**Assignment -2**

Python Programming

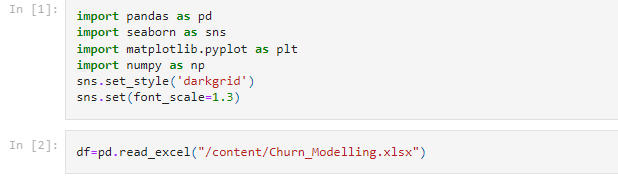
|  |  |
| --- | --- |
| Assignment Date | 23 september 2022 |
| Student Name | Shiny .R |
| Student Roll Number | 311419205036 |
| Maximum Marks | 2 Marks |

# Data Visualization and Pre-processing

**Question-1:**

1. Load the dataset

|  |
| --- |
| **Solution:** |
| **import** pandas **as** pd  **import** seaborn **as** sns  **import** matplotlib.pyplot **as** plt  **import** numpy **as** np  sns**.**set\_style('darkgrid')  sns**.**set(font\_scale**=**1.3)  In [2]:  df**=**pd**.**read\_excel("/content/Churn\_Modelling.xlsx") |
|  |
|  |
|  |
|  |
|  |
|  |
|  |  |



**Question-2:**

2. Perform Below Visualizations.

● Univariate Analysis

● Bi - Variate Analysis

● Multi - Variate Analysis

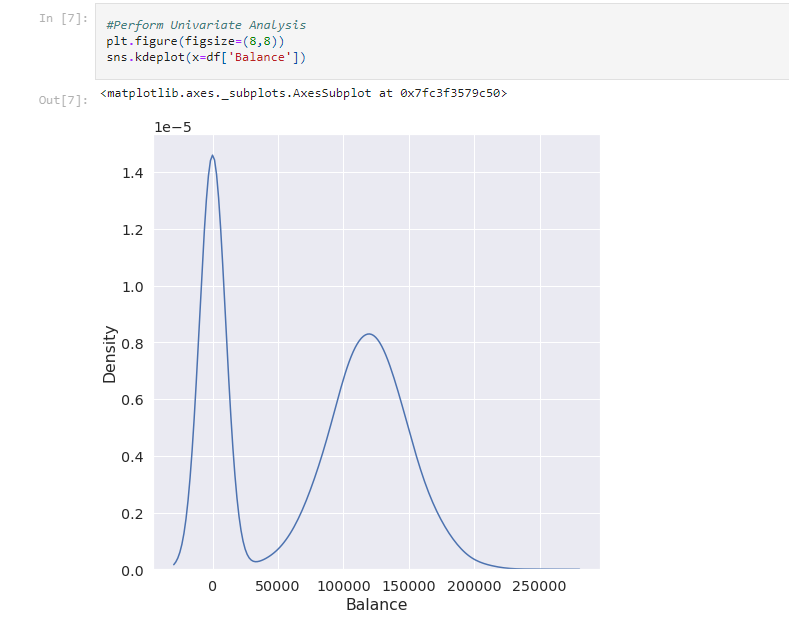
|  |
| --- |
| **Solution:** |

.

*#Perform Univariate Analysis*

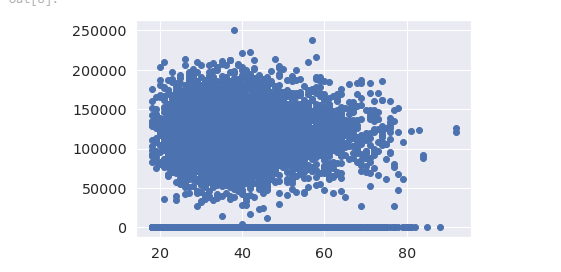
plt**.**figure(figsize**=**(8,8))

sns**.**kdeplot(x**=**df['Balance'])



*#Perform Bivariate Analysis*

plt**.**scatter(df**.**Age,df**.**Balance)



*#Perform Bivariate Analysis*

df**.**corr()

**

*#Perform Bivariate Analysis*

**import** statsmodels.api **as** sm

*#define response variable*

y **=** df['CreditScore']

*#define explanatory variable*

x **=** df[['EstimatedSalary']]

*#add constant to predictor variables*

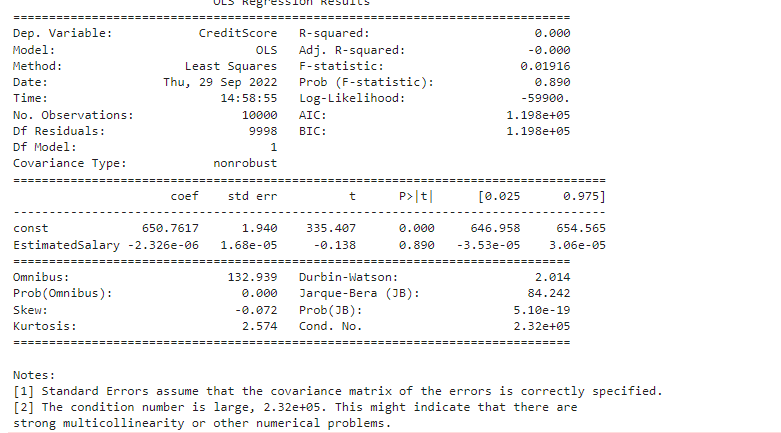
x **=** sm**.**add\_constant(x)

*#fit linear regression model*

model **=** sm**.**OLS(y, x)**.**fit()

*#view model summary*

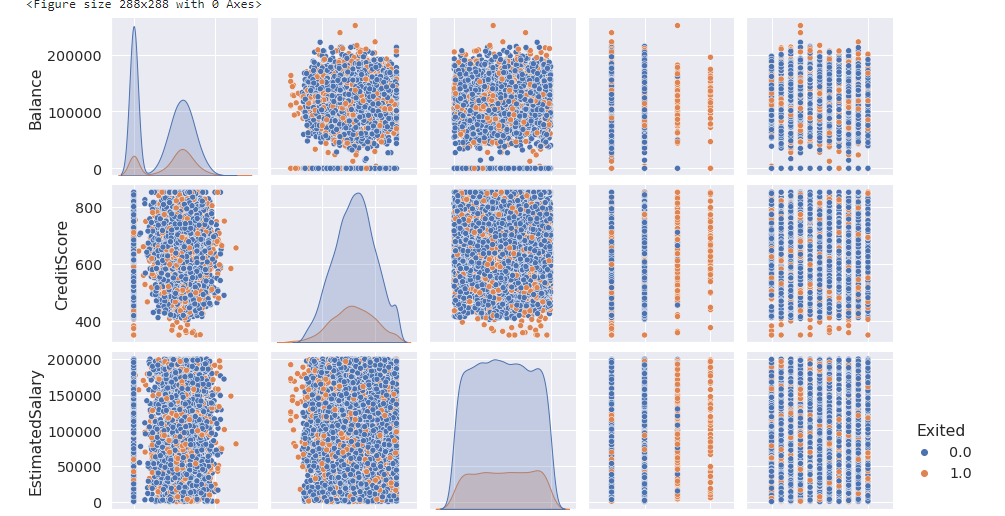
print(model**.**summary())

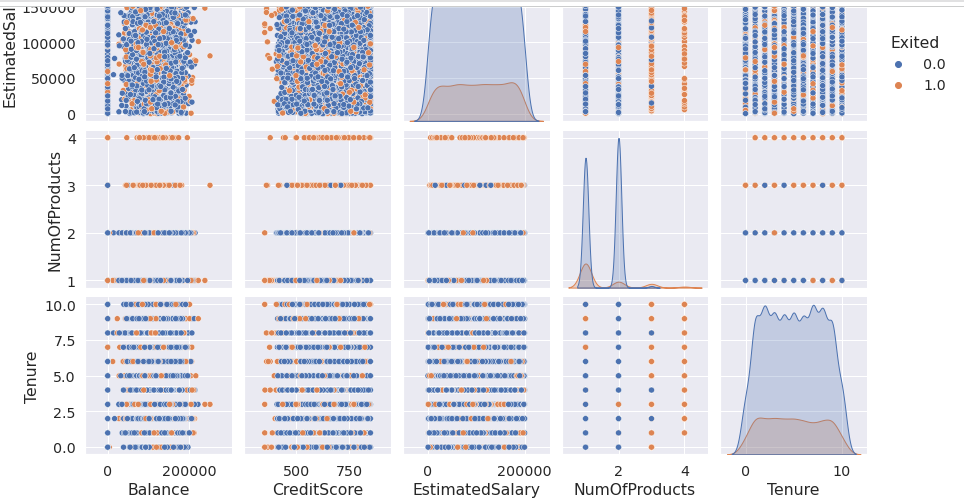
****

*#Perform Multivariate Analysis*

plt**.**figure(figsize**=**(4,4))

sns**.**pairplot(data**=**df[["Balance","CreditScore","EstimatedSalary","NumOfProducts","Tenure","Exited"]],hue**=**"Exited")

****

****

**Question-3:**

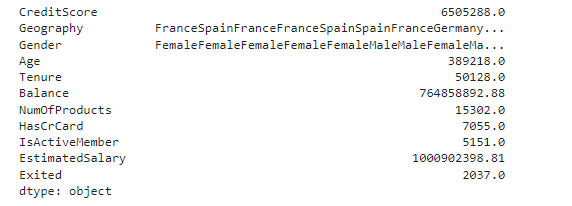
3. Perform descriptive statistics on the dataset.

**Solution:**

*#Perform Descriptive Statistics*

df**=**pd**.**DataFrame(df)

print(df**.**sum())

****

*#Perform Descriptive Statistics*

print("----Sum Value-----")

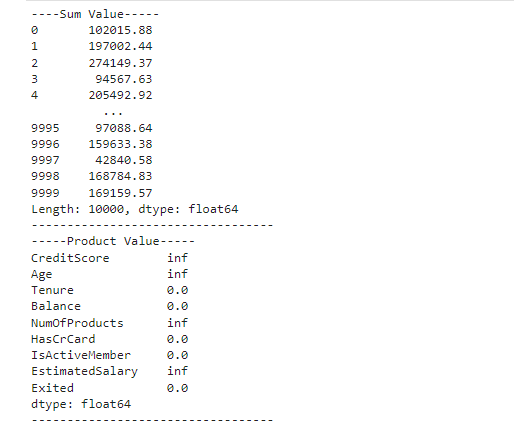
print(df**.**sum(1))

print("----------------------------------")

print("-----Product Value-----")

print(df**.**prod())

print("----------------------------------")



*#Perform Descriptive Statistics*

print("----------Mean Value-----------")

print(df**.**mean())

print("-------------------------------")

print("----------Median Value---------")

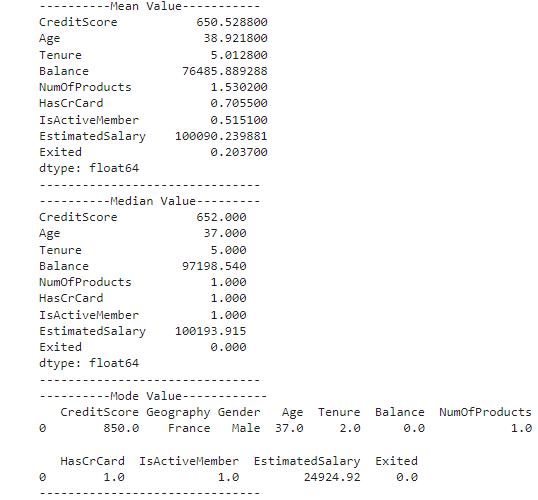
print(df**.**median())

print("-------------------------------")

print("----------Mode Value------------")

print(df**.**mode())

print("-------------------------------")

**

**Question-4:**

4.Handle the Missing values

**Solution:**

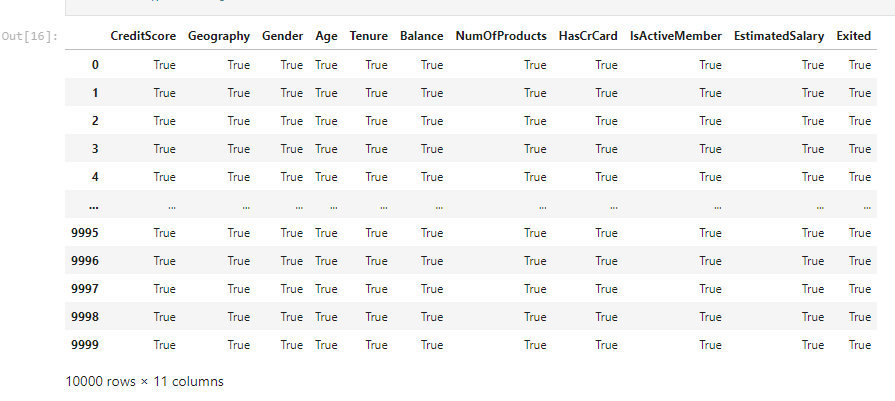
*#Handling with missing Values*

df**.**isnull()**.**values;

*#Checking values are null*

*#Handling with missing Values*

df**.**notnull()*#Checking values are not null*



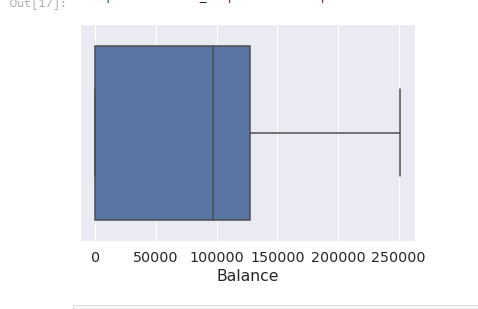
**Question-5:**

5. Find the outliers and replace the outliers

**Solution:**

*#Find outliers & replace the outliers*

sns**.**boxplot(df['Balance'])

****

*#Find outliers & replace the outliers*

print(np**.**where(df['Balance']**>**100000))

(array([ 2, 4, 5, ..., 9987, 9993, 9999]),)

In [19]:

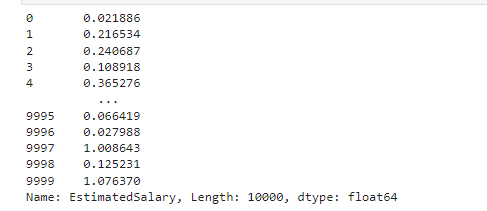
*#Find outliers & replace the outliers*

**from** scipy **import** stats

**import** numpy **as** np

z **=** np**.**abs(stats**.**zscore(df["EstimatedSalary"]))

print(z)

****

**Question-6:**

6. Check for Categorical columns and perform encoding

**Solution:**

*#Check for categorical columns & performs encoding*

**from** sklearn.preprocessing **import** LabelEncoder

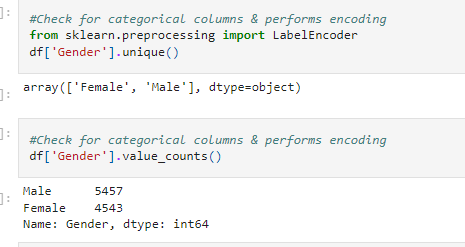
df['Gender']**.**unique()

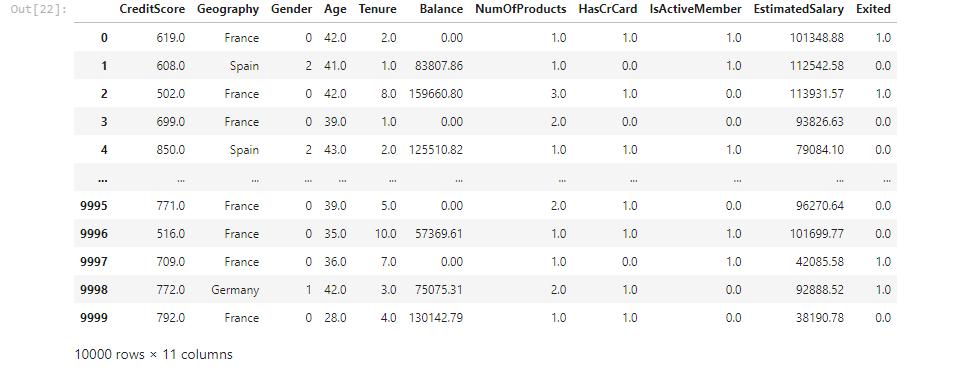
df['Gender']**.**value\_counts()

encoding**=**LabelEncoder()

df["Gender"]**=**encoding**.**fit\_transform(df**.**iloc[:,1]**.**values)

df

****

****

**Question-7:**

**7.**Split the data into dependent and independent variables.

**Solution:**

*#Split the data into Dependent & Independent Variables*

print("----------Dependent Variables----------")

X**=**df**.**iloc[:,1:4]

print(X)

print("---------------------------------------")

print("---------Independent Variables---------")

Y**=**df**.**iloc[:,4]

print(Y)

print("---------------------------------------")

**Question-8:**

8. Scale the independent variables

**Solution:**

*#Split the data into Dependent & Independent Variables*

print("----------Dependent Variables----------")

X**=**df**.**iloc[:,1:4]

print(X)

print("---------------------------------------")

print("---------Independent Variables---------")

Y**=**df**.**iloc[:,4]

print(Y)

print("---------------------------------------")

**Question-9:**

9. Split the data into training and testing

**Solution:**

*#Split the data into training & testing*

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

In [34]:

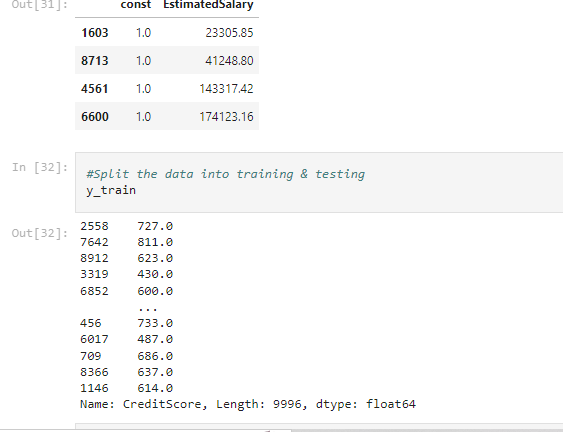
*#Split the data into training & testing*

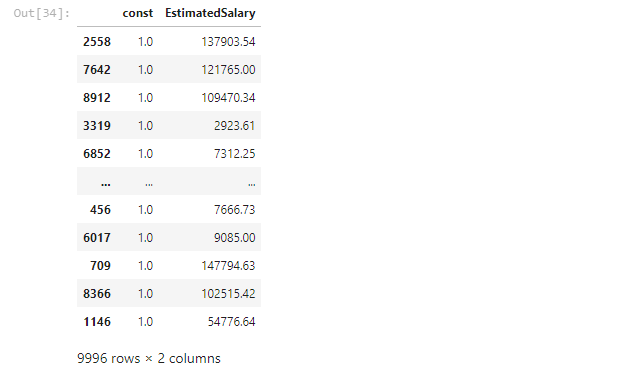
x\_train, x\_test, y\_train, y\_test **=** train\_test\_split(x, y, test\_size**=**4,random\_state**=**4)

x\_train

x\_test

y\_train

y\_test

****