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).

Dataset

The company collected the transactional data of customers who purchased products from the Walmart Stores during Black Friday. The dataset has the following features:
Dataset link: **Walmart_data.csv**

User_ID:	User ID
Product_ID:	Product ID
Gender:	Sex of User
Age:	Age in bins
Occupation:	Occupation(Masked)
City_Category:	Category of the City (A,B,C)
StayInCurrentCityYears:	Number of years stay in current city
Marital_Status:	Marital Status
ProductCategory:	Product Category (Masked)
Purchase:	Purchase Amount

Solution:

Before starting the analysis let's import the required libraries.

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

1. Import the dataset and do usual data analysis steps like checking the structure & characteristics of the dataset:

```
[2] import io
    import requests
    url="https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/walmart_data.csv?1641285094"
    s=requests.get(url).content
    df=pd.read_csv(io.StringIO(s.decode('utf-8')))
```

```
[3] df
```

	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
...	...	...	...	...	...	...	...	...	...	...
550063	1006033	P00372445	M	51-55	13	B	1	1	20	368
550064	1006035	P00375436	F	26-35	1	C	3	0	20	371
550065	1006036	P00375436	F	26-35	15	B	4+	1	20	137
550066	1006038	P00375436	F	55+	1	C	2	0	20	365
550067	1006039	P00371644	F	46-50	0	B	4+	1	20	490

550068 rows × 10 columns

a. The data type of all columns in the "customers" table.

```
[4] 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
```

Insights:

i. The data set has total 10 columns, out of which 5 integer and 5 string columns are present.

ii. The data set is taking 42+ MB size in the memory.

b. You can find the number of rows and columns given in the dataset

```
[6]  print(f'The number of rows present in the data set: {df.shape[0]}')
     print(f'The number of columns present in the data set: {df.shape[1]}')

     The number of rows present in the data set: 550068
     The number of columns present in the data set: 10
```

The data set have 550068 number of rows and 10 columns.

c. Check for the missing values and find the number of missing values in each column:

```
[8]  for i in df.columns:
       print(f'The number of null values present in {i} :', sum(df[i].isna()))

     The number of null values present in User_ID : 0
     The number of null values present in Product_ID : 0
     The number of null values present in Gender : 0
     The number of null values present in Age : 0
     The number of null values present in Occupation : 0
     The number of null values present in City_Category : 0
     The number of null values present in Stay_In_Current_City_Years : 0
     The number of null values present in Marital_Status : 0
     The number of null values present in Product_Category : 0
     The number of null values present in Purchase : 0
```

Insights:

There are no null values are present in any of the columns.

```
[10] for i in df.columns:
       print(f'The number of unique values present in {i} :', df[i].nunique())

     The number of unique values present in User_ID : 5891
     The number of unique values present in Product_ID : 3631
     The number of unique values present in Gender : 2
     The number of unique values present in Age : 7
     The number of unique values present in Occupation : 21
     The number of unique values present in City_Category : 3
     The number of unique values present in Stay_In_Current_City_Years : 5
     The number of unique values present in Marital_Status : 2
     The number of unique values present in Product_Category : 20
     The number of unique values present in Purchase : 18105
```

```
[33] for i in df.columns:
       print(f'The unique values present in {i} :', df[i].unique())
       print('-'*50)
```

```
The unique values present in User_ID : [1000001 1000002 1000003 ... 1004113 1005391 1001529]
--------------------------------------------------
The unique values present in Product_ID : ['P00069042' 'P00248942' 'P00087842' ... 'P00370293' 'P00371644'
 'P00370853']
--------------------------------------------------
The unique values present in Gender : ['F' 'M']
--------------------------------------------------
The unique values present in Age : ['0-17' '55+' '26-35' '46-50' '51-55' '36-45' '18-25']
--------------------------------------------------
The unique values present in Occupation : [10 16 15  7 20  9  1 12 17  0  3  4 11  8 19  2 18  5 14 13  6]
--------------------------------------------------
The unique values present in City_Category : ['A' 'C' 'B']
--------------------------------------------------
The unique values present in Stay_In_Current_City_Years : ['2' '4+' '3' '1' '0']
--------------------------------------------------
The unique values present in Marital_Status : [0 1]
--------------------------------------------------
The unique values present in Product_Category : [ 3  1 12  8  5  4  2  6 14 11 13 15  7 16 18 10 17  9 20 19]
--------------------------------------------------
The unique values present in Purchase : [ 8370 15200  1422 ...   135   123   613]
--------------------------------------------------
```

Insights:

1. The user_id column has 5891 unique values present.
2. Product Id has 3631 unique values present.
3. There are two type of genders Male and Female.
4. Age group is given in 7 bins. Which have minimum group of 0 to 17 and maximum age group is 55+
5. 3 types of city is present in the data set.
5. Marital Status is given in binary 0 and 1.

<u>Statistical Summary</u>:

1. Categorical Variables:

```
for i in df.columns[:-1]:
  df[i] = df[i].astype('category')

df.describe(include = 'category')
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category
count	550068	550068	550068	550068	550068	550068	550068	550068	550068
unique	5891	3631	2	7	21	3	5	2	20
top	1001680	P00265242	M	26-35	4	B	1	0	5
freq	1026	1880	414259	219587	72308	231173	193821	324731	150933

Insights:

1. <u>User_ID:</u> Out of 550068 rows 5891 customers are unique which conclude that same customer buy product from Walmart multiple times which is expected.

2. <u>Product_ID</u>: Walmart have 3631 unique products present among which P00265242 being the highest seller which sold for 1880 times.

3. Gender: Out of male and female 414259 times males purchased from Walmart, means Walmart has more male customer. Which indicating a significant disparity in purchase behavior between males and females during the Black Friday event.

4. Age: There are 7 age group present among which 26-35 group has the highest purchase value. Means this age group visits Walmart for 219587 times.

5. Marital Status: (324731/550068) *100 = 59% of transaction are done by unmarried among the Walmart customer list.

2. Numerical Variable:

```
[40] df.describe()
```

	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

Insight:

1. The minimum purchase amount is $12 and maximum is $23961.
2. The mean value is $9263.96 which is notably differ from median of $8047. Which indicates the right skewed distribution.
3. 50% percentile purchase is done between $5823 to $12054.

Duplicate Detection:

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

     False    550068
     dtype: int64
```

There is no duplicate row present in the data.

Decoding Marital Status:

```
df['Mod_Marital_Status'] = df['Marital_Status']
df['Mod_Marital_Status'] = df['Mod_Marital_Status'].replace({'0': 'Unmarried', '1': 'Married'})
df['Mod_Marital_Status'].unique()
```

```
['Unmarried', 'Married']
Categories (2, object): ['Unmarried', 'Married']
```

```
[57] df.head()
```

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

Creating one column to decode Marital Status in Married and Unmarried.

2. Detect Null values and outliers

```
Q1 = df['Purchase'].quantile(0.25)
Q3 = df['Purchase'].quantile(0.75)
print(f'The quartile 1 value is {Q1}')
print(f'The quartile 3 value is {Q3}')
IQR = Q3 - Q1
print(f'The inter quartile value is {IQR}')
print(f'The amount below {Q1-1.5*IQR} and the amount above {Q3 + 1.5*IQR} will be treated as outliers.')
```

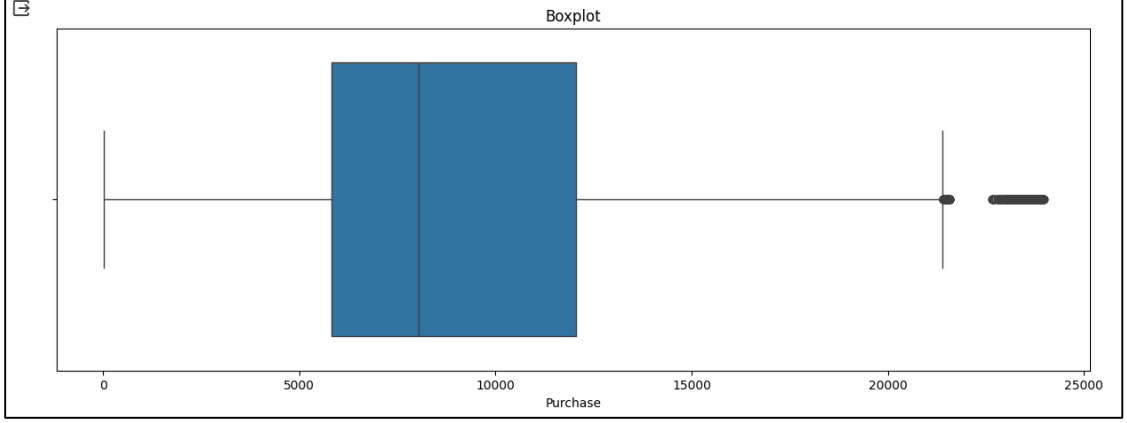
```
The quartile 1 value is 5823.0
The quartile 3 value is 12054.0
The inter quartile value is 6231.0
The amount below -3523.5 and the amount above 21400.5 will be treated as outliers.
```

Insights:

1. The quartile 1 value is $5823, and quartile 3 value is $12054.
2. The IQR = $6231. So, the purchase amount below Q1 – 1.5*IQR = -$3523.5 and above $21400.5 will be treated as outliers.

```
[73] ucnt = df[df['Purchase'] > 21400.5]['User_ID'].count()
     lcnt = df[df['Purchase'] < -3523.5]['User_ID'].count()
     print(f'The count of lower outliers are: {lcnt}')
     print(f'The count of upper outliers are: {ucnt}')

     The count of lower outliers are: 0
     The count of upper outliers are: 2677
```

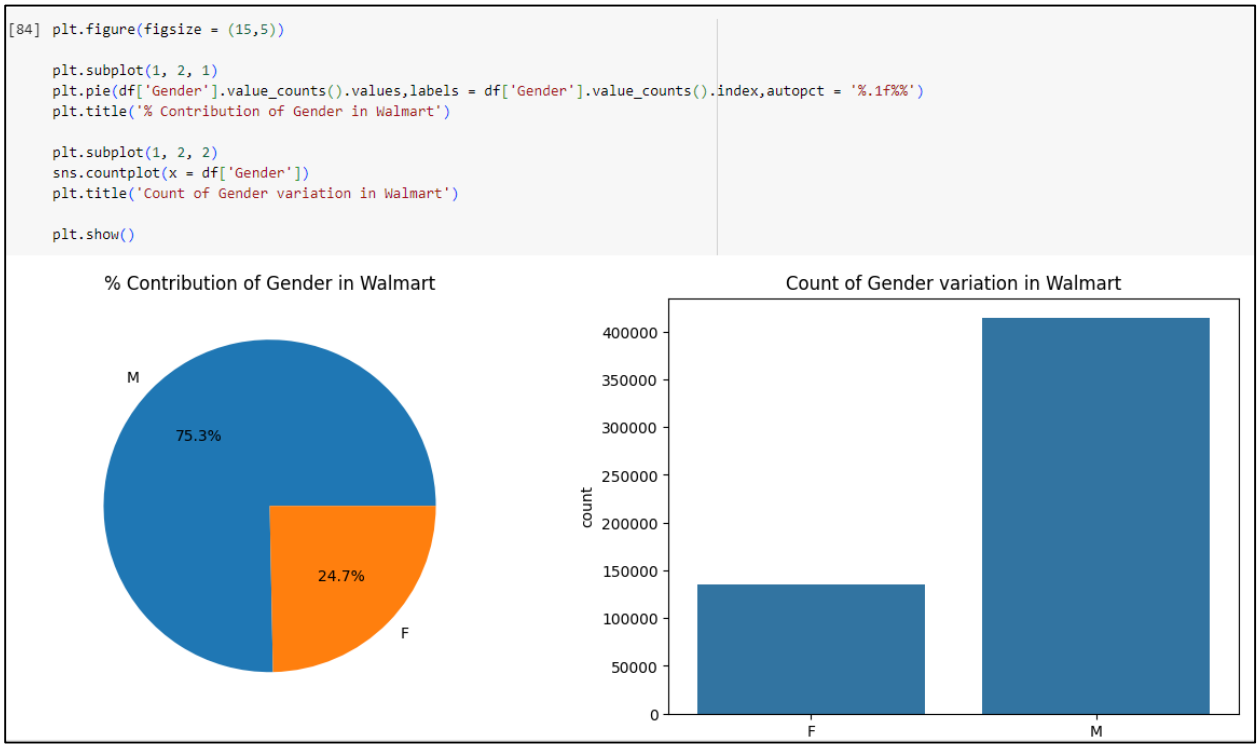
```
plt.figure(figsize = (15, 5))
sns.boxplot(x = df['Purchase'])
plt.title('Boxplot')
plt.show()
```



3. Categorical Variable Univariate analysis:

i. Gender:

```
[84] plt.figure(figsize = (15,5))

     plt.subplot(1, 2, 1)
     plt.pie(df['Gender'].value_counts().values,labels = df['Gender'].value_counts().index,autopct = '%.1f%%')
     plt.title('% Contribution of Gender in Walmart')

     plt.subplot(1, 2, 2)
     sns.countplot(x = df['Gender'])
     plt.title('Count of Gender variation in Walmart')

     plt.show()
```

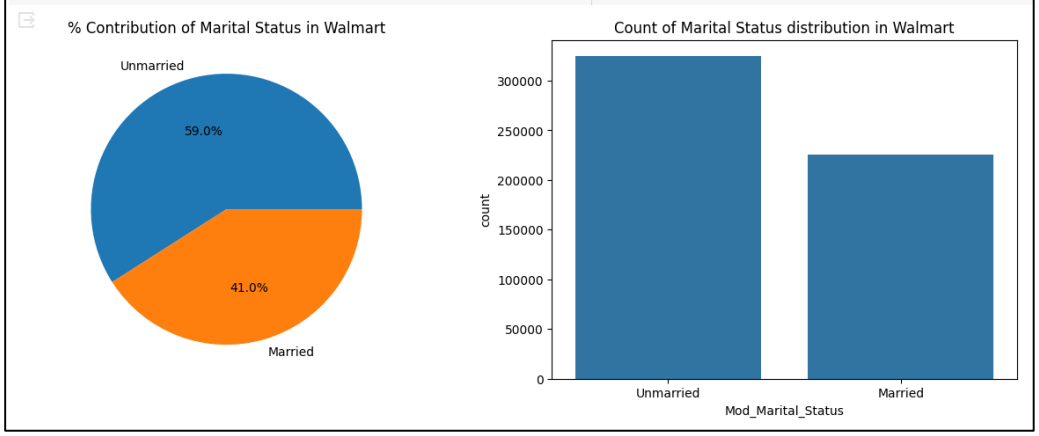


Insight:

75.3% transaction is done by male customer whereas 24.7% transaction is done by female customers.

ii. Marital Status:

```
[85] plt.figure(figsize = (15,5))

     plt.subplot(1, 2, 1)
     plt.pie(df['Mod_Marital_Status'].value_counts().values,labels = df['Mod_Marital_Status'].value_counts().index,autopct = '%.1f%%')
     plt.title('% Contribution of Marital Status in Walmart')

     plt.subplot(1, 2, 2)
     sns.countplot(x = df['Mod_Marital_Status'])
     plt.title('Count of Marital Status distribution in Walmart')

     plt.show()
```

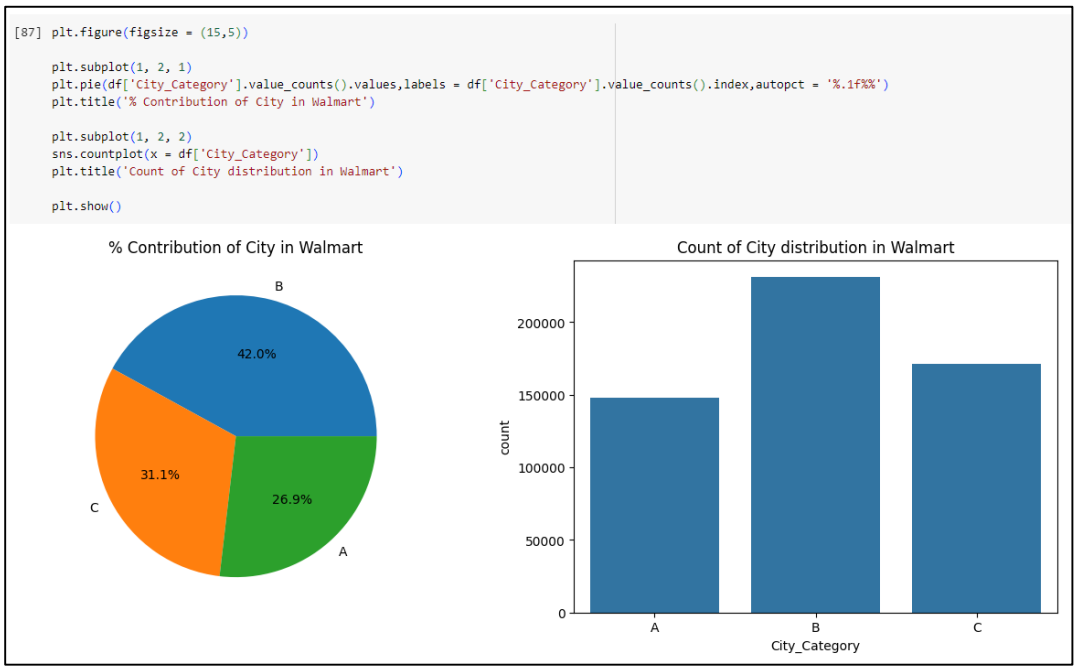


Insights:

59% of transaction is done by unmarried persons whereas 41% transactions are done by married couples.

iii. City Category:

```
[87] plt.figure(figsize = (15,5))

     plt.subplot(1, 2, 1)
     plt.pie(df['City_Category'].value_counts().values,labels = df['City_Category'].value_counts().index,autopct = '%.1f%%')
     plt.title('% Contribution of City in Walmart')

     plt.subplot(1, 2, 2)
     sns.countplot(x = df['City_Category'])
     plt.title('Count of City distribution in Walmart')

     plt.show()
```

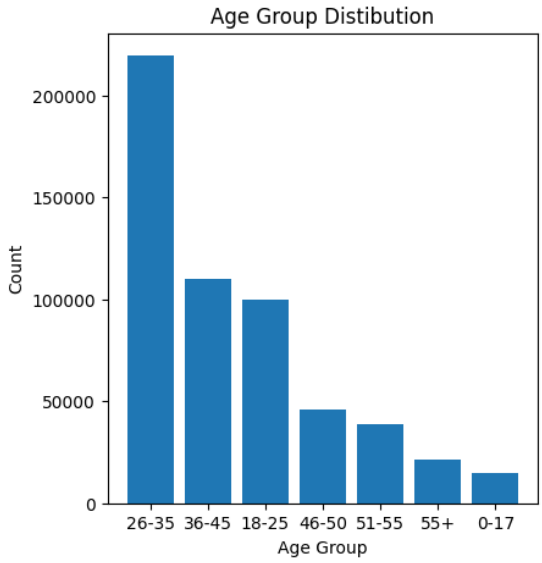


Insights:

City B has the most number of transaction of 42% followed by city C 31.1% and city A 26.9%

iv. Age Group:

```
[108] plt.figure(figsize = (10, 5))

      plt.subplot(1,2,1)
      df1 = df['Age'].value_counts()
      plt.bar(x = df1.index, height = df1.values)
      plt.title('Age Group Distibution')
      plt.xlabel('Age Group')
      plt.ylabel('Count')

      plt.subplot(1,2,2)
      age_per = [['26-35','40%'], ['36-45','20%'], ['18-25','18%'], ['46-50','8%'], ['51-55','7%'], ['55+','4%'], ['0-17','3%']]
      plt.table(cellText = age_per, colLabels = ['Age Grp', '% Contri'], cellLoc = 'center', colLoc = 'center', bbox =[0, 0, 1, 1])
      plt.axis('off')

      plt.show()
```



Age Grp	% Contri
26-35	40%
36-45	20%
18-25	18%
46-50	8%
51-55	7%
55+	4%
0-17	3%

Insights:

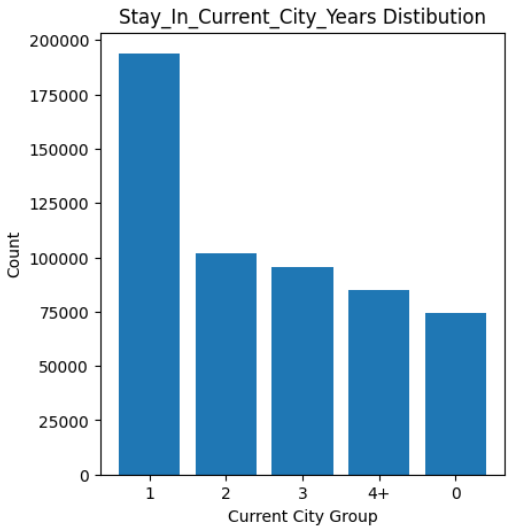
1. The age group 26-35 are the highest contributor in Walmart's Black Friday sales which is 40%. By which we can say adult customers have much interest in promos and discounts.

2. The age group of 36-45 and 18-25 contributes 20% and 18% respectively into the Black Friday sales. So we can say that Walmart has a diversity in customer age groups.

3. The age group of 51-55, 55+ and 0-17 are contribute least in the Black Friday Sales, which implies that Walmart should focus in this range to attract elderly customers and children.

v. Customer Sales in City Distribution:

```
[110] plt.figure(figsize = (10, 5))

      plt.subplot(1,2,1)
      df2 = df['Stay_In_Current_City_Years'].value_counts()
      plt.bar(x = df2.index, height = df2.values)
      plt.title('Stay_In_Current_City_Years Distibution')
      plt.xlabel('Current City Group')
      plt.ylabel('Count')

      plt.subplot(1,2,2)
      age_per = [['1','35%'],['2','19%'],['3','17%'],['4+','15%'],['0','14%']]
      plt.table(cellText = age_per, colLabels = ['Current_City Grp', '% Contri'], cellLoc = 'center', colLoc = 'center', bbox =[0, 0, 1, 1])
      plt.axis('off')

      plt.show()
```



Current_City Grp	% Contri
1	35%
2	19%
3	17%
4+	15%
0	14%

Insights:

1. The customers who lived in the city for last 1 years contribute more in Black Friday sales around 35%. Which implies that the new customers are more attractive to Black Friday Sales as it is there first time sales.

2. Customers who lived 2, 3 and 4+ years the sales are decreasing year by year like 19% to 17% to 15% which implies that customers are lose their interest over the years. Walmart should launch some additional promos for old customers to attract them more.

3. The people who not lived in the city or may be just arrived contributed 14% of the transaction which is acceptable.

4. Data Exploration:

a. What products are different age groups buying?

```
df3 = df['Product_Category'].value_counts().head(5).reset_index()
df3
```

	index	Product_Category
0	5	150933
1	1	140378
2	8	113925
3	11	24287
4	2	23864

Next steps: Generate code with df3 View recommended plots

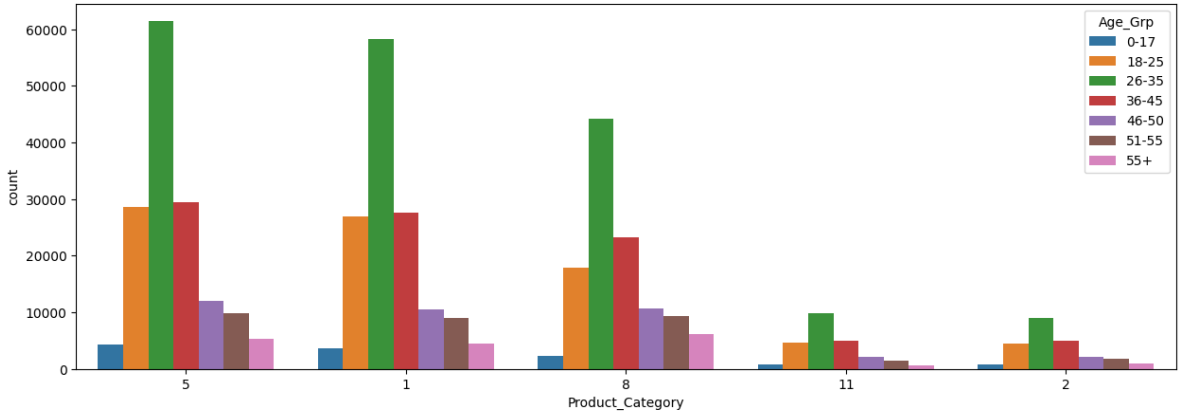
```
[137] df4 = pd.merge(df3, df, left_on = 'index', right_on = 'Product_Category', how = 'inner')[['index',  'Product_Category_x', 'Age']]
```

```
[138] df4.rename(columns = {'index' : 'Product_Category', 'Product_Category_x': 'Count_Product_Category', 'Age' : 'Age_Grp'}, inplace = True)
```

```
[151] df4['Product_Category'] = df4['Product_Category'].astype('str')
```

```
plt.figure(figsize = (15, 5))
sns.countplot(df4, x = 'Product_Category', hue = 'Age_Grp')
```

```
<Axes: xlabel='Product_Category', ylabel='count'>
```



Insights:

For the above analysis we can conclude:

1. Product Category 5,1,8,11,2 are the most saleable products in Black Friday. Which contributes 95 % of the transactions.

2. Age grp 26-35 are interested in all the 5 type of products among which product category 5,1 and 8 brings more contribution from this age group.

3. Age group 18-25 and 36-45 contributes almost same sales.

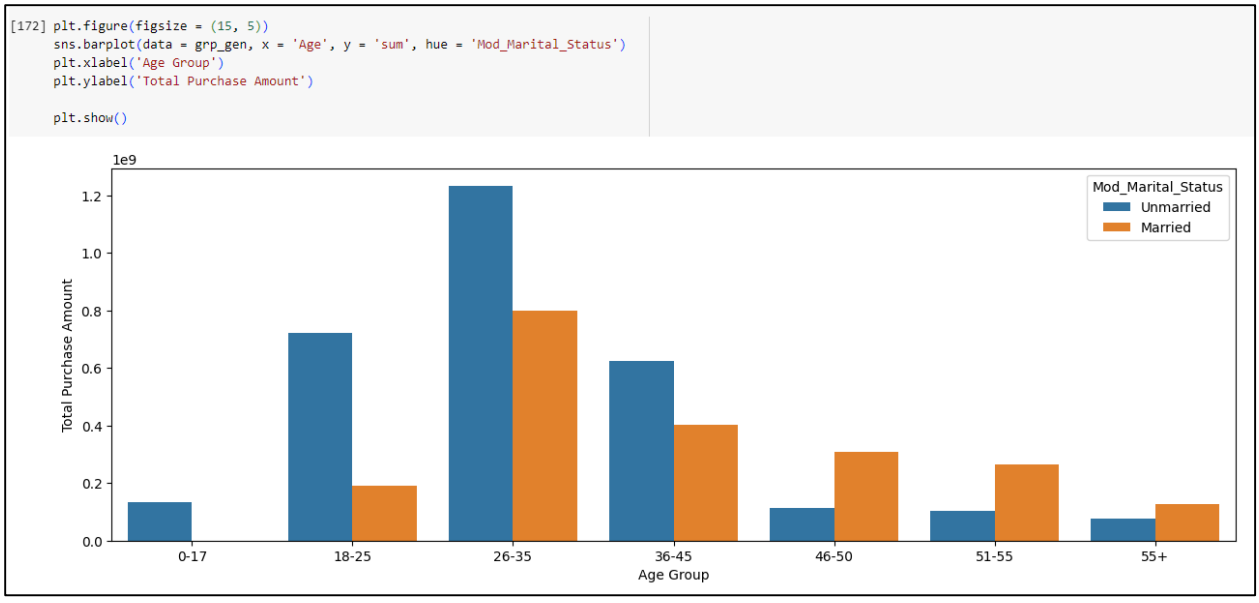
4. Walmart should focus on age group of 0-17 and 55+ to bring more profit on their sales.

b. Is there a relationship between age, marital status, and the amount spent?

```
grp_gen = df.groupby(['Mod_Marital_Status', 'Age'])['Purchase'].agg(['sum', 'count']).reset_index()
grp_gen['Contri_Per_Purchase'] = round(grp_gen['sum']/grp_gen['count'],2)
grp_gen
```

	Mod_Marital_Status	Age	sum	count	Contri_Per_Purchase
0	Unmarried	0-17	134913183	15102	8933.46
1	Unmarried	18-25	723920602	78544	9216.75
2	Unmarried	26-35	1233330102	133296	9252.57
3	Unmarried	36-45	624110760	66377	9402.52
4	Unmarried	46-50	113658360	12690	8956.53
5	Unmarried	51-55	103792394	10839	9575.83
6	Unmarried	55+	75202046	7883	9539.77
7	Married	0-17	0	0	NaN
8	Married	18-25	189928073	21116	8994.51
9	Married	26-35	798440476	86291	9252.88
10	Married	36-45	402459124	43636	9223.10
11	Married	46-50	307185043	33011	9305.54
12	Married	51-55	263307250	27662	9518.74
13	Married	55+	125565329	13621	9218.51

```
[172] plt.figure(figsize = (15, 5))
      sns.barplot(data = grp_gen, x = 'Age', y = 'sum', hue = 'Mod_Marital_Status')
      plt.xlabel('Age Group')
      plt.ylabel('Total Purchase Amount')

      plt.show()
```



Insights:

1. The unmarried customers in the age group of 18-25, 26-35 and 36-45 contributed 80% of the unmarried sales during Black Friday in Walmart. Among which 26-35 age group has highest contribution.

2. Married couple of age group 26-35 and 36-45 contributed highest among married couples. The age group 0-17 only contributes as unmarried which is expected.

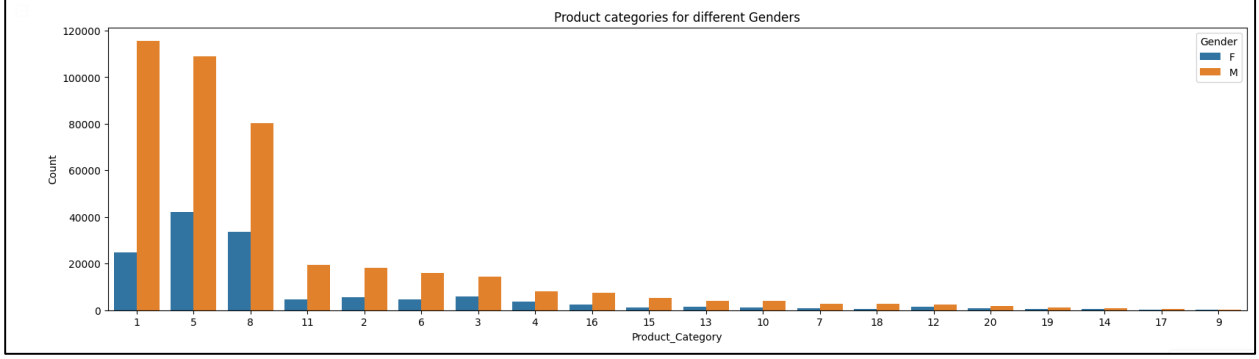
3. As there is low probability unmarried people according to age we can see a low number of unmarried customers over married customers as age increase after 45. Which clearly depicts that as age increased the unmarried Walmart's customers are decreased.

c. Are there preferred product categories for different genders?

```
[189] df5 = df[['Gender', 'Product_Category']]
      df5['Product_Category'] = df5['Product_Category'].astype('str')
      df_grp = df5.groupby(['Gender', 'Product_Category'])
      df6 = df_grp.value_counts().reset_index()
```

```
[203] df6.rename(columns = {0:'Count'}, inplace =True)
      df6.sort_values(by = 'Count', ascending = False, inplace = True)
```

```
[205] plt.figure(figsize = (20, 5))
      sns.barplot(data = df6, x = 'Product_Category', y = 'Count', hue = 'Gender')
      plt.title('Product categories for different Genders')
      plt.show()
```



Insights:

1. Product categories 1,5,8 all are popular among both male and female customers.

2. Category 5 is more popular in female group among rest of the products.

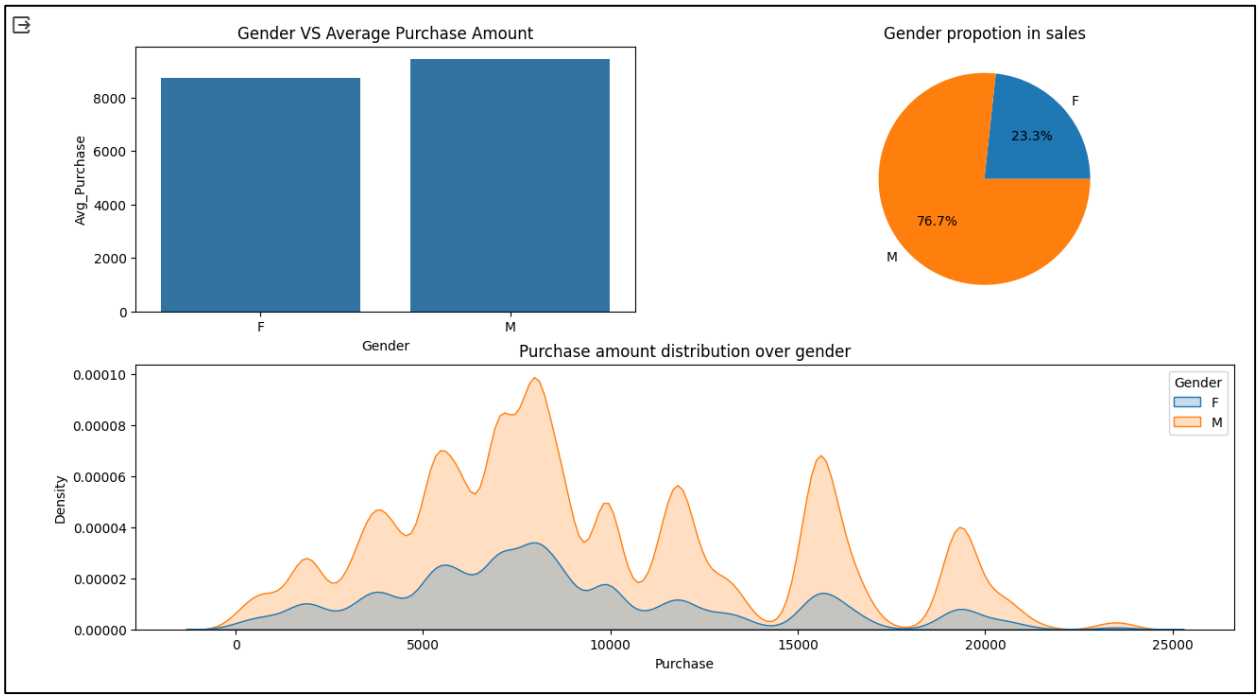
3. In all categories we can see the male customers purchase significantly more than female customers.

5. Gender VS Purchase Amount

```
[211] g_grp = df.groupby('Gender')['Purchase'].agg(['sum', 'count']).reset_index()
      g_grp['Avg_Purchase'] = g_grp['sum']/g_grp['count']
      g_grp
```

	Gender	sum	count	Avg_Purchase
0	F	1186232642	135809	8734.565765
1	M	3909580100	414259	9437.526040

```
plt.figure(figsize = (15,8))
plt.subplot(2,2,1)
sns.barplot(data = g_grp, x = 'Gender', y = 'Avg_Purchase' )
plt.title('Gender VS Average Purchase Amount')

plt.subplot(2,2,2)
plt.pie(g_grp['sum'],labels = g_grp['Gender'],autopct = '%.1f%%')
plt.title('Gender propotion in sales')

plt.subplot(2,1,2)
sns.kdeplot(data = df, x = 'Purchase', hue = 'Gender', fill = True)
plt.title('Purchase amount distribution over gender')

plt.show()
```



Insights:

1. The average purchase amount more male $9437.52 is slightly higher than female customers $8734.56.
2. In terms of sales 76.7% sales is done by male customer.
3. The purchase amount distribution for both male and female are not normally distributed.

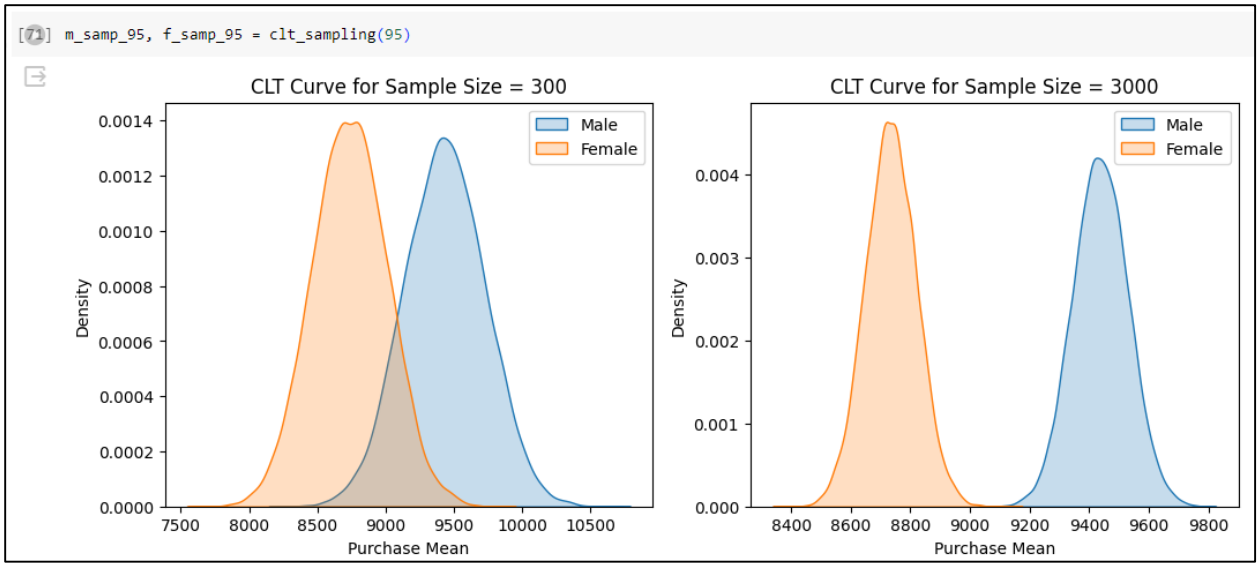
Construction of Confidence Interval:

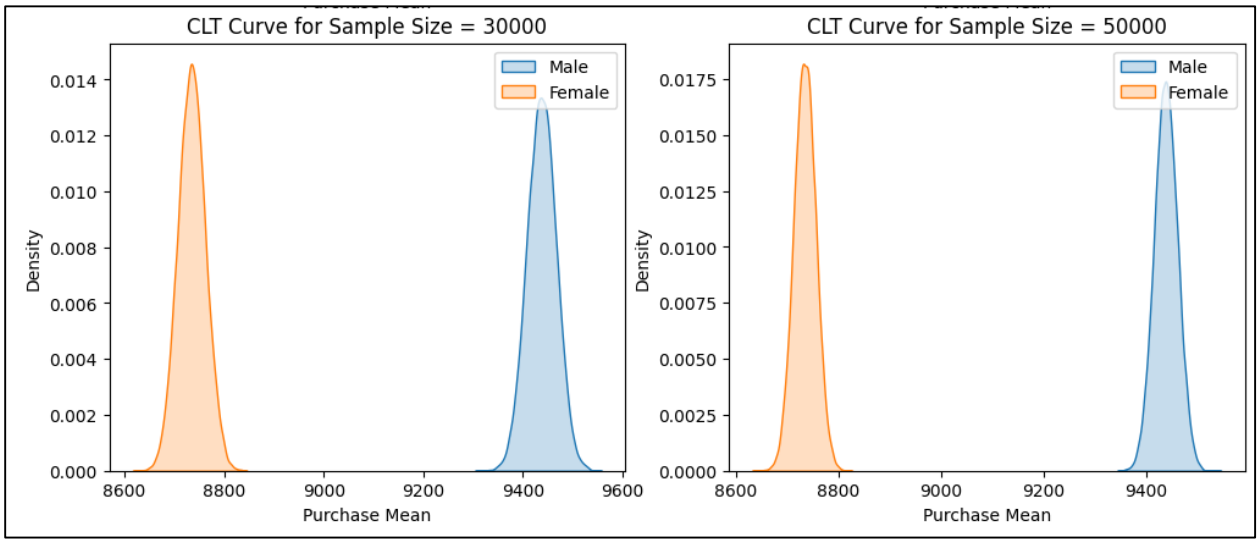
Step1: As we have seen from above the gender distribution over purchase is not following normal distribution so we need to use CLT (Central Limit Theorem).

Step2: After building the curve using CLT, we will create a confidence interval predicting population mean at 95%.

```
[46] def clt_sampling(ci):
       plt.figure(figsize = (12, 10))
       df_male = df.loc[df['Gender'] == 'M', 'Purchase']
       df_female = df.loc[df['Gender'] == 'F', 'Purchase']
       sample_size = [(300,1), (3000, 2),(30000, 3), (50000,4)]

       bootstarp_samples = 20000
       male_samples = {}
       female_samples = {}

       for i,k in sample_size:
         male_means = []
         female_means = []

         for j in range(bootstarp_samples):
           male_bootstrapped_samples = np.random.choice(df_male,size = i)
           female_bootstrapped_samples = np.random.choice(df_female,size = i)

           male_sample_mean = np.mean(male_bootstrapped_samples)
           female_sample_mean = np.mean(female_bootstrapped_samples)

           male_means.append(male_sample_mean)
           female_means.append(female_sample_mean)

         male_samples[f'{ci}%_{i}'] = male_means
         female_samples[f'{ci}%_{i}'] = female_means
         sam_df = pd.DataFrame(data = {'male_means':male_means,'female_means':female_means})

         plt.subplot(2,2,k)
         sns.kdeplot(data = sam_df,x = 'male_means',fill = True, label = 'Male')
         plt.xlabel('Purchase Mean')
         sns.kdeplot(data = sam_df,x = 'female_means',fill = True, label = 'Female')
         plt.xlabel('Purchase Mean')

         plt.title(f'CLT Curve for Sample Size = {i}')

       plt.show()
       return male_samples, female_samples
```

```
[71] m_samp_95, f_samp_95 = clt_sampling(95)
```






```
[48] def confidence_interval(data,ci):
         l_ci = (100-ci)/2
         u_ci = (100+ci)/2

         interval = np.percentile(data,[l_ci,u_ci]).round(0)

         return interval
```

```
[58] for i,j,k in [(m_samp_95,f_samp_95,95)]:
         m_ci = ['Male']
         f_ci = ['Female']

         for m in i:
             m_range = confidence_interval(i[m],k)
             m_ci.append(f"CI = ${m_range[0]:.0f} - ${m_range[1]:.0f}, Range = {(m_range[1] - m_range[0]):.0f}")

         for f in j:
             f_range = confidence_interval(j[f],k)
             f_ci.append(f"CI = ${f_range[0]:.0f} - ${f_range[1]:.0f}, Range = {(f_range[1] - f_range[0]):.0f}")

     cnt = 1
     for i in [300, 3000, 30000, 50000]:
       print(f'The CI range when choose {i} samples for Male: {m_ci[cnt]}')
       print(f'The CI range when choose {i} samples for Female: {f_ci[cnt]}')
       cnt += 1
       print('*'*80)
```

```
The CI range when choose 300 samples for Male: CI = $8867 - $10016, Range = 1149
The CI range when choose 300 samples for Female: CI = $8201 - $9285, Range = 1084
********************************************************************************
The CI range when choose 3000 samples for Male: CI = $9254 - $9620, Range = 366
The CI range when choose 3000 samples for Female: CI = $8565 - $8904, Range = 339
********************************************************************************
The CI range when choose 30000 samples for Male: CI = $9381 - $9496, Range = 115
The CI range when choose 30000 samples for Female: CI = $8681 - $8788, Range = 107
********************************************************************************
The CI range when choose 50000 samples for Male: CI = $9393 - $9483, Range = 90
The CI range when choose 50000 samples for Female: CI = $8692 - $8777, Range = 85
********************************************************************************
```

Insights:

1. Sample Size: Here we can see the importance of sample size. As sample size increases the we can see the confidence interval decreases or become narrower which is more precise. So, in business if sample size is more then analysis will be more precise and one can estimate the prediction more accurately.

2. Confidence Interval: From above all the 4 graphs we can say that except sample size 300 none of the other graphs are overlapped. Which means as sample size increases we can see a large gap in the spending behaviors among males and females. The 50000 sample graph clearly depicts that there is a significant gap between male and female purchase amount.

3. Population Mean: From the higher sample (50,000) graph we can say that the population mean will be sampling means as n (sample size) is high which leads to very minimum standard error. So the population mean lies between $9393 to $9483 for male and $8692 to $8777 for female.

Recommendation:

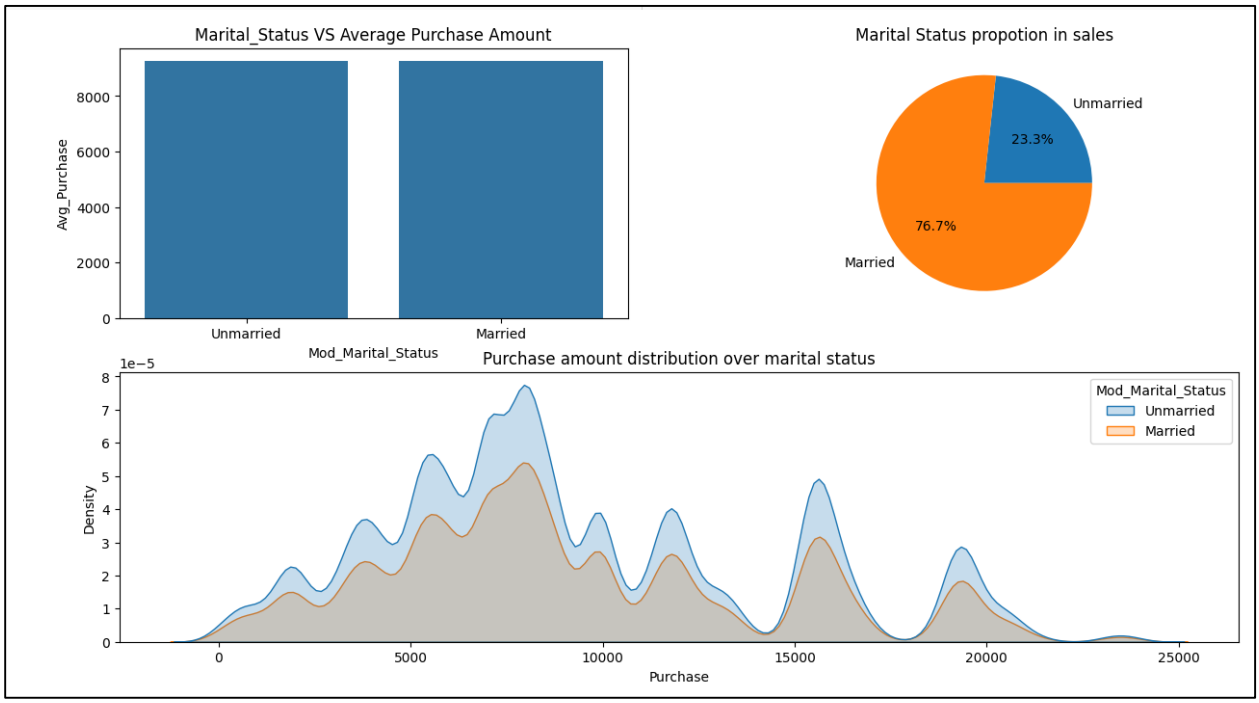
After all the above analysis we can see that the spending behaviors of males is higher than females. To overcome this situation Walmart can launch some campaigns, loyalty programs also can sell more gender specific products to attract male and females both. This can improve the revenue of the company.

6. Marital Status VS Purchase Amount

```
m_grp = df.groupby('Mod_Marital_Status')['Purchase'].agg(['sum', 'count']).reset_index()
m_grp['Avg_Purchase'] = m_grp['sum']/m_grp['count']
m_grp
```

	Mod_Marital_Status	sum	count	Avg_Purchase
0	Unmarried	3008927447	324731	9265.907619
1	Married	2086885295	225337	9261.174574

```
plt.figure(figsize = (15,8))
plt.subplot(2,2,1)
sns.barplot(data = m_grp, x = 'Mod_Marital_Status', y = 'Avg_Purchase' )
plt.title('Marital_Status VS Average Purchase Amount')

plt.subplot(2,2,2)
plt.pie(g_grp['sum'],labels = m_grp['Mod_Marital_Status'],autopct = '%.1f%%')
plt.title('Marital Status propotion in sales')

plt.subplot(2,1,2)
sns.kdeplot(data = df, x = 'Purchase', hue = 'Mod_Marital_Status', fill = True)
plt.title('Purchase amount distribution over marital status')

plt.show()
```



Insights:

1. More than 20% transaction is done by unmarried persons while more than 75% transaction done by married persons.

2. The average purchase amount is almost same for both the category.

3. The distribution graph for both married and unmarried doesn't follow the normal distribution.

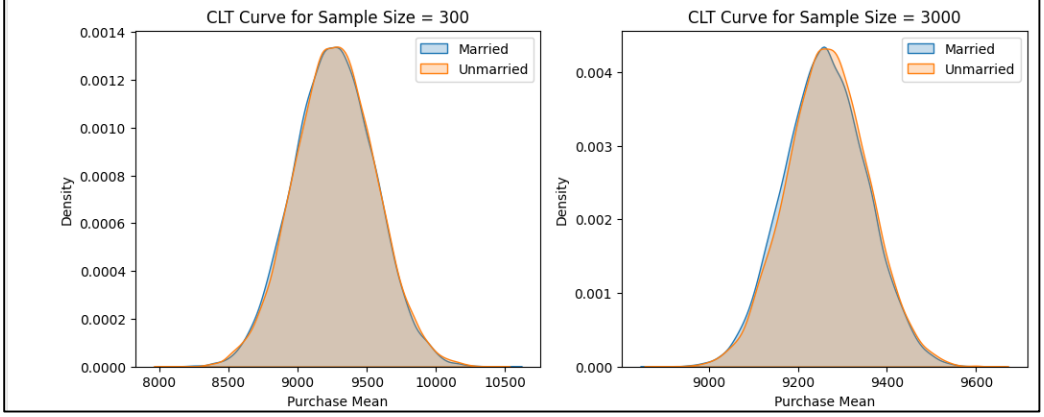
Construction of Confidence Interval:

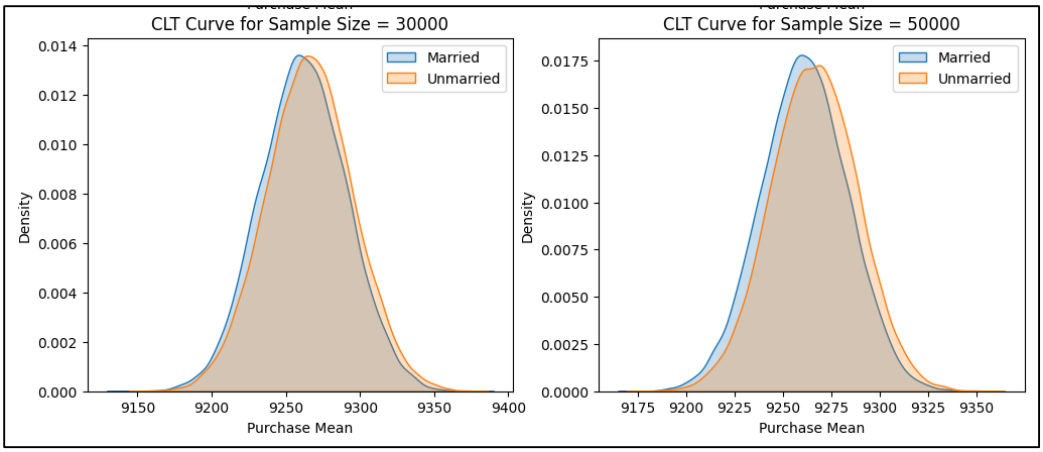
Step1: As we have seen from above the marital status distribution over purchase is not following normal distribution so we need to use CLT (Central Limit Theorem).

Step2: After building the curve using CLT, we will create a confidence interval predicting population mean at 95%.

```
def clt_sampling_marital(ci):
  plt.figure(figsize = (12, 10))
  df_married = df.loc[df['Mod_Marital_Status'] == 'Married', 'Purchase']
  df_unmarried = df.loc[df['Mod_Marital_Status'] == 'Unmarried', 'Purchase']
  sample_size = [(300,1), (3000, 2),(30000, 3), (50000,4)]

  bootstarp_samples = 20000
  married_samples = {}
  unmarried_samples = {}

  for i,k in sample_size:
    married_means = []
    unmarried_means = []

    for j in range(bootstarp_samples):
      married_bootstrapped_samples = np.random.choice(df_married,size = i)
      unmarried_bootstrapped_samples = np.random.choice(df_unmarried,size = i)

      married_sample_mean = np.mean(married_bootstrapped_samples)
      unmarried_sample_mean = np.mean(unmarried_bootstrapped_samples)

      married_means.append(married_sample_mean)
      unmarried_means.append(unmarried_sample_mean)

    married_samples[f'{ci}%_{i}'] = married_means
    unmarried_samples[f'{ci}%_{i}'] = unmarried_means
    sam_df = pd.DataFrame(data = {'married_means':married_means,'unmarried_means':unmarried_means})

    plt.subplot(2,2,k)
    sns.kdeplot(data = sam_df,x = 'married_means',fill = True, label = 'Married')
    plt.xlabel('Purchase Mean')
    sns.kdeplot(data = sam_df,x = 'unmarried_means',fill = True, label = 'Unmarried')
    plt.xlabel('Purchase Mean')

    plt.title(f'CLT Curve for Sample Size = {i}')
    plt.legend()

  plt.show()
  return married_samples, unmarried_samples
```

```
m_samp_95,u_samp_95 = clt_sampling_marital(95)
```






```
[75] for i,j,k in [(m_samp_95,u_samp_95,95)]:
         m_ci = ['Married']
         f_ci = ['Unmarried']

         for m in i:
             m_range = confidence_interval(i[m],k)
             m_ci.append(f"CI = ${m_range[0]:.0f} - ${m_range[1]:.0f}, Range = {(m_range[1] - m_range[0]):.0f}")

         for f in j:
             f_range = confidence_interval(j[f],k)
             f_ci.append(f"CI = ${f_range[0]:.0f} - ${f_range[1]:.0f}, Range = {(f_range[1] - f_range[0]):.0f}")

     cnt = 1
     for i in [300, 3000, 30000, 50000]:
       print(f'The CI range when choose {i} samples for Married: {m_ci[cnt]}')
       print(f'The CI range when choose {i} samples for Unmarried: {f_ci[cnt]}')
       cnt += 1
       print('*'*80)
```

```
The CI range when choose 300 samples for Married: CI = $8700 - $9829, Range = 1129
The CI range when choose 300 samples for Unmarried: CI = $8704 - $9843, Range = 1139
********************************************************************************
The CI range when choose 3000 samples for Married: CI = $9083 - $9442, Range = 359
The CI range when choose 3000 samples for Unmarried: CI = $9089 - $9447, Range = 358
********************************************************************************
The CI range when choose 30000 samples for Married: CI = $9205 - $9319, Range = 114
The CI range when choose 30000 samples for Unmarried: CI = $9209 - $9324, Range = 115
********************************************************************************
The CI range when choose 50000 samples for Married: CI = $9217 - $9304, Range = 87
The CI range when choose 50000 samples for Unmarried: CI = $9222 - $9310, Range = 88
********************************************************************************
```

Insights:

1. Sample Size: Here we can see the importance of sample size. As sample size increases the we can see the confidence interval decreases or become narrower which is more precise. So, in business if sample size is more then analysis will be more precise and one can estimate the prediction more accurately.

2. Confidence Interval: From above all the 4 graphs we can say that all of the other graphs are overlapped. Which means there is no significance difference between average spending behavior among married and unmarried customers.

3. Population Mean: From the higher sample (50,000) graph we can say that the population mean will be sampling means as n (sample size) is high which leads to very minimum standard error. So the population mean lies between $9217 to $9304 for married and $9222 to $9310 for unmarried.

4. Both type spend same: From all the overlapping graphs implies that there is no difference in the spending behavior between married and unmarried per transaction.

Recommendation:

Here by judging the spending behavior for both the married and unmarried we can say there is no need of any specific changes from Walmart side, instead of that Walmart can increase their sales by focusing both the groups.

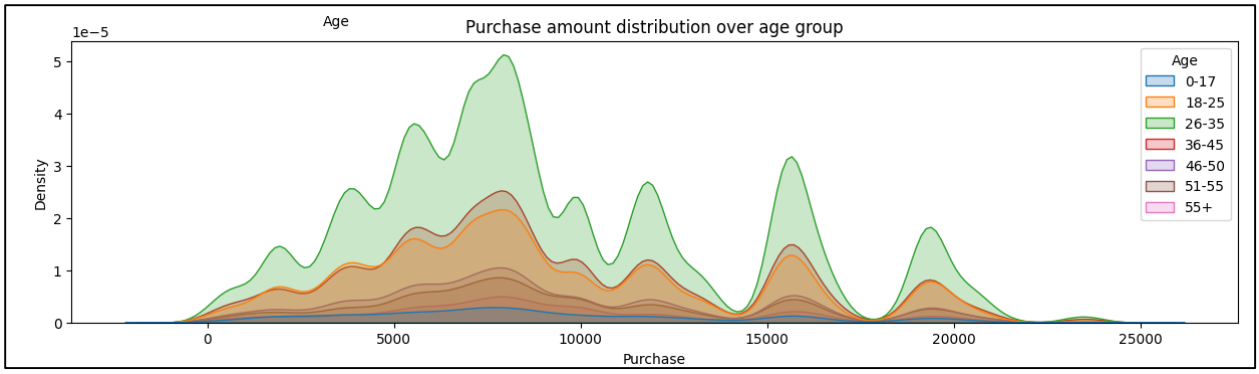
7. Age VS Purchase Amount

AGE VS Customer Purchase

```
[76] a_grp = df.groupby('Age')['Purchase'].agg(['sum', 'count']).reset_index()
     a_grp['Avg_Purchase'] = a_grp['sum']/a_grp['count']
     a_grp
```

	Age	sum	count	Avg_Purchase
0	0-17	134913183	15102	8933.464640
1	18-25	913848675	99660	9169.663606
2	26-35	2031770578	219587	9252.690633
3	36-45	1026569884	110013	9331.350695
4	46-50	420843403	45701	9208.625697
5	51-55	367099644	38501	9534.808031
6	55+	200767375	21504	9336.280459

```
[77] plt.figure(figsize = (15,8))
     plt.subplot(2,2,1)
     sns.barplot(data = a_grp, x = 'Age', y = 'Avg_Purchase' )
     plt.title('Age Group VS Average Purchase Amount')

     plt.subplot(2,2,2)
     plt.pie(a_grp['sum'],labels = a_grp['Age'],autopct = '%.1f%%')
     plt.title('Age Group propotion in sales')

     plt.subplot(2,1,2)
     sns.kdeplot(data = df, x = 'Purchase', hue = 'Age', fill = True)
     plt.title('Purchase amount distribution over age group')

     plt.show()
```



Insights:

1. Age group 26-35 are the valuable customers as they contribute the 40% sales in Black Friday. Another 40% contribution is done by age group 18-25 and 36-45.

2. 0-17 age group customers only contribute 2.6% of the sales which is expected as per the income. The elderly customers are also contributing less in sales.

3. Though 51-55 age group contributing 7.2% in the sales but the average purchasing power is higher among all the groups, which can be a best point for Walmart to look upon, company can improve their strategies to attract these high values customers.

4. The distribution curve is not normally distributed among all the age groups.

Construction of Confidence Interval:

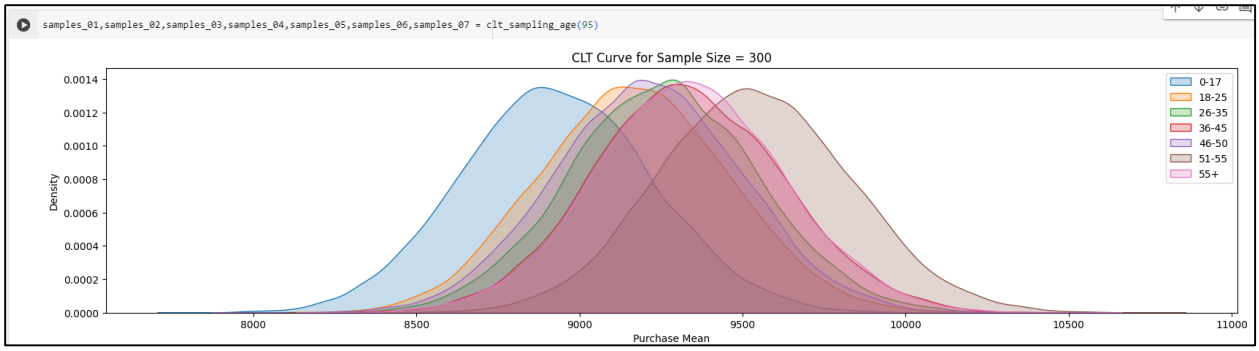
Step1: As we have seen from above the marital status distribution over purchase is not following normal distribution so we need to use CLT (Central Limit Theorem).

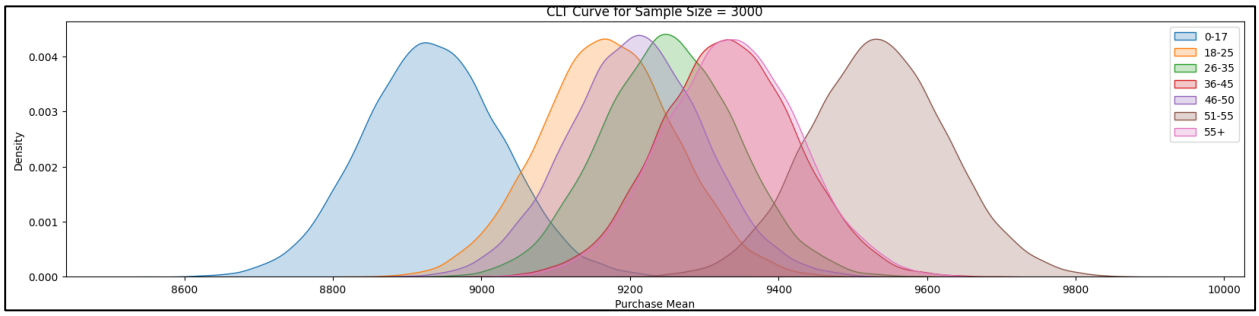
Step2: After building the curve using CLT, we will create a confidence interval predicting population mean at 95%.

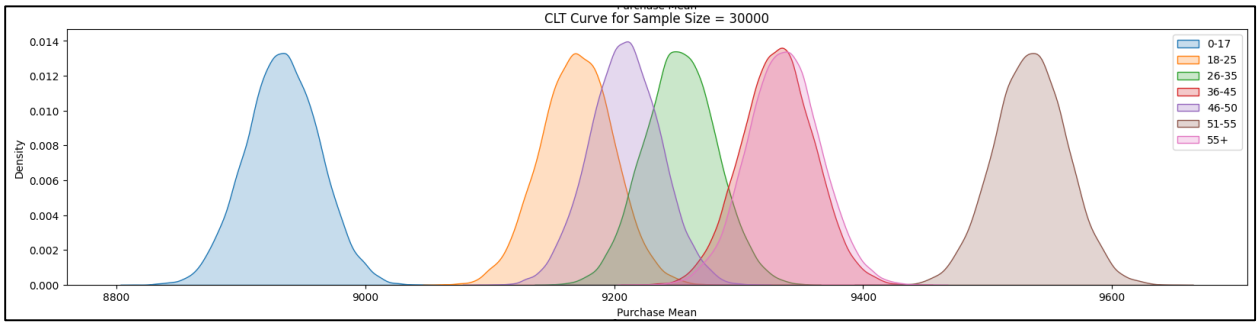
```
[83] def clt_sampling_age(ci):
       plt.figure(figsize = (20, 20))
       df_01= df.loc[df['Age'] == '0-17', 'Purchase']
       df_02= df.loc[df['Age'] == '18-25', 'Purchase']
       df_03= df.loc[df['Age'] == '26-35', 'Purchase']
       df_04= df.loc[df['Age'] == '36-45', 'Purchase']
       df_05= df.loc[df['Age'] == '46-50', 'Purchase']
       df_06= df.loc[df['Age'] == '51-55', 'Purchase']
       df_07= df.loc[df['Age'] == '55+', 'Purchase']

       sample_size = [(300,1), (3000, 2),(30000, 3), (50000,4)]

       bootstarp_samples = 20000
       samples_01, samples_02, samples_03, samples_04, samples_05, samples_06, samples_07  = {}, {}, {}, {}, {}, {}, {}

       for i,k in sample_size:
         l_01, l_02, l_03, l_04, l_05, l_06, l_07 = [], [], [], [], [], [], []

         for j in range(bootstarp_samples):
           bootstrapped_samples_01 = np.random.choice(df_01,size = i)
           bootstrapped_samples_02 = np.random.choice(df_02,size = i)
           bootstrapped_samples_03 = np.random.choice(df_03,size = i)
           bootstrapped_samples_04 = np.random.choice(df_04,size = i)
           bootstrapped_samples_05 = np.random.choice(df_05,size = i)
           bootstrapped_samples_06 = np.random.choice(df_06,size = i)
           bootstrapped_samples_07 = np.random.choice(df_07,size = i)

           sample_mean_01 = np.mean(bootstrapped_samples_01)
           sample_mean_02 = np.mean(bootstrapped_samples_02)
           sample_mean_03 = np.mean(bootstrapped_samples_03)
           sample_mean_04 = np.mean(bootstrapped_samples_04)
           sample_mean_05 = np.mean(bootstrapped_samples_05)
           sample_mean_06 = np.mean(bootstrapped_samples_06)
           sample_mean_07 = np.mean(bootstrapped_samples_07)

           l_01.append(sample_mean_01)
           l_02.append(sample_mean_02)
           l_03.append(sample_mean_03)
           l_04.append(sample_mean_04)
           l_05.append(sample_mean_05)
           l_06.append(sample_mean_06)
           l_07.append(sample_mean_07)
```

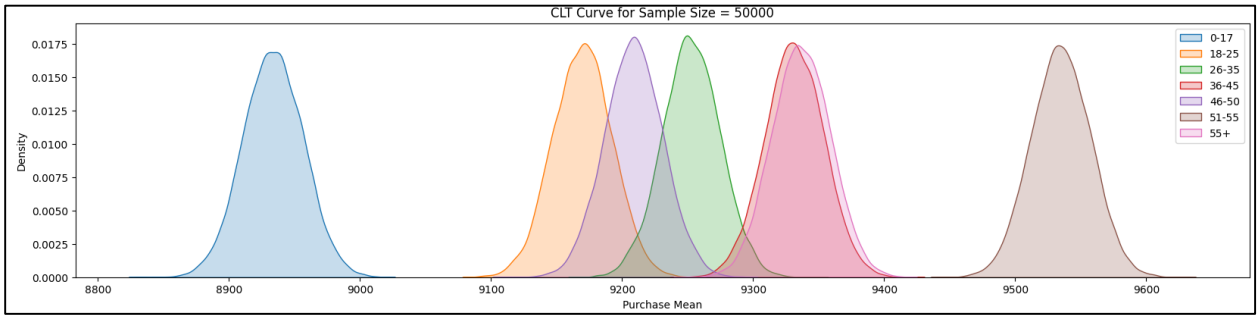
```
       samples_01[f'{ci}%_{i}'] = l_01
       samples_02[f'{ci}%_{i}'] = l_02
       samples_03[f'{ci}%_{i}'] = l_03
       samples_04[f'{ci}%_{i}'] = l_04
       samples_05[f'{ci}%_{i}'] = l_05
       samples_06[f'{ci}%_{i}'] = l_06
       samples_07[f'{ci}%_{i}'] = l_07

       sam_df = pd.DataFrame(data = {'0-17':l_01,'18-25':l_02, '26-35': l_03, '36-45' : l_04, '46-50' : l_05, '51-55' : l_06, '55+' : l_07})

       plt.subplot(4,1,k)
       sns.kdeplot(data = sam_df,x = '0-17',fill = True, label = '0-17')
       plt.xlabel('Purchase Mean')
       sns.kdeplot(data = sam_df,x = '18-25',fill = True, label = '18-25')
       plt.xlabel('Purchase Mean')
       sns.kdeplot(data = sam_df,x = '26-35',fill = True, label = '26-35')
       plt.xlabel('Purchase Mean')
       sns.kdeplot(data = sam_df,x = '36-45',fill = True, label = '36-45')
       plt.xlabel('Purchase Mean')
       sns.kdeplot(data = sam_df,x = '46-50',fill = True, label = '46-50')
       plt.xlabel('Purchase Mean')
       sns.kdeplot(data = sam_df,x = '51-55',fill = True, label = '51-55')
       plt.xlabel('Purchase Mean')
       sns.kdeplot(data = sam_df,x = '55+',fill = True, label = '55+')
       plt.xlabel('Purchase Mean')

       plt.title(f'CLT Curve for Sample Size = {i}')
       plt.legend()

     plt.show()
     return samples_01, samples_02, samples_03, samples_04,samples_05, samples_06, samples_07
```

```
samples_01,samples_02,samples_03,samples_04,samples_05,samples_06,samples_07 = clt_sampling_age(95)
```










```
[87] ci_1,ci_2,ci_3,ci_4,ci_5,ci_6,ci_7 = ['0-17'],['18-25'],['26-35'],['36-45'],['46-50'],['51-55'],['55+']

     samples = [(samples_01,ci_1),(samples_02,ci_2),(samples_03,ci_3),(samples_04,ci_4),(samples_05,ci_5),(samples_06,ci_6),(samples_07,ci_7)]

     for s,c in samples:
         for i in s:
             s_range = confidence_interval(s[i],95)
             c.append(f"CI = ${s_range[0]:.0f} - ${s_range[1]:.0f}, Range = {(s_range[1] - s_range[0]):.0f}")

     cnt = 1
     for i in [300, 3000, 30000, 50000]:
       print(f'The CI range when choose {i} samples for 0-17: {ci_1[cnt]}')
       print(f'The CI range when choose {i} samples for 18-25: {ci_2[cnt]}')
       print(f'The CI range when choose {i} samples for 26-35: {ci_3[cnt]}')
       print(f'The CI range when choose {i} samples for 36-45: {ci_4[cnt]}')
       print(f'The CI range when choose {i} samples for 46-50: {ci_5[cnt]}')
       print(f'The CI range when choose {i} samples for 51-55: {ci_6[cnt]}')
       print(f'The CI range when choose {i} samples for 55+: {ci_7[cnt]}')
       cnt += 1
       print('*'*80)
```

```
The CI range when choose 300 samples for 0-17: CI = $8367 - $9514, Range = 1147
The CI range when choose 300 samples for 18-25: CI = $8608 - $9739, Range = 1131
The CI range when choose 300 samples for 26-35: CI = $8695 - $9826, Range = 1131
The CI range when choose 300 samples for 36-45: CI = $8768 - $9911, Range = 1143
The CI range when choose 300 samples for 46-50: CI = $8651 - $9775, Range = 1124
The CI range when choose 300 samples for 51-55: CI = $8967 - $10121, Range = 1154
The CI range when choose 300 samples for 55+: CI = $8769 - $9912, Range = 1143
**********************************************************************************
The CI range when choose 3000 samples for 0-17: CI = $8752 - $9115, Range = 363
The CI range when choose 3000 samples for 18-25: CI = $8991 - $9350, Range = 359
The CI range when choose 3000 samples for 26-35: CI = $9076 - $9434, Range = 358
The CI range when choose 3000 samples for 36-45: CI = $9151 - $9512, Range = 361
The CI range when choose 3000 samples for 46-50: CI = $9029 - $9389, Range = 360
The CI range when choose 3000 samples for 51-55: CI = $9354 - $9718, Range = 364
The CI range when choose 3000 samples for 55+: CI = $9158 - $9516, Range = 358
**********************************************************************************
The CI range when choose 30000 samples for 0-17: CI = $8875 - $8992, Range = 117
The CI range when choose 30000 samples for 18-25: CI = $9113 - $9227, Range = 114
The CI range when choose 30000 samples for 26-35: CI = $9196 - $9310, Range = 114
The CI range when choose 30000 samples for 36-45: CI = $9274 - $9388, Range = 114
The CI range when choose 30000 samples for 46-50: CI = $9153 - $9264, Range = 111
The CI range when choose 30000 samples for 51-55: CI = $9477 - $9592, Range = 115
The CI range when choose 30000 samples for 55+: CI = $9279 - $9394, Range = 115
**********************************************************************************
The CI range when choose 50000 samples for 0-17: CI = $8889 - $8979, Range = 90
The CI range when choose 50000 samples for 18-25: CI = $9126 - $9213, Range = 87
The CI range when choose 50000 samples for 26-35: CI = $9209 - $9297, Range = 88
The CI range when choose 50000 samples for 36-45: CI = $9287 - $9375, Range = 88
The CI range when choose 50000 samples for 46-50: CI = $9165 - $9253, Range = 88
The CI range when choose 50000 samples for 51-55: CI = $9491 - $9579, Range = 88
The CI range when choose 50000 samples for 55+: CI = $9293 - $9380, Range = 87
**********************************************************************************
```

Insights:

1. Sample Size: Here we can see the importance of sample size. As sample size increases the we can see the confidence interval decreases or become narrower which is more precise. So, in business if sample size is more then analysis will be more precise and one can estimate the prediction more accurately.

2. Confidence Interval: From above all the 4 graphs we can say that some of the age groups are overlapped. By combining we can conclude as below:

i. Age group 0-17 has the lowest spending per transaction.

ii. Age group 18-25, 26-35, 46-50 are overlapping and they have same kind of purchasing characteristics.

Iii. Age group 36-45 and 55+ are overlapped depicts same kind of spending pattern.

Iv. Age group 50-55 has the highest average purchasing power.

3. Population Mean: From the higher sample (50,000) graph we can say that the population mean will be sampling means as n (sample size) is high which leads to very minimum standard error. So the population mean lies as follows,

a. 0-17: $8889 - $8979

b. 18-25: $9126 - $9213

c. 26-35: $9209 - $9297

d. 36-45: $9287 - $9375

e. 46-50: $9165 - $9253

f. 51-55: $9491 - $9579

g. 55+: $9293 - $9380

8. Recommendations:

1. Male Shoppers:

From all above analysis we have seen that the male customers among all the age groups leads Walmart to high sales. So Walmart should strategies to increase these sales among male's customers while creating a competitive environment for female customers also. Walmart should enrich their stocks with variety of products with promotions which attract both male and female customers.

2. Focus on 26-35 Age group:

From all above analysis the Black Friday sales is kind of festival for age group 26-35 because they bring the highest revenue by purchasing the items. Walmart should encourage this age group by giving attractive offers on purchase range.

3. Age group 0-17:

We have seen that the age group 0-17 contributing lowest in the sales. Which somewhat expected as well. But here Walmart have a great chance of generating revenue. Walmart can start campaign for student and give attractive discounts to this age group.

4. Engaging elderly customers:

We have seen that age group 50-55 leads highest average transaction. So, Walmart needs to focus this age group to generate more revenue, by doing

loyalty programs, attractive coupons, home delivery, different queue system for elders.

5. Loyalty for old customers:

While analyzing we have seen that the purchasing pattern decreases among the years of spending in a city. Walmart should focus in this section. Company needs to give attention to retain their old customers. By campaigning, loyalty programs Walmart can attract the old customers.

6. Sales after Black Friday:

Though the above analysis is sales on Black Friday but Walmart should focus on their daily sales as well after Black Friday Sales. By which one customer not only visit during offer season also revisit throughout the years.