**Assignment -2**

Python Programming

|  |  |
| --- | --- |
| Assignment Date | 25 September 2022 |
| Student Name | Yashika U |
| Student Roll Number | 511919104025 |
| Maximum Marks | 2 Marks |

**Data Visualization and pre-processing**

**Question-1:**

1. Download dataset

**Solution:**

Dataset has been downloaded successfully

**Question-2:**

1. Load the dataset

**Solution:**

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

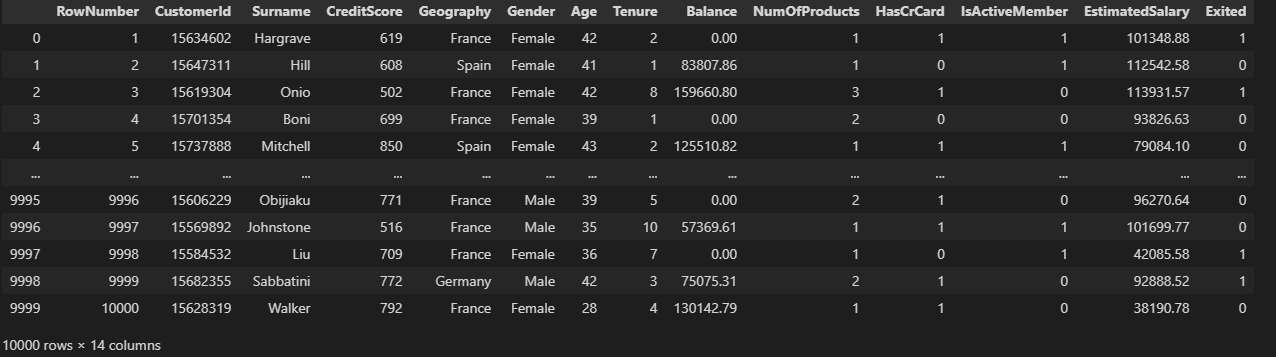
import seaborn as sns

%matplotlib inline

df = pd.read\_csv("Churn\_Modelling.csv")

df

**Output:**



**Question-3:**

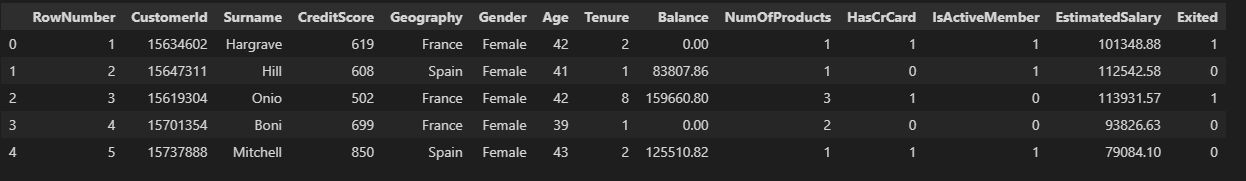
3. Perform below visualizations.

3.1. Univarient Analysis

**Solution-1:**

df.head()

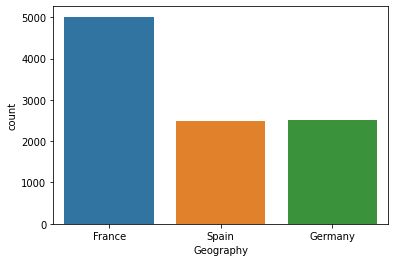
**Output:**



**Solution-2:**

sns.countplot(x='Geography',data=df)ggfddg

**Output:**



**Solution-3:**

sns.countplot(x='Gender',data=df)

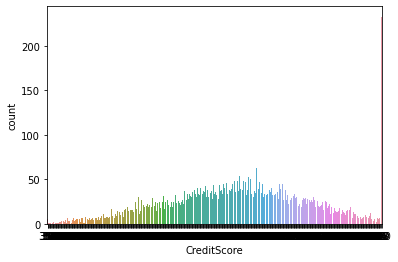
**Output:**



**Solution-4:**

sns.countplot(x='CreditScore',data=df)

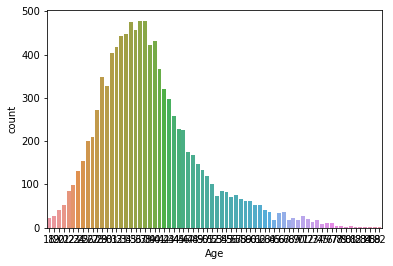
**Output:**



**Solution-5:**

sns.countplot(x='Age',data=df)

**Output:**

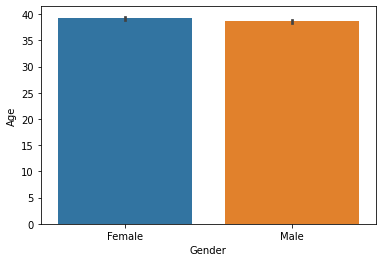


2.2.Bi- variant Analysis

**Solution-1:**

sns.barplot(x='Gender',y='Age',data=df)

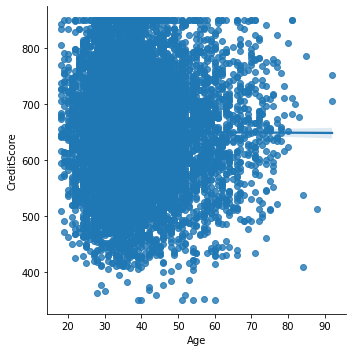
**Output:**



**Solution-2:**

sns.lmplot(x='Age',y='CreditScore',data=df)

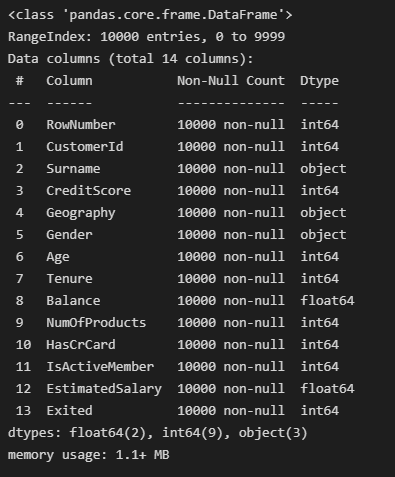
**Output:**



**Solution-3:**

df.info()

**Output:**

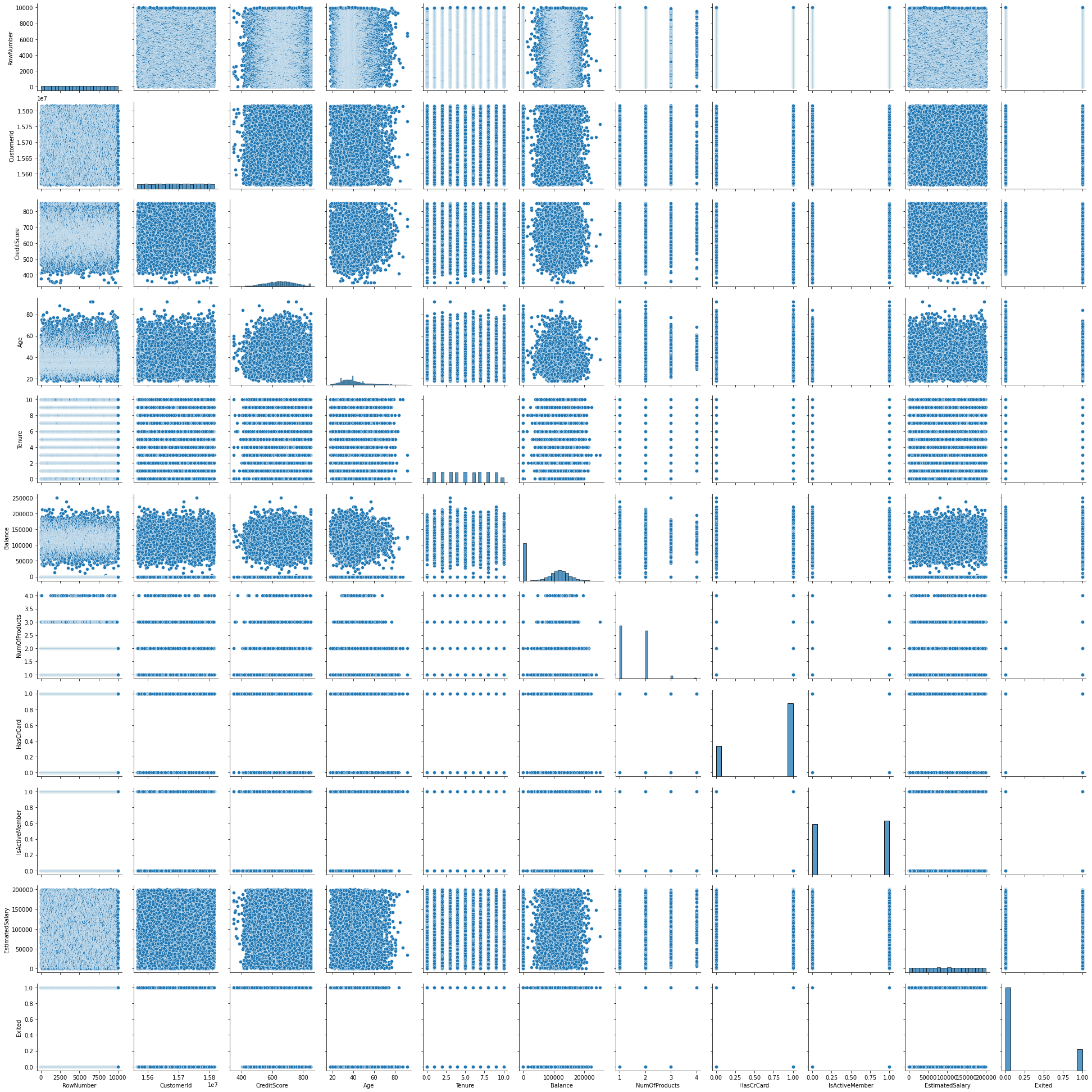


3.3.Multi-variate Analysis

**Solution:**

sns.pairplot(df)

**Output:**



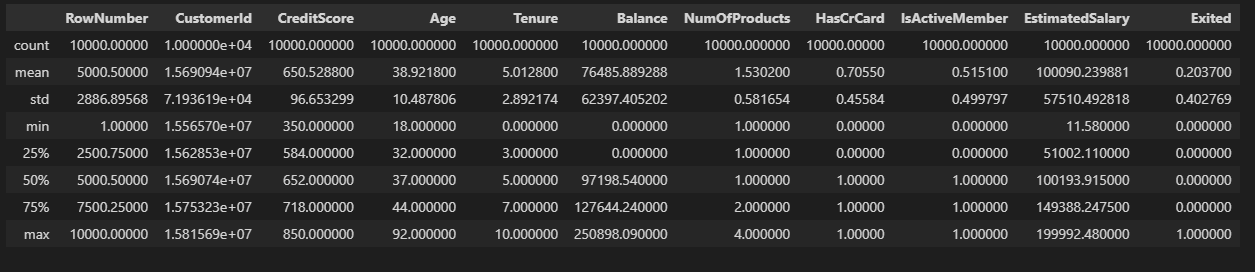
**Question-4:**

4. Perform descriptive statistics on the dataset

**Solution-1:**

df.describe()

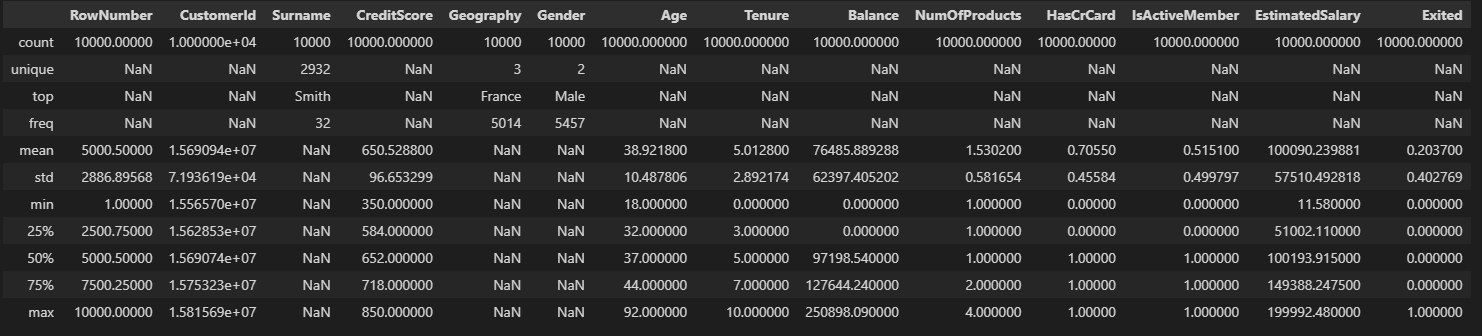
**Output:**



**Solution-2:**

df. describe(include='all')

**Output:**



**Question-5:**

5. Handle the missing values

**Solution-1:**

df.isnull()

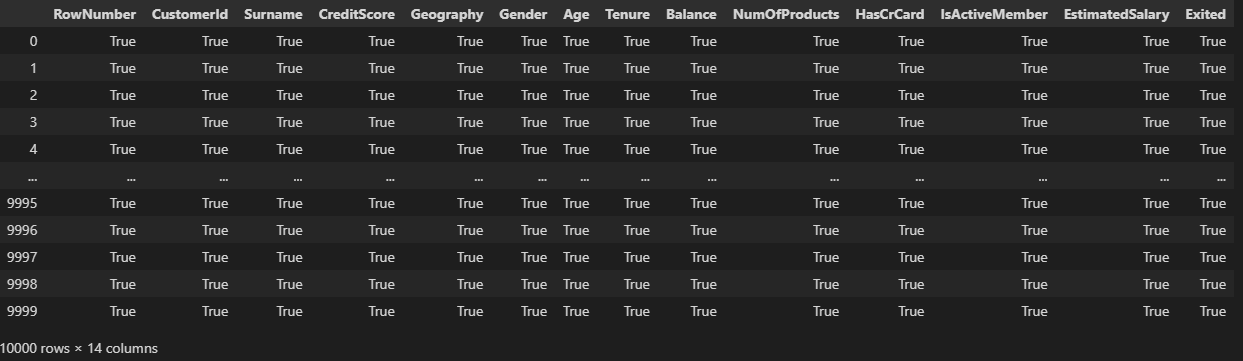
**Output:**



**Solution-2:**

df.notnull()

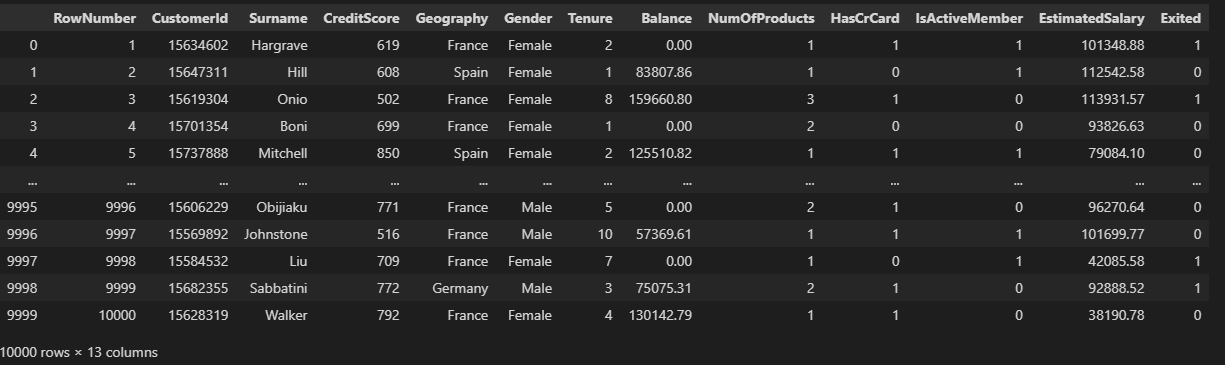
**Output:**

****

**Solution-3:**

df.dropna()

**Output:**

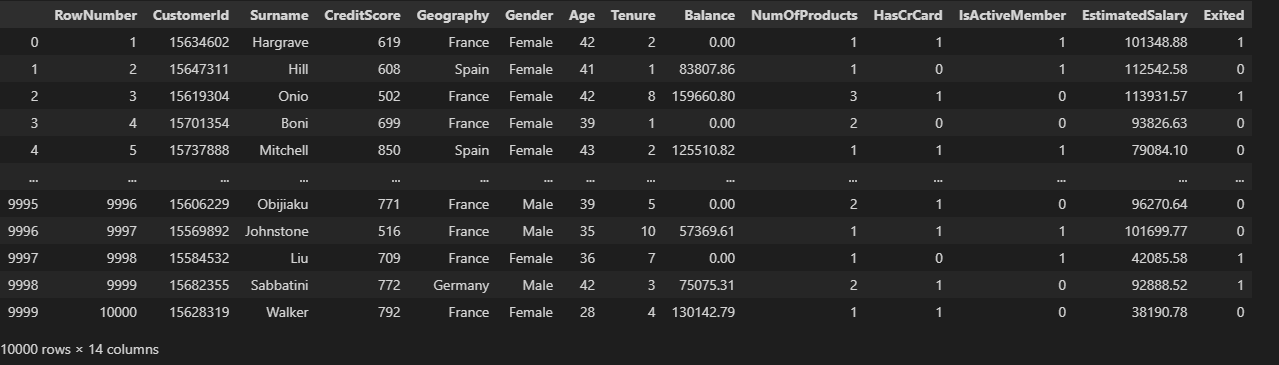


**Question-6:**

1. Find the outliers and replace the outliers

**Solution-1:**

df.interpolate(method ='linear', limit\_direction ='backward', limit = 1)

**Output:**

**Solution-2:**

def box\_scatter(data, x, y):

fig, (ax1, ax2) = plt.subplots(nrows=2, ncols=1, figsize=(16,6))

sns.boxplot(data=data, x=x, ax=ax1)

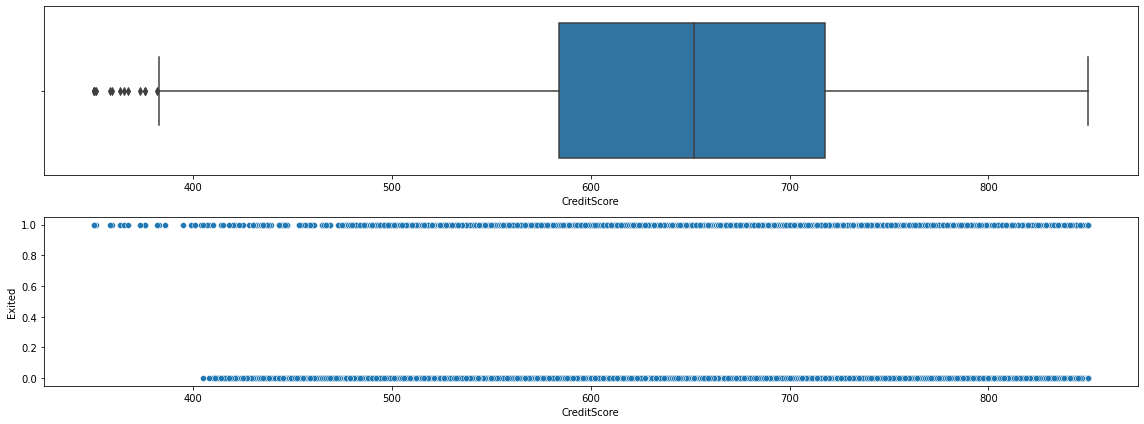
sns.scatterplot(data=data, x=x,y=y,ax=ax2)

box\_scatter(df,'CreditScore','Exited');

plt.tight\_layout()

print(f"# of Bivariate Outliers: {len(df.loc[df['CreditScore'] < 400])}")

**Output:**

****

**Solution-3:**

for i in df:

if df[i].dtype=='int64' or df[i].dtypes=='float64':

q1=df[i].quantile(0.25)

q3=df[i].quantile(0.75)

iqr=q3-q1

upper=q3+1.5\*iqr

lower=q1-1.5\*iqr

df[i]=np.where(df[i] >upper, upper, df[i])

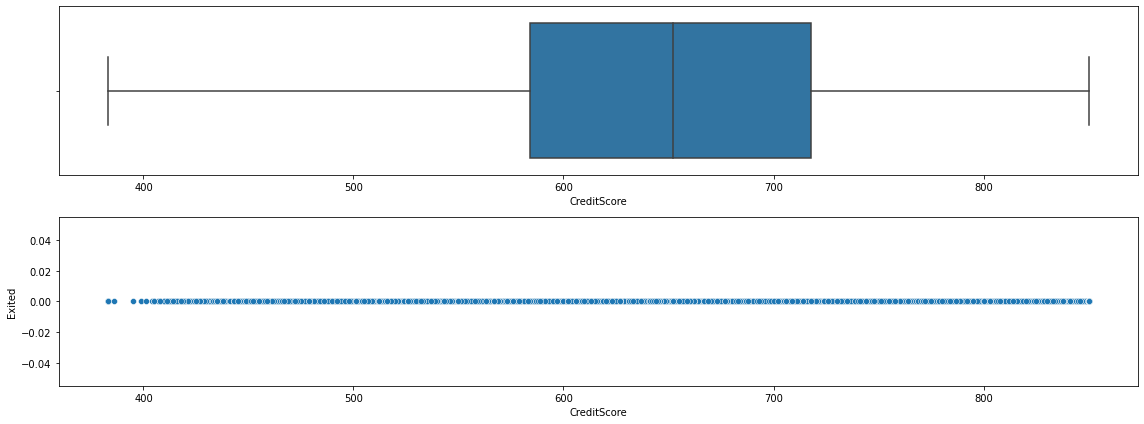
df[i]=np.where(df[i] <lower, lower, df[i])

box\_scatter(df,'CreditScore','Exited');

plt.tight\_layout()

print(f"# of Bivariate Outliers: {len(df.loc[df['CreditScore'] < 400])}")

**Output:**



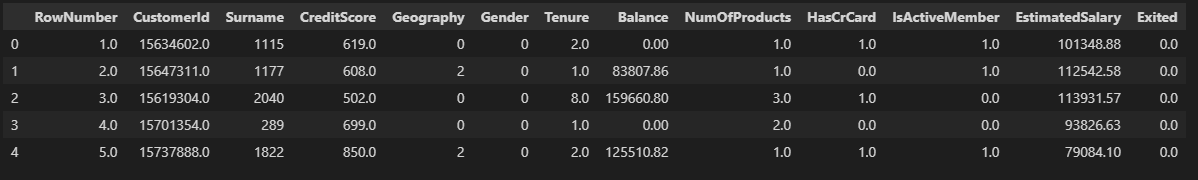
**Question-7:**

1. Check for Categorical columns and perform encoding.

**Solution:**

df.head()

**Output:**



**Question-8:**

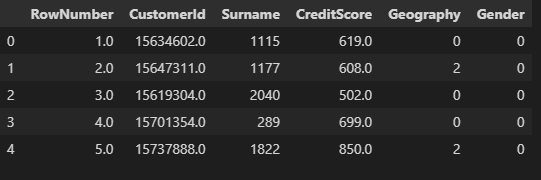
8. Split the data into dependent and independent variables.

**Solution-1:**

x=df.iloc[:,:6]

x.head()

**Output:**

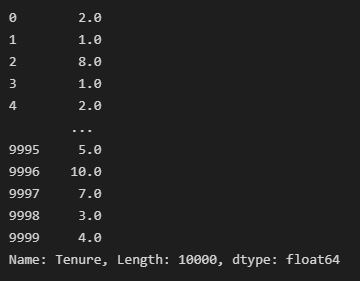


**Solution-2:**

Y = df.iloc[:, 6]

print(Y)

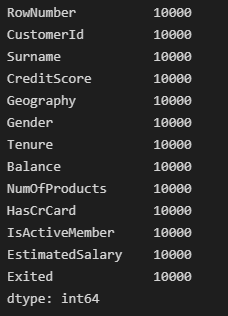
**Output:**



**Solution-3:**

df.count(0)

**Output:**



]

**Question-9:**

1. Scale the independent variables

**Solution:**

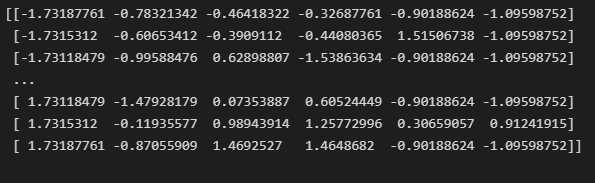
from sklearn.preprocessing import StandardScaler

scaler=StandardScaler()

x=scaler.fit\_transform(x)

print(x)

**Output:**



**Question-10:**

10. Split the data into training and testing

**Solution-1:**

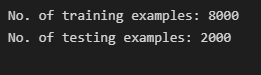
training\_data = df.sample(frac=0.8, random\_state=25)

testing\_data = df.drop(training\_data.index)

print(f"No. of training examples: {training\_data.shape[0]}")

print(f"No. of testing examples: {testing\_data.shape[0]}")

**Output:**



**Solution-2:**

from sklearn.model\_selection import train\_test\_split

training\_data, testing\_data = train\_test\_split(df, test\_size=0.2, random\_state=25)

print(f"No. of training examples: {training\_data.shape[0]}")

print(f"No. of testing examples: {testing\_data.shape[0]}")

**Output:**

