Importing all the required libraries

```
import pandas as pd
import numpy as np
from scipy.stats import binom,poisson,norm
import matplotlib.pyplot as plt
import seaborn as sns
```

Downloading the dataset

```
!gdown https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/walmart_data.csv
```

```
Downloading...
From: https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/walmart_data.csv
To: /content/walmart_data.csv
100% 23.0M/23.0M [00:00<00:00, 287MB/s]
```

```
!ls
```

```
sample_data  walmart_data.csv
```

Reading the CSV file and storing it in a variable

```
df = pd.read_csv("walmart_data.csv")
df.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

Checking for the details of the dataframe, there are a total of 10 columns, and out of that 5 are of type integer and 5 are type of object and there are a total of 550068 entries

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550068 entries, 0 to 550067
Data columns (total 10 columns):
 #   Column                      Non-Null Count   Dtype 
---  ------                      --------------   ----- 
 0   User_ID                     550068 non-null  int64 
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  int64 
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  int64 
 8   Product_Category            550068 non-null  int64 
 9   Purchase                    550068 non-null  int64 
dtypes: int64(5), object(5)
memory usage: 42.0+ MB
```

Statistical Summary of the data of type integers, it shows the mean, standar deviation and percentile count each column. The Mean of purchase is 9264.

```
df.describe()
```

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	5.500680e+05	550068.000000	550068.000000	550068.000000	550068.000000
mean	1.003029e+06	8.076707	0.409653	5.404270	9263.968713
std	1.727592e+03	6.522660	0.491770	3.936211	5023.065394
min	1.000001e+06	0.000000	0.000000	1.000000	12.000000
25%	1.001516e+06	2.000000	0.000000	1.000000	5823.000000
50%	1.003077e+06	7.000000	0.000000	5.000000	8047.000000
75%	1.004478e+06	14.000000	1.000000	8.000000	12054.000000
max	1.006040e+06	20.000000	1.000000	20.000000	23961.000000

Statistical Summary of the data of type Objects, it shows the count, unique values count.

```
df.describe(include="object")
```

	Product_ID	Gender	Age	City_Category	Stay_In_Current_City_Years
count	550068	550068	550068	550068	550068
unique	3631	2	7	3	5
top	P00265242	M	26-35	B	1
freq	1880	414259	219587	231173	193821

Checking for null values and there are no null values found in any of the columns

```
df.isna().any()
```

```
User_ID                       False
Product_ID                    False
Gender                        False
Age                           False
Occupation                    False
City_Category                 False
Stay_In_Current_City_Years    False
Marital_Status                False
Product_Category              False
Purchase                      False
dtype: bool
```

Checking for duplicates and there are no duplicate rows found.

```
df[df.duplicated()]
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase

Checking for number of unique values in each column.

```
df.nunique()
```

```
User_ID                       5891
Product_ID                    3631
Gender                           2
Age                              7
Occupation                      21
City_Category                    3
Stay_In_Current_City_Years       5
Marital_Status                   2
Product_Category                20
Purchase                     18105
dtype: int64
```

In the given dataset Males are more than Females

```
df.value_counts("Gender")
```

```
Gender
M    414259
F    135809
dtype: int64
```

In the given dataset, unmarrried people are more than the married people

```
df.value_counts("Marital_Status")
```

```
Marital_Status
0    324731
1    225337
dtype: int64
```

In the given dataset, majority of the people live in city B

```
df.value_counts("City_Category")
```

```
City_Category
B    231173
C    171175
A    147720
dtype: int64
```

In the given dataset, majority of the people falls under the age category of 26 to 35

```
df.value_counts("Age")
```

```
Age
26-35    219587
36-45    110013
18-25     99660
46-50     45701
51-55     38501
55+       21504
0-17      15102
dtype: int64
```

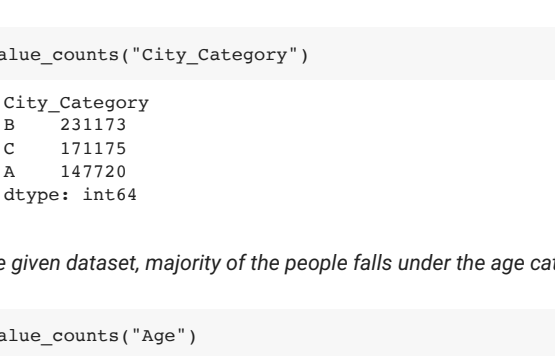
```
df.value_counts("Stay_In_Current_City_Years")
```

```
Stay_In_Current_City_Years
1     193821
2     101838
3      95285
4+     84726
0      74398
dtype: int64
```

From the below plot, we can see that most of the people stay mostly 1 year in the current city and maximum number of people stay in city B and the Minimum number of people stay in City A

```
sns.countplot(data=df,x="Stay_In_Current_City_Years", order = df.value_counts("Stay_In_Current_City_Years").index, hue="City_Category")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebdbdfb50>
```



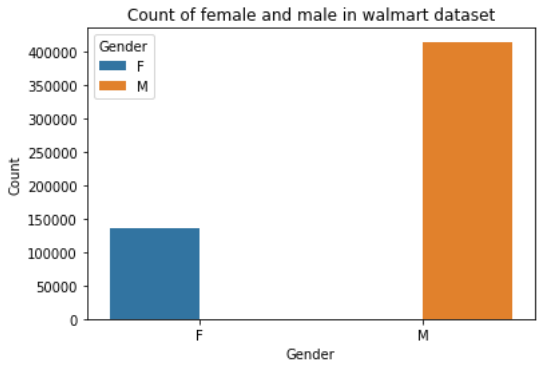

```
data = (df.groupby("Gender")['User_ID'].count()).reset_index()
data
```

	Gender	User_ID
0	F	135809
1	M	414259

From the below plot, we can see that males are more than the females

```
sns.barplot(data=data, x = 'Gender', y = 'User_ID', hue='Gender')
plt.ylabel('Count')
plt.title("Count of female and male in walmart dataset")
```

```
Text(0.5, 1.0, 'Count of female and male in walmart dataset')
```

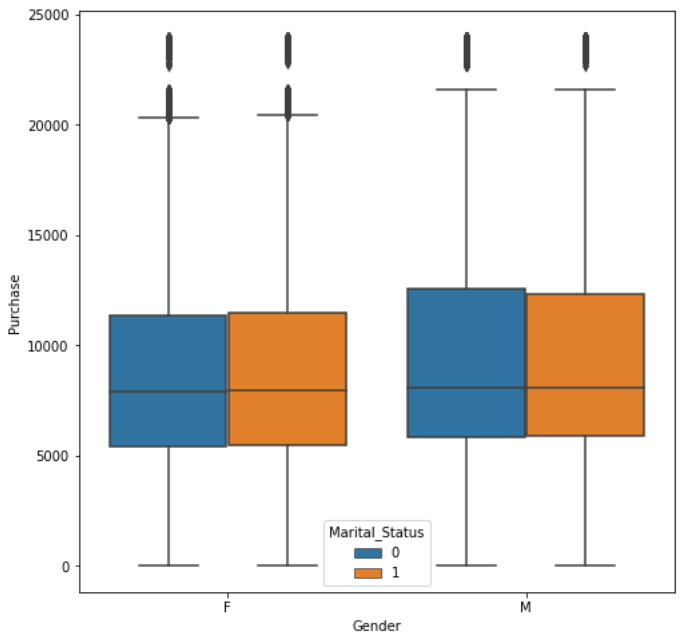


From the below plot we can see the average purchase of Married and Unmarried males and Married and Unmarried females

- From the given dataset both married and unmarried males and females people are having similar average purchase rates.
- From the below box plot we can see there are outliers and these outliers might affect the average purchase rate

```
plt.figure(figsize=(8,8))
sns.boxplot(y="Purchase",x="Gender",hue = "Marital_Status",data=df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebd5f8e80>
```

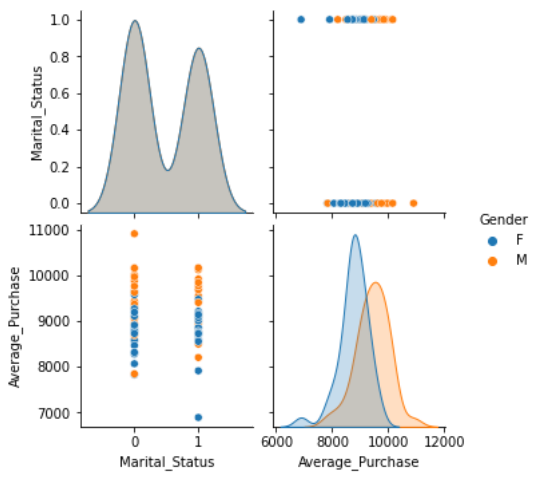



```
gender_marital_age_data = df.groupby(["City_Category","Age","Gender","Marital_Status"]).agg(Average_Purchase = ("Purchase","mean")).reset_index()
gender_marital_age_data
```

	City_Category	Age	Gender	Marital_Status	Average_Purchase
0	A	0-17	F	0	7826.252246
1	A	0-17	M	0	9655.655424
2	A	18-25	F	0	8558.911988
3	A	18-25	F	1	6892.483344
4	A	18-25	M	0	9044.066667
...	...	...	...	...	...
73	C	51-55	M	1	9837.798026
74	C	55+	F	0	8726.412274
75	C	55+	F	1	9134.650930
76	C	55+	M	0	9758.533138
77	C	55+	M	1	9400.414121

78 rows × 5 columns

```
sns.pairplot(data=gender_marital_age_data, hue='Gender')
plt.show()
```




```
gender_marital_data = df.groupby(["City_Category","Gender","Marital_Status"]).agg(Average_Purchase = ("Purchase","mean")).reset_index()
gender_marital_data
```

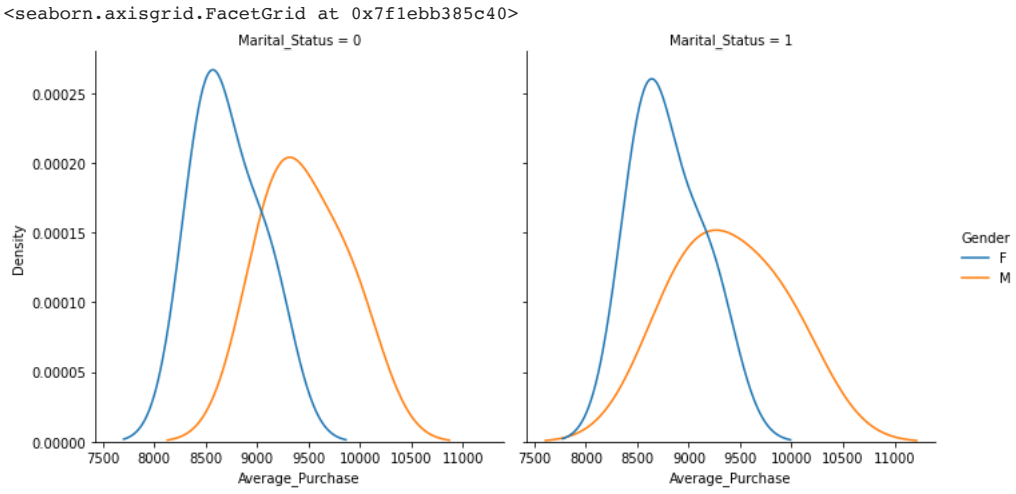
	City_Category	Gender	Marital_Status	Average_Purchase
0	A	F	0	8579.690979
1	A	F	1	8579.736254
2	A	M	0	9100.477475
3	A	M	1	8883.525545
4	B	F	0	8480.546732
5	B	F	1	8629.901199
6	B	M	0	9372.347212

From the below plot derived from the given dataset, we can see the following:

- The Average purchase of married females and unmarried females are almost same.
- The Average purchase of married males are slightly higher than the unmarried males.

```
sns.displot(data=gender_marital_data, x="Average_Purchase", hue="Gender", col="Marital_Status", kind="kde")
```

```
<seaborn.axisgrid.FacetGrid at 0x7f1ebb385c40>
```




```
age_data = df.groupby(["Age"]).agg(Average_Purchase = ("Purchase","mean")).reset_index()
age_data
```

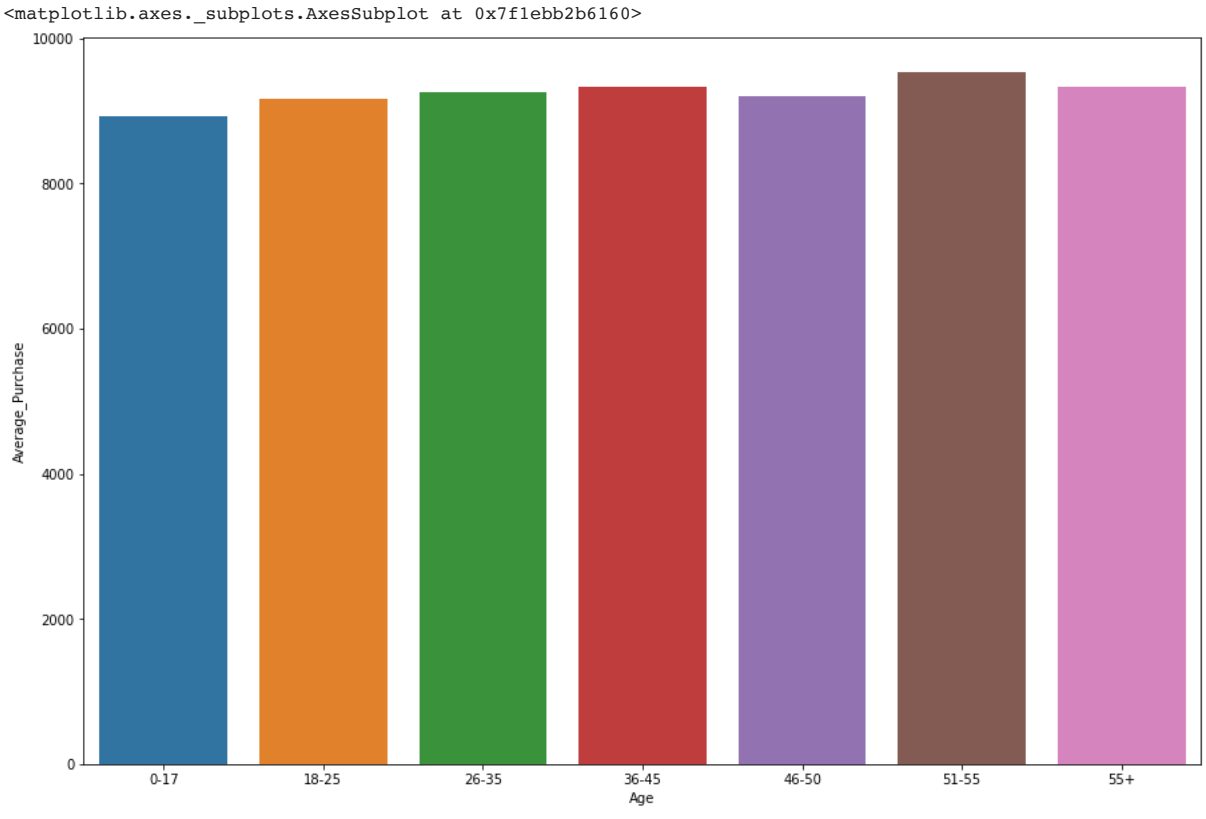
	Age	Average_Purchase
0	0-17	8933.464640
1	18-25	9169.663606
2	26-35	9252.690633
3	36-45	9331.350695
4	46-50	9208.625697
5	51-55	9534.808031
6	55+	9336.280459

From the below plot derived from the given dataset, we can see the following:

- The average purchase of all age group people are more or less similar.

```
plt.figure(figsize=(15,10))
sns.barplot(data=age_data, x = 'Age', y = 'Average_Purchase')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebb2b6160>
```



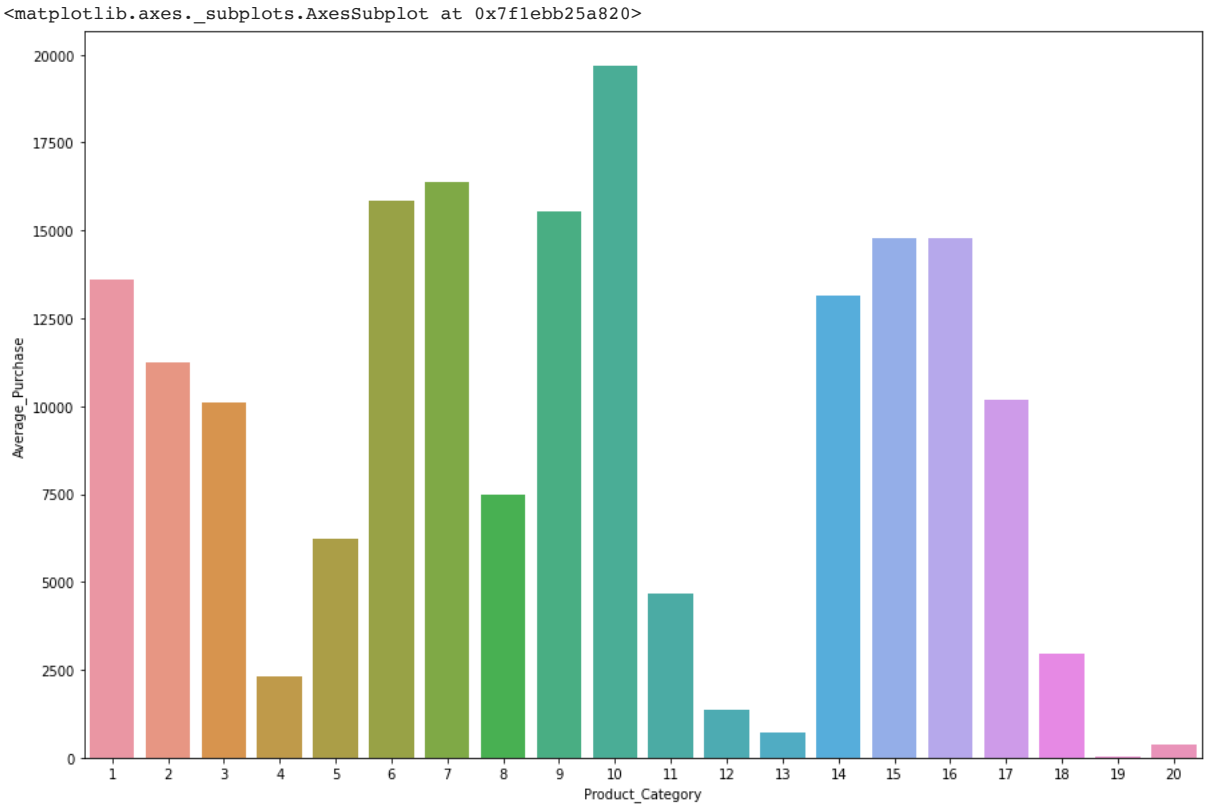

```
product_data = df.groupby(["Product_Category"]).agg(Average_Purchase = ("Purchase","mean")).reset_index()
product_data
```

	Product_Category	Average_Purchase
0	1	13606.218596
1	2	11251.935384
2	3	10096.705734
3	4	2329.659491
4	5	6240.088178
5	6	15838.478550
6	7	16365.689600
7	8	7498.958078
8	9	15537.375610
9	10	19675.570927
10	11	4685.268456
11	12	1350.859894

From the below plot, we can see that the average purchase rate of product category 10 is high compared to anyother product category

```
plt.figure(figsize=(15,10))
sns.barplot(data=product_data, x = 'Product_Category', y = 'Average_Purchase')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebb25a820>
```

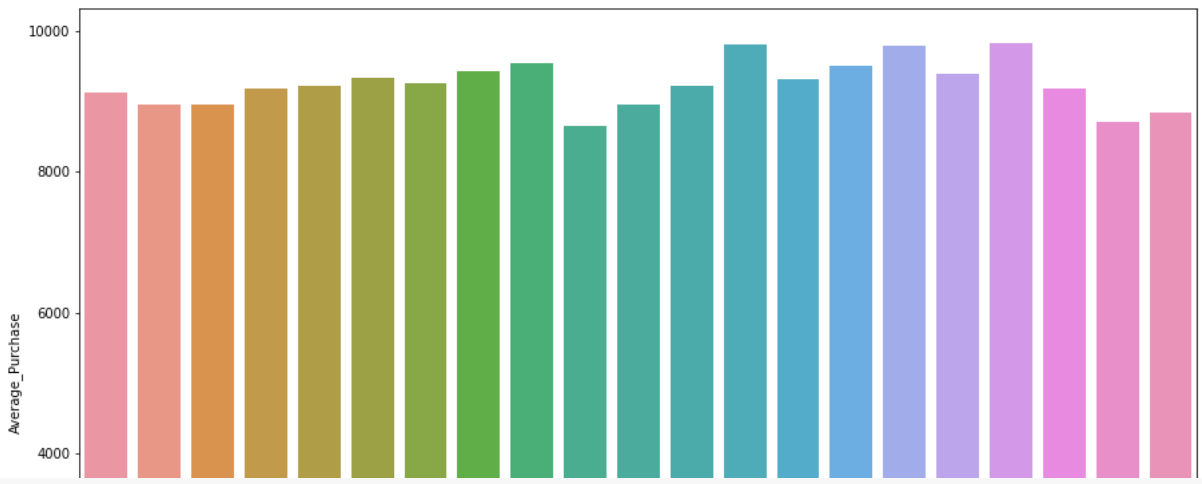



```
occupation_data = df.groupby(["Occupation"]).agg(Average_Purchase = ("Purchase","mean")).reset_index()
occupation_data
```

	Occupation	Average_Purchase
0	0	9124.428588
1	1	8953.193270
2	2	8952.481683
3	3	9178.593088
4	4	9213.980251
5	5	9333.149298
6	6	9256.535691
7	7	9425.728223
8	8	9532.592497
9	9	8637.743761
10	10	8959.355375
11	11	9213.845848
12	12	9796.640239
13	13	9306.351061
14	14	9500.702772
15	15	9778.891163
16	16	9394.464349
17	17	9821.478236
18	18	9169.655844
19	19	8710.627231
20	20	8836.494905

```
plt.figure(figsize=(15,10))
sns.barplot(data=occupation_data, x = 'Occupation', y = 'Average_Purchase')
```

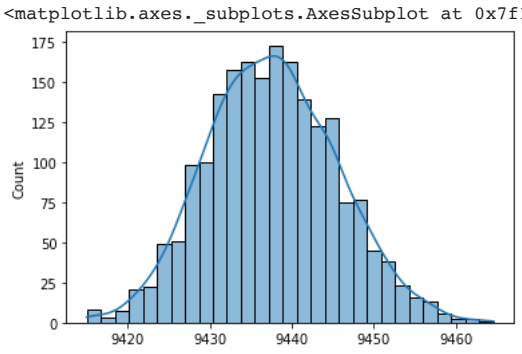
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebb0f17c0>
```




```
Male_Transaction_Series = df.loc[df["Gender"] == "M"]["Purchase"]
M_len = len(Male_Transaction_Series)
bootstrap_male_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Male_Transaction_Series, size = M_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_male_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_male_transaction,kde=True)
```

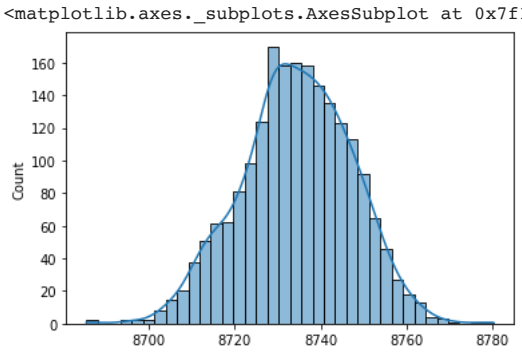
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebb1fc700>
```




```
Female_Transaction_Series = df.loc[df["Gender"] == "F"]["Purchase"]
F_len = len(Female_Transaction_Series)
bootstrap_female_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Female_Transaction_Series, size = F_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_female_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_female_transaction,kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebaf97790>
```



The confidence intervals when it's 90%, 95%, 99% it does not overlap each other with the male and female average purchase

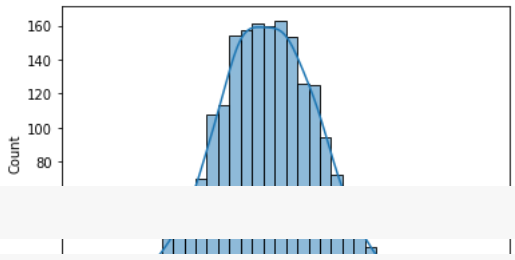
```
left = round(np.percentile(bootstrap_female_transaction, 5),2)
right = round(np.percentile(bootstrap_female_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by female lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_male_transaction, 5),2)
right = round(np.percentile(bootstrap_male_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by male lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_female_transaction, 2.5),2)
right = round(np.percentile(bootstrap_female_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by female lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_male_transaction, 2.5),2)
right = round(np.percentile(bootstrap_male_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by male lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_female_transaction, 0.5),2)
right = round(np.percentile(bootstrap_female_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by female lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_male_transaction, 0.5),2)
right = round(np.percentile(bootstrap_male_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by male lies between [{left}, {right}]")
```

```
With 90% confidence, the mean purchase by female lies between [8712.48, 8754.66]
With 90% confidence, the mean purchase by male lies between [9425.05, 9451.01]
With 95% confidence, the mean purchase by female lies between [8709.05, 8758.29]
With 95% confidence, the mean purchase by male lies between [9422.85, 9453.16]
With 99% confidence, the mean purchase by female lies between [8702.55, 8764.39]
With 99% confidence, the mean purchase by male lies between [9418.42, 9457.88]
```

```
Married_Transaction_Series = df.loc[df["Marital_Status"] == 1]["Purchase"]
Married_len = len(Married_Transaction_Series)
bootstrap_married_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Married_Transaction_Series, size = Married_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_married_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_married_transaction,kde=True)
```

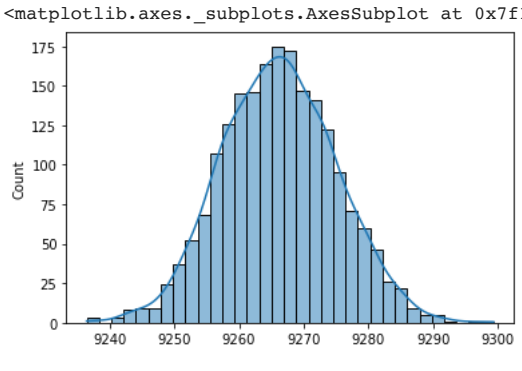
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebae63e20>
```




```
Unmarried_Transaction_Series = df.loc[df["Marital_Status"] == 0]["Purchase"]
Unmarried_len = len(Unmarried_Transaction_Series)
bootstrap_unmarried_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Unmarried_Transaction_Series, size = Unmarried_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_unmarried_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_unmarried_transaction,kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebadbf040>
```



The confidence intervals when it's 90%, 95%, 99% it does overlap each other with the married and unmarried peoples average purchase

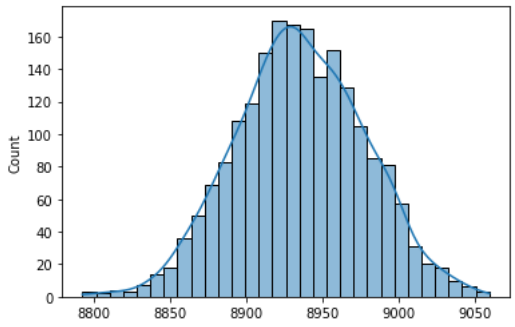
```
left = round(np.percentile(bootstrap_unmarried_transaction, 5),2)
right = round(np.percentile(bootstrap_unmarried_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people who are not married lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_married_transaction, 5),2)
right = round(np.percentile(bootstrap_married_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people who are married lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_unmarried_transaction, 2.5),2)
right = round(np.percentile(bootstrap_unmarried_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people who are not married lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_married_transaction, 2.5),2)
right = round(np.percentile(bootstrap_married_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people who are married lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_unmarried_transaction, 0.5),2)
right = round(np.percentile(bootstrap_unmarried_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people who are not married lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_married_transaction, 0.5),2)
right = round(np.percentile(bootstrap_married_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people who are married lies between [{left}, {right}]")
```

```
With 90% confidence, the mean purchase by the people who are not married lies between [9252.11, 9280.79]
With 90% confidence, the mean purchase by the people who are married lies between [9243.09, 9278.14]
With 95% confidence, the mean purchase by the people who are not married lies between [9249.39, 9283.77]
With 95% confidence, the mean purchase by the people who are married lies between [9240.2, 9281.19]
With 99% confidence, the mean purchase by the people who are not married lies between [9243.23, 9288.79]
With 99% confidence, the mean purchase by the people who are married lies between [9235.12, 9287.08]
```

```
Age017_Transaction_Series = df.loc[df["Age"] == "0-17"]["Purchase"]
Age017_len = len(Age017_Transaction_Series)
bootstrap_age017_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Age017_Transaction_Series, size = Age017_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_age017_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_age017_transaction,kde=True)
```

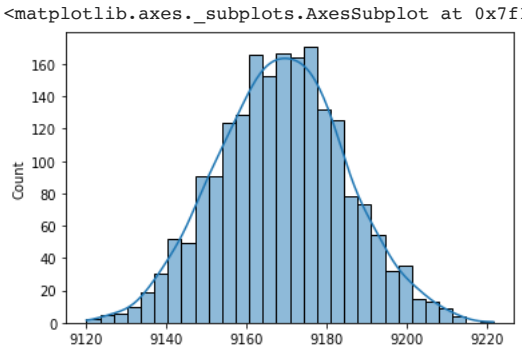
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebd622b80>
```




```
Age1825_Transaction_Series = df.loc[df["Age"] == "18-25"]["Purchase"]
Age1825_len = len(Age1825_Transaction_Series)
bootstrap_age1825_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Age1825_Transaction_Series, size = Age1825_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_age1825_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_age1825_transaction,kde=True)
```

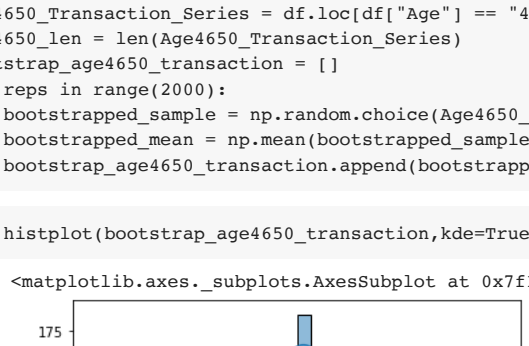
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebd428ca0>
```



```
Age2635_Transaction_Series = df.loc[df["Age"] == "26-35"]["Purchase"]
Age2635_len = len(Age2635_Transaction_Series)
bootstrap_age2635_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Age2635_Transaction_Series, size = Age2635_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_age2635_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_age2635_transaction,kde=True)
```

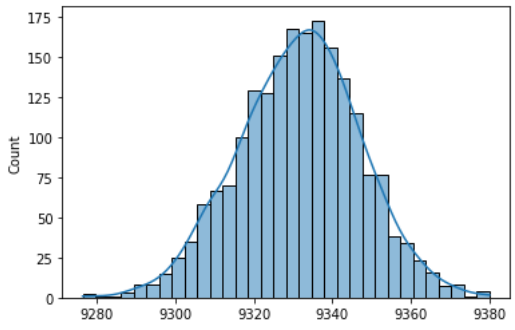
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebad2c850>
```




```
Age3645_Transaction_Series = df.loc[df["Age"] == "36-45"]["Purchase"]
Age3645_len = len(Age3645_Transaction_Series)
bootstrap_age3645_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Age3645_Transaction_Series, size = Age3645_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_age3645_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_age3645_transaction,kde=True)
```

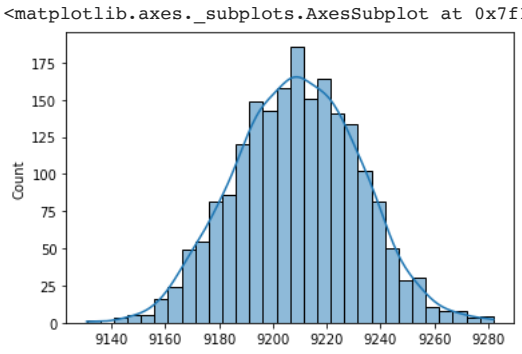
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebac67520>
```




```
Age4650_Transaction_Series = df.loc[df["Age"] == "46-50"]["Purchase"]
Age4650_len = len(Age4650_Transaction_Series)
bootstrap_age4650_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Age4650_Transaction_Series, size = Age4650_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_age4650_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_age4650_transaction,kde=True)
```

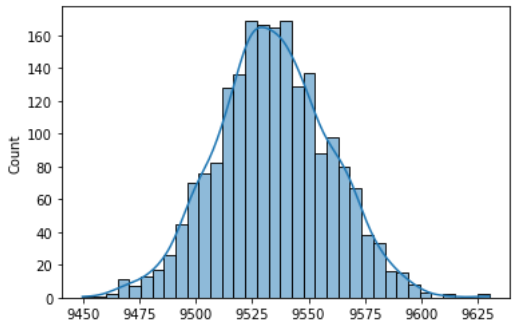
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebb138fa0>
```




```
Age5155_Transaction_Series = df.loc[df["Age"] == "51-55"]["Purchase"]
Age5155_len = len(Age5155_Transaction_Series)
bootstrap_age5155_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Age5155_Transaction_Series, size = Age5155_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_age5155_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_age5155_transaction,kde=True)
```

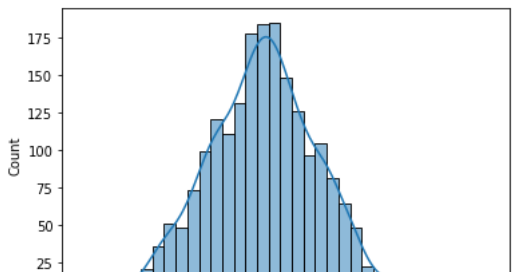
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebaaf1880>
```




```
Age55plus_Transaction_Series = df.loc[df["Age"] == "55+"]["Purchase"]
Age55plus_len = len(Age55plus_Transaction_Series)
bootstrap_age55plus_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(Age55plus_Transaction_Series, size = Age55plus_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_age55plus_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_age55plus_transaction,kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1ebaa39400>
```



The confidence intervals when it's 90%, 95%, 99% it does not overlap each other with any age groups average purchase, but it does overlap with the age group of 18-25 and 46-50

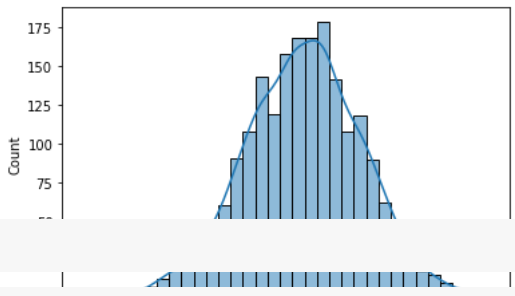
```
left = round(np.percentile(bootstrap_age017_transaction, 5),2)
right = round(np.percentile(bootstrap_age017_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in the age group 0-17 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age1825_transaction, 5),2)
right = round(np.percentile(bootstrap_age1825_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in the age group 18-25 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age2635_transaction, 5),2)
right = round(np.percentile(bootstrap_age2635_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in the age group 26-35 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age3645_transaction, 5),2)
right = round(np.percentile(bootstrap_age3645_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in the age group 36-45 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age4650_transaction, 5),2)
right = round(np.percentile(bootstrap_age4650_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in the age group 46-50 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age5155_transaction, 5),2)
right = round(np.percentile(bootstrap_age5155_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in the age group 51-55 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age55plus_transaction, 5),2)
right = round(np.percentile(bootstrap_age55plus_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in the age group 55+ lies between [{left}, {right}]")

left = round(np.percentile(bootstrap_age017_transaction, 2.5),2)
right = round(np.percentile(bootstrap_age017_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in the age group 0-17 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age1825_transaction, 2.5),2)
right = round(np.percentile(bootstrap_age1825_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in the age group 18-25 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age2635_transaction, 2.5),2)
right = round(np.percentile(bootstrap_age2635_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in the age group 26-35 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age3645_transaction, 2.5),2)
right = round(np.percentile(bootstrap_age3645_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in the age group 36-45 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age4650_transaction, 2.5),2)
right = round(np.percentile(bootstrap_age4650_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in the age group 46-50 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age5155_transaction, 2.5),2)
right = round(np.percentile(bootstrap_age5155_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in the age group 51-55 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age55plus_transaction, 2.5),2)
right = round(np.percentile(bootstrap_age55plus_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in the age group 55+ lies between [{left}, {right}]")

left = round(np.percentile(bootstrap_age017_transaction, 0.5),2)
right = round(np.percentile(bootstrap_age017_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in the age group 0-17 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age1825_transaction, 0.5),2)
right = round(np.percentile(bootstrap_age1825_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in the age group 18-25 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age2635_transaction, 0.5),2)
right = round(np.percentile(bootstrap_age2635_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in the age group 26-35 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age55plus_transaction, 0.5),2)
right = round(np.percentile(bootstrap_age55plus_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in the age group 55+ lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age5155_transaction, 0.5),2)
right = round(np.percentile(bootstrap_age5155_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in the age group 51-55 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age4650_transaction, 0.5),2)
right = round(np.percentile(bootstrap_age4650_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in the age group 46-50 lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_age3645_transaction, 0.5),2)
right = round(np.percentile(bootstrap_age3645_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in the age group 36-45 lies between [{left}, {right}]")
```

```
With 90% confidence, the mean purchase by the people in the age group 0-17 lies between [8865.96, 9003.71]
With 90% confidence, the mean purchase by the people in the age group 18-25 lies between [9142.48, 9194.87]
With 90% confidence, the mean purchase by the people in the age group 26-35 lies between [9235.38, 9269.67]
With 90% confidence, the mean purchase by the people in the age group 36-45 lies between [9305.7, 9356.4]
With 90% confidence, the mean purchase by the people in the age group 46-50 lies between [9170.82, 9245.77]
With 90% confidence, the mean purchase by the people in the age group 51-55 lies between [9494.16, 9575.42]
With 90% confidence, the mean purchase by the people in the age group 55+ lies between [9277.74, 9391.28]
With 95% confidence, the mean purchase by the people in the age group 0-17 lies between [8854.0, 9018.67]
With 95% confidence, the mean purchase by the people in the age group 18-25 lies between [9137.86, 9200.87]
With 95% confidence, the mean purchase by the people in the age group 26-35 lies between [9230.78, 9272.84]
With 95% confidence, the mean purchase by the people in the age group 36-45 lies between [9301.07, 9361.22]
With 95% confidence, the mean purchase by the people in the age group 46-50 lies between [9165.01, 9253.57]
With 95% confidence, the mean purchase by the people in the age group 51-55 lies between [9485.6, 9582.82]
With 95% confidence, the mean purchase by the people in the age group 55+ lies between [9268.1, 9401.38]
With 99% confidence, the mean purchase by the people in the age group 0-17 lies between [8823.57, 9040.65]
With 99% confidence, the mean purchase by the people in the age group 18-25 lies between [9128.87, 9208.8]
With 99% confidence, the mean purchase by the people in the age group 26-35 lies between [9224.75, 9278.11]
With 99% confidence, the mean purchase by the people in the age group 55+ lies between [9248.58, 9426.31]
With 99% confidence, the mean purchase by the people in the age group 51-55 lies between [9469.16, 9596.02]
With 99% confidence, the mean purchase by the people in the age group 46-50 lies between [9152.32, 9269.24]
With 99% confidence, the mean purchase by the people in the age group 36-45 lies between [9290.89, 9371.0]
```

```
CityA_Transaction_Series = df.loc[df["City_Category"] == "A"]["Purchase"]
CityA_len = len(CityA_Transaction_Series)
bootstrap_citya_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(CityA_Transaction_Series, size = CityA_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_citya_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_citya_transaction,kde=True)
```

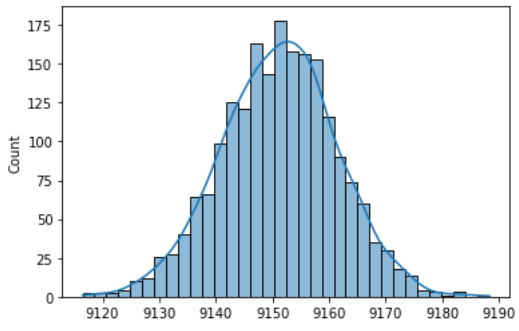
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1eba9f45b0>
```




```
CityB_Transaction_Series = df.loc[df["City_Category"] == "B"]["Purchase"]
CityB_len = len(CityB_Transaction_Series)
bootstrap_cityb_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(CityB_Transaction_Series, size = CityB_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_cityb_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_cityb_transaction,kde=True)
```

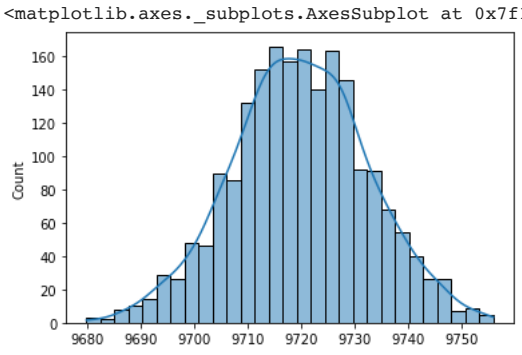
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1eba8cc880>
```




```
CityC_Transaction_Series = df.loc[df["City_Category"] == "C"]["Purchase"]
CityC_len = len(CityC_Transaction_Series)
bootstrap_cityc_transaction = []
for reps in range(2000):
    bootstrapped_sample = np.random.choice(CityC_Transaction_Series, size = CityC_len)
    bootstrapped_mean = np.mean(bootstrapped_sample)
    bootstrap_cityc_transaction.append(bootstrapped_mean)
```

```
sns.histplot(bootstrap_cityc_transaction,kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1eba814fa0>
```



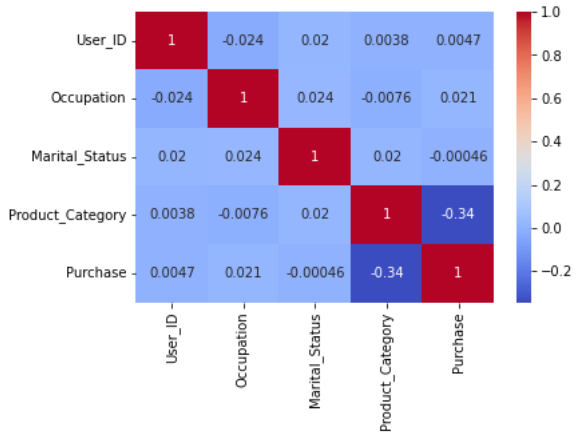
The confidence intervals when it's 90%, 95%, 99% it does not overlap each other with the any cities average purchase

```
left = round(np.percentile(bootstrap_citya_transaction, 5),2)
right = round(np.percentile(bootstrap_citya_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in City A lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_cityb_transaction, 5),2)
right = round(np.percentile(bootstrap_cityb_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in City B lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_cityc_transaction, 5),2)
right = round(np.percentile(bootstrap_cityc_transaction, 95),2)
print(f"With 90% confidence, the mean purchase by the people in City C lies between [{left}, {right}]")
```

```
left = round(np.percentile(bootstrap_citya_transaction, 2.5),2)
right = round(np.percentile(bootstrap_citya_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in City A lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_cityb_transaction, 2.5),2)
right = round(np.percentile(bootstrap_cityb_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in City B lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_cityc_transaction, 2.5),2)
right = round(np.percentile(bootstrap_cityc_transaction, 97.5),2)
print(f"With 95% confidence, the mean purchase by the people in City C lies between [{left}, {right}]")

left = round(np.percentile(bootstrap_citya_transaction, 0.5),2)
right = round(np.percentile(bootstrap_citya_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in City A lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_cityb_transaction, 0.5),2)
right = round(np.percentile(bootstrap_cityb_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in City B lies between [{left}, {right}]")
left = round(np.percentile(bootstrap_cityc_transaction, 0.5),2)
right = round(np.percentile(bootstrap_cityc_transaction, 99.5),2)
print(f"With 99% confidence, the mean purchase by the people in City C lies between [{left}, {right}]")
```

```
With 90% confidence, the mean purchase by the people in City A lies between [8890.69, 8932.1]
With 90% confidence, the mean purchase by the people in City B lies between [9134.28, 9167.41]
With 90% confidence, the mean purchase by the people in City C lies between [9698.79, 9740.84]
With 95% confidence, the mean purchase by the people in City A lies between [8886.68, 8936.42]
With 95% confidence, the mean purchase by the people in City B lies between [9130.92, 9170.97]
With 95% confidence, the mean purchase by the people in City C lies between [9694.4, 9745.1]
With 99% confidence, the mean purchase by the people in City A lies between [8879.19, 8945.02]
With 99% confidence, the mean purchase by the people in City B lies between [9125.04, 9175.92]
With 99% confidence, the mean purchase by the people in City C lies between [9687.45, 9751.93]
```

```
sns.heatmap(df.corr(), cmap= "coolwarm", annot=True)
plt.show()
```


User_ID
Occupation
Marital_Status
Product_Category
Purchase
1
-0.024
0.02
0.0038
0.0047
-0.024
1
0.024
-0.0076
0.021
0.02
0.024
1
0.02
-0.00046
0.0038
-0.0076
0.02
1
-0.34
0.0047
0.021
-0.00046
-0.34
1
1.0
0.8
0.6
0.4
0.2
0.0
−0.2