Data Science Minor Project at InternsElite

Submitted by Rajvardhan Mall(October Batch)

Domain :-- Banking

PROBLEM STATEMENT :-- Happy Bank provides various credit cards to customers.The manager of Happy Bank is disturbed by more and more customers leaving their credit card services.The team did a customer survey to check customer attrition.Various customer attributes like Customer_Age, Credit_Limit, Dependent_Count. The team would really appreciate it if one could predict for them who is gonna get churned so they can proactively go to the customer to provide them better services and turn customers' decisions in the opposite direction.

```
import numpy as np

import pandas as pd

from sklearn.preprocessing import StandardScaler

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

sns.set(color_codes = True)

import timeit

from warnings import filterwarnings

filterwarnings('ignore')

df= pd.read_csv('BankChurners.csv')
```

Q1) Display a sample of five rows of the data frame.

```
df.head(5)
```

Solution :-

	CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Category	Card_Category	Months_on_book	...
0	768805383	Existing Customer	45	M	3	High School	Married	$60K - 80K	Blue	39	...
1	818770008	Existing Customer	49	F	5	Graduate	Single	Less than $40K	Blue	44	...
2	713982108	Existing Customer	51	M	3	Graduate	Married	$80K - 120K	Blue	36	...
3	769911858	Existing Customer	40	F	4	High School	Unknown	Less than $40K	Blue	34	...
4	709106358	Existing Customer	40	M	3	Uneducated	Married	$60K - 80K	Blue	21	...

5 rows × 21 columns

Q2) Check the shape of the data(number of rows and columns).

```
df.shape
```

Solution :-

(10127, 21)

Q3) Check the percentage of missing values in each column of the data frame.

```
df.isnull().sum()/(len(df))*100
```

Solution :-

```
CLIENTNUM                 0.0
Attrition_Flag            0.0
Customer_Age              0.0
Gender                    0.0
Dependent_count           0.0
Education_Level           0.0
Marital_Status            0.0
Income_Category           0.0
Card_Category             0.0
Months_on_book            0.0
Total_Relationship_Count  0.0
Months_Inactive_12_mon    0.0
Contacts_Count_12_mon     0.0
Credit_Limit              0.0
Total_Revolving_Bal       0.0
Avg_Open_To_Buy           0.0
Total_Amt_Chng_Q4_Q1      0.0
Total_Trans_Amt           0.0
Total_Trans_Ct            0.0
Total_Ct_Chng_Q4_Q1       0.0
Avg_Utilization_Ratio     0.0
dtype: float64
```

Since, we can see that the percentage of missing values in each column of the data frame is "ZERO".So, this will give us a accurate analysis of the data frame(df).

Q4) Check if there are any duplicate rows.

```
df.duplicated()
```

Solution :-

```
0        False
1        False
2        False
3        False
4        False
        ...
10122    False
10123    False
```

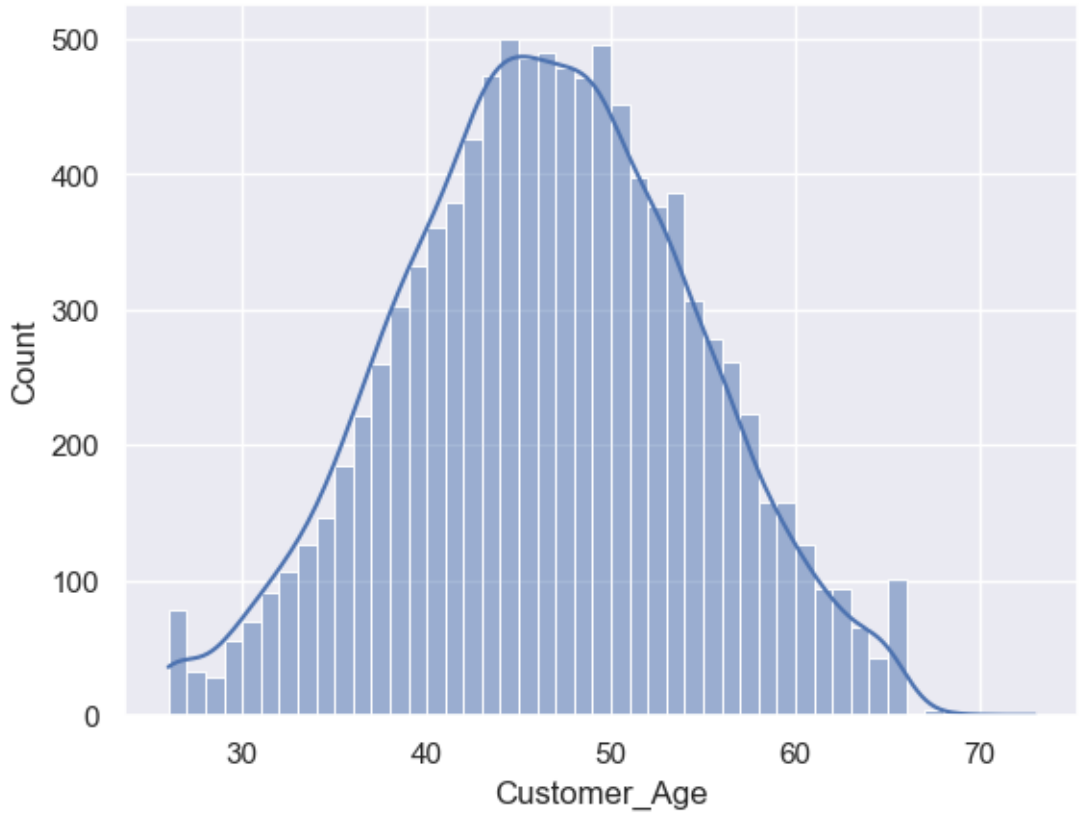
```
10124    False
10125    False
10126    False
Length: 10127, dtype: bool
```

Using the '.duplicated()' method in pandas to identify duplicate rows in a Data Frame. We can specify the subset of columns to consider when identifying duplicates. If we found any duplicate row, then we may remove that duplicate row for accurate analysis. So, we can use 'drop_duplicates()' method in pandas to remove duplicate rows from a Data Frame, ensuring that only unique rows are retained.

Q5) Check the distribution of the Customer_Age column.Check the basis statistics like mean,median, and standard deviation of the age column.

```
sns.histplot(df.Customer_Age, kde=True)
plt.show()
```

Solution :-



Conclusion:-- We can see that,most of credit card holders are of age gap 42 to 53.

df.describe()

Solution :-

	CLIENTNUM	Customer_Age	Dependent_count	Months_on_book	Total_Relationship_Count	Months_Inactive_12_mon	Contacts_Count_12_mon	Credit_Limi
count	1.012700e+04	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.000000	10127.00000
mean	7.391776e+08	46.325960	2.346203	35.928409	3.812580	2.341167	2.455317	8631.95369
std	3.690378e+07	8.016814	1.298908	7.986416	1.554408	1.010622	1.106225	9088.77665
min	7.080821e+08	26.000000	0.000000	13.000000	1.000000	0.000000	0.000000	1438.30000
25%	7.130368e+08	41.000000	1.000000	31.000000	3.000000	2.000000	2.000000	2555.00000
50%	7.179264e+08	46.000000	2.000000	36.000000	4.000000	2.000000	2.000000	4549.00000
75%	7.731435e+08	52.000000	3.000000	40.000000	5.000000	3.000000	3.000000	11067.50000
max	8.283431e+08	73.000000	5.000000	56.000000	6.000000	6.000000	6.000000	34516.00000

Hence, we can see the mean,median and standard deviation of age column are 46.325960,46.000000 and 10127.000000 respectively

Q6) Plot 2 box plots and 2 pie chart of the parameter of your on choice and write your intuition about it.

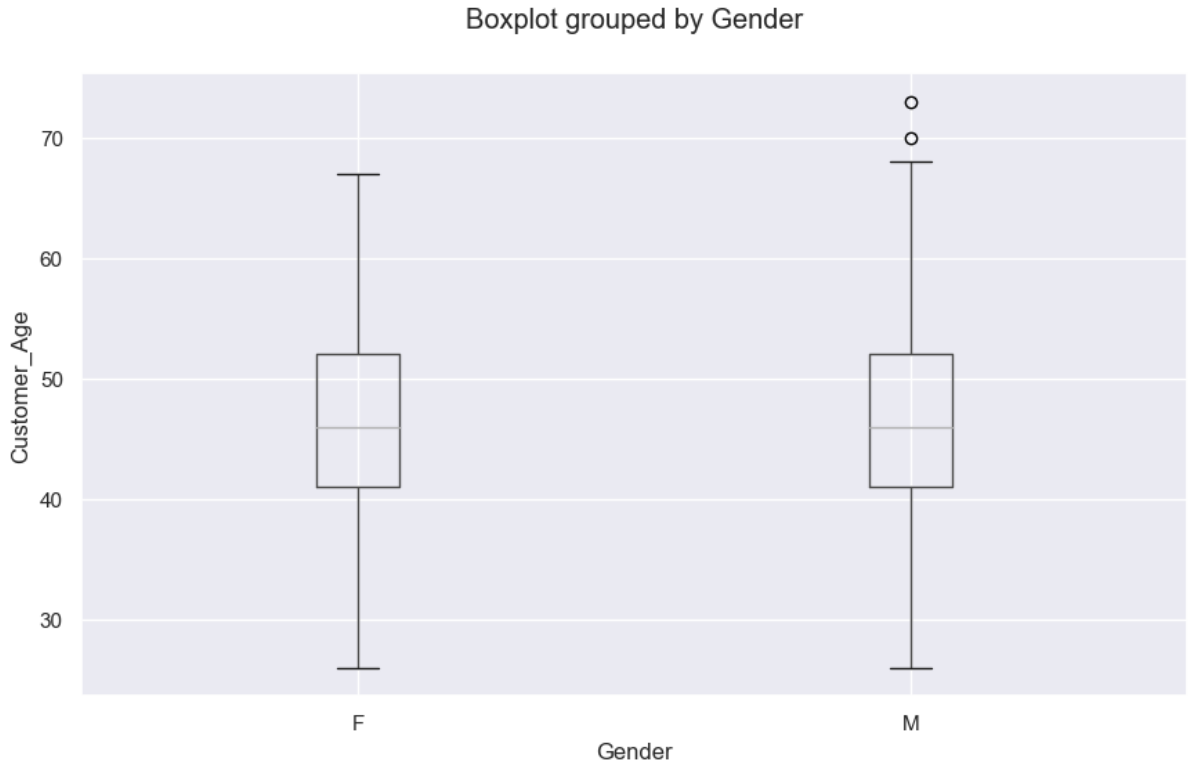
#box plot for Gender

ax = df.boxplot(column="Customer_Age", by="Gender", figsize=(10,6))

ax.set_ylabel("Customer_Age")

ax.set_title("")

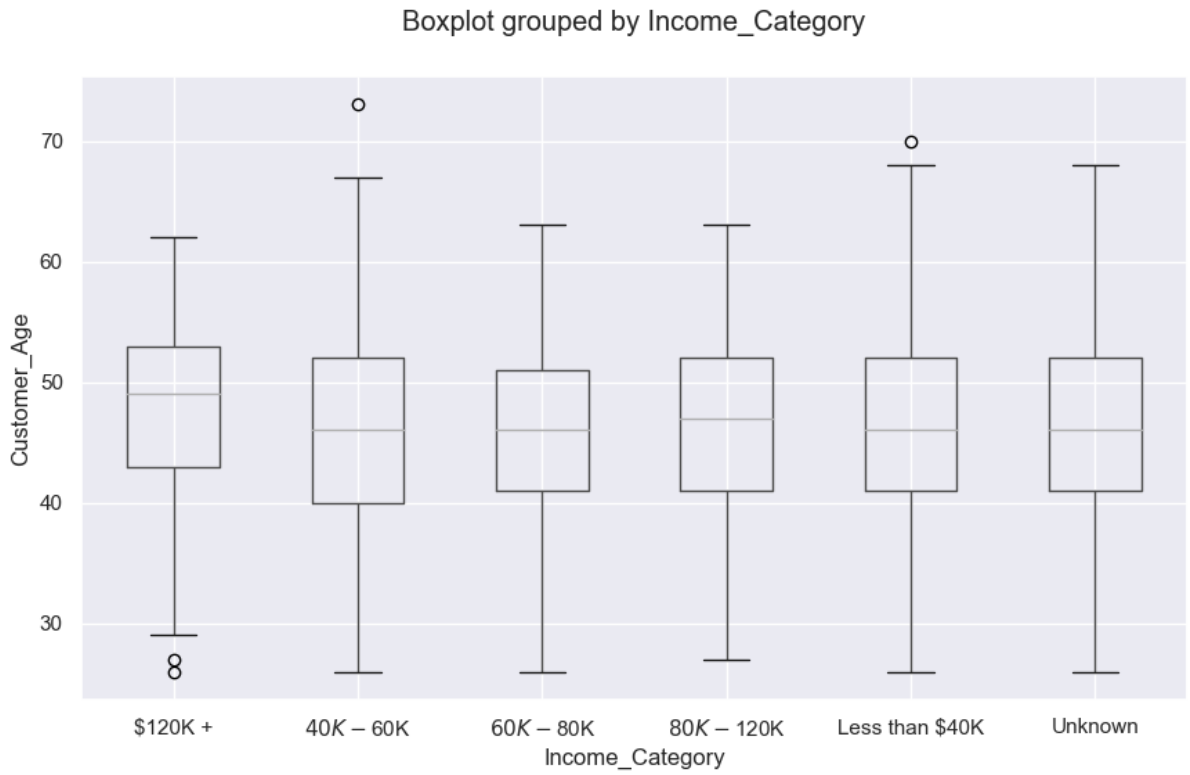
Solution :-



Conclusion:-- a)In above boxplot we see that,50% of male and female who have a credit card are of age from 42 to 53. b)46 is the median age of customers who has credit card. c)25% of customers are below age 41. d)75% of customers are below age 53.

```
#Box plot for Income_Categoty

ax = df.boxplot(column="Customer_Age", by="Income_Category", figsize=(10,6))

ax.set_ylabel("Customer_Age")

ax.set_title("")
```

Solution :-



Conclusion:-- a)Customer having Annual Income>$120k are very less in count. b)In all the Income Categories, 46 is the median age of customers. c)Almost in all the Income Categories 50% of the customers are of age 42 to 53.

df.describe(include='all')

Solution :-

	CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Category	Card_Category	Months_on_boo
count	1.012700e+04	10127	10127.000000	10127	10127.000000	10127	10127	10127	10127	10127.0000(
unique	NaN	2	NaN	2	NaN	7	4	6	4	Na
top	NaN	Existing Customer	NaN	F	NaN	Graduate	Married	Less than $40K	Blue	Na
freq	NaN	8500	NaN	5358	NaN	3128	4687	3561	9436	Na
mean	7.391776e+08	NaN	46.325960	NaN	2.346203	NaN	NaN	NaN	NaN	35.9284(
std	3.690378e+07	NaN	8.016814	NaN	1.298908	NaN	NaN	NaN	NaN	7.9864
min	7.080821e+08	NaN	26.000000	NaN	0.000000	NaN	NaN	NaN	NaN	13.0000(
25%	7.130368e+08	NaN	41.000000	NaN	1.000000	NaN	NaN	NaN	NaN	31.0000(
50%	7.179264e+08	NaN	46.000000	NaN	2.000000	NaN	NaN	NaN	NaN	36.0000(
75%	7.731435e+08	NaN	52.000000	NaN	3.000000	NaN	NaN	NaN	NaN	40.0000(
max	8.283431e+08	NaN	73.000000	NaN	5.000000	NaN	NaN	NaN	NaN	56.0000(

11 rows × 21 columns

#pie chart for Gender

labels = ['Male','Female']

sizes = [4769,5358] #from above chart, we have, total number of males and females

plt.figure(figsize=(6,6))

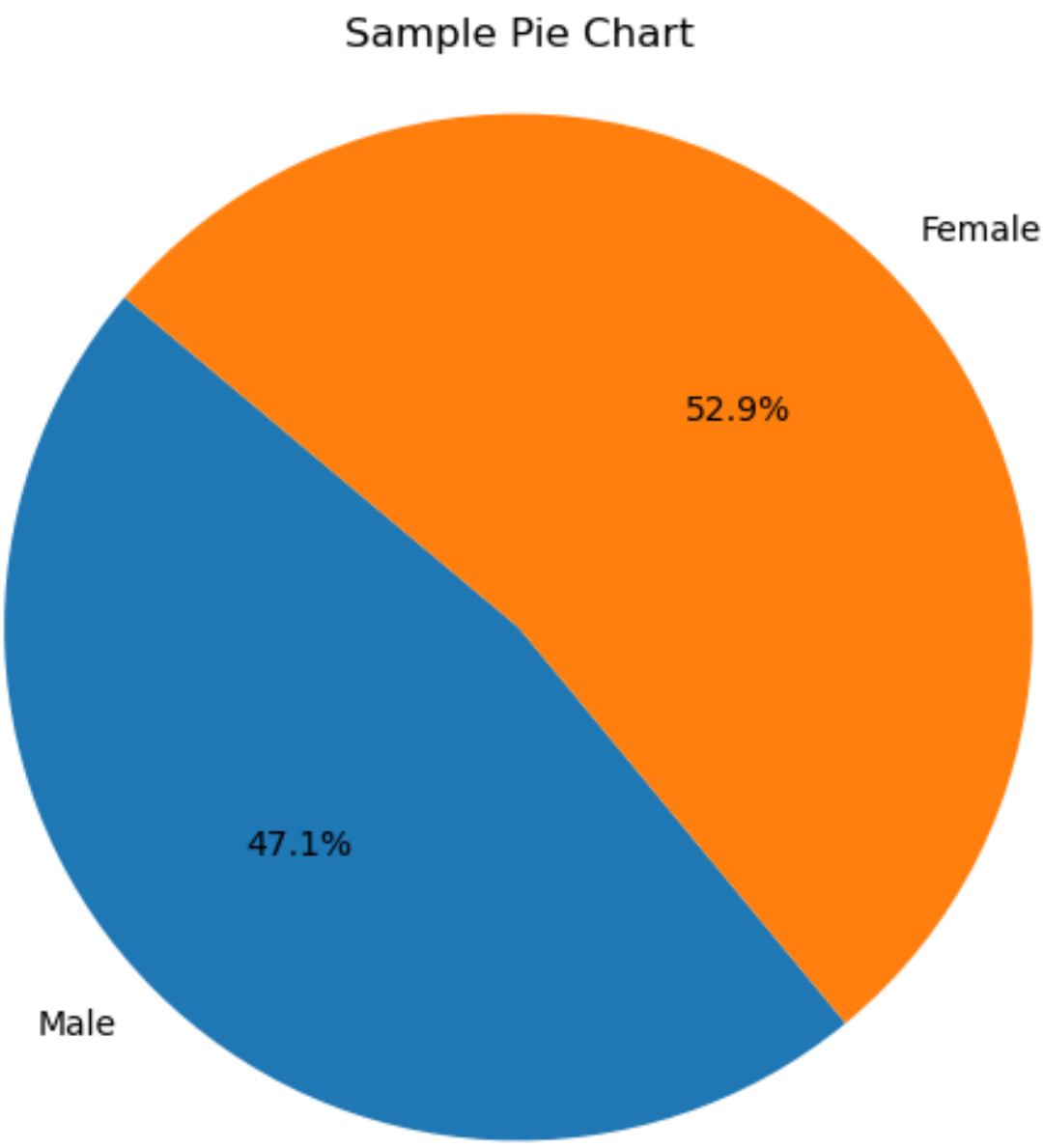
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)

plt.axis('equal')

plt.title('Sample Pie Chart')

plt.show()

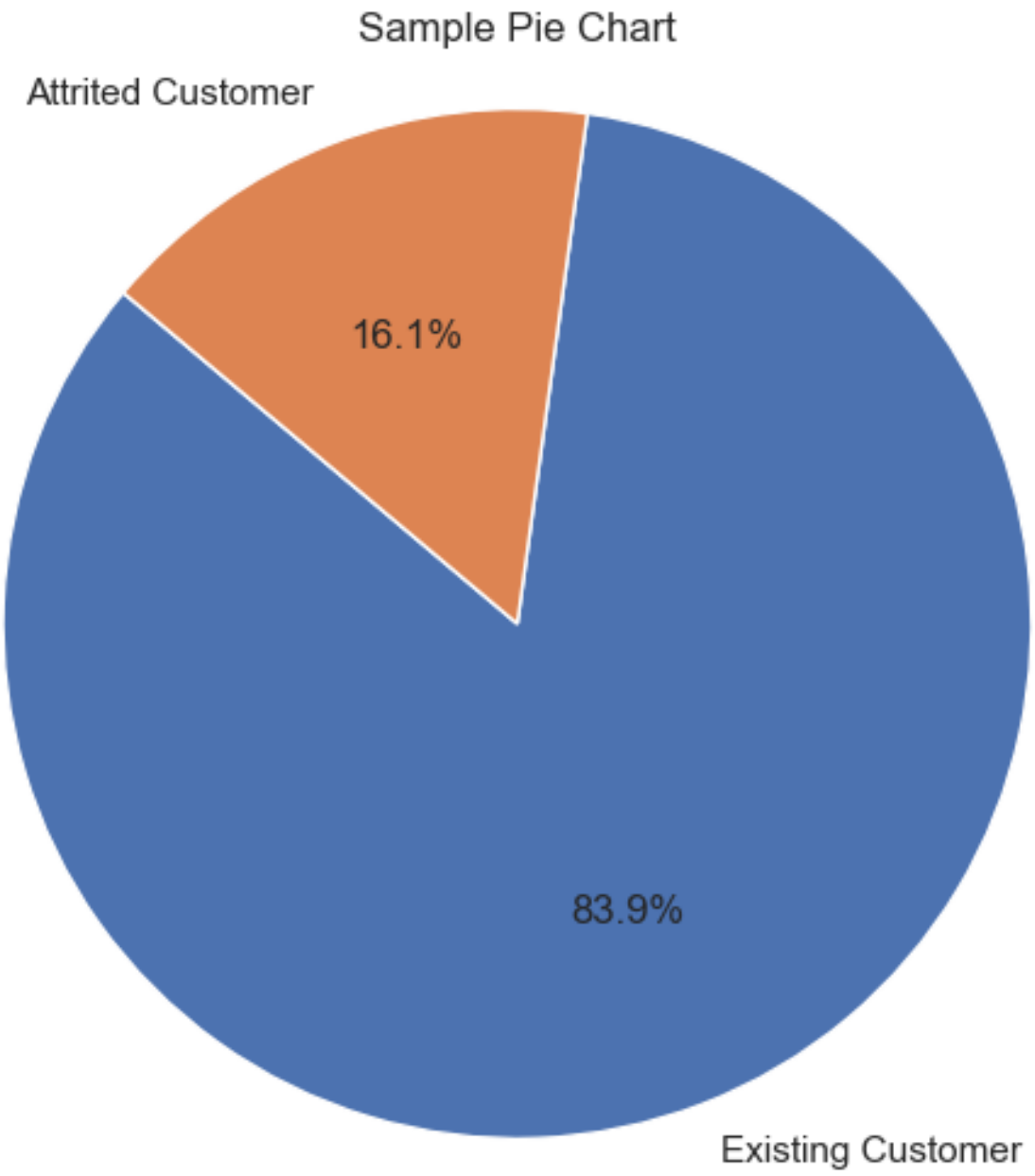
Solution :-



Conclusion:-- From above pie chart we see that we have more female customers than male.

```
labels = ['Existing Customer','Attrited Customer']
sizes = [8500,1627] #from above chart, we have total number of Existing Customers and Attrited Customers
plt.figure(figsize=(6,6))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=140)
plt.axis('equal')
plt.title('Sample Pie Chart')
plt.show()
```

Solution :-



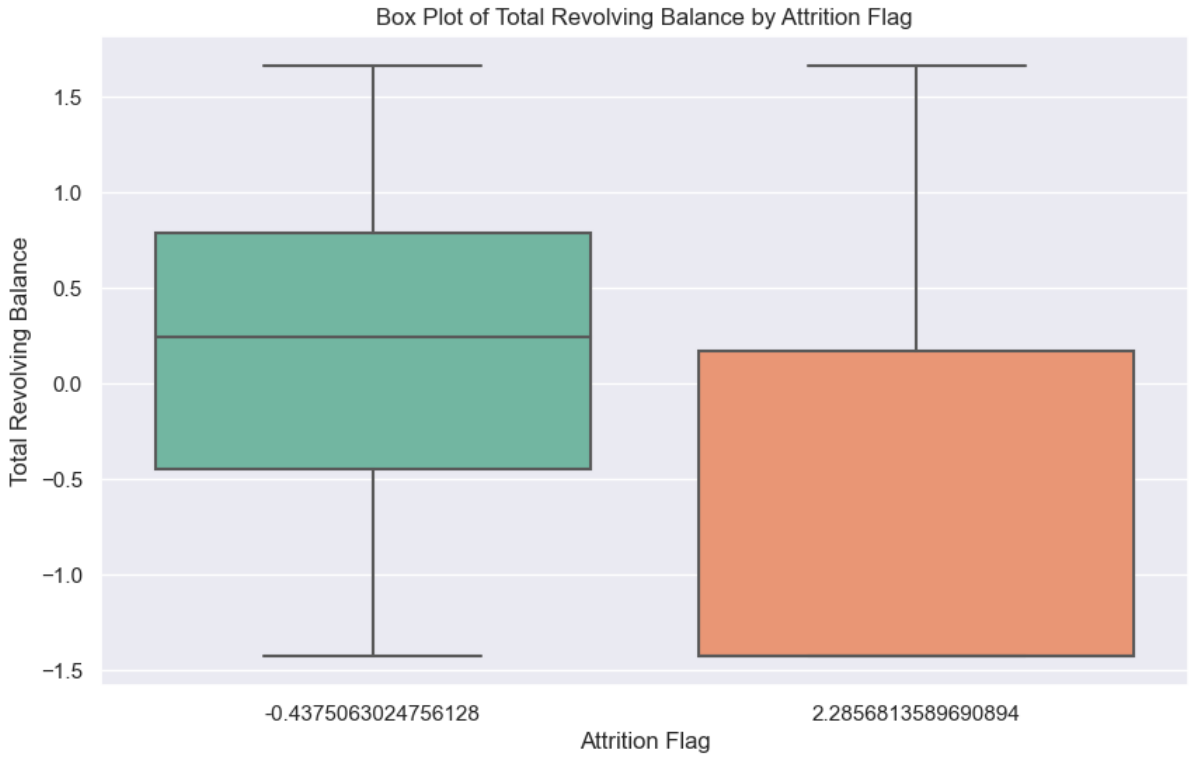
Conclusion:-- We have more Existing Customers than Attrited Cuatomers.

Q7) Plot a Box-plot of Total_Revolving_Bal and Card_Category by characterizing with Attrition_Flag.Write your intuitions about it.

```
data = df[['Attrition_Flag', 'Total_Revolving_Bal']]

plt.figure(figsize=(10, 6))

sns.boxplot(x='Attrition_Flag', y='Total_Revolving_Bal', data=data, palette='Set2')

plt.xlabel('Attrition Flag')

plt.ylabel('Total Revolving Balance')
```

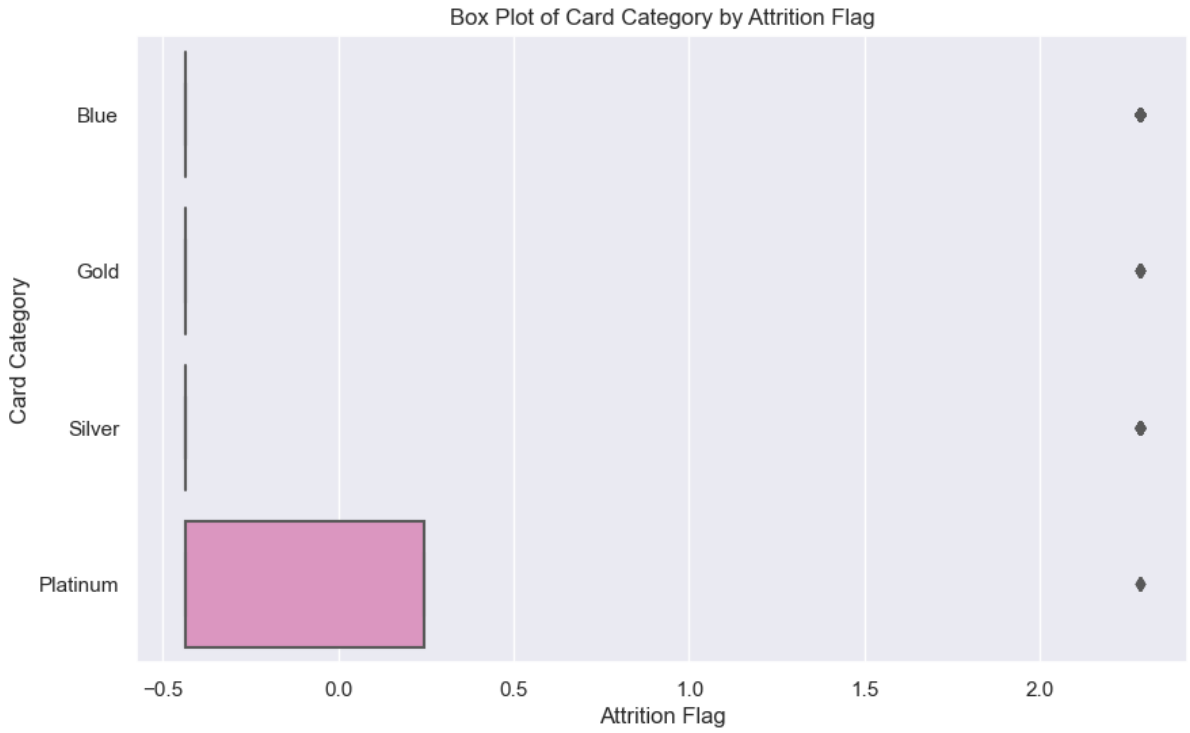
```
plt.title('Box Plot of Total Revolving Balance by Attrition Flag')
plt.show()
```

Solution :-




```
data = df[['Attrition_Flag', 'Card_Category']]
plt.figure(figsize=(10, 6))
sns.boxplot(x='Attrition_Flag', y='Card_Category', data=data, palette='Set2')
plt.xlabel('Attrition Flag')
plt.ylabel('Card Category')
plt.title('Box Plot of Card Category by Attrition Flag')
plt.show()
```

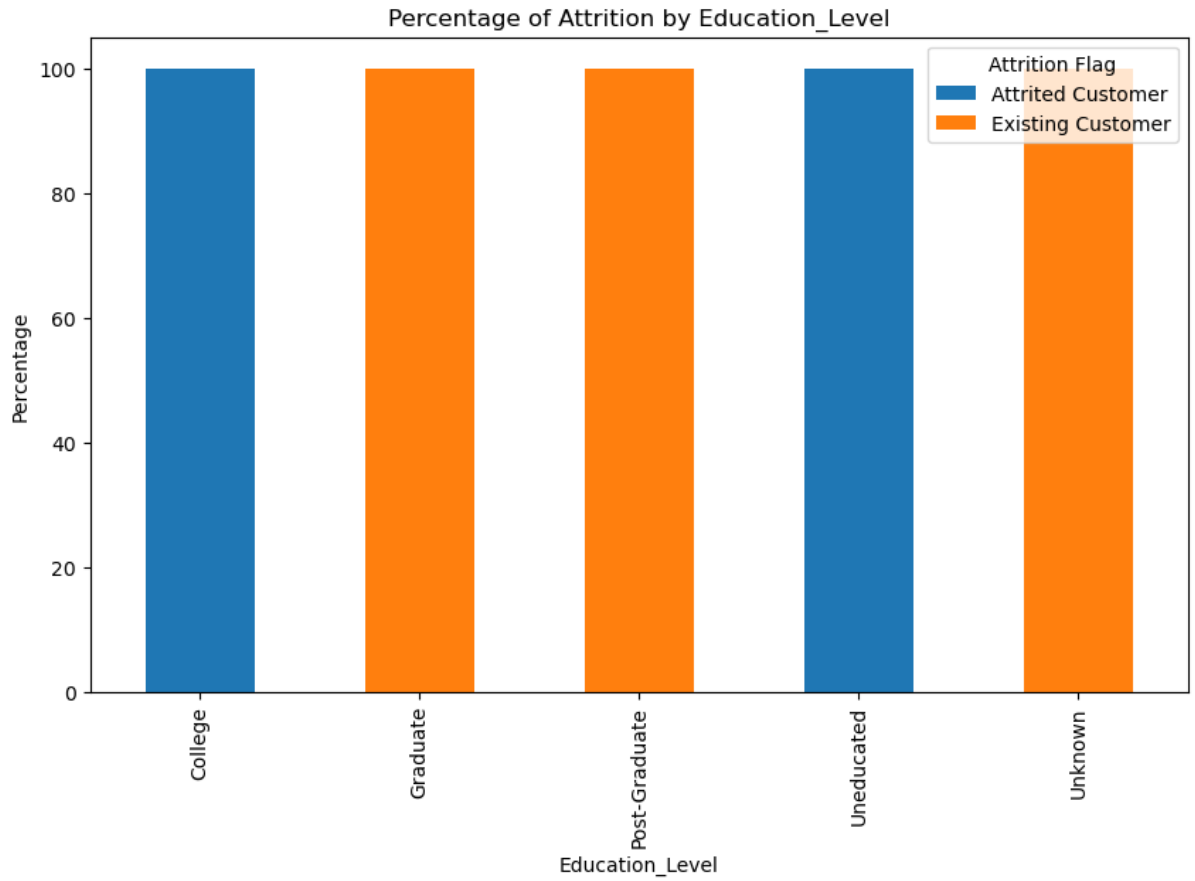
Solution :-



Q8) Plot a percentage segment bar graph between Education_Level and Attrition_Flag of the customers.

```
data = {
   "Education_Level": ["Graduate", "Uneducated", "Unknown", "College", "Post-Graduate"],
   "Attrition_Flag": ["Existing Customer", "Attrited Customer", "Existing Customer", "Attrited Customer", "Existing Customer"],
}
df = pd.DataFrame(data)
percentage_data = df.groupby("Education_Level")["Attrition_Flag"].value_counts(normalize=True).unstack().fillna(0) * 100
ax = percentage_data.plot(kind="bar", stacked=True, figsize=(10, 6))
plt.title("Percentage of Attrition by Education_Level")
plt.xlabel("Education_Level")
plt.ylabel("Percentage")
plt.legend(title="Attrition Flag", loc="upper right")
plt.show()
```

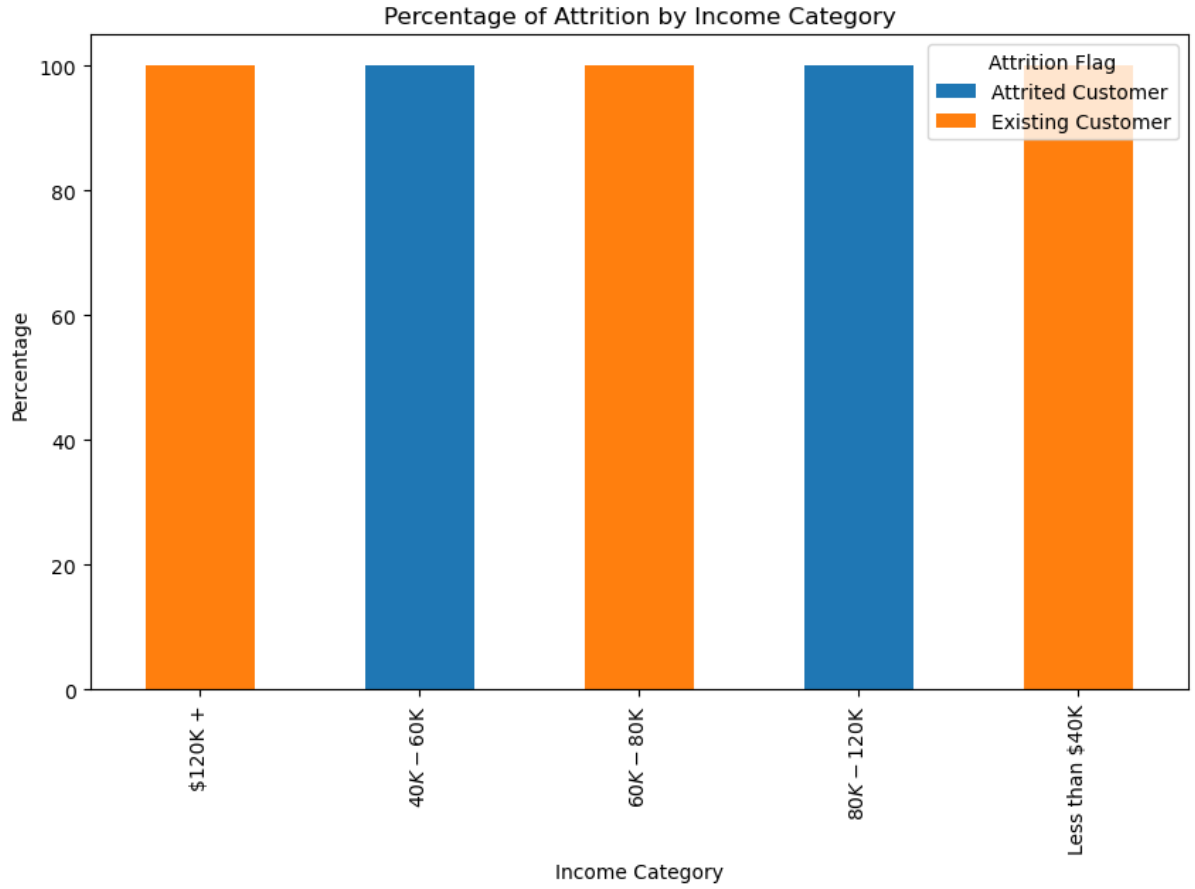
Solution :-



Q9) Plot a percentage segment bar graph between Income_Category and Attrition_Flag of the customers.

```
data = {
    "Income_Category": ["Less than $40K", "$40K- $60K", "$60K- $80K", "$80K- $120K", "$120K +"],
    "Attrition_Flag": ["Existing Customer", "Attrited Customer", "Existing Customer", "Attrited Customer", "Existing Customer"],
}
df = pd.DataFrame(data)
percentage_data = df.groupby("Income_Category")["Attrition_Flag"].value_counts(normalize=True).unstack().fillna(0) * 100
ax = percentage_data.plot(kind="bar", stacked=True, figsize=(10, 6))
plt.title("Percentage of Attrition by Income Category")
plt.xlabel("Income Category")
plt.ylabel("Percentage")
plt.legend(title="Attrition Flag", loc="upper right")
plt.show()
```

Solution :-



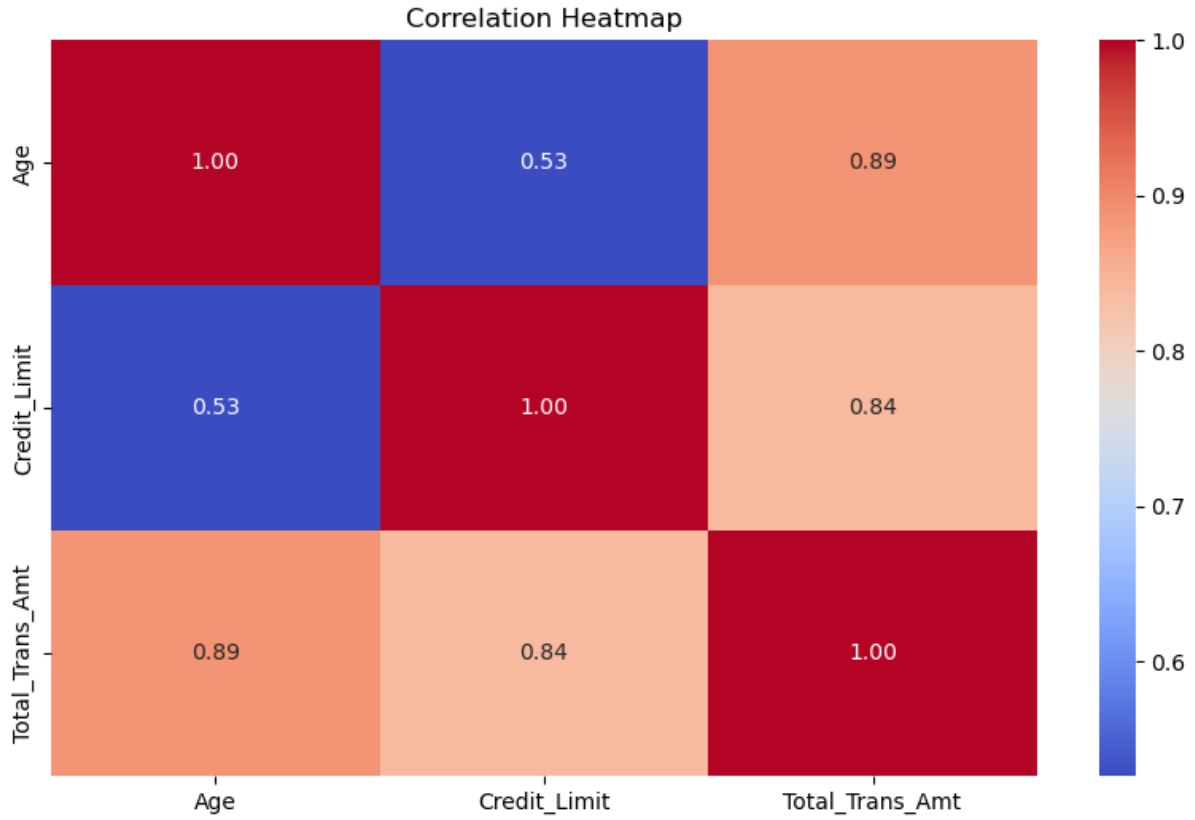
Q10) Drop CLIENTNUM column.Make a sub data frame which consists of all the numerical columns(i.e.int64,float64) along with the Attrition_Flag column.Plot a heatmap to view the correlation using seaborn.

```
data = {
    "CLIENTNUM": [1, 2, 3, 4, 5],
    "Attrition_Flag": ["Existing Customer", "Attrited Customer", "Existing Customer", "Existing Customer", "Attrited Customer"],
    "Age": [35, 42, 28, 55, 33],
    "Credit_Limit": [5000.0, 8000.0, 12000.0, 15000.0, 7000.0],
    "Total_Trans_Amt": [2300, 4500, 3500, 7000, 2800],
}
df = pd.DataFrame(data)
df.drop("CLIENTNUM", axis=1, inplace=True)
numerical_df = df.select_dtypes(include=["int64", "float64"])
plt.figure(figsize=(10, 6))
sns.heatmap(numerical_df.corr(), annot=True, cmap="coolwarm", fmt=".2f")
```

plt.title("Correlation Heatmap")

plt.show()

Solution :-

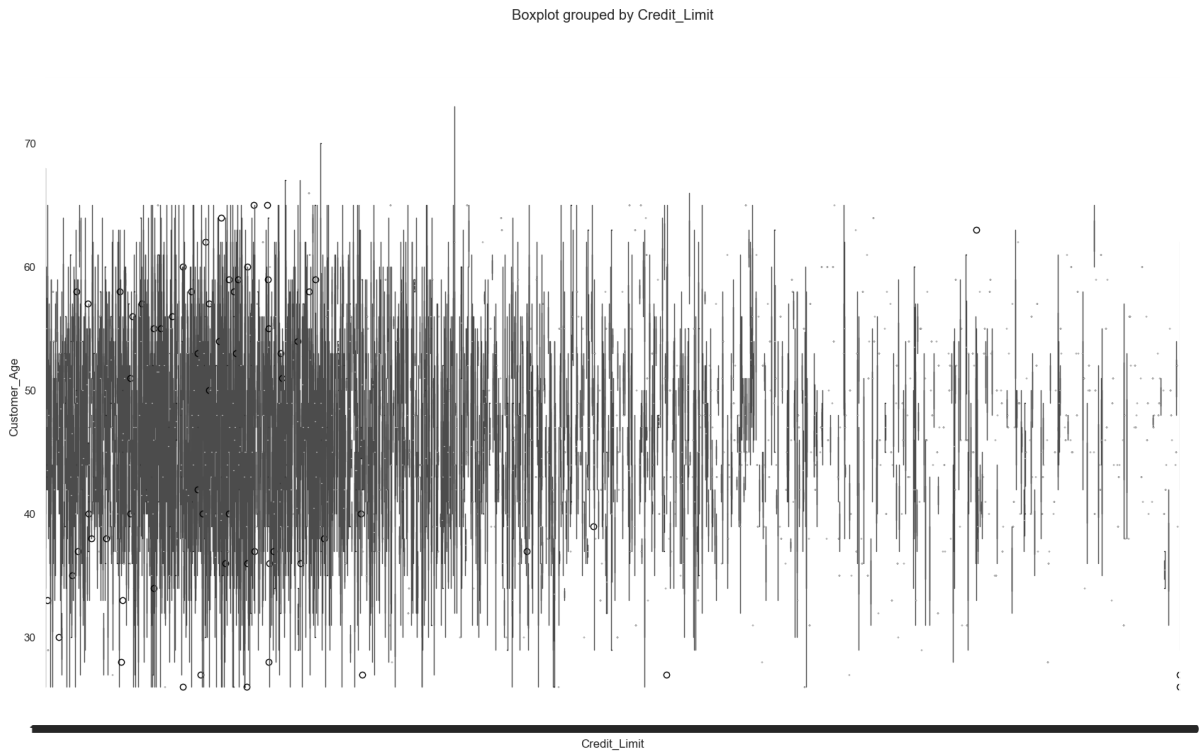


Q11) Plot a boxplot for the Credit_Limit column and check if it contains any outlier or not.

ax = df.boxplot(column="Customer_Age", by="Credit_Limit", figsize=(20,12))
ax.set_ylabel("Customer_Age")
ax.set_title("")

Solution :-

Text(0.5, 1.0, '')



It does not contains any outliers.

Q12) Map the Attrition_Flag values to 0 and 1(i.e.Existing Customer=o and Attrited Customer=1).Standardize the columns.

```
data = {
   "Attrition_Flag": ["Existing Customer", "Attrited Customer", "Existing Customer", "Existing Customer", "Attrited Customer"],
   "Age": [35, 42, 28, 55, 33],
   "Credit_Limit": [5000.0, 8000.0, 12000.0, 15000.0, 7000.0],
   "Total_Trans_Amt": [2300, 4500, 3500, 7000, 2800],
}
df = pd.DataFrame(data)
df['Attrition_Flag'] = df['Attrition_Flag'].map({'Existing Customer': 0, 'Attrited Customer': 1})
target = df['Attrition_Flag']
df = df.drop(columns=['Attrition_Flag'])
scaler = StandardScaler()
df = pd.DataFrame(scaler.fit_transform(df), columns=df.columns)
df['Attrition_Flag'] = target
print(df)
```

Solution :-

	Age	Credit_Limit	Total_Trans_Amt	Attrition_Flag
0	-0.384988	-1.218467	-1.034270	0
1	0.363600	-0.387694	0.288633	1
2	-1.133576	0.720003	-0.312686	0
3	1.753835	1.550777	1.791932	0
4	-0.598871	-0.664619	-0.733610	1