
[28]
```

```
Python
```

```
... (7000, 3)  
(3000, 3)  
(7000,)  
(3000,)
```

```

    print(np.mean(X_train))
    print(np.mean(X_test))
[29] Python

...
Tenure      4.989429
Balance    75204.853421
NumOfProducts  1.534286
dtype: float64
Tenure      5.067333
Balance    79474.972977
NumOfProducts  1.520667
dtype: float64

#3 Descriptive stastical analysis
df.describe()
[30] Python

...

df.NumOfProducts.value_counts()
[31] Python

...
1    5084
2    4590
3     266
4      60
Name: NumOfProducts, dtype: int64

df.corr()
[32] Python

...

df['Age'].mean()
[33] Python

...
38.9218

df['Gender'].value_counts()
[34] Python

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

#2 Visualization

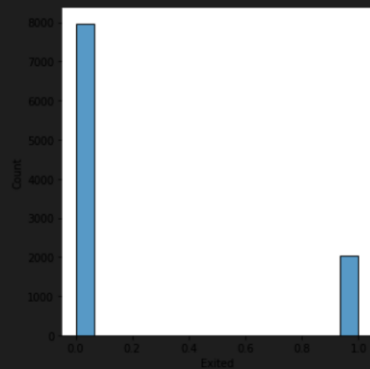
#univariate analysis
sns.displot(df.Exited)
[61] Python

...
<seaborn.axisgrid.FacetGrid at 0x7f8a972818d0>

```

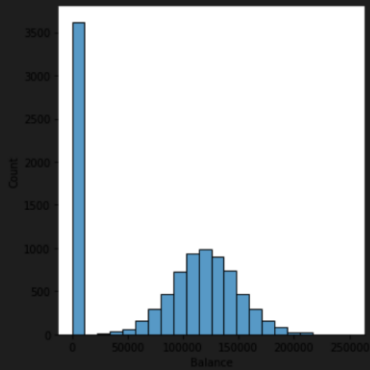
```
<seaborn.axisgrid.FacetGrid at 0x7f8a972818d0>
```

```
</>
```



```
sns.displot(df.Balance)
```

```
</>
```



```
df.hist(figsize=(10,10))
```

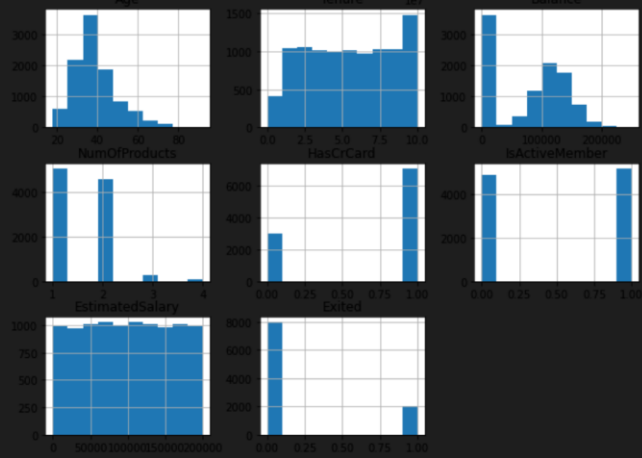
```
[37]
```

Python

```
<> array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9ca3a150>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c9ee790>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c9a4d90>],  
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c96a3d0>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c924cd0>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c8e9550>],  
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c8a1d10>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c869290>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c8692d0>],  
[<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c81c550>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c7fdb50>,  
<matplotlib.axes._subplots.AxesSubplot object at 0x7f8a9c74b690>]],  
dtype=object)
```

```
</>
```

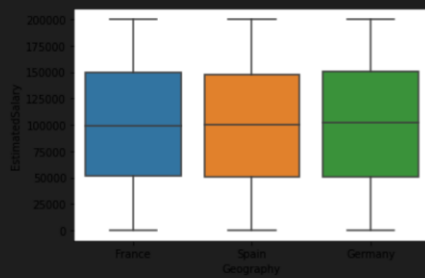




```
#Boxplot
sns.boxplot(x='Geography',y='EstimatedSalary',data=df)
plt.show()
```

[62]

Python



```
# Bivariate Analysis on continuous variable
sns.lineplot(df.Age,df.Exited)
```

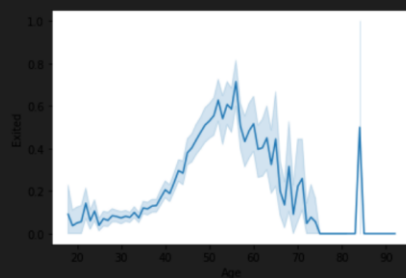
...

/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 argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7f8a9c438650>

</>

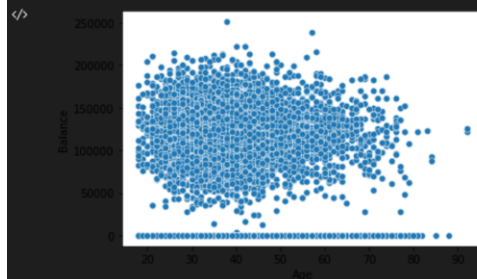


```
sns.scatterplot(df.Age,df.Balance)
```

```
... /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 argument will be `data`, and passing other arguments without an explicit keyword will result in an error
or misinterpretation.
```

```
FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8a9c308b10>
```



```
# Bivariate Analysis on categorical variable
sns.barplot(df.Tenure.value_counts())
```

```
[41]
```

```
Python
```

```
... /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 will be `data`, and passing other arguments without an explicit keyword will result in an error or
misinterpretation.
```

```
FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8a9c313150>
```

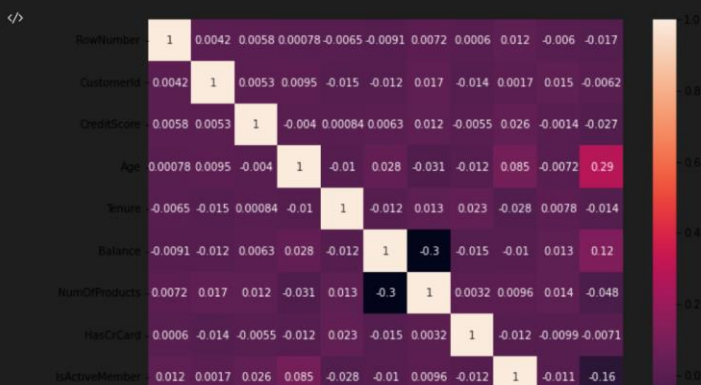


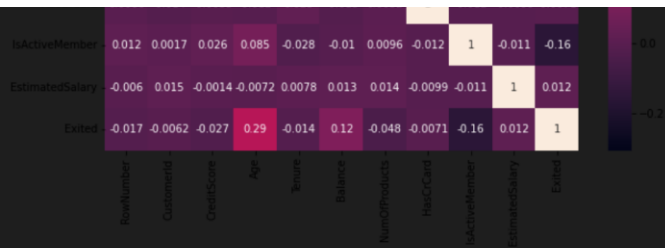
```
fig=plt.figure(figsize=(10,8))
sns.heatmap(df.corr(),annot=True)
```

```
[42]
```

```
Python
```

```
... <matplotlib.axes._subplots.AxesSubplot at 0x7f8a9c313050>
```





```
sns.barplot(df.NumOfProducts.value_counts().index,df.NumOfProducts.value_counts())
```

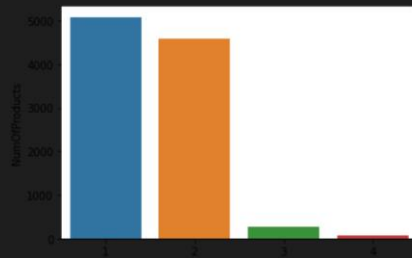
[43]

Python

```
... /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 argument will be `data`, and passing other arguments without an explicit keyword will result in an error or
misinterpretation.
FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8a9c04b6d0>
```

</>



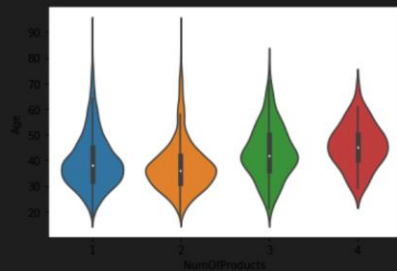
```
sns.violinplot(x='NumOfProducts',y='Age',data=df,size=8)
plt.show
```

[65]

Python

```
... <function matplotlib.pyplot.show(*args, **kw)>
```

</>



```
sns.boxplot(df.Age)
```

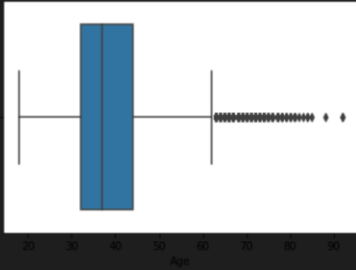
[45]

Python

```
... /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 will be `data`, and passing other arguments without an explicit keyword will result in an error or
misinterpretation.
FutureWarning
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f8a9a8329d0>

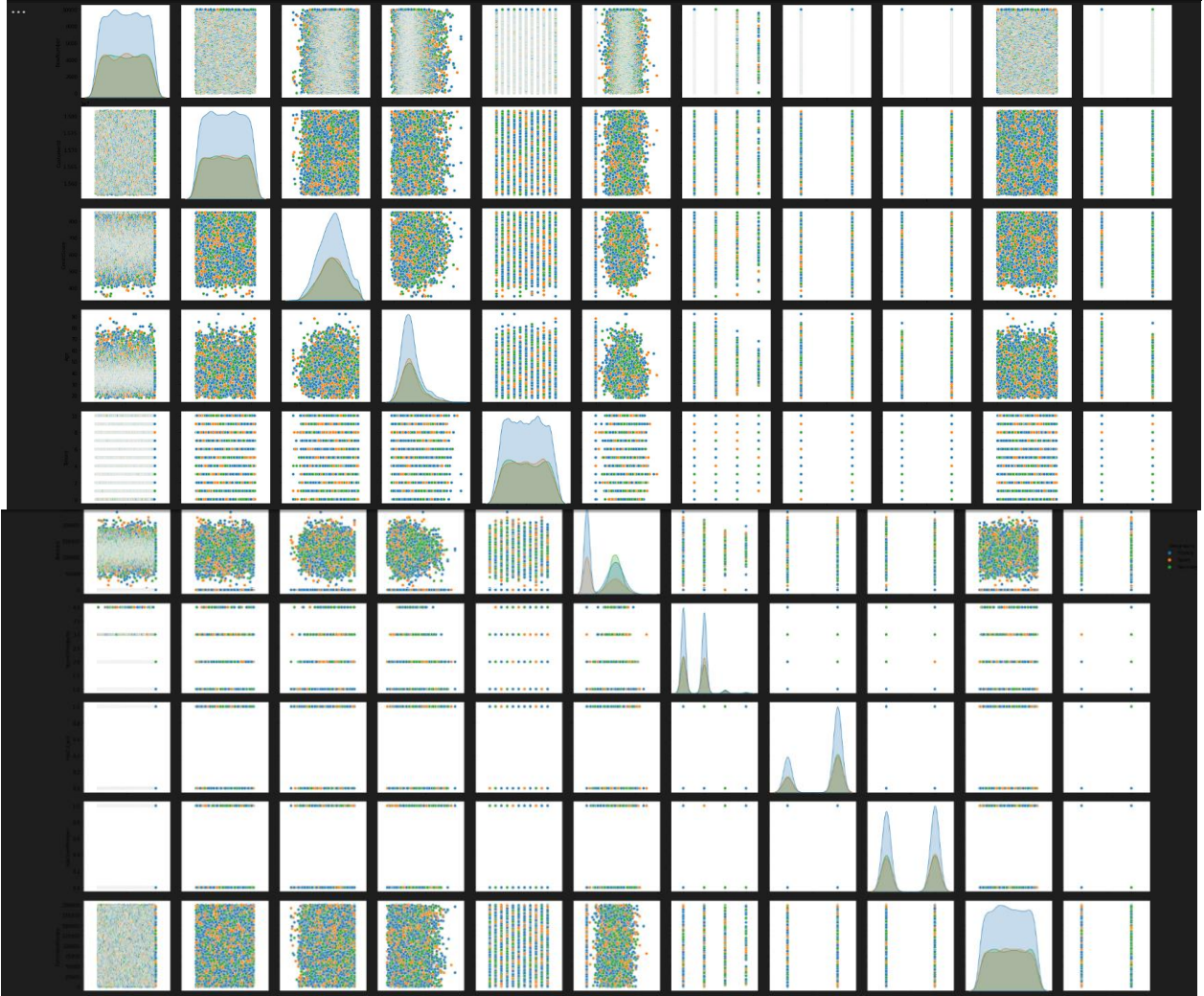
</>



```
#Multivariate analysis
sns.pairplot(data=df,hue='Geography',height=3)
plt.show()
```

[66]

Python





```
#5 Find the outlier and replace the outlier
#find the limits
upper_limit=df['Age'].mean()+3*df['Age'].std()
lower_limit=df['Age'].mean()-3*df['Age'].std()
print('upper_limit',upper_limit)
print('lower_limit',lower_limit)
```

[47]

Python

```
... upper_limit 70.38521935511383
     lower_limit 7.458380644886169
```

```
df.loc[(df['Age']>upper_limit)|(df['Age']<lower_limit)]
```

[48]

Python

...

```
#trimming - delete the outlier
new_df=df.loc[(df['Age']<upper_limit)&(df['Age']>lower_limit)]
print('Before removing outlier:',len(df))
print('After removing outlier:',len(new_df))
```

[49]

Python

```
... Before removing outlier: 10000
     After removing outlier: 9867
```

```
sns.boxplot(new_df['Age'])
```

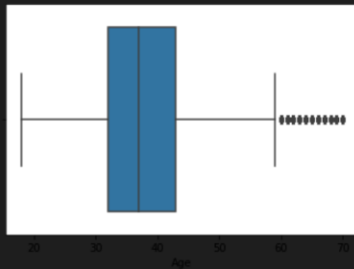
[50]

Python

```
... /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 will be `data`, and passing other arguments without an explicit keyword will result in an error or
misinterpretation.
FutureWarning
```

<matplotlib.axes._subplots.AxesSubplot at 0x7f8a97f01790>

</>



```
#copying-change the outlier value to upper or lower limits values
new_df=df.copy()
new_df.loc[(new_df['Age']>upper_limit),'Age']=upper_limit
new_df.loc[(new_df['Age']<lower_limit),'Age']=lower_limit
```

[51]

Python

```
sns.boxplot(new_df['Age'])
```

Python

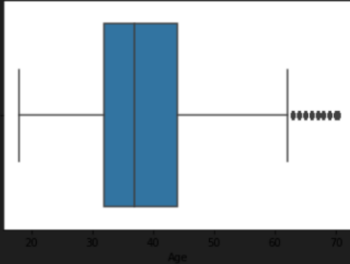
[52]

```
... /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 will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
```

FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7f8a98031ad0>

</>



```
len(new_df)
```

Python

[53]

```
... 10000
```

```
#IQR method
q1=df['Age'].quantile(0.25)
q3=df['Age'].quantile(0.75)
iqr=q3-q1
```

Python

[54]

```
q1,q3,iqr
```

Python

[55]

```
... (32.0, 44.0, 12.0)
```

```
upper_limit=q3+(1.5*iqr)
lower_limit=q3-(1.5*iqr)
lower_limit,upper_limit
```

Python

[56]

```
... (26.0, 62.0)
```

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

Python

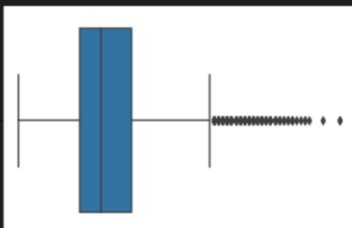
[57]

```
... /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 will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
```

FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7f8a98077ed0>

</>



```
#copying-change the outlier value to upper or lower limits values
new_df=df.copy()
new_df.loc[(new_df['Age']>upper_limit),'Age']=upper_limit
new_df.loc[(new_df['Age']<lower_limit),'Age']=lower_limit
```

[58]

Python

```
sns.boxplot(new_df['Age'])
```

[59]

Python

```
... /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 will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
```

```
FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8a97e1a690>
```

</>

