Business Case: Walmart - Confidence Interval and CLT

About Walmart

Walmart is an American multinational retail corporation that operates a chain of supercenters, discount departmental stores, and grocery stores from the United States. Walmart has more than 100 million customers worldwide.

Business Problem

The Management team at Walmart Inc. wants to analyze the customer purchase behavior (specifically, purchase amount) against the customer's gender and the various other factors to help the business make better decisions. They want to understand if the spending habits differ between male and female customers: Do women spend more on Black Friday than men? (Assume 50 million customers are male and 50 million are female).

In [222…

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

In [223…

```
!wget "https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/w
```

```
--2024-06-01 11:03:58--  https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/
001/293/original/walmart_data.csv?1641285094
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)... 3.162.130.11
1, 3.162.130.14, 3.162.130.97, ...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)|3.162.130.11
1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 23027994 (22M) [text/plain]
Saving to: 'walmart_data.csv?1641285094.2'

walmart_data.csv?16 100%[===================>]  21.96M   108MB/s    in 0.2s

2024-06-01 11:03:58 (108 MB/s) - 'walmart_data.csv?1641285094.2' saved [23027994/2302799
4]
```

In [224…

```
df = pd.read_csv('walmart_data.csv?1641285094') # loading the dataset
df.head(10)
```

Out[224]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status
0	1000001	P00069042	F	0-17	10	A	2	0
1	1000001	P00248942	F	0-17	10	A	2	0
2	1000001	P00087842	F	0-17	10	A	2	0
3	1000001	P00085442	F	0-17	10	A	2	0

4	1000002	P00285442	M	55+	16	C	4+	0
5	1000003	P00193542	M	26-35	15	A	3	0
6	1000004	P00184942	M	46-50	7	B	2	1
7	1000004	P00346142	M	46-50	7	B	2	1
8	1000004	P0097242	M	46-50	7	B	2	1
9	1000005	P00274942	M	26-35	20	A	1	1

In [ ]:
```
sns.heatmap(df.corr(numeric_only=True), cmap= "Blues", annot=True)
plt.title('Correlation between all factors')
plt.show()
```

In [ ]:
```
df.shape
```

Out[ ]:
```
(550068, 10)
```

insights:

- total no of rows = 550068
- total no of columns = 10

---

data type of all columns

In [ ]:
```
df.dtypes
```

Out[ ]:
```
User_ID                        int64
Product_ID                    object
Gender                        object
Age                           object
Occupation                     int64
City_Category                 object
Stay_In_Current_City_Years    object
Marital_Status                 int64
Product_Category               int64
Purchase                       int64
dtype: object
```

insights:

- Apart from Purchase Column, all the other data types are of categorical type. -I will change the datatypes of all such columns to object

---

In [ ]:
```
# Converting all columns (except Purchase) to categorical type in the DataFrame
col = ['User_ID','Occupation','Marital_Status','Product_Category']
for i in col:
  df[i] = df[i].astype('object')
```

In [ ]:
```
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  object
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  object
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  object
 8   Product_Category            550068 non-null  object
 9   Purchase                    550068 non-null  int64 
dtypes: int64(1), object(9)
memory usage: 42.0+ MB
```

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

Out[ ]:
```
5891
```

In [ ]:
```
df['Product_ID'].nunique()
```

Out[ ]:
```
3631
```

Analysing basic metrics

In [ ]:
```
df.describe()
```

Out[ ]:

	Purchase
count	550068.000000
mean	9263.968713
std	5023.065394
min	12.000000
25%	5823.000000
50%	8047.000000
75%	12054.000000
max	23961.000000

In [ ]:
```
#Checking the characteristics of the data:
df.describe(include='all')
```

Out[ ]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marita
count	550068.0	550068	550068	550068	550068.0	550068	550068	5
unique	5891.0	3631	2	7	21.0	3	5	
top	1001680.0	P00265242	M	26-35	4.0	B	1	
freq	1026.0	1880	414259	219587	72308.0	231173	193821	3
mean	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
std	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
min	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

25%	NaN	NaN	NaN	NaN	NaN	NaN	NaN
50%	NaN	NaN	NaN	NaN	NaN	NaN	NaN
75%	NaN	NaN	NaN	NaN	NaN	NaN	NaN
max	NaN	NaN	NaN	NaN	NaN	NaN	NaN

insights:

- There are **5891** unique users, and **userid 1001680.0** being with the highest count.
- There are **3631** unique product IDs in the dataset.**P00265242** is the most sold Product ID.
- There are **7 unique age groups** and most of the purchase belongs to age **26-35 group**.
- The customers belongs to **21** distinct occupation for the purchases being made with Occupation **4 being the highest**.
- There are **3 unique city_categories** with **category B being the highest**.
- The range of purchasing behavior is quite extensive, as evidenced by a **minimum purchase of 12** and a **maximum of 23,961**. -The mean purchase amount stands at **9,264**, and **75%** of purchases are at or below **12,054**, indicating that the majority of purchases fall below the 12,000 threshold.
- 5 unique values for Stay_in_current_citi_years with 1 being the highest. Marital status unmarried contribute more in terms of the count for the purchase.

In [ ]:
```
df.isnull().sum()
```

Out[ ]:
```
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
```

insights:

- Dataset doesn't contain any missing values.

In [ ]:
```
col_unique = ['User_ID','Product_ID','Gender','Age','Occupation',
              'City_Category','Stay_In_Current_City_Years','Marital_Status',
              'Product_Category',       'Purchase']
df[col_unique].nunique()
```

Out[ ]:
```
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 [ ]:
```
df.duplicated().value_counts()
```

```
False    550068
```

Out[ ]: Name: count, dtype: int64

Non-Graphical Analysis:

In [ ]:
```
cate_col = ['Gender', 'Age', 'City_Category','Marital_Status', 'Stay_In_Current_City_Yea
result = df[cate_col].melt().groupby(['variable', 'value'])[['value']].count()/len(df)
result = result.rename(columns={'value': 'value_percent'})
result['value_percent'] = result['value_percent'] * 100
result
```

Out[ ]:

variable	value	value_percent
Age	0-17	2.745479
	18-25	18.117760
	26-35	39.919974
	36-45	19.999891
	46-50	8.308246
	51-55	6.999316
	55+	3.909335
City_Category	A	26.854862
	B	42.026259
	C	31.118880
Gender	F	24.689493
	M	75.310507
Marital_Status	0	59.034701
	1	40.965299
Stay_In_Current_City_Years	0	13.525237
	1	35.235825
	2	18.513711
	3	17.322404
	4+	15.402823

insights:

- 40% of the purchase done by aged 26-35.
- maximum percent of purchase done by city_category B.
- 75% of the purchase count are done by Male and 25% by Female.
- 60% single, 40% married contributes to purchase count.
- 35% people staying from a year and 15% living for more than 4 years in current city.

In [ ]:
```
df.duplicated().value_counts()
```

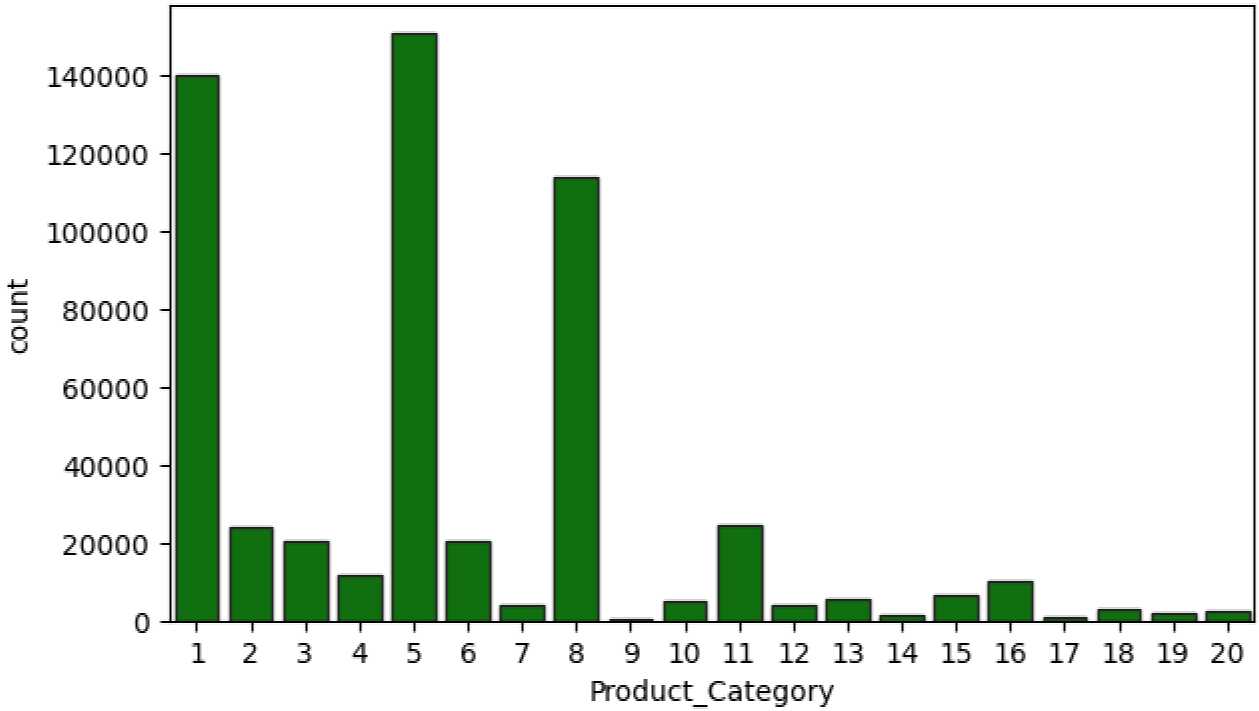
Out[ ]:
```
False    550068
Name: count, dtype: int64
```

insights:

- In our dataset doesn't contain duplicates value.

Univariate Analysis

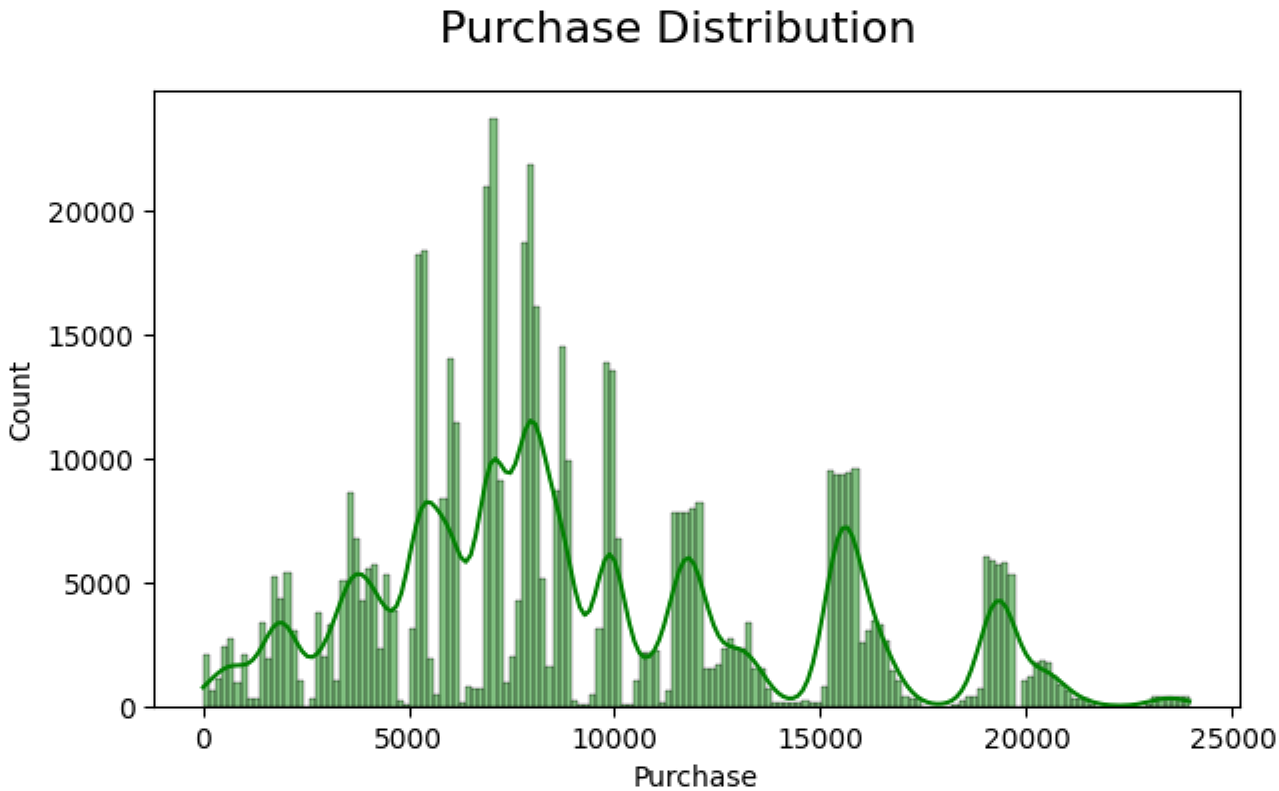
In [ ]:
```
plt.figure(figsize=(7,4))
sns.countplot(data=df, x='Product_Category',color='green',edgecolor="0.15")
plt.show()
```



insights

- Product_categories 1, 5, and 8 have higher counts, while category 9 and 17 records the lowest number of purchases

In [ ]:
```
plt.figure(figsize=(7,4))
sns.histplot(data=df, x='Purchase',color='green',kde=True)
plt.suptitle('Purchase Distribution',fontsize=16)
plt.show()
```

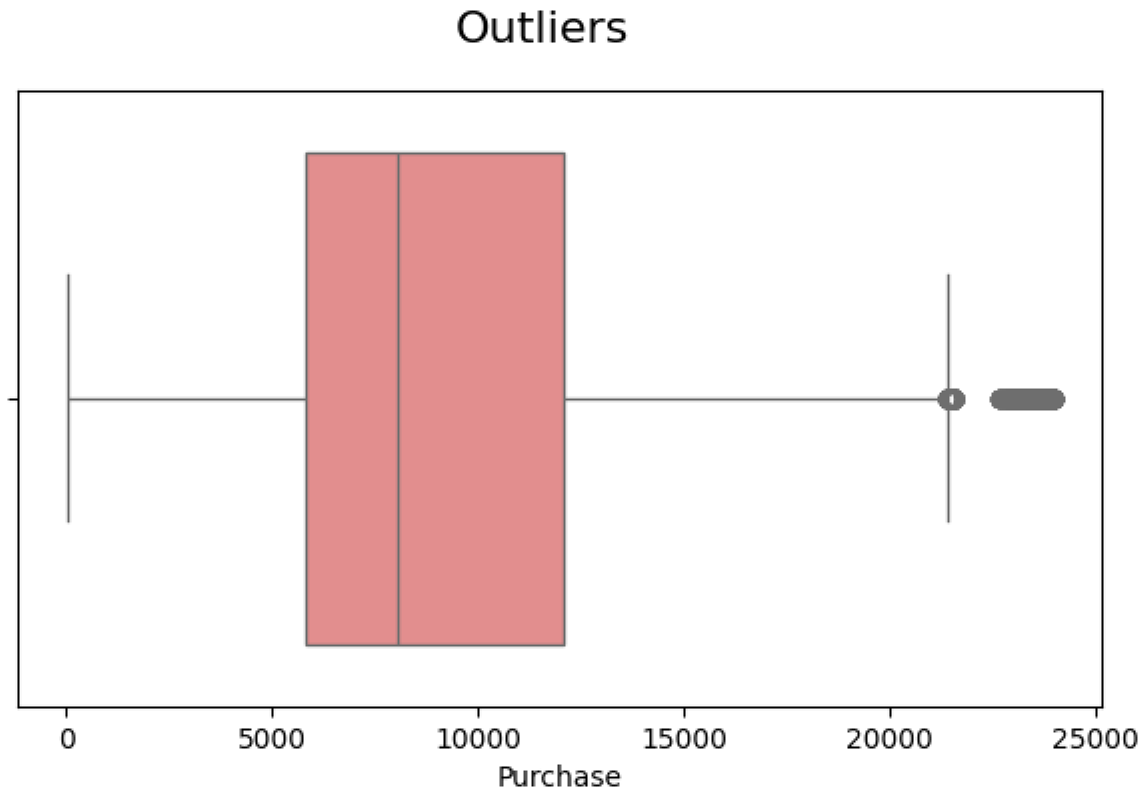


- There is a higher count of purchase values falling within the range of 5000 to 10000.

2.Detect Outliers

Finding the outliers for every continuous variable in the dataset.

In [ ]:
```
plt.figure(figsize=(7,4))
sns.boxplot(data=df, x='Purchase', color='lightcoral',width=0.8)
plt.suptitle('Outliers',fontsize=16)
plt.show()
```



Remove/clip the data between the 5 percentile and 95 percentile

In [ ]:
```
remove_purchase = np.clip(df['Purchase'],np.percentile(df['Purchase'],5),np.percentile(d
```

In [ ]:
```
plt.figure(figsize=(7,4))
sns.boxplot(data=df, x=remove_purchase, color='lightcoral',width=0.8)
plt.suptitle('Removed_Outliers',fontsize=16)
plt.show()
```

insights:

- Clearly we can see that data has been removed between the 5 percentile and 95 percentile

In [ ]:
```
#Replacing the values in marital_status column
df['Marital_Status']=df['Marital_Status'].replace({0:'Single',1:'Married'})
```

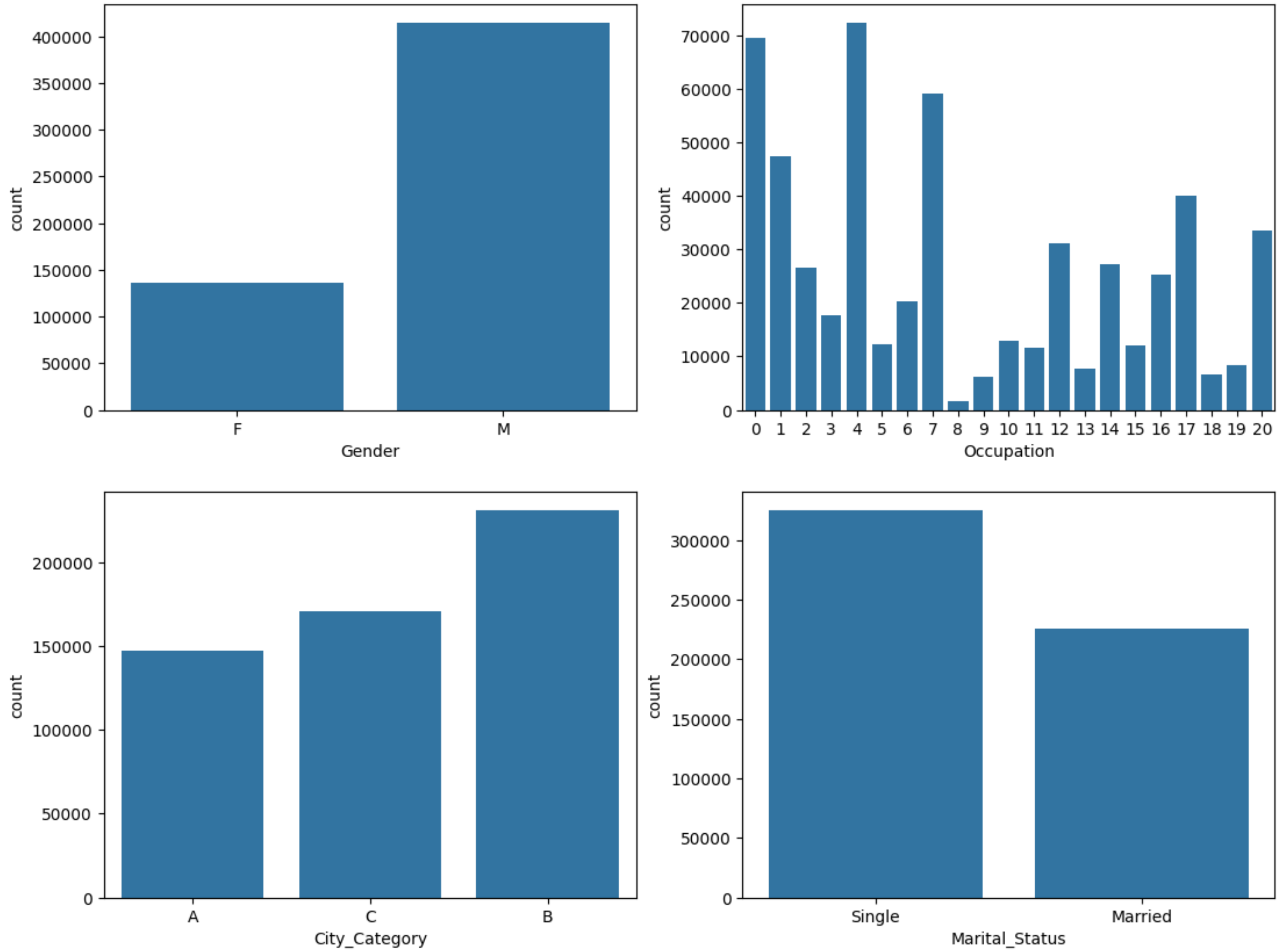
In [ ]:
```
df.head(10)
```

Out[ ]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status
0	1000001	P00069042	F	0-17	10	A	2	Single
1	1000001	P00248942	F	0-17	10	A	2	Single
2	1000001	P00087842	F	0-17	10	A	2	Single
3	1000001	P00085442	F	0-17	10	A	2	Single
4	1000002	P00285442	M	55+	16	C	4+	Single
5	1000003	P00193542	M	26-35	15	A	3	Single
6	1000004	P00184942	M	46-50	7	B	2	Married
7	1000004	P00346142	M	46-50	7	B	2	Married
8	1000004	P0097242	M	46-50	7	B	2	Married
9	1000005	P00274942	M	26-35	20	A	1	Married

In [ ]:

```
fig,axs = plt.subplots(2,2,figsize=(13,10))
sns.countplot(data=df,x='Gender',ax=axs[0,0])
sns.countplot(data=df,x='Occupation',ax=axs[0,1])
sns.countplot(data=df,x='City_Category',ax=axs[1,0])
sns.countplot(data=df,x='Marital_Status',ax=axs[1,1])
plt.show()
```



- Occupation categories 4, 0, and 7 exhibit notably higher purchase volumes, while 8 records the lowest number of purchases.
- most of the user belogs to category B.
- most of the users are singls
- most of rhe users are male

In [ ]:

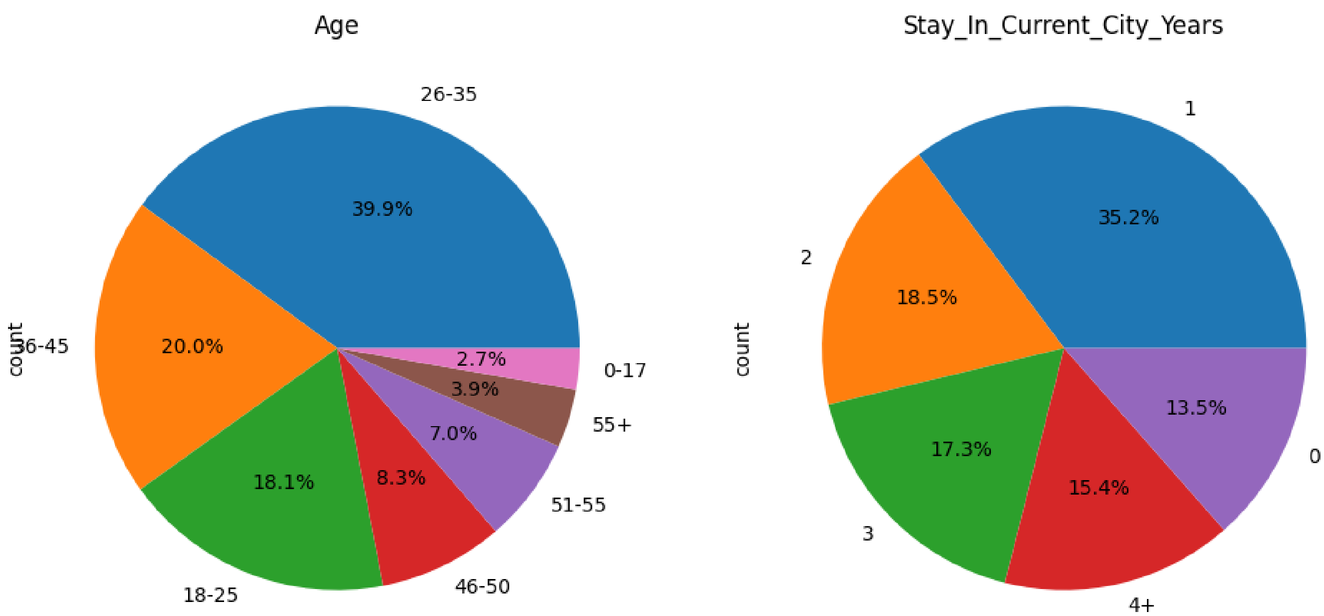
```
df.head(10)
```

Out[ ]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status
0	1000001	P00069042	F	0-17	10	A	2	single
1	1000001	P00248942	F	0-17	10	A	2	single
2	1000001	P00087842	F	0-17	10	A	2	single
3	1000001	P00085442	F	0-17	10	A	2	single
4	1000002	P00285442	M	55+	16	C	4+	single

5	1000003	P00193542	M	26-35	15	A	3	single
6	1000004	P00184942	M	46-50	7	B	2	married
7	1000004	P00346142	M	46-50	7	B	2	married
8	1000004	P0097242	M	46-50	7	B	2	married
9	1000005	P00274942	M	26-35	20	A	1	married

In [ ]:
```
fig,axs =plt.subplots(1,2,figsize=(12,6))
df['Age'].value_counts().plot(kind='pie',autopct='%1.1f%%',ax=axs[0])
axs[0].set_title('Age')
df['Stay_In_Current_City_Years'].value_counts().plot(kind='pie',autopct='%1.1f%%',ax=axs
axs[1].set_title('Stay_In_Current_City_Years')
plt.show()
```

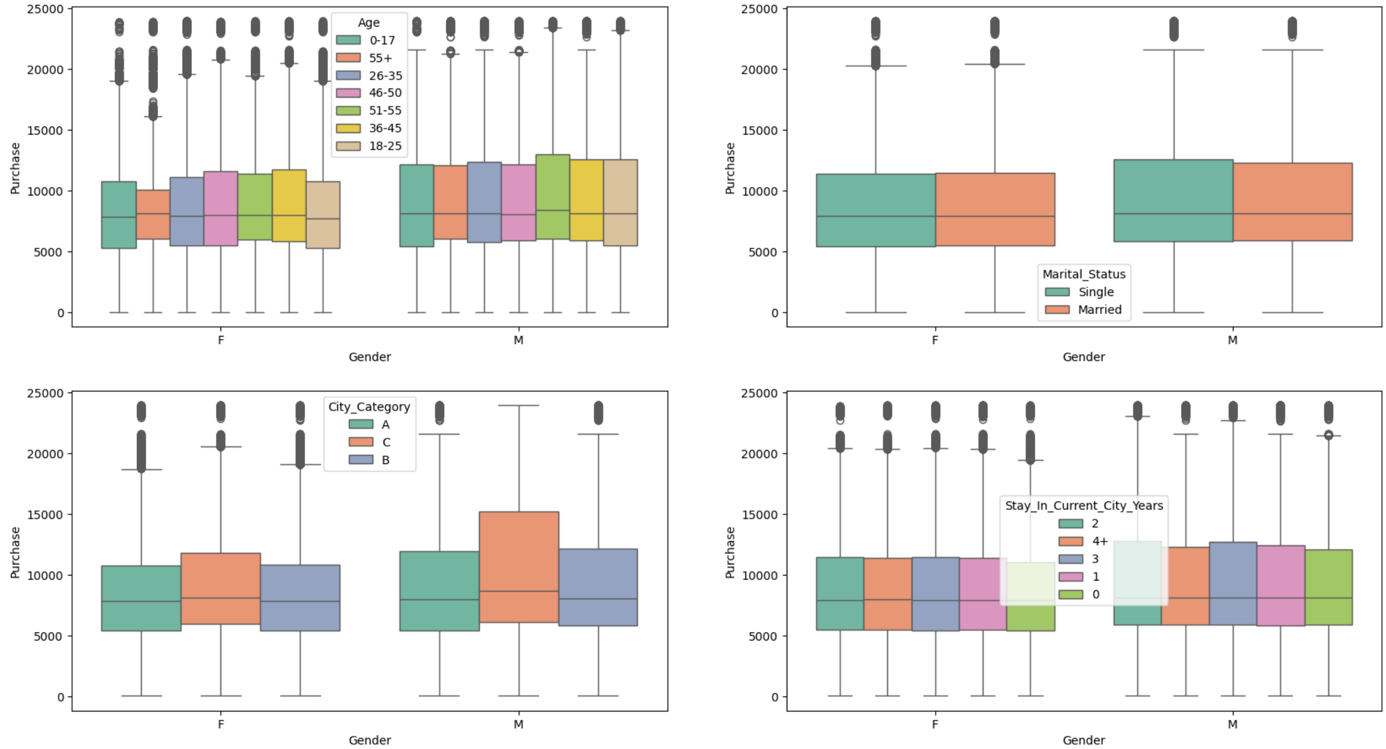


insights:

- agre group 26-35 are approx 40% which is highest and min. % of buyers are of age 55+
- 35.2% are living in city since 1 year .

Bivariate analysis

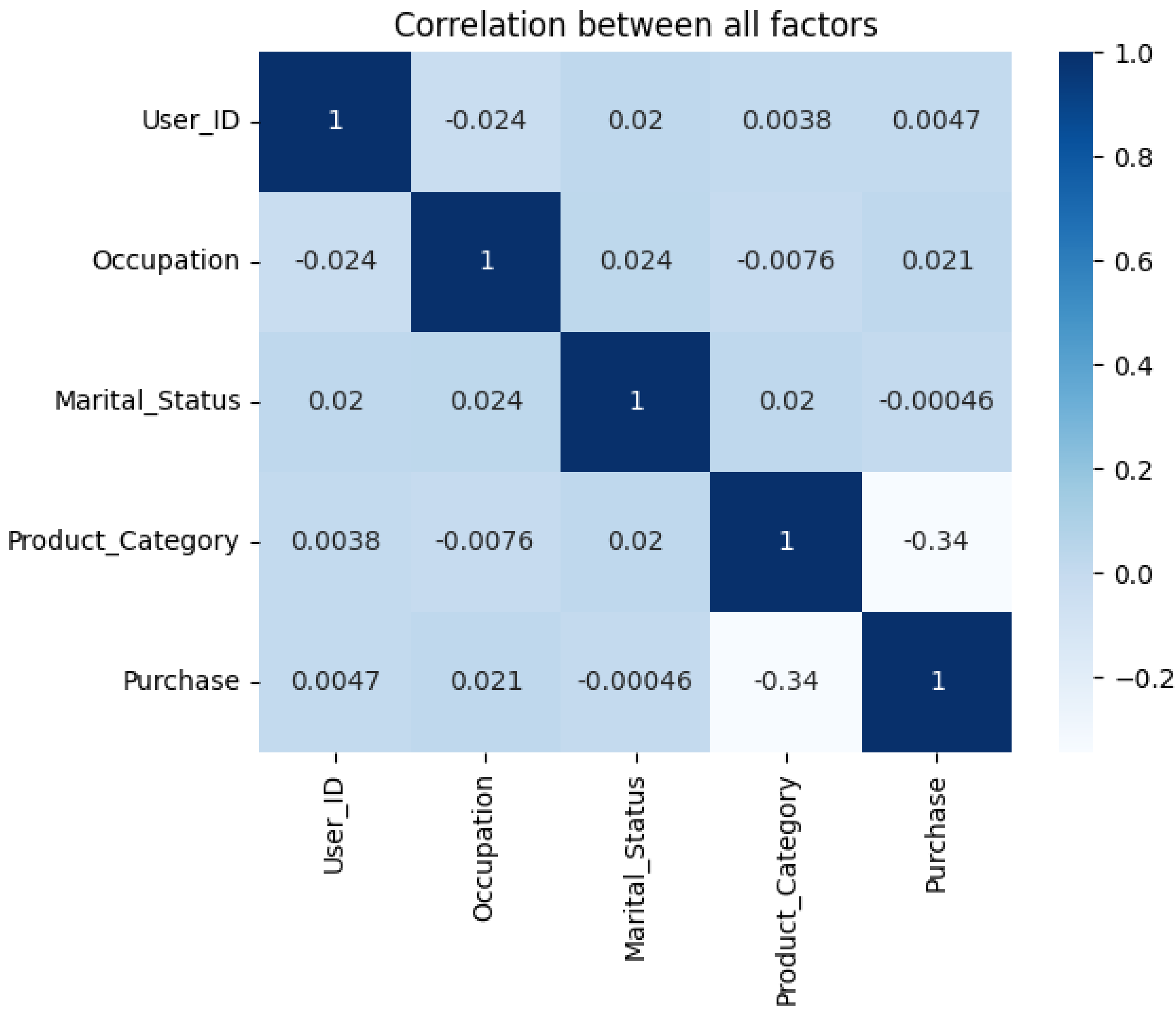
In [ ]:
```
fig,axs=plt.subplots(2,2,figsize=(20,6))
sns.boxplot(data=df,x='Gender',y='Purchase',hue='Age',ax=axs[0,0],palette='Set2')
sns.boxplot(data=df,x='Gender',y='Purchase',hue='Marital_Status',ax=axs[0,1],palette='Se
sns.boxplot(data=df,x='Gender',y='Purchase',hue='City_Category',ax=axs[1,0],palette='Set
sns.boxplot(data=df,x='Gender',y='Purchase',hue='Stay_In_Current_City_Years',ax=axs[1,1]
fig.subplots_adjust(top=1.5)
plt.show()
```



Correlation: Heatmaps

In [225…

```
sns.heatmap(df.corr(numeric_only=True), cmap= "Blues", annot=True)
plt.title('Correlation between all factors')
plt.show()
```



Average amount spent per males and females

In [ ]:
```
amount_df = df.groupby(['User_ID','Gender'])['Purchase'].sum().reset_index()
amount_df
```

Out[ ]:

	User_ID	Gender	Purchase
0	1000001	F	334093
1	1000002	M	810472
2	1000003	M	341635
3	1000004	M	206468
4	1000005	M	821001
...	...	...	...
5886	1006036	F	4116058
5887	1006037	F	1119538
5888	1006038	F	90034
5889	1006039	F	590319
5890	1006040	M	1653299

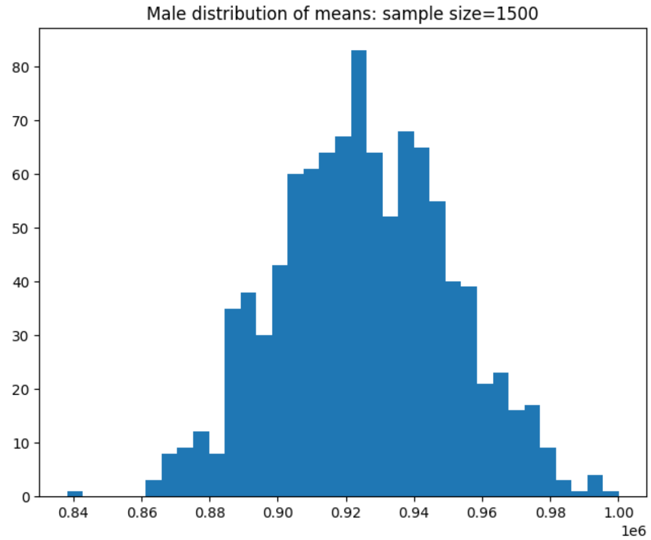
5891 rows × 3 columns

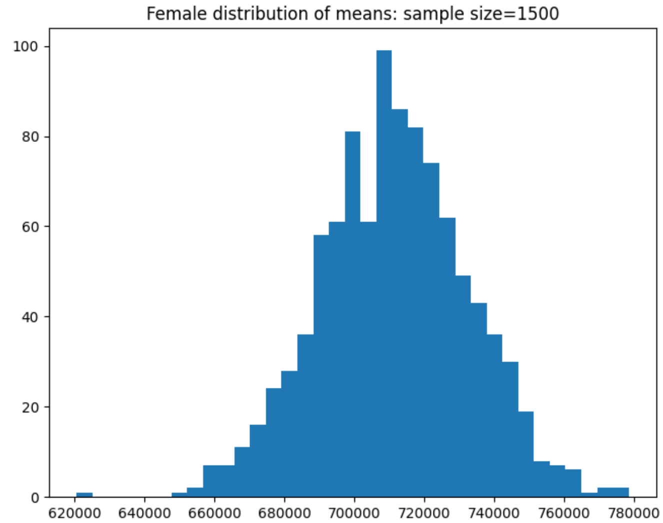
In [ ]:
```
amount_df['Gender'].value_counts()
```
Out[ ]:
```
Gender
M    4225
F    1666
Name: count, dtype: int64
```
In [ ]:
```
M_avg=amount_df[amount_df['Gender']=='M']['Purchase'].mean()
F_avg=amount_df[amount_df['Gender']=='F']['Purchase'].mean()
print('Average amount spent by Male:',M_avg)
print('Average amount spent by Female:',F_avg)
```
```
Average amount spent by Male: 925344.4023668639
Average amount spent by Female: 712024.3949579832
```
insights:

- we can observed that Male customers spent more money than Female customers.

Is the confidence interval computed using the entire dataset wider for one of the genders? Why is this the case?

In [ ]:
```
alpha = 0.05
ci = 1 - alpha/2
z = norm.ppf(ci)
print(z)
```
```
1.959963984540054
```
In [ ]:
```
Male_df = amount_df[amount_df['Gender']=='M']
Female_df = amount_df[amount_df['Gender']=='F']
```
In [ ]:
```
genders = ['M','F']
male_sample_size=1500
female_sample_size=1500
num_repitition = 1000
male_means=[]
female_means=[]
for i in range (num_repitition):
    male_mean= Male_df.sample(male_sample_size, replace=True)['Purchase'].mean()
    female_mean=Female_df.sample(female_sample_size, replace=True)['Purchase'].mean()
    male_means.append(male_mean)
    female_means.append(female_mean)

fig, axs=plt.subplots(1,2, figsize=(17,6))
axs[0].hist(male_means,  bins=35,)
axs[0].set_title('Male distribution of means: sample size=1500')
axs[1].hist(female_means,  bins=35)
axs[1].set_title('Female distribution of means: sample size=1500')
plt.show()
```





In [ ]:

```
sample_mean_male = np.mean(male_means)
sample_mean_female = np.mean(female_means)

sample_std_male=pd.Series(male_means).std()
sample_std_female=pd.Series(female_means).std()

sample_std_error_male = sample_std_male/np.sqrt(1000)
sample_std_error_female = sample_std_female/np.sqrt(1000)

upper_limit_male = ci*sample_std_error_male + sample_mean_male
lower_limit_male = sample_mean_male - ci*sample_std_error_male

upper_limit_female = ci*sample_std_error_female + sample_mean_female
lower_limit_female = sample_mean_female - ci*sample_std_error_female

print('Population avg spend amount for Male: {:.2f}'.format(Male_df['Purchase'].mean()))
print('Population avg spend amount for Female: {:.2f}'.format(Female_df['Purchase'].mean

print('\nMale- Sample mean: {:.2f}'.format(sample_mean_male))
print('Female- Sample mean: {:.2f}'.format(np.mean(female_means)))

print('\nSample std for Male: {:.2f}'.format(pd.Series(male_means).std()))
print('Sample std for Female: {:.2f}'.format(pd.Series(female_means).std()))

print('\nSample std error for Male: {:.2f}'.format(pd.Series(male_means).std()/np.sqrt(1
print('Sample std error for Female: {:.2f}'.format(pd.Series(female_means).std()/np.sqrt

print('\nMale at 95% CI: ',[lower_limit_male, upper_limit_male])
print('Female at 95% CI: ',[lower_limit_female, upper_limit_female])
```

```
Population avg spend amount for Male: 925344.40
Population avg spend amount for Female: 712024.39

Male- Sample mean: 925529.74
Female- Sample mean: 711289.96

Sample std for Male: 25151.49
Sample std for Female: 21382.53

Sample std error for Male: 795.36
Sample std error for Female: 676.17

Male at 95% CI:  [924754.2661356989, 926305.2179523013]
Female at 95% CI:  [710630.6895181899, 711949.23053781]
```

using the Central Limit Theorem for the population we can say that:

- Average amount spend by male customers is 925344.40
- Average amount spend by female customers is 712024.39

using the Confidence interval at 95%, we can say that:

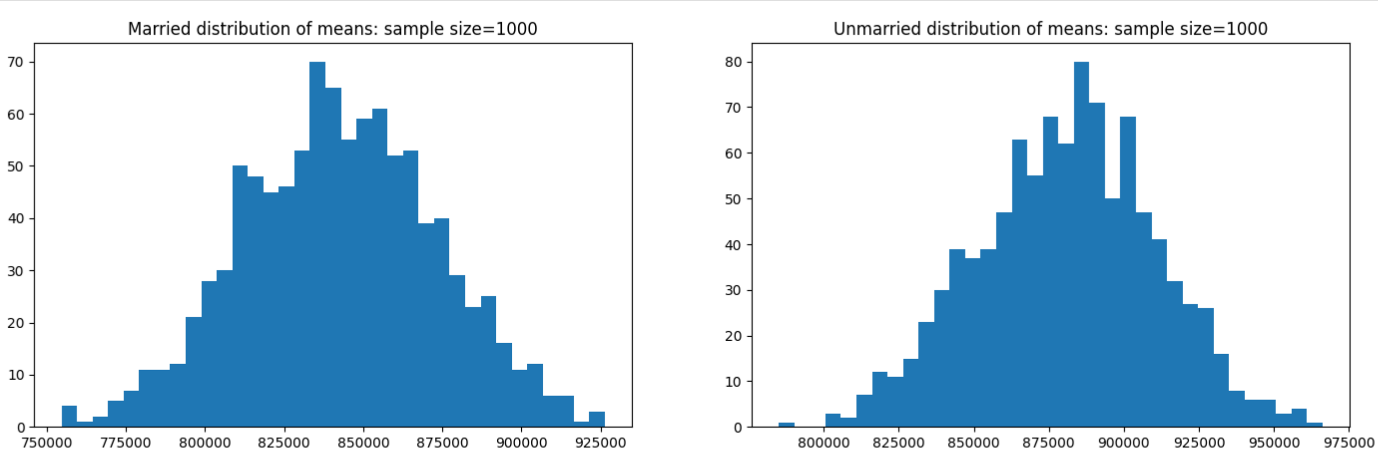
- Average amount spend by male customers lie in the range 924754.26 - 926305.21

- Average amount spend by female customers lie in range 710630.68- 711949.23

How does Marital_Status affect the amount spent?

In [ ]:
```
df.head()
df['Marital_Status']=df['Marital_Status'].replace({'Single':0,'Married':1})
```

In [ ]:
```
avg_marital = df.groupby(['User_ID','Marital_Status'])[['Purchase']].sum()
avg_marital = avg_marital.reset_index()

#Marital wise distribution
df_married=avg_marital[avg_marital['Marital_Status']==1]
df_unmarried=avg_marital[avg_marital['Marital_Status']==0]

sample_size=1000
num_repitition =1000

married_means=[]
unmarried_means=[]

for i in range (num_repitition):
    married_mean = df_married.sample(sample_size, replace=True)['Purchase'].mean()
    unmarried_mean = df_unmarried.sample(sample_size, replace=True)['Purchase'].mean()

    married_means.append(married_mean)
    unmarried_means.append(unmarried_mean)

fig, axs=plt.subplots(nrows=1, ncols=2, figsize=(17,5))
axs[0].hist(married_means,  bins=35)
axs[0].set_title('Married distribution of means: sample size=1000')
axs[1].hist(unmarried_means,  bins=35)
axs[1].set_title('Unmarried distribution of means: sample size=1000')
plt.show()
```



In [ ]:
```
avg_marital['Marital_Status'].value_counts(normalize=True)*100
```

Out[ ]:
```
Marital_Status
0    58.003735
1    41.996265
Name: proportion, dtype: float64
```

In [ ]:
```
# Calculating 95% confidence interval for avg expenses for married/Unmarried for sample
alpha = 0.05
ci = 1 - alpha/2
z = norm.ppf(ci)
print(z)
sample_married_mean = np.mean(married_means)
sample_unmarried_mean = np.mean(unmarried_means)

sample_std_married = pd.Series(married_means).std()
sample_std_unmarried = pd.Series(unmarried_means).std()

sample_std_error_married = sample_std_married/np.sqrt(1000)
sample_std_error_unmarried = sample_std_unmarried/np.sqrt(1000)

upper_limit_married = ci*sample_std_error_married + sample_married_mean
lower_limit_married = sample_married_mean - ci*sample_std_error_married

upper_limit_unmarried = ci*sample_std_error_unmarried + sample_unmarried_mean
lower_limit_unmarried = sample_unmarried_mean - ci*sample_std_error_female

print('Population avg spend amount for Married: {:.2f}'.format(df_married['Purchase'].me
print('Population avg spend amount for Unmarried: {:.2f}'.format(df_unmarried['Purchase'

print('\nMarried- Sample mean: {:.2f}'.format(sample_married_mean))
print('Unmarried- Sample mean: {:.2f}'.format(sample_unmarried_mean))

print('\nSample std for Married: {:.2f}'.format(sample_std_married))
print('Sample std for Unmarried: {:.2f}'.format(sample_std_unmarried))

print('\nSample std error for Married: {:.2f}'.format(sample_std_error_married))
print('Sample std error for Unnmarried: {:.2f}'.format(sample_std_error_unmarried))

print('\nMarried at 95% CI: ',[lower_limit_married, upper_limit_married])
print('Unmarried at 95% CI: ',[lower_limit_unmarried, upper_limit_unmarried])
```

```
1.959963984540054
Population avg spend amount for Married: 843526.80
Population avg spend amount for Unmarried: 880575.78

Married- Sample mean: 842680.90
Unmarried- Sample mean: 880942.13

Sample std for Married: 30664.86
Sample std for Unmarried: 29703.98

Sample std error for Married: 969.71
Sample std error for Unnmarried: 939.32

Married at 95% CI:  [841735.4299221095, 843626.3605338903]
Unmarried at 95% CI:  [880282.8627341898, 881857.9723673475]
```

insights: using the Central Limit Theorem for the population we can say that:

- Average amount spend by married customers is 843526.80
- Average amount spend by unmarried customers is 880575.78

by using the Confidence interval at 90%, we can say that:

- Average amount spend by married customers lie in the range 841735.42, 843626.36

- Average amount spend by unmarried customers lie in range 880282.86, 881857.97

How does Age affect the amount spent

In [ ]:

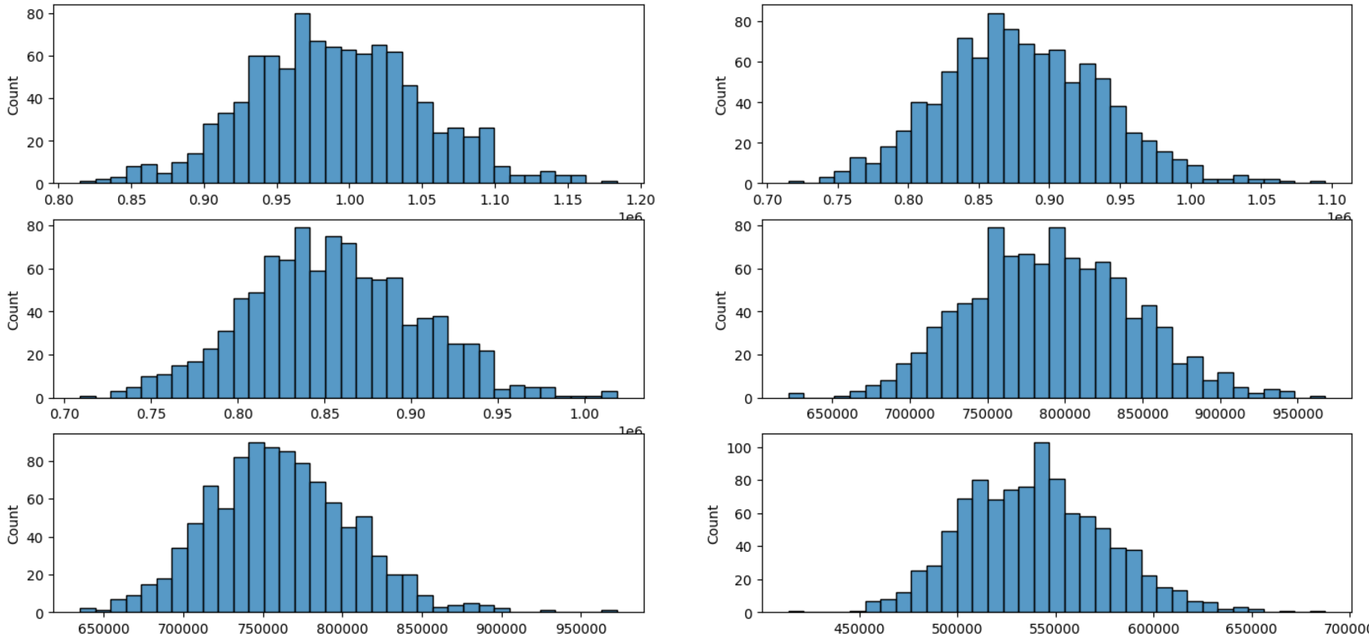
```
avg_age = df.groupby(['User_ID', 'Age'])[['Purchase']].sum()
avg_age = avg_age.reset_index()
avg_age['Age'].value_counts()
```

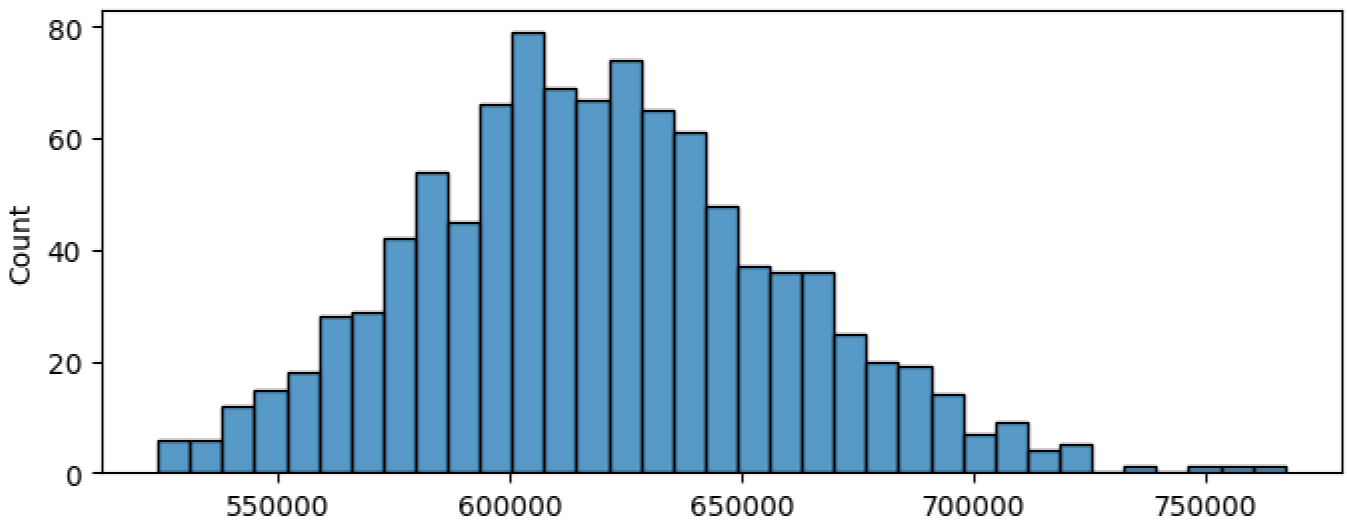
Out[ ]:

```
Age
26-35    2053
36-45    1167
18-25    1069
46-50     531
51-55     481
55+       372
0-17      218
Name: count, dtype: int64
```

In [ ]:

```
samp_size=300
num_repitition =1000

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

for i in age_intervals:
    for j in range(num_repitition):
        mean = avg_age[avg_age['Age']==i].sample(samp_size, replace=True)['Purchase'].me
        age_means[i].append(mean)

fig, axis = plt.subplots(3,2, figsize=(17, 8))

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

plt.figure(figsize=(8, 3))
sns.histplot(age_means['0-17'],bins=35)
plt.show()
```





In [ ]:
```python
population_means = {}
for i in age_intervals:
    population_means[i] = []
    population_m = avg_age[avg_age['Age']==i]['Purchase'].mean()
    population_means[i].append(population_m)
    print("Population mean for age group '{}': {:.2f}".format(i, population_m))
```

```
Population mean for age group '26-35': 989659.32
Population mean for age group '36-45': 879665.71
Population mean for age group '18-25': 854863.12
Population mean for age group '46-50': 792548.78
Population mean for age group '51-55': 763200.92
Population mean for age group '55+': 539697.24
Population mean for age group '0-17': 618867.81
```

- 26-35 age group tends to spend the most, indicating a prime demographic for high-value products and premium offerings.
- 55+ age group spends the least on average

Recommendations

- Customers in the age 18-45 spend more money than the others, So company should focus on acquisition of customers who are in the age 18-45
- Unmarried customers spend more money than married customers, So company should focus on acquisition of Unmarried customers.
- Men spent more money than women, So company should focus on retaining the male customers and getting more male customers.
- With the age group between 26 and 45 contributing to the majority of sales, Walmart should specifically cater to the preferences and needs of this demographic.This could include offering exclusive deals on products that are popular among this age group.
- Focusing advertising and promotional efforts on customers from City Type B who have been staying for 1 year can be a profitable strategy. This specific target audience appears to exhibit favorable spending patterns, making them a promising group for campaigns designed to boost sales and engagement.
- Targetting Unmarried males and married females with advertisements specific to them can fetch new customers from the group and engage the existing customers more.
- The high purchasing frequency observed for products in Product Categories 1, 5, and 8 suggests strong demand for items within these categories. Focusing on increasing the availability and promotion of products in these categories could be a profitable strategy for the company, as it aligns with consumer preferences and buying patterns.