```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
import warnings as wr
wr.filterwarnings('ignore')


df=pd.read_csv('/content/walmart_data.txt')
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

```
shape=df.shape
print(f'Total no of rows: {shape[0]}')
print(f'Total no of columns: {shape[1]}')
```

```
Total no of rows: 550068
Total no of columns: 10
```

```
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
```

```
df.isna().sum(axis=0)
```

	0
User_ID	0
Product_ID	0
Gender	0
Age	0
Occupation	0
City_Category	0
Stay_In_Current_City_Years	0
Marital_Status	0
Product_Category	0
Purchase	0

dtype: int64

No null value present in the data

```
df.duplicated().sum()
```

```
df.duplicated().sum()
```

```
0
```

No duplicated values present in the data

```
df['User_ID'].value_counts()
```

User_ID	count
1001680	1026
1004277	979
1001941	898
1001181	862
1000889	823
...	...
1002690	7
1002111	7
1005810	7
1004991	7
1000708	6

5891 rows × 1 columns

dtype: int64

Most number of orders have been done by the customer_id 1001680

```
df['User_ID'].nunique()
```

```
5891
```

There are 5891 unique user_ids are present in the data

```
df['Product_ID'].value_counts()
```

Product_ID	count
P00265242	1880
P00025442	1615
P00110742	1612
P00112142	1562
P00057642	1470
...	...
P00314842	1
P00298842	1
P00231642	1
P00204442	1
P00066342	1

3631 rows × 1 columns

dtype: int64

Most number of oders have been placed for the product_id P00265242. A total of 1880 orders have been recieved for the product_id P00265242.

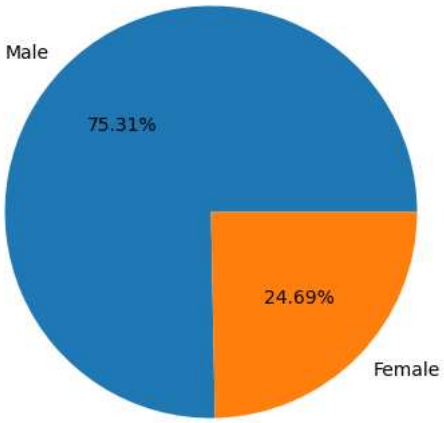
3631 unique product_ids present in the data

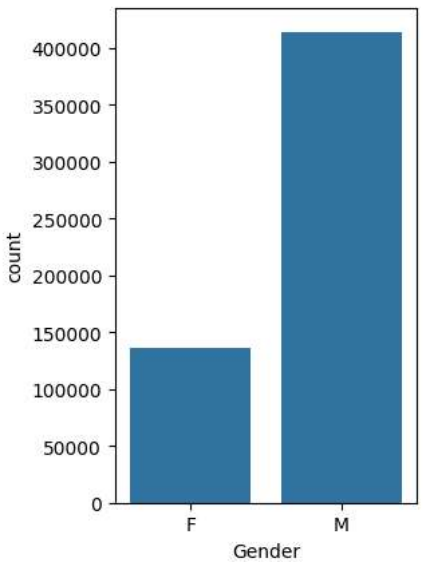
```
gender_wise_pur=df['Gender'].value_counts()
gender_wise_pur
```

Gender	count
M	414259
F	135809

dtype: int64

```
# purchases based on gender


plt.figure(figsize=(14,5))
plt.subplot(1,2,1)
plt.pie(gender_wise_pur.values,labels=['Male','Female'],autopct='%.2f%%')
plt.show()

plt.subplot(1,2,2)
sns.countplot(x=df['Gender'])
plt.show()
```





75.31% of customers of walmart are male customers. Rest 24.69% are female customers.

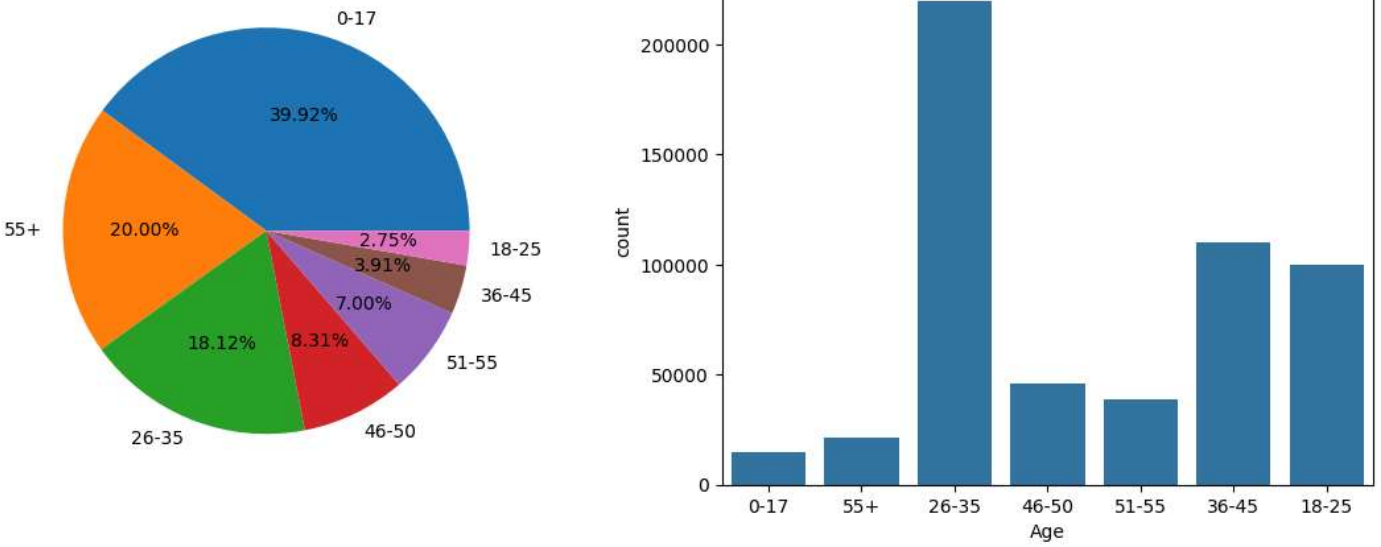
Men prefer walmart store more than female with the total number orders being placed 414259.

```
df['Age'].value_counts()
```

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

dtype: int64

```
# bar plot for age related ditribution


plt.figure(figsize=(14,5))
plt.subplot(1,2,1)
plt.pie(df['Age'].value_counts().values,labels=df['Age'].unique(),autopct='%.2f%%')

plt.subplot(1,2,2)

sns.countplot(x=df['Age'])
plt.show()
```



The customers are divided into 7 age groups according to their age.

customers aging between 26-35 are prefering walmart store a lot than other age group customers.

40% of the buyers fall under the age group of 26-35 which is the highest amongst all age groups.

Approximately 0.22 million records are present for age group 26-35 followed by 0.11 million records for group 36-45.

Age group 0-17 and 55+ are the least frequent buyers which is only 3%(15102) and 4%(21504) of the data respectively.

We can observe that most buyers are in within the age of 18-45 before and after this range we can see less buyers.

```
df['Occupation'].value_counts()
```

Occupation	count
4	72308
0	69638
7	59133
1	47426
17	40043
20	33562
12	31179
14	27309
2	26588
16	25371
6	20355
3	17650
10	12930
5	12177
15	12165
11	11586
19	8461
13	7728
18	6622
9	6291
8	1546

dtype: int64

```
df['Occupation'].nunique()
```

```
21
```

21 unique occupation are there.

The occupation type 4 is having highest number of orders with order count 72308.

```
df['City_Category'].value_counts()
```

City_Category	count
B	231173
C	171175
A	147720

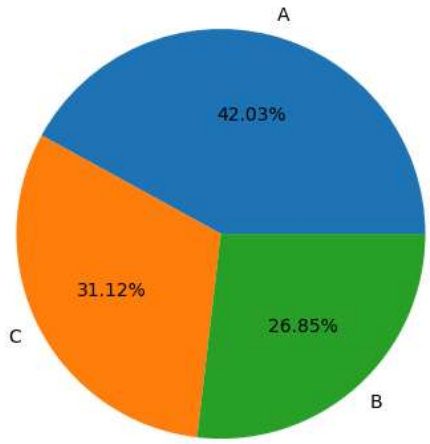
dtype: int64

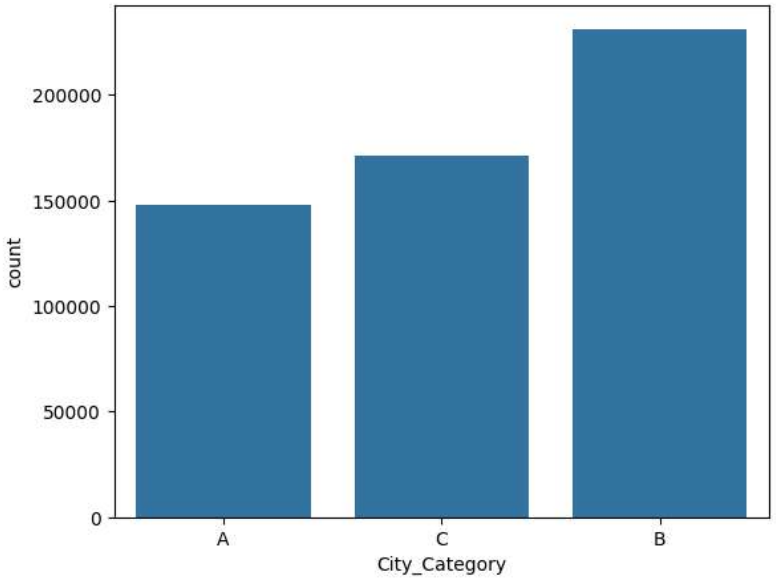
Walmart is having customers from 3 different cities.

Among them city B customers are having highest number purchases from walmart store.

```
# based on city category
plt.figure(figsize=(14,5))
plt.subplot(1,2,1)
plt.pie(df['City_Category'].value_counts().values,labels=df['City_Category'].unique(),autopct='%.2f%%')

plt.subplot(1,2,2)
```

```
sns.countplot(x=df['City_Category'])
plt.show()
```





Buyers from city B is 42.03% in proportion while from A and C are 26.85% and 31.13% respectively Based on Purcha

```
df['Marital_Status'].value_counts()
```

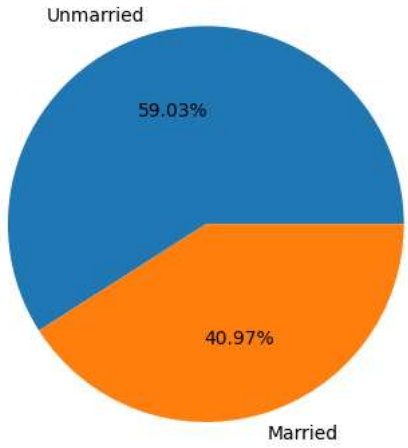
Marital_Status	count
0	324731
1	225337

dtype: int64

Here 0 means unmarried. 1 means married.

unmarried customers are more than the married customers.

```
# plotting pie chart based on marital_status

plt.pie(df['Marital_Status'].value_counts().values,labels=['Unmarried','Married'],autopct='%.2f%%')
plt.show()
```



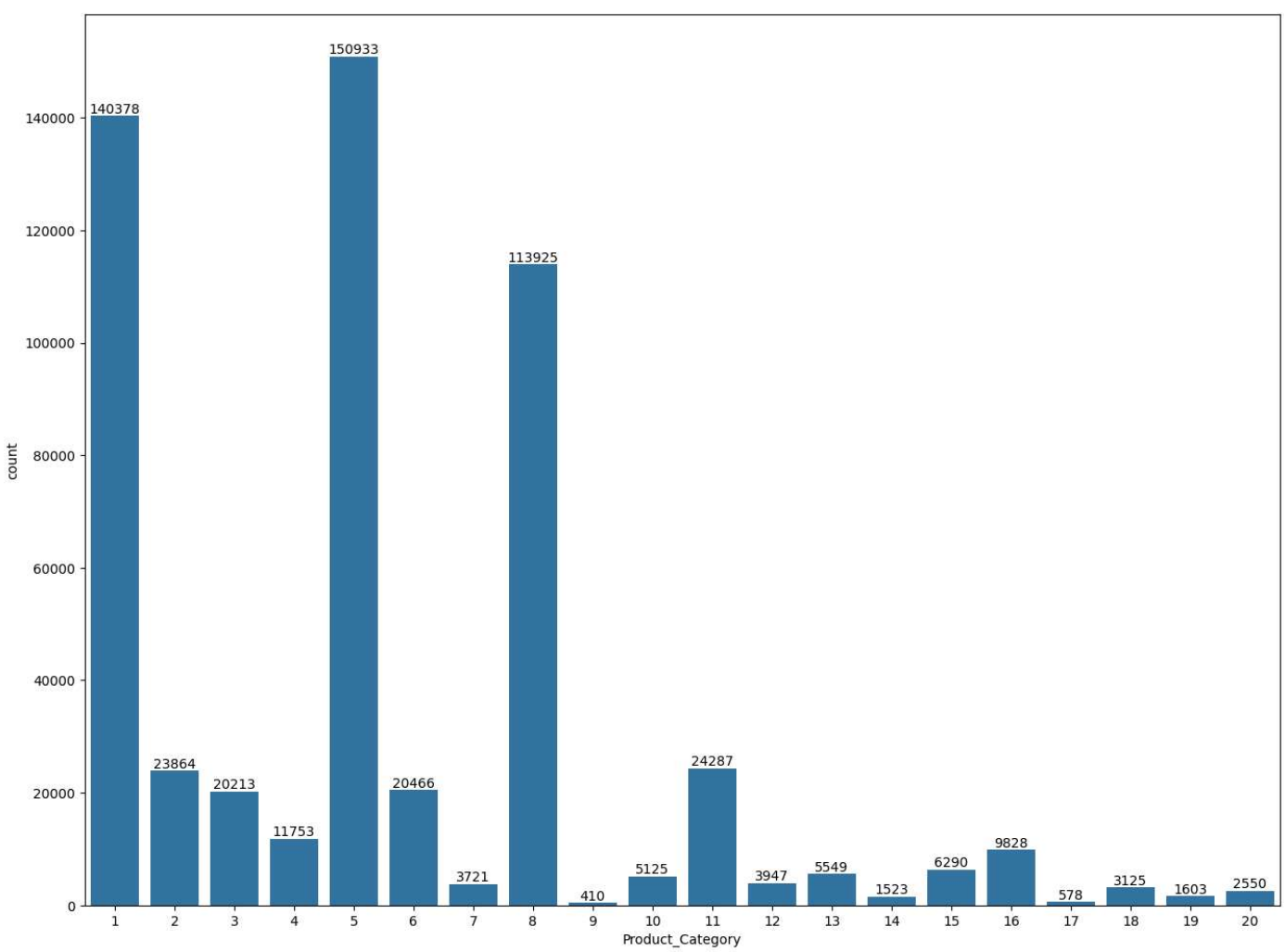
walmart is having 59.03% unmarried customers and remaining 40.97% are married customers.

```
df['Product_Category'].value_counts()
```

Product_Category	count
5	150933
1	140378
8	113925
11	24287
2	23864
6	20466
3	20213
4	11753
16	9828
15	6290
13	5549
10	5125
12	3947
7	3721
18	3125
20	2550
19	1603
14	1523
17	578
9	410

dtype: int64

```
plt.figure(figsize=(16,12))
prod_cat=sns.countplot(x=df['Product_Category'])
for i in prod_cat.containers:
  prod_cat.bar_label(i)
plt.show()
```



From the above plot we can colclude that:

walmart is having 20 unique product categories.

Most frequent categories of the product which was bought are 5,1,8 in descreasing order.

And least categories of the product which was bought are 19,17,14 in ascending order

```
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

```
df.describe(include=['O'])
```

	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

Product_ID:'P00265242' is the most frequent product purchased by customers with '1880' frequency.

Gender:'Male' Most of the users are Male with '414259' counts.

Age: '26-35' is the most frequent customers Age group with frequency '219587'.

City_Category: Customers from 'City_Category:B' made most of the transactions.

```
df.groupby(['Gender'])['Purchase'].describe()
```

Gender	count	mean	std	min	25%	50%	75%	max
F	135809.0	8734.565765	4767.233289	12.0	5433.0	7914.0	11400.0	23959.0
M	414259.0	9437.526040	5092.186210	12.0	5863.0	8098.0	12454.0	23961.0

```
df.groupby(['Marital_Status'])['Purchase'].describe()
```

Marital_Status	count	mean	std	min	25%	50%	75%	max
0	324731.0	9265.907619	5027.347859	12.0	5605.0	8044.0	12061.0	23961.0
1	225337.0	9261.174574	5016.897378	12.0	5843.0	8051.0	12042.0	23961.0

```
df.groupby(['Age'])['Purchase'].describe()
```

Age	count	mean	std	min	25%	50%	75%	max
0-17	15102.0	8933.464640	5111.114046	12.0	5328.0	7986.0	11874.0	23955.0
18-25	99660.0	9169.663606	5034.321997	12.0	5415.0	8027.0	12028.0	23958.0
26-35	219587.0	9252.690633	5010.527303	12.0	5475.0	8030.0	12047.0	23961.0
36-45	110013.0	9331.350695	5022.923879	12.0	5876.0	8061.0	12107.0	23960.0
46-50	45701.0	9208.625697	4967.216367	12.0	5888.0	8036.0	11997.0	23960.0
51-55	38501.0	9534.808031	5087.368080	12.0	6017.0	8130.0	12462.0	23960.0
55+	21504.0	9336.280459	5011.493996	12.0	6018.0	8105.5	11932.0	23960.0

```
df.groupby(['City_Category'])['Purchase'].describe()
```

City_Category	count	mean	std	min	25%	50%	75%	max
A	147720.0	8911.939216	4892.115238	12.0	5403.0	7931.0	11786.0	23961.0
B	231173.0	9151.300563	4955.496566	12.0	5460.0	8005.0	11986.0	23960.0
C	171175.0	9719.920993	5189.465121	12.0	6031.5	8585.0	13197.0	23961.0

There are more single people than married people.

Most of the customers are between the ages of 26 and 35.

The majority of our customers come from city category B but customers come from City category C spent more as mean is 9719.

Male customers tend to spend more than female customers, as the mean is higher for male customers.

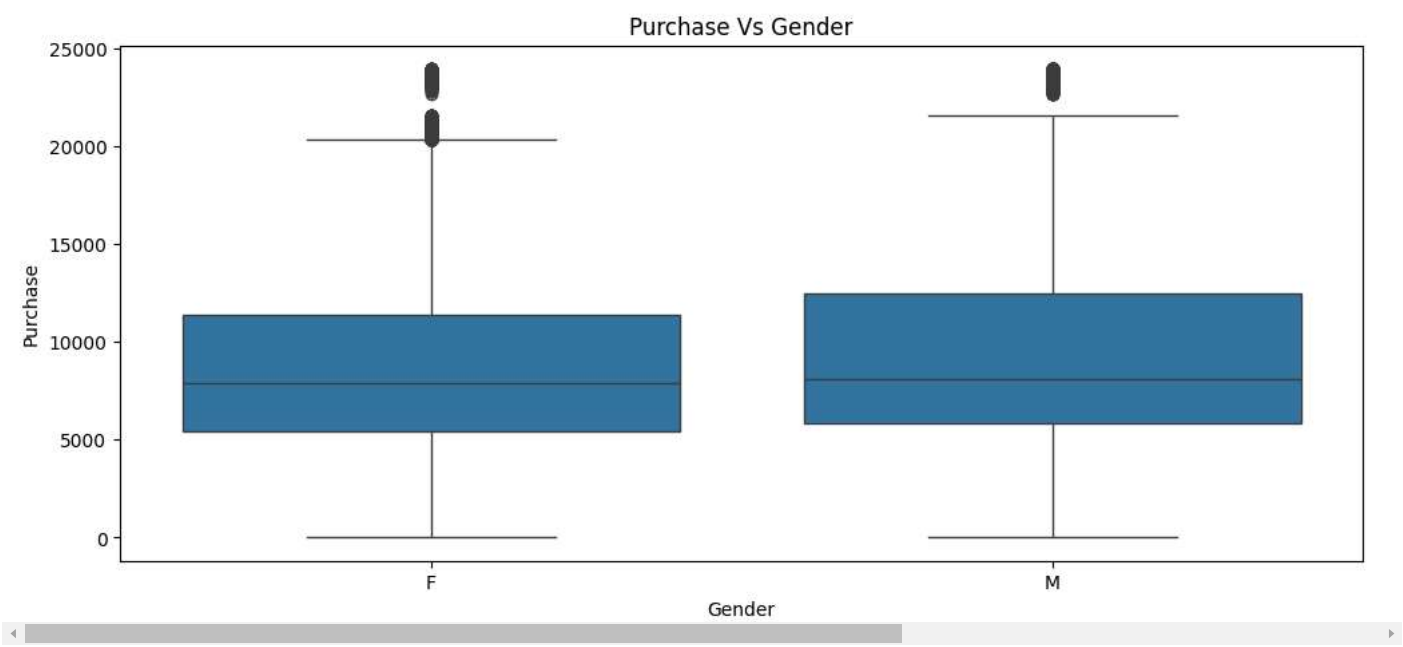
The majority of users come from City Category C, but more people from City Category B tend to purchase, which suggests the same users visit the outlet multiple times in City Category B.

```
Start coding or generate with AI.
```

Univariate analysis

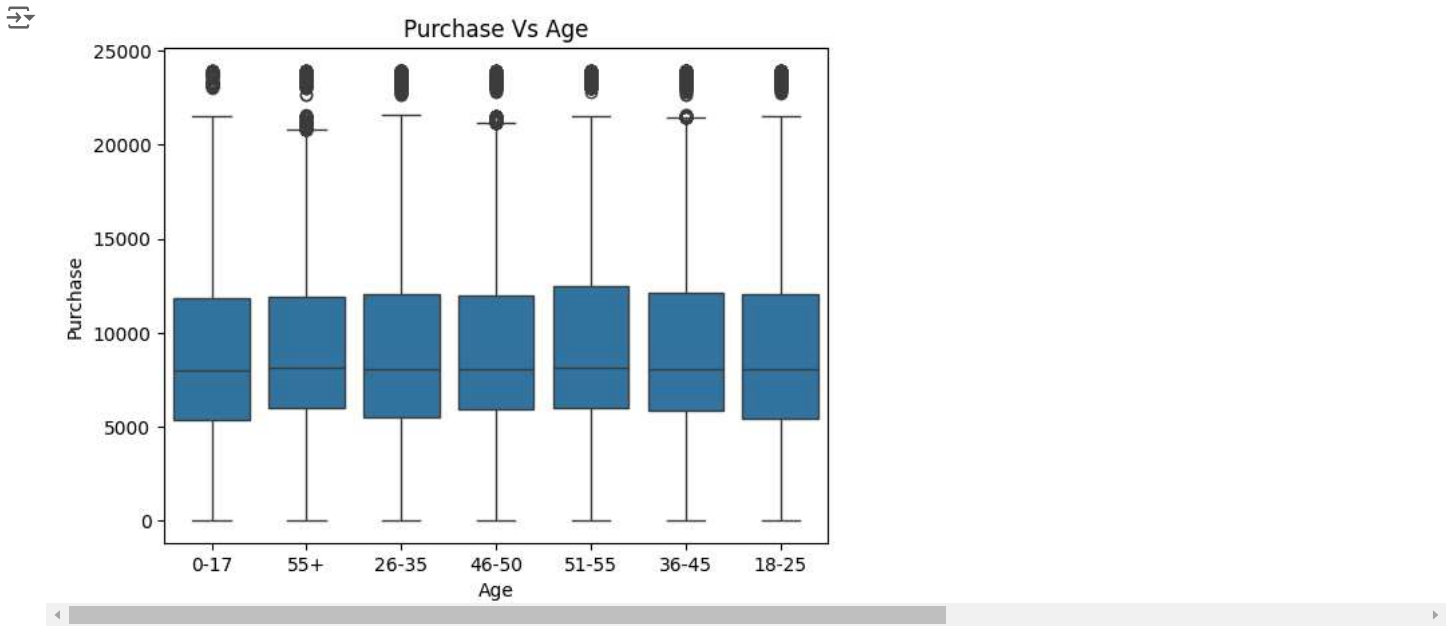
Based on Gender and Purchase Habits

```
plt.figure(figsize=(12,5))
sns.boxplot(data=df,x='Gender',y='Purchase')
plt.title('Purchase Vs Gender')
plt.show()
```



Above figure shows that Male spending behaviour is more than Females

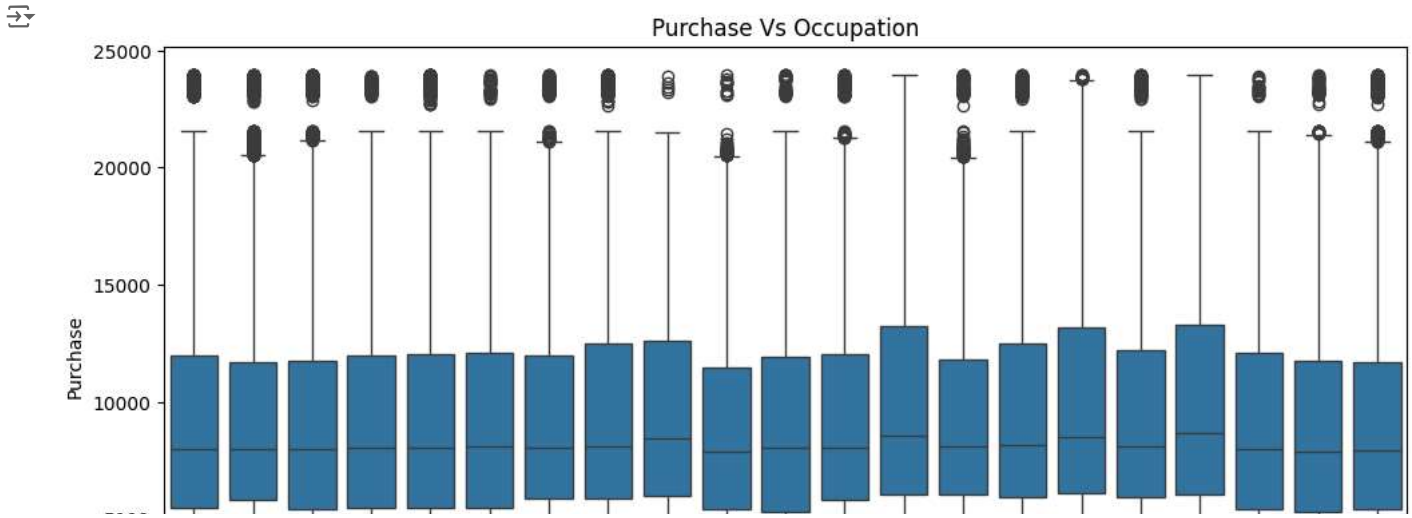
```
# Purchase Habit Based on Age Group

sns.boxplot(data=df,x='Age',y='Purchase')
plt.title('Purchase Vs Age')
plt.show()
```



Above figure shows there is no significance different in puchase habits based on age group

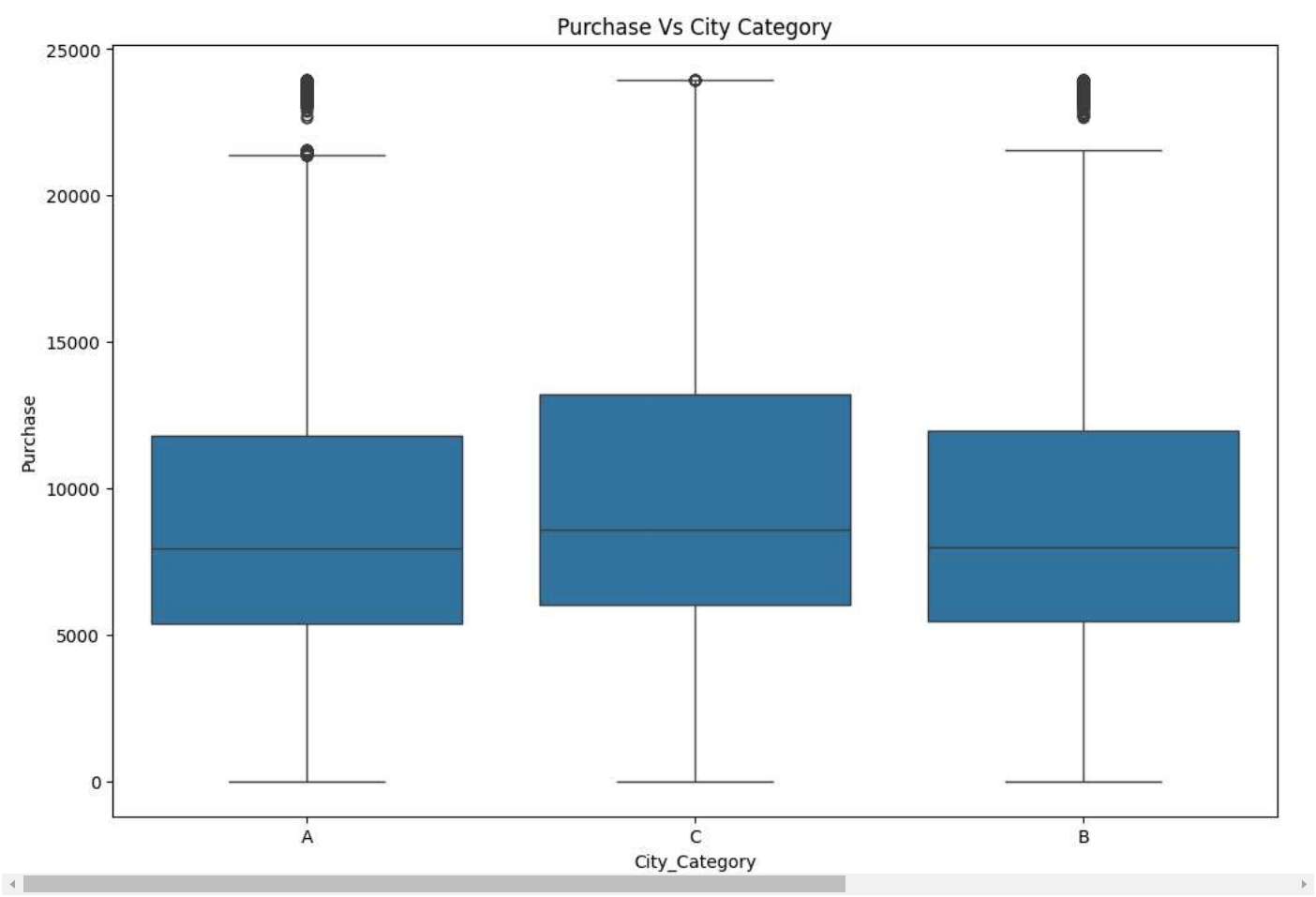
Purchase Habits based on occupation

```
plt.figure(figsize = (12,6))
sns.boxplot(data=df,x='Occupation',y='Purchase')
plt.title('Purchase Vs Occupation')
plt.show()
```



Purchase habit based on City Category

```
plt.figure(figsize = (12,8))
sns.boxplot(data=df,x='City_Category',y='Purchase')
plt.title('Purchase Vs City Category')
plt.show()
```

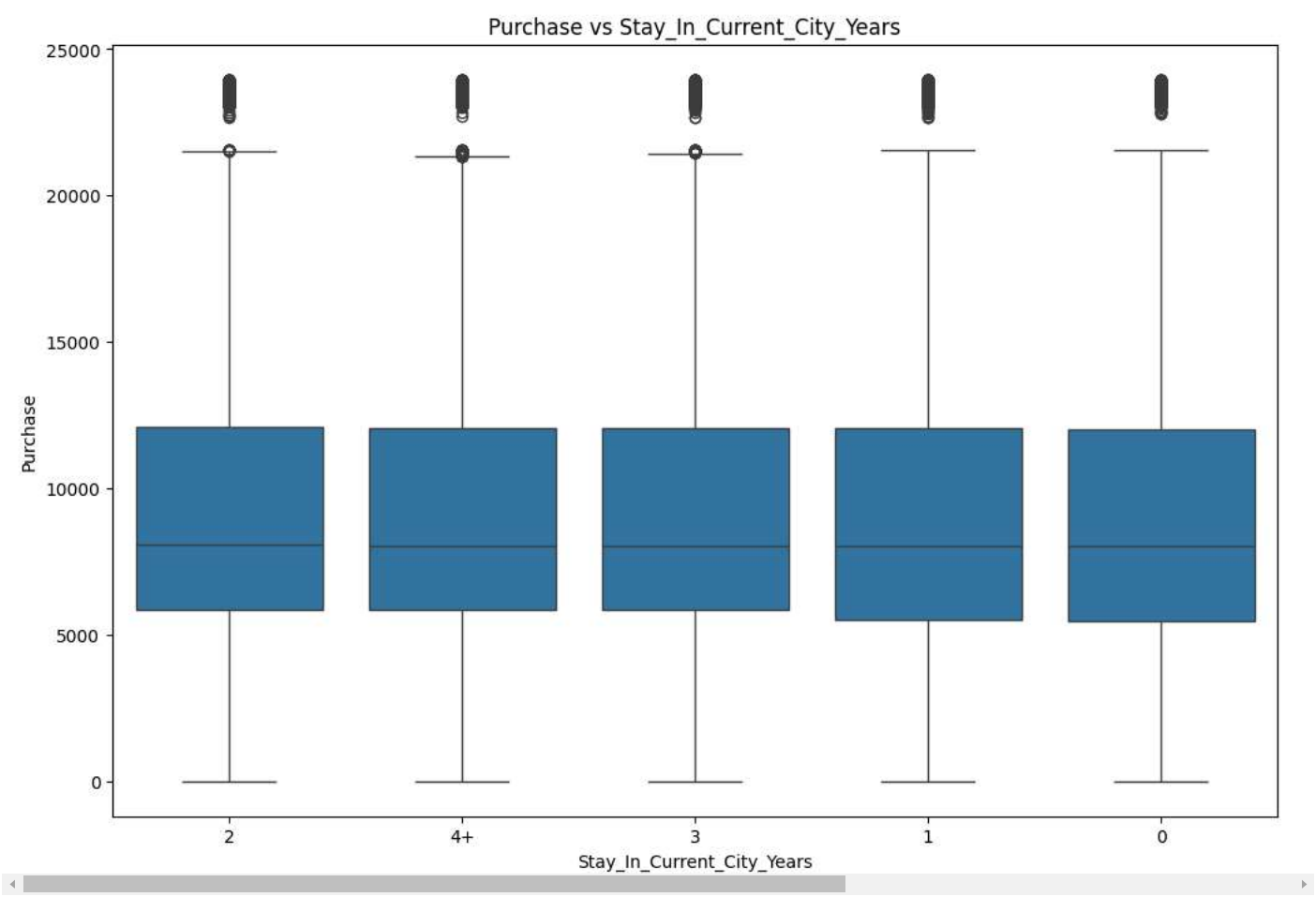


City "C" has higher median value for purchase hence higher spending habit followed by B and A

City "C" has no outliers compare to "B" and "A"

Purchase habit based on stay year in city

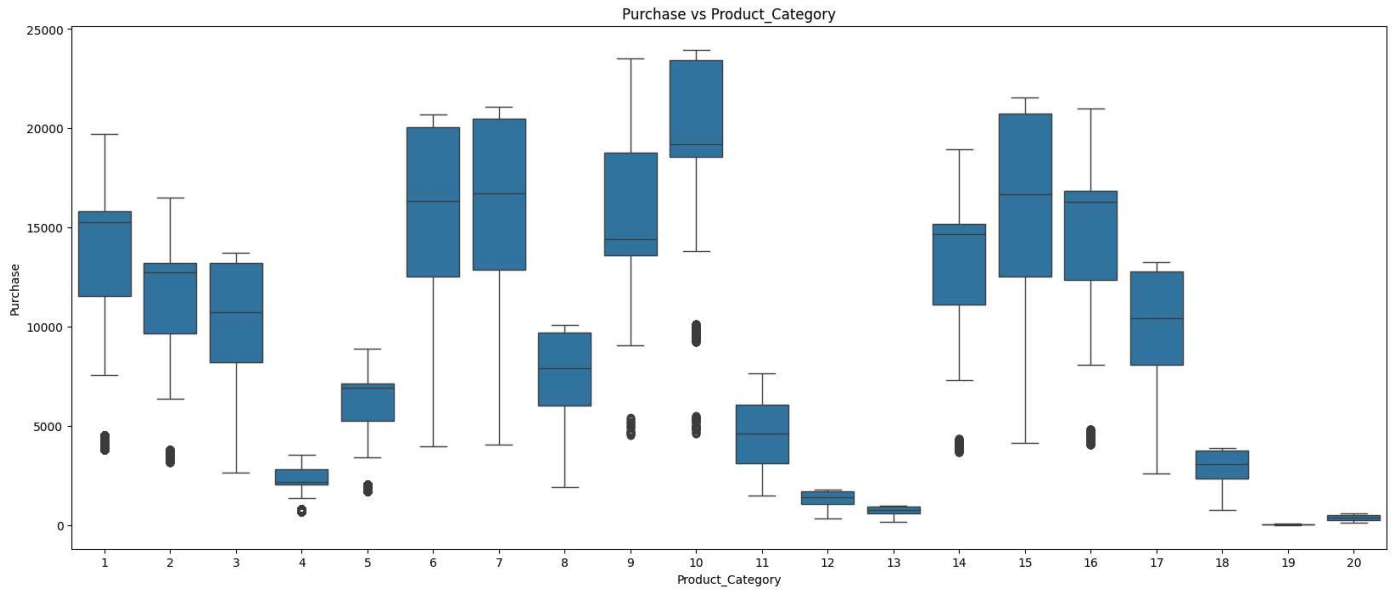
```
plt.figure(figsize = (12,8))
sns.boxplot(data = df, y ='Purchase', x = 'Stay_In_Current_City_Years')
plt.title('Purchase vs Stay_In_Current_City_Years')
plt.show()
```



Above figure shows that the median, maximum order value and minimum order values for all the cities are almost same, hence we can say that stays in year in cities has almost negligible impact on purchasing behaviour.

Purchase habit based on Product categories

```
plt.figure(figsize = (20,8))
sns.boxplot(data = df, y ='Purchase', x = 'Product_Category')
plt.title('Purchase vs Product_Category')
plt.show()
```



Above data and figure shows that product category 10 and 19 are the most and least preferred respectively while purchasing

Central Limit Theorem

```
from scipy.stats import norm
from scipy import stats
```

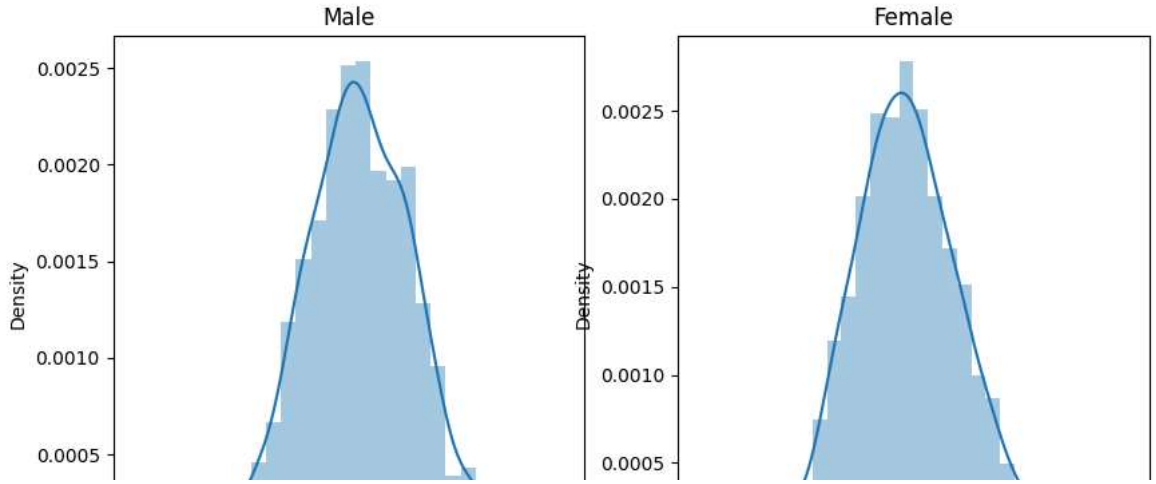
```
male_data=df[df['Gender']=='M']
male_data
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purcha
4	1000002	P00285442	M	55+	16	C	4+	0	8	7
5	1000003	P00193542	M	26-35	15	A	3	0	1	15
6	1000004	P00184942	M	46-50	7	B	2	1	1	19
7	1000004	P00346142	M	46-50	7	B	2	1	1	15
8	1000004	P0097242	M	46-50	7	B	2	1	1	15
...	...	...	...	...	...	...	...	...	...	
550057	1006023	P00370853	M	26-35	0	C	2	1	19	
550058	1006024	P00372445	M	26-35	12	A	0	1	20	

```
female_data=df[df['Gender']=='F']
female_data
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purcha
0	1000001	P00069042	F	0-17	10	A	2	0	3	8
1	1000001	P00248942	F	0-17	10	A	2	0	1	15
2	1000001	P00087842	F	0-17	10	A	2	0	12	1
3	1000001	P00085442	F	0-17	10	A	2	0	12	1
14	1000006	P00231342	F	51-55	9	A	1	0	5	5
...	...	...	...	...	...	...	...	...	...	
550061	1006029	P00372445	F	26-35	1	C	1	1	20	
550064	1006035	P00375436	F	26-35	1	C	3	0	20	

```
male_means=[]
female_means=[]
for i in range(1000):
  male_mean=male_data['Purchase'].sample(1000).mean()
  female_mean=female_data['Purchase'].sample(1000).mean()
  male_means.append(male_mean)
  female_means.append(female_mean)


plt.figure(figsize=(10,5))
plt.subplot(1,2,1)
sns.distplot(male_means)
plt.title('Male')
plt.subplot(1,2,2)
sns.distplot(female_means)
plt.title('Female')
plt.show()
```



1. The means sample seems to be normally distributed for both males and females. Also, we can see the mean of the sample means are closer to the population mean as per central limit theorem.

```
#Taking the values for z at 90%, 95% and 99% confidence interval as:


Z_value_90_perc=norm.ppf(0.90)
Z_value_90_perc
```

```
1.2815515655446004
```

calculating 90% confidence interval for 1000 samples:

```
print('Avg spend of male population:',np.mean(male_data['Purchase']))
print('Avg spend of female population:',np.mean(female_data['Purchase']))
print('Standard deviation of male population:',np.std(male_data['Purchase']))
print('Standard deviation of female population:',np.std(female_data['Purchase']))
male_Standard_of_error=np.std(male_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',male_Standard_of_error)
female_Standard_of_error=np.std(female_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',female_Standard_of_error)



Upper_Limit_male=Z_value_90_perc*male_Standard_of_error+np.mean(male_means)
Lower_Limit_male=np.mean(male_means)-Z_value_90_perc*male_Standard_of_error

Upper_limit_female=Z_value_90_perc*female_Standard_of_error+np.mean(female_means)
Lower_limit_female=np.mean(female_means)-Z_value_90_perc*female_Standard_of_error

print('90% confidence interval for male population:',Lower_Limit_male,'to',Upper_Limit_male)
print('90% confidence interval for female population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of male population: 9437.526040472265
Avg spend of female population: 8734.565765155476
Standard deviation of male population: 5092.180063635943
Standard deviation of female population: 4767.215738016988
Standard of error: 161.0288725679074
Standard of error: 150.75259829534235
90% confidence interval for male population: 9226.761085262717 to 9639.494692737286
90% confidence interval for female population: 8543.14528864469 to 8929.539745355312
```

Calculating 95% confidence interval for sample size 1000:

```
Z_value_95_perc=norm.ppf(0.95)
Z_value_95_perc
```

```
1.6448536269514722
```

```
print('Avg spend of male population:',np.mean(male_data['Purchase']))
print('Avg spend of female population:',np.mean(female_data['Purchase']))
print('Standard deviation of male population:',np.std(male_data['Purchase']))
print('Standard deviation of female population:',np.std(female_data['Purchase']))
male_Standard_of_error=np.std(male_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',male_Standard_of_error)
female_Standard_of_error=np.std(female_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',female_Standard_of_error)



Upper_Limit_male=Z_value_95_perc*male_Standard_of_error+np.mean(male_means)
Lower_Limit_male=np.mean(male_means)-Z_value_95_perc*male_Standard_of_error

Upper_limit_female=Z_value_95_perc*female_Standard_of_error+np.mean(female_means)
Lower_limit_female=np.mean(female_means)-Z_value_95_perc*female_Standard_of_error

print('95% confidence interval for male population:',Lower_Limit_male,'to',Upper_Limit_male)
print('95% confidence interval for female population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of male population: 9437.526040472265
Avg spend of female population: 8734.565765155476
Standard deviation of male population: 5092.180063635943
Standard deviation of female population: 4767.215738016988
Standard of error: 161.0288725679074
Standard of error: 150.75259829534235
95% confidence interval for male population: 9168.258963912773 to 9697.99681408723
95% confidence interval for female population: 8488.376558921549 to 8984.308475078453
```

Calculating 99% confidence interval for sample size 1000:

```
Z_value_99_perc=norm.ppf(0.99)
Z_value_99_perc
```

```
2.3263478740408408
```

```
print('Avg spend of male population:',np.mean(male_data['Purchase']))
print('Avg spend of female population:',np.mean(female_data['Purchase']))
print('Standard deviation of male population:',np.std(male_data['Purchase']))
print('Standard deviation of female population:',np.std(female_data['Purchase']))
male_Standard_of_error=np.std(male_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',male_Standard_of_error)
female_Standard_of_error=np.std(female_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',female_Standard_of_error)



Upper_Limit_male=Z_value_99_perc*male_Standard_of_error+np.mean(male_means)
Lower_Limit_male=np.mean(male_means)-Z_value_99_perc*male_Standard_of_error

Upper_limit_female=Z_value_99_perc*female_Standard_of_error+np.mean(female_means)
Lower_limit_female=np.mean(female_means)-Z_value_99_perc*female_Standard_of_error

print('99% confidence interval for male population:',Lower_Limit_male,'to',Upper_Limit_male)
print('99% confidence interval for female population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of male population: 9437.526040472265
Avg spend of female population: 8734.565765155476
Standard deviation of male population: 5092.180063635943
Standard deviation of female population: 4767.215738016988
Standard of error: 161.0288725679074
Standard of error: 150.75259829534235
99% confidence interval for male population: 9058.518713642456 to 9807.737064357547
99% confidence interval for female population: 8385.639530449498 to 9087.045503550504
```

Observation:

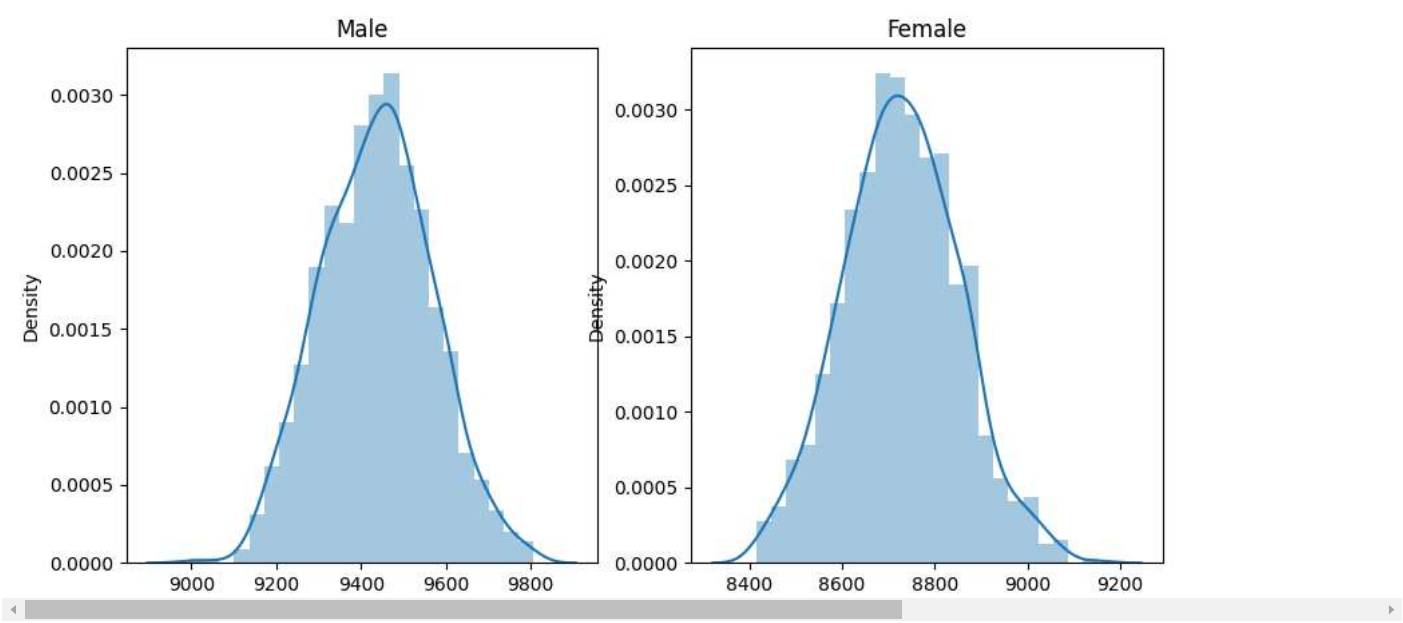
Now using the Confidence interval at 99%, we can say that:

Average amount spend by male customers lie in the range 9058.518713642456 - 9807.737064357547

Average amount spend by female customers lie in range 8385.639530449498 - 9087.045503550504

Calculating 90% confidence interval for sample size 1500:

```
male_means_1500=[]
female_means_1500=[]
for i in range(1000):
  male_mean_1500=male_data['Purchase'].sample(1500).mean()
  female_mean_1500=female_data['Purchase'].sample(1500).mean()
  male_means_1500.append(male_mean_1500)
  female_means_1500.append(female_mean_1500)


plt.figure(figsize=(10,5))
plt.subplot(1,2,1)
sns.distplot(male_means_1500)
plt.title('Male')
plt.subplot(1,2,2)
sns.distplot(female_means_1500)
plt.title('Female')
plt.show()
```




```
Z_value_90_perc=norm.ppf(0.90)
Z_value_90_perc
```

```
1.2815515655446004
```

```
print('Avg spend of male population:',np.mean(male_data['Purchase']))
print('Avg spend of female population:',np.mean(female_data['Purchase']))
print('Standard deviation of male population:',np.std(male_data['Purchase']))
print('Standard deviation of female population:',np.std(female_data['Purchase']))
male_Standard_of_error=np.std(male_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',male_Standard_of_error)
female_Standard_of_error=np.std(female_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',female_Standard_of_error)
```

```
Upper_Limit_male=Z_value_90_perc*male_Standard_of_error+np.mean(male_means)
Lower_Limit_male=np.mean(male_means)-Z_value_90_perc*male_Standard_of_error

Upper_limit_female=Z_value_90_perc*female_Standard_of_error+np.mean(female_means)
Lower_limit_female=np.mean(female_means)-Z_value_90_perc*female_Standard_of_error

print('90% confidence interval for male population:',Lower_Limit_male,'to',Upper_Limit_male)
print('90% confidence interval for female population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of male population: 9437.526040472265
Avg spend of female population: 8734.565765155476
Standard deviation of male population: 5092.180063635943
Standard deviation of female population: 4767.215738016988
Standard of error: 131.47952388234287
Standard of error: 123.08898107411797
90% confidence interval for male population: 9264.630099331525 to 9601.625678668477
90% confidence interval for female population: 8578.597640603175 to 8894.087393396827
```

```
Start coding or generate with AI.
```

Calculating 95% confidence interval for sample size 1500:

```
Z_value_95_perc=norm.ppf(0.95)
Z_value_95_perc
```

```
1.6448536269514722
```

```
print('Avg spend of male population:',np.mean(male_data['Purchase']))
print('Avg spend of female population:',np.mean(female_data['Purchase']))
print('Standard deviation of male population:',np.std(male_data['Purchase']))
print('Standard deviation of female population:',np.std(female_data['Purchase']))
male_Standard_of_error=np.std(male_data['Purchase'])/np.sqrt(1500)
```

```
print('Standard of error:',male_Standard_of_error)
female_Standard_of_error=np.std(female_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',female_Standard_of_error)



Upper_Limit_male=Z_value_95_perc*male_Standard_of_error+np.mean(male_means)
Lower_Limit_male=np.mean(male_means)-Z_value_95_perc*male_Standard_of_error

Upper_limit_female=Z_value_95_perc*female_Standard_of_error+np.mean(female_means)
Lower_limit_female=np.mean(female_means)-Z_value_95_perc*female_Standard_of_error

print('95% confidence interval for male population:',Lower_Limit_male,'to',Upper_Limit_male)
print('95% confidence interval for female population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of male population: 9437.526040472265
Avg spend of female population: 8734.565765155476
Standard deviation of male population: 5092.180063635943
Standard deviation of female population: 4767.215738016988
Standard of error: 131.47952388234287
Standard of error: 123.08898107411797
95% confidence interval for male population: 9216.863317272277 to 9649.392460727726
95% confidence interval for female population: 8533.879160042477 to 8938.805873957524
```

Calculating 99% confidence interval for sample size 1500:

```
Z_value_99_perc=norm.ppf(0.99)
Z_value_99_perc
```

```
2.3263478740408408
```

```
print('Avg spend of male population:',np.mean(male_data['Purchase']))
print('Avg spend of female population:',np.mean(female_data['Purchase']))
print('Standard deviation of male population:',np.std(male_data['Purchase']))
print('Standard deviation of female population:',np.std(female_data['Purchase']))
male_Standard_of_error=np.std(male_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',male_Standard_of_error)
female_Standard_of_error=np.std(female_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',female_Standard_of_error)



Upper_Limit_male=Z_value_99_perc*male_Standard_of_error+np.mean(male_means)
Lower_Limit_male=np.mean(male_means)-Z_value_99_perc*male_Standard_of_error

Upper_limit_female=Z_value_99_perc*female_Standard_of_error+np.mean(female_means)
Lower_limit_female=np.mean(female_means)-Z_value_99_perc*female_Standard_of_error

print('99% confidence interval for male population:',Lower_Limit_male,'to',Upper_Limit_male)
print('99% confidence interval for female population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of male population: 9437.526040472265
Avg spend of female population: 8734.565765155476
Standard deviation of male population: 5092.180063635943
Standard deviation of female population: 4767.215738016988
Standard of error: 131.47952388234287
Standard of error: 123.08898107411797
99% confidence interval for male population: 9127.260778136411 to 9738.994999863591
99% confidence interval for female population: 8449.994727560374 to 9022.690306439628
```

Observations:

By increasing the sample size we can see confidence interval is more closer to the population mean.

```
# Confidence Interval for marital_status


unmarried_data=df[df['Marital_Status']==0]
unmarried_data.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

```
married_data=df[df['Marital_Status']==1]
married_data
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purcha
6	1000004	P00184942	M	46-50	7	B	2	1	1	19
7	1000004	P00346142	M	46-50	7	B	2	1	1	15
8	1000004	P0097242	M	46-50	7	B	2	1	1	15
9	1000005	P00274942	M	26-35	20	A	1	1	8	7
10	1000005	P00251242	M	26-35	20	A	1	1	5	5
...	...	...	...	...	...	...	...	...	...	
550060	1006026	P00371644	M	36-45	6	C	1	1	20	
550061	1006029	P00372445	F	26-35	1	C	1	1	20	

```
unmarried_sample_data=[]
married_sample_data=[]
for i in range(1000):
  unmarried_mean=unmarried_data['Purchase'].sample(1000).mean()
  married_mean=married_data['Purchase'].sample(1000).mean()
  unmarried_sample_data.append(unmarried_mean)
  married_sample_data.append(married_mean)
```

Calculating 90% confidence interval for sample size 1000:

```
Z_value_90_perc=norm.ppf(0.90)
Z_value_90_perc
```

```
1.2815515655446004
```

```
print('Avg spend of unmarried population:',np.mean(unmarried_data['Purchase']))
print('Avg spend of married population:',np.mean(married_data['Purchase']))
print('Standard deviation of unmarried population:',np.std(unmarried_data['Purchase']))
print('Standard deviation of married population:',np.std(married_data['Purchase']))
unmarried_Standard_of_error=np.std(unmarried_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',unmarried_Standard_of_error)
married_Standard_of_error=np.std(married_data['Purchase'])/np.sqrt(1000)
```

```
print('Standard of error:',married_Standard_of_error)



Upper_Limit_male=Z_value_90_perc*unmarried_Standard_of_error+np.mean(unmarried_data['Purchase'])
Lower_Limit_male=np.mean(unmarried_data['Purchase'])-Z_value_90_perc*unmarried_Standard_of_error

Upper_limit_female=Z_value_90_perc*married_Standard_of_error+np.mean(married_data['Purchase'])
Lower_limit_female=np.mean(married_data['Purchase'])-Z_value_90_perc*married_Standard_of_error

print('90% confidence interval for unmarried population:',Lower_Limit_male,'to',Upper_Limit_male)
print('90% confidence interval for married population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of unmarried population: 9265.907618921507
Avg spend of married population: 9261.174574082374
Standard deviation of unmarried population: 5027.340117880186
Standard deviation of married population: 5016.886245793184
Standard of error: 158.9784534484078
Standard of error: 158.64787298677794
90% confidence interval for unmarried population: 9062.16853301684 to 9469.646704826173
90% confidence interval for married population: 9057.859144085847 to 9464.4900040789
```

Calculating 95% confidence interval for sample size 1000:

```
Z_value_95_perc=norm.ppf(0.95)
Z_value_95_perc
```

```
1.6448536269514722
```

```
print('Avg spend of unmarried population:',np.mean(unmarried_data['Purchase']))
print('Avg spend of married population:',np.mean(married_data['Purchase']))
print('Standard deviation of unmarried population:',np.std(unmarried_data['Purchase']))
print('Standard deviation of married population:',np.std(married_data['Purchase']))
unmarried_Standard_of_error=np.std(unmarried_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',unmarried_Standard_of_error)
married_Standard_of_error=np.std(married_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',married_Standard_of_error)



Upper_Limit_male=Z_value_95_perc*unmarried_Standard_of_error+np.mean(unmarried_data['Purchase'])
Lower_Limit_male=np.mean(unmarried_data['Purchase'])-Z_value_95_perc*unmarried_Standard_of_error

Upper_limit_female=Z_value_95_perc*married_Standard_of_error+np.mean(married_data['Purchase'])
Lower_limit_female=np.mean(married_data['Purchase'])-Z_value_95_perc*married_Standard_of_error

print('95% confidence interval for unmarried population:',Lower_Limit_male,'to',Upper_Limit_male)
print('95% confidence interval for married population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of unmarried population: 9265.907618921507
Avg spend of married population: 9261.174574082374
Standard deviation of unmarried population: 5027.340117880186
Standard deviation of married population: 5016.886245793184
Standard of error: 158.9784534484078
Standard of error: 158.64787298677794
95% confidence interval for unmarried population: 9004.411333159756 to 9527.403904683257
95% confidence interval for married population: 9000.222044791935 to 9522.127103372812
```

Calculating 99% confidence interval for sample size 1000:

```
Z_value_99_perc=norm.ppf(0.99)
Z_value_99_perc
```

```
2.3263478740408408
```

```
print('Avg spend of unmarried population:',np.mean(unmarried_data['Purchase']))
print('Avg spend of married population:',np.mean(married_data['Purchase']))
print('Standard deviation of unmarried population:',np.std(unmarried_data['Purchase']))
print('Standard deviation of married population:',np.std(married_data['Purchase']))
unmarried_Standard_of_error=np.std(unmarried_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',unmarried_Standard_of_error)
married_Standard_of_error=np.std(married_data['Purchase'])/np.sqrt(1000)
print('Standard of error:',married_Standard_of_error)
```

```
Upper_Limit_male=Z_value_99_perc*unmarried_Standard_of_error+np.mean(unmarried_data['Purchase'])
Lower_Limit_male=np.mean(unmarried_data['Purchase'])-Z_value_99_perc*unmarried_Standard_of_error

Upper_limit_female=Z_value_99_perc*married_Standard_of_error+np.mean(married_data['Purchase'])
Lower_limit_female=np.mean(married_data['Purchase'])-Z_value_99_perc*married_Standard_of_error

print('99% confidence interval for unmarried population:',Lower_Limit_male,'to',Upper_Limit_male)
print('99% confidence interval for married population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of unmarried population: 9265.907618921507
Avg spend of married population: 9261.174574082374
Standard deviation of unmarried population: 5027.340117880186
Standard deviation of married population: 5016.886245793184
Standard of error: 158.9784534484078
Standard of error: 158.64787298677794
99% confidence interval for unmarried population: 8896.068431723503 to 9635.74680611951
99% confidence interval for married population: 8892.10443203848 to 9630.244716126266
```

Calculating 90% confidence interval for sample size 1500

```
unmarried_sample_data_1500=[]
married_sample_data_1500=[]
for i in range(1000):
  unmarried_mean_1500=unmarried_data['Purchase'].sample(1500).mean()
  married_mean_1500=married_data['Purchase'].sample(1500).mean()
  unmarried_sample_data_1500.append(unmarried_mean_1500)
  married_sample_data_1500.append(married_mean_1500)


Z_value_90_perc=norm.ppf(0.90)
Z_value_90_perc
```

```
1.2815515655446004
```

```
print('Avg spend of unmarried population:',np.mean(unmarried_data['Purchase']))
print('Avg spend of married population:',np.mean(married_data['Purchase']))
print('Standard deviation of unmarried population:',np.std(unmarried_data['Purchase']))
print('Standard deviation of married population:',np.std(married_data['Purchase']))
unmarried_Standard_of_error_1500=np.std(unmarried_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',unmarried_Standard_of_error_1500)
married_Standard_of_error_1500=np.std(married_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',married_Standard_of_error_1500)



Upper_Limit_male=Z_value_90_perc*unmarried_Standard_of_error_1500+np.mean(unmarried_data['Purchase'])
Lower_Limit_male=np.mean(unmarried_data['Purchase'])-Z_value_90_perc*unmarried_Standard_of_error_1500

Upper_limit_female=Z_value_90_perc*married_Standard_of_error_1500+np.mean(married_data['Purchase'])
Lower_limit_female=np.mean(married_data['Purchase'])-Z_value_90_perc*married_Standard_of_error_1500

print('90% confidence interval for unmarried population:',Lower_Limit_male,'to',Upper_Limit_male)
print('90% confidence interval for married population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of unmarried population: 9265.907618921507
Avg spend of married population: 9261.174574082374
Standard deviation of unmarried population: 5027.340117880186
Standard deviation of married population: 5016.886245793184
Standard of error: 129.80536368180262
Standard of error: 129.53544586516034
90% confidence interval for unmarried population: 9099.555351879006 to 9432.259885964007
90% confidence interval for married population: 9095.16822064036 to 9427.180927524387
```

Calculating 95% confidence interval for sample size 1500:

```
Z_value_95_perc=norm.ppf(0.95)
Z_value_95_perc
```

```
1.6448536269514722
```

```
print('Avg spend of unmarried population:',np.mean(unmarried_data['Purchase']))
print('Avg spend of married population:',np.mean(married_data['Purchase']))
```

```
print('Standard deviation of unmarried population:',np.std(unmarried_data['Purchase']))
print('Standard deviation of married population:',np.std(married_data['Purchase']))
unmarried_Standard_of_error_1500=np.std(unmarried_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',unmarried_Standard_of_error_1500)
married_Standard_of_error_1500=np.std(married_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',married_Standard_of_error_1500)



Upper_Limit_male=Z_value_95_perc*unmarried_Standard_of_error_1500+np.mean(unmarried_data['Purchase'])
Lower_Limit_male=np.mean(unmarried_data['Purchase'])-Z_value_95_perc*unmarried_Standard_of_error_1500

Upper_limit_female=Z_value_95_perc*married_Standard_of_error_1500+np.mean(married_data['Purchase'])
Lower_limit_female=np.mean(married_data['Purchase'])-Z_value_95_perc*married_Standard_of_error_1500

print('95% confidence interval for unmarried population:',Lower_Limit_male,'to',Upper_Limit_male)
print('95% confidence interval for married population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of unmarried population: 9265.907618921507
Avg spend of married population: 9261.174574082374
Standard deviation of unmarried population: 5027.340117880186
Standard deviation of married population: 5016.886245793184
Standard of error: 129.80536368180262
Standard of error: 129.53544586516034
95% confidence interval for unmarried population: 9052.396795671739 to 9479.418442171274
95% confidence interval for married population: 9048.107726132288 to 9474.241422032459
```

Calculating 99% confidence interval for sample size 1500:

```
Z_value_99_perc=norm.ppf(0.99)
Z_value_99_perc
```

```
2.3263478740408408
```

```
print('Avg spend of unmarried population:',np.mean(unmarried_data['Purchase']))
print('Avg spend of married population:',np.mean(married_data['Purchase']))
print('Standard deviation of unmarried population:',np.std(unmarried_data['Purchase']))
print('Standard deviation of married population:',np.std(married_data['Purchase']))
unmarried_Standard_of_error_1500=np.std(unmarried_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',unmarried_Standard_of_error_1500)
married_Standard_of_error_1500=np.std(married_data['Purchase'])/np.sqrt(1500)
print('Standard of error:',married_Standard_of_error_1500)



Upper_Limit_male=Z_value_99_perc*unmarried_Standard_of_error_1500+np.mean(unmarried_data['Purchase'])
Lower_Limit_male=np.mean(unmarried_data['Purchase'])-Z_value_99_perc*unmarried_Standard_of_error_1500

Upper_limit_female=Z_value_99_perc*married_Standard_of_error_1500+np.mean(married_data['Purchase'])
Lower_limit_female=np.mean(married_data['Purchase'])-Z_value_99_perc*married_Standard_of_error_1500

print('99% confidence interval for unmarried population:',Lower_Limit_male,'to',Upper_Limit_male)
print('99% confidence interval for married population:',Lower_limit_female,'to',Upper_limit_female)
```

```
Avg spend of unmarried population: 9265.907618921507
Avg spend of married population: 9261.174574082374
Standard deviation of unmarried population: 5027.340117880186
Standard deviation of married population: 5016.886245793184
Standard of error: 129.80536368180262
Standard of error: 129.53544586516034
99% confidence interval for unmarried population: 8963.935187081246 to 9567.880050761767
99% confidence interval for married population: 8959.830064981026 to 9562.51908318372
```

Observation:

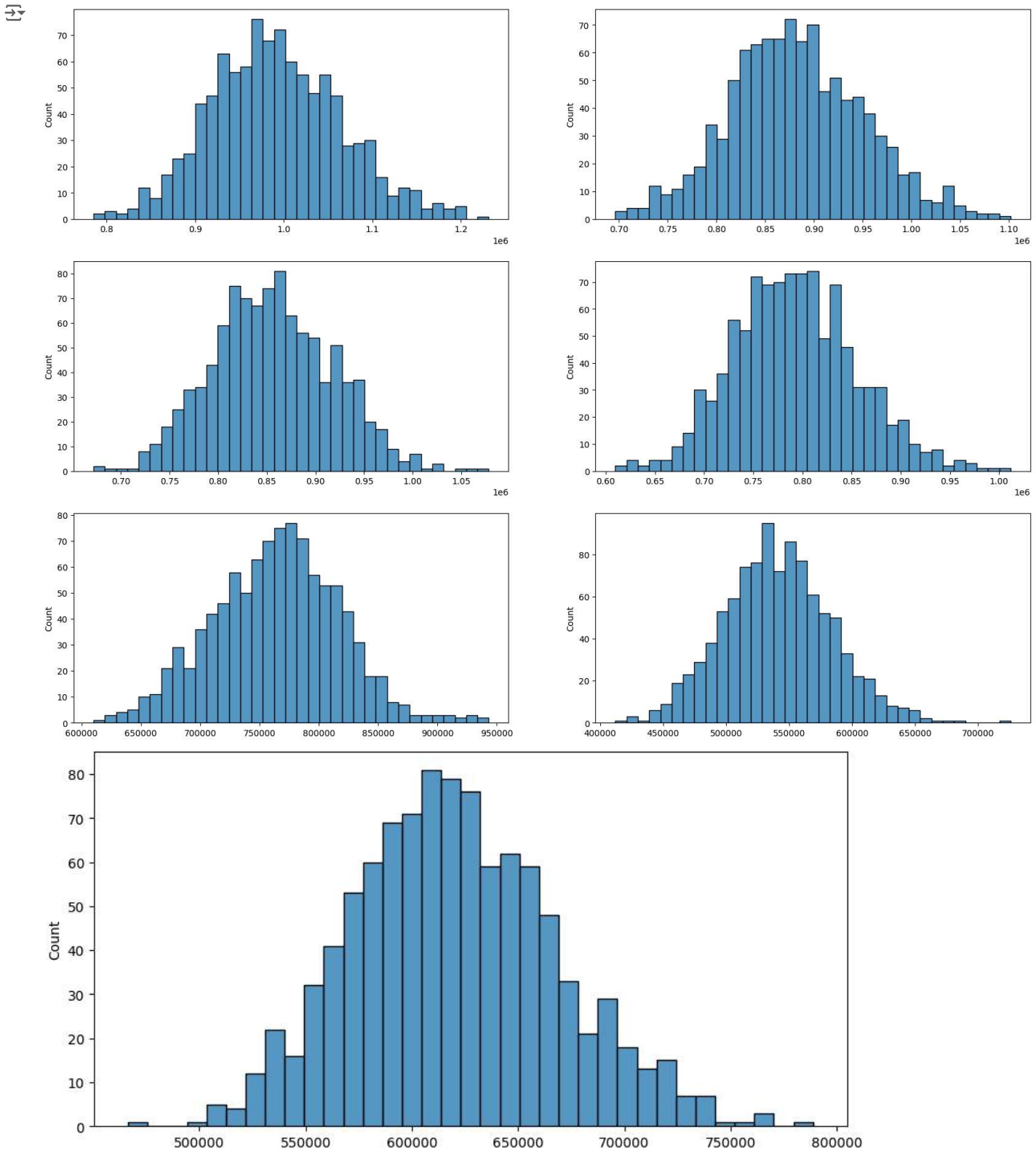
For married and singles, it can be seen with larger sample size the sample mean gets closer to tthe population mean. And at greater confidence interval, the range increases.

```
avgamt_age = df.groupby(['User_ID', 'Age'])[['Purchase']].sum()
avgamt_age = avgamt_age.reset_index()

avgamt_age['Age'].value_counts()
```

Age	count
26-35	2053
36-45	1167
18-25	1069
46-50	531
51-55	481
55+	372
0-17	218

dtype: int64

```
sample_size = 200
num_repitions = 1000

all_sample_means = {}

age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
    all_sample_means[i] = []

for i in age_intervals:
    for j in range(num_repitions):

        mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
        all_sample_means[i].append(mean)



fig, axis = plt.subplots(nrows=3, ncols=2, figsize=(20, 15))

sns.histplot(all_sample_means['26-35'],bins=35,ax=axis[0,0])
sns.histplot(all_sample_means['36-45'],bins=35,ax=axis[0,1])
sns.histplot(all_sample_means['18-25'],bins=35,ax=axis[1,0])
sns.histplot(all_sample_means['46-50'],bins=35,ax=axis[1,1])
sns.histplot(all_sample_means['51-55'],bins=35,ax=axis[2,0])
sns.histplot(all_sample_means['55+'],bins=35,ax=axis[2,1])

plt.show()

plt.figure(figsize=(10, 5))
sns.histplot(all_sample_means['0-17'],bins=35)
plt.show()
```



Observations:

The means sample seems to be normally distributed for all age groups. Also, we can see the mean of the sample means are closer to the population mean as per central limit theorem.

Calculating 90% confidence interval for avg expenses for different age groups for sample size 200:

```
Z_value_90_perc=norm.ppf(0.90)
Z_value_90_perc
```

```
1.2815515655446004
```

```
sample_size = 200
num_repitions = 1000

all_population_means={}
all_sample_means = {}

age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
    all_sample_means[i] = []
    all_population_means[i]=[]
    population_mean=avgamt_age[avgamt_age['Age']==i]['Purchase'].mean()
    all_population_means[i].append(population_mean)

print("All age group population mean: \n", all_population_means)
print("\n")

for i in age_intervals:
    for j in range(num_repitions):

        mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
        all_sample_means[i].append(mean)


for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:

    new_df = avgamt_age[avgamt_age['Age']==val]

    std_error = Z_value_90_perc*new_df['Purchase'].std()/np.sqrt(len(new_df))
    sample_mean = new_df['Purchase'].mean()
    lower_lim = sample_mean - std_error
    upper_lim = sample_mean + std_error

print("For age {} confidence interval of means: ({:.2f}, {:.2f})".format(val, lower_lim, upper_lim))
```

```
All age group population mean:
 {'26-35': [989659.3170969313], '36-45': [879665.7103684661], '18-25': [854863.119738073], '46-50': [792548.7815442561], '51-55': [76320

For age 0-17 confidence interval of means: (559232.93, 678502.69)
```

Calculating 95% confidence interval for avg expenses for different age groups for sample size 200:

```
Z_value_95_perc=norm.ppf(0.95)
Z_value_95_perc
```

```
1.6448536269514722
```

```
sample_size = 200
num_repitions = 1000

all_means = {}

age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
    all_means[i] = []

for i in age_intervals:
    for j in range(num_repitions):
        mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
        all_means[i].append(mean)
for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:

    new_df = avgamt_age[avgamt_age['Age']==val]
```

```
    std_error = Z_value_95_perc*new_df['Purchase'].std()/np.sqrt(len(new_df))
    sample_mean = new_df['Purchase'].mean()
    lower_lim = sample_mean - std_error
    upper_lim = sample_mean + std_error

print("For age {} confidence interval of means: ({:.2f}, {:.2f})".format(val, lower_lim, upper_lim))
```

```
For age 0-17 confidence interval of means: (542327.27, 695408.35)
```

Calculating 99% confidence interval for avg expenses for different age groups for sample size 200:

```
Z_value_99_perc=norm.ppf(0.99)
Z_value_99_perc
```

```
2.3263478740408408
```

```
sample_size = 200
num_repitions = 1000

all_means = {}

age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for i in age_intervals:
    all_means[i] = []

for i in age_intervals:
    for j in range(num_repitions):
        mean = avgamt_age[avgamt_age['Age']==i].sample(sample_size, replace=True)['Purchase'].mean()
        all_means[i].append(mean)
for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:

    new_df = avgamt_age[avgamt_age['Age']==val]

    std_error = Z_value_99_perc*new_df['Purchase'].std()/np.sqrt(len(new_df))
    sample_mean = new_df['Purchase'].mean()
    lower_lim = sample_mean - std_error
    upper_lim = sample_mean + std_error

    print("For age {} confidence interval of means: ({:.2f}, {:.2f})".format(val, lower_lim, upper_lim))
```

```
For age 26-35 confidence interval of means: (936693.49, 1042625.15)
For age 36-45 confidence interval of means: (812821.30, 946510.12)
For age 18-25 confidence interval of means: (791683.38, 918042.86)
For age 46-50 confidence interval of means: (698731.51, 886366.06)
For age 51-55 confidence interval of means: (679157.45, 847244.39)
For age 55+ confidence interval of means: (465219.71, 614174.78)
For age 0-17 confidence interval of means: (510615.06, 727120.56)
```

Observation

1. We can see the sample means are closer to the population mean for the differnt age groups. And, with greater confidence interval we have the upper limit and lower limit range increases. As we have seen for gender and marital status, by increasing the sample size we can have the mean of the sample means closer to the population.

Recommendations

1. Men spent more money than women, company can focus on retaining the male customers and getting more male customers.
2. Product_Category - 1, 5, 8 have highest purchasing frequency. it means these are the products in these categories are in more demand. Company can focus on selling more of these products.
3. Unmarried customers spend more money than married customers, So company should focus on acquisition of Unmarried customers.
4. Customers in the age 26-35 spend more money than the others, So company should focus on acquisition of customers who are in the age 26-35.
5. We have more customers aged 26-35 in the city category B and A, company can focus more on these customers for these cities to increase the business.
6. Male customers living in City_Category C spend more money than other male customers living in B or C, Selling more products in the City_Category C will help the company increase the revenue.
7. Some of the Product category like 19,20,13 have very less purchase. Company can think of dropping it.

8. The top 10 users who have purchased more company should give more offers and discounts so that they can be retained and can be helpful for companies business.
9. The occupation which are contributing more company can think of offering credit cards or other benefits to those customers by liasing with some financial partners to increase the sales.
10. The top products should be given focus in order to maintain the quality in order to further increase the sales of those products.
11. People who are staying in city for an year have contributed to 35% of the total purchase amount. Company can focus on such customer base who are neither too old nor too new residents in the city.
12. We have highest frequency of purchase order between 5k and 10k, company can focus more on these mid range products to increase the sales.

```
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