WALMART CUSTOMER PURCHASE BEHAVIOUR ANALYSIS

BUSINESS PROBLEM :

Walmart is an American multinational retail corporation that operates a chain of supercenters, discount departmental stores, and grocery stores in the United States. Walmart has more than 100 million customers worldwide.
The Management team at Walmart Inc. wants to analyze the customer purchase behavior (precisely, 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?

Basic Metrics :

```
1 data.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
```

```
1 data.shape
```

```
(550068, 10)
```

```
1 data.isna().sum()
```

```
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
```

- Dataset has **550068** records with **10** features for each record as it can be seen in the shape of data *(550068,10)*.
- There are no missing values in any of the columns.We have **five** columns with *object* datatype and **five** columns with *integer* datatype.

```
1 data['User_ID'].nunique()
```

```
5891
```

```
1 data['Gender'].value_counts()
```

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

```
1 male_data=data[data['Gender']=='M']
2 male_data['User_ID'].nunique()
```

```
4225
```

```
1 female_data=data[data['Gender']=='F']
2 female_data['User_ID'].nunique()
```

```
1666
```

- Ratio of Male to Female Customer purchases is close to **3:1**.
- There are **5891** unique users who have made purchases on Black Friday.
- There are **4225** Unique Male Customers and **1666** Unique Female Customers on Black Friday Sale.
- **414259** purchases have been made by **4225** Male customers which is **98** purchases on an average by a Male Customer.
- **135809** purchases have been made by **1666** Female Customers which is **82** purchases on an average by a Female Customer.

```
1 data['Product_ID'].value_counts().head(5)
```

```
P00265242    1880
P00025442    1615
P00110742    1612
P00112142    1562
P00057642    1470
Name: Product_ID, dtype: int64
```

- The above table shows the **top 5** most frequently bought products among the customers.

```
1 data['Product_Category'].value_counts()
```

```
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
Name: Product_Category, dtype: int64
```

• There 20 different product categories among which category 5, 1, 8 are the most sought after product categories.

```
1 data.describe()
```

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

- data.describe() gives a descriptive Statistical analysis on the continuous data.

This gives us the range of values in each numerical columns, the mean of the data, different percentile values , median and the standard deviation.

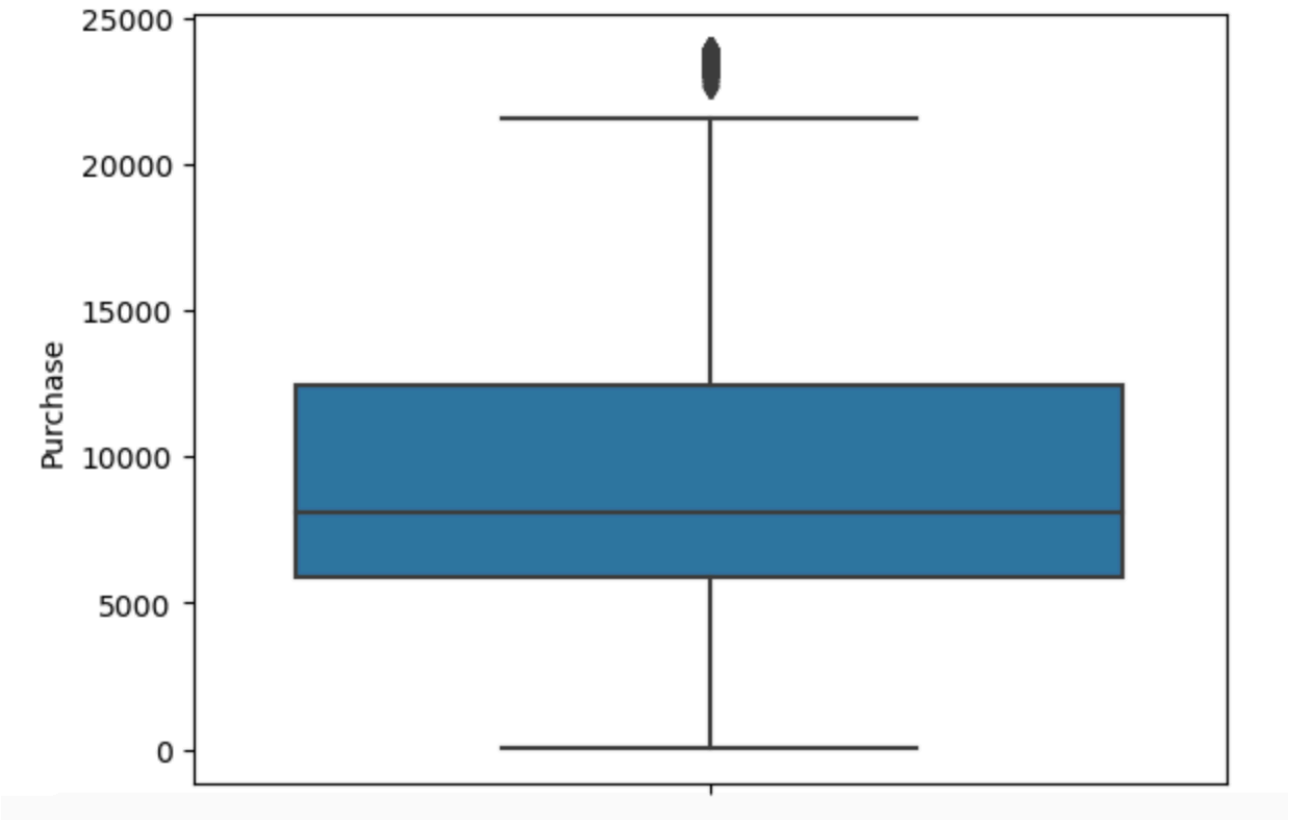
- Mean of the occupation years of customers is nearly 8 years with the lowest being 0 and the highest being 20 years of occupation.
- Mean Purchase value is 9263 with lowest being 12 and highest being 23961.
- 50 percentile of purchases are below 8047 and 50 percentile of customers have been working for less than 7 years.

```
1 male_data.describe()
```

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	4.142590e+05	414259.00000	414259.000000	414259.000000	414259.00000
mean	1.002996e+06	8.51475	0.406386	5.301512	9437.52604
std	1.706494e+03	6.55379	0.491159	4.006275	5092.18621
min	1.000002e+06	0.00000	0.000000	1.000000	12.00000
25%	1.001505e+06	3.00000	0.000000	1.000000	5863.00000
50%	1.003041e+06	7.00000	0.000000	5.000000	8098.00000
75%	1.004411e+06	15.00000	1.000000	8.000000	12454.00000
max	1.006040e+06	20.00000	1.000000	20.000000	23961.00000

- Purchase amount per transaction for male customers,
    - Actual Minimum : 12
    - Actual Maximum : 23961
    - 25th Percentile : 5863 (Q1)
    - 50th Percentile : 8098 (Q2)
    - 75th Percentile : 12454 (Q3)
    - IQR=Q3-Q1 => 6595

- Minimum Considered : Q1-1.5*IQR or Actual Minimum.Here Actual Minimum is considered.



- Maximum Considered : Q3+1.5*IQR or Actual Maximum.
- Q3 + 1.5*IQR => 22346.5
- Any Purchase Amount above 22346.5 is considered outliers in this data.
- Out of 414259 Purchases by male customers, 1812 purchase amounts falls in Outliers.

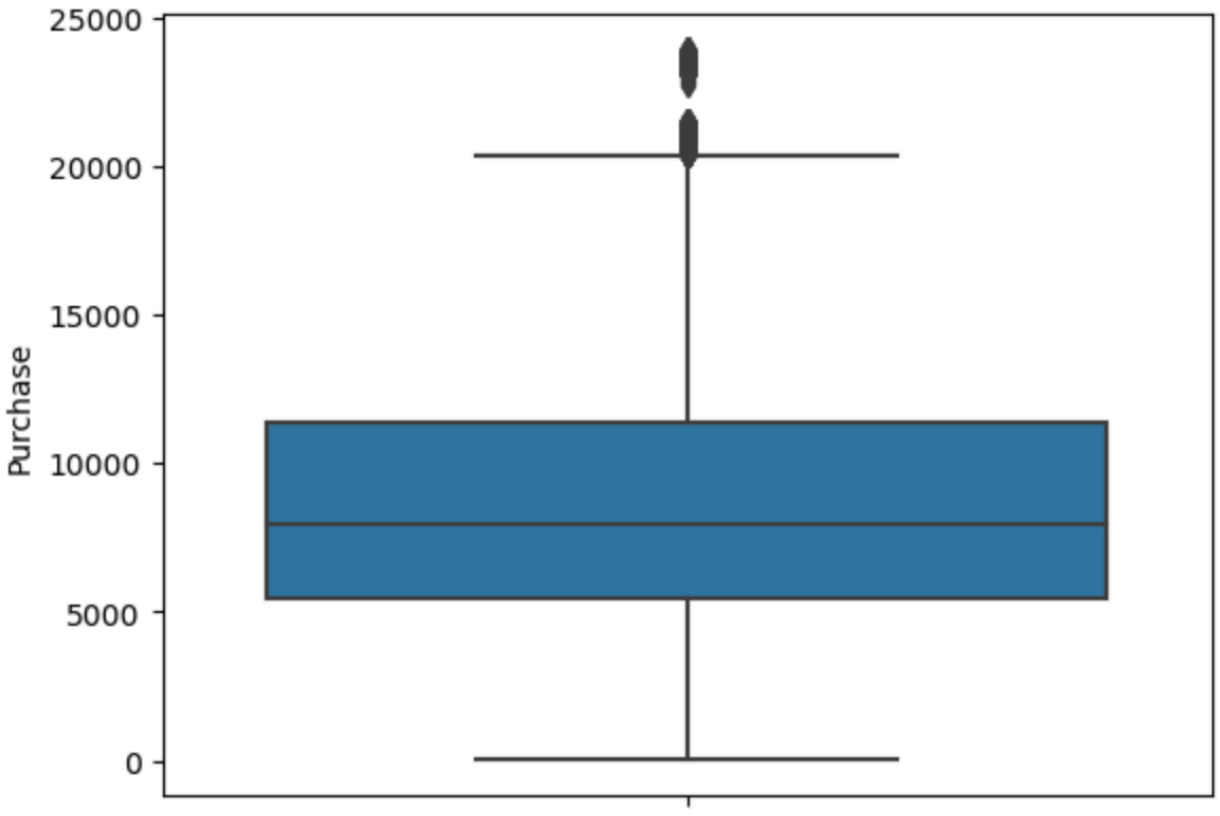
```
1 (male_data['Purchase']>22346.5).sum()
```

```
1812
```

```
1 female_data.describe()
```

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	1.358090e+05	135809.000000	135809.000000	135809.000000	135809.000000
mean	1.003130e+06	6.740540	0.419619	5.717714	8734.565765
std	1.786631e+03	6.239639	0.493498	3.696752	4767.233289
min	1.000001e+06	0.000000	0.000000	1.000000	12.000000
25%	1.001569e+06	1.000000	0.000000	3.000000	5433.000000
50%	1.003159e+06	4.000000	0.000000	5.000000	7914.000000
75%	1.004765e+06	11.000000	1.000000	8.000000	11400.000000
max	1.006039e+06	20.000000	1.000000	20.000000	23959.000000

- Purchase amount per transaction for female customers,
    - Actual Minimum : 12
    - Actual Maximum : 23959
    - 25th Percentile : 5833 (Q1)
    - 50th Percentile : 7914 (Q2)



- 75th Percentile : 11400 (Q3)
- IQR=Q3-Q1 => 5567
- Minimum Considered : Q1-1.5*IQR or Actual Minimum.Here Actual Minimum is considered.
- Maximum Considered : Q3+1.5*IQR or Actual Maximum.
- Q3 + 1.5*IQR => 19750.5
- Any Purchase Amount above 19750.5 is considered outliers in this data.
- Out of 135809 Purchases by female customers, 2997 purchase amounts falls in Outliers.

```
1 male_data['Purchase'].mean()
```

```
9437.526040472265
```

```
1 male_data['Purchase'].std()
```

```
5092.186209777949
```

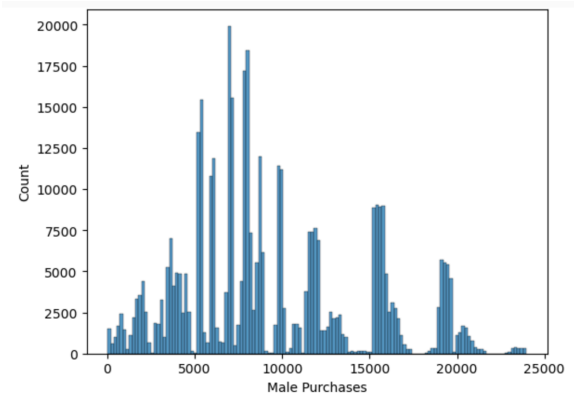
```
1 female_data['Purchase'].mean()
```

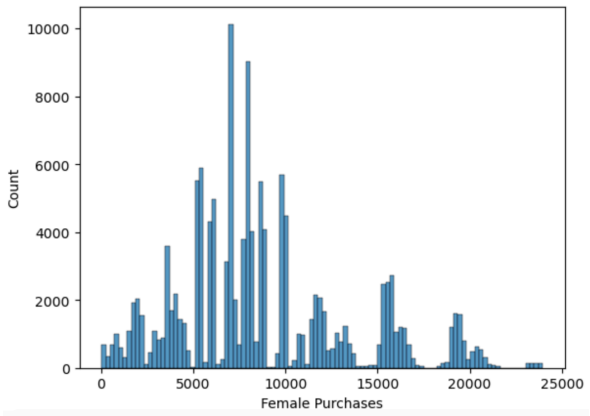
```
8734.565765155476
```

```
1 female_data['Purchase'].std()
```

```
4767.233289291444
```

- Population Mean Male= 9438
- Population Standard Deviation Male = 5092
- Population Mean Female= 8734
- Population Standard Deviation Female = 4767





- The Distribution of the population male & female Purchase Amounts is not normal.

Let us take 1000 samples with a sample size of 30.

```
sample_means_male=[]
sample_means_female=[]
```

```
for i in range(1000):
    sample_m=np.random.choice(male_data['Purchase'],size=30)
    sample_means_male.append(np.mean(sample_m))
    sample_f=np.random.choice(female_data['Purchase'],size=30)
    sample_means_female.append(np.mean(sample_f))
```

```
np.mean(sample_means_male)
```

```
9432.439522222223
```

```
np.mean(sample_means_female)
```

```
8740.699822222221
```

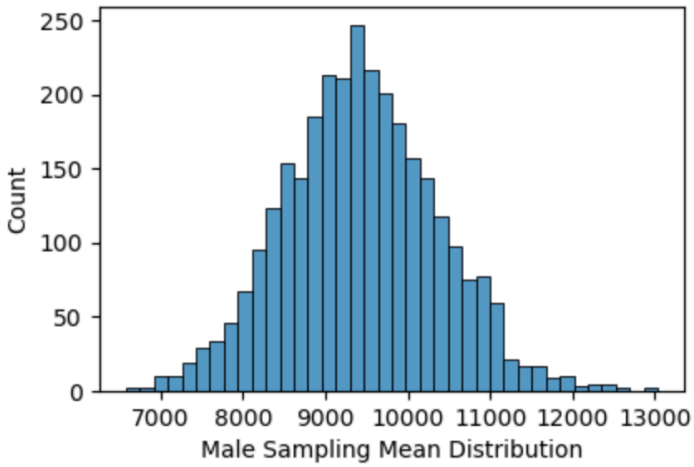
```
np.std(sample_means_male)/np.sqrt(30)
```

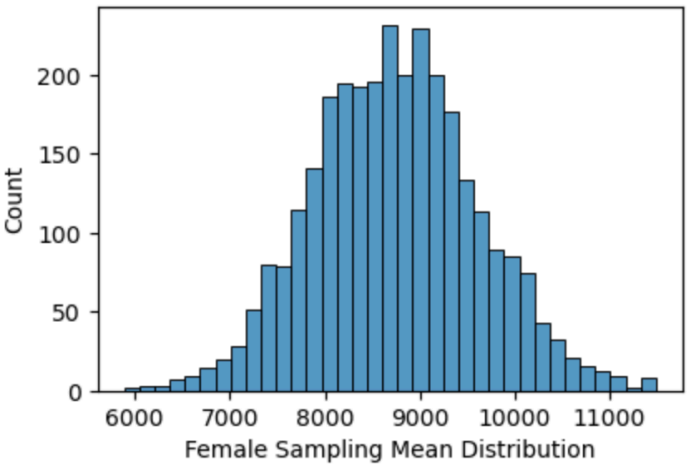
```
170.53739954877656
```

```
np.std(sample_means_female)/np.sqrt(30)
```

```
158.2194811800615
```

- Male Sample Mean : 9433 -> According to Central Limit Theorem, the Sampling distribution is a true estimate of the Population Mean.
- Male Standard Error : 170.53 -> Sample_means_male.std()/ np.sqrt(sample size).
- Female Sample Mean : 8741 -> According to Central Limit Theorem, the Sampling distribution is a true estimate of the Population Mean.
- Female Standard Error : 158.22 -> Sample_means_female.std()/ np.sqrt(sample size).





- The distribution of Sampling means is a Normal Distribution.
- Now we can estimate Confidence Interval at 90%, 95% and 99%.

```
# Calculating Confidence Interval At 90%.
```

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

```
norm(loc=np.mean(sample_means_male),scale=np.std(sample_means_male)/np.sqrt(30)).interval(0.90)
```

```
(9151.930462043545, 9712.9485824009)
```

```
# Calculating Confidence Interval At 95%.
```

```
norm(loc=np.mean(sample_means_male),scale=np.std(sample_means_male)/np.sqrt(30)).interval(0.95)
```

```
(9098.192361089503, 9766.686683354943)
```

```
# Calculating Confidence Interval At 99%.
```

```
norm(loc=np.mean(sample_means_male),scale=np.std(sample_means_male)/np.sqrt(30)).interval(0.99)
```

```
(8993.164291113457, 9871.714753330989)
```

- Confidence Interval of the Purchase Amounts of Male Customer Purchases :
    - 90% Confidence : [9151.93 to 9712.94]
    - 95% Confidence : [9098.19 to 9766.69]
    - 99% Confidence : [8993.16 to 9871.71]

- While taking 100 samples with a sample size of 30, the sampling means follows a normal distribution and 99% of the means of the samples lies between 8993.16 to 9871.71

- Only 1% of Sample means lies outside this range. This Can be similarly inferred for 90% and 95% Confidence.

```
# Calculating Confidence Interval At 90%.
```

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

```
norm(loc=np.mean(sample_means_female),scale=np.std(sample_means_female)/np.sqrt(30)).interval(0.90)
```

```
(8480.451934748817, 9000.947709695625)
```

```
# Calculating Confidence Interval At 95%.
```

```
norm(loc=np.mean(sample_means_female),scale=np.std(sample_means_female)/np.sqrt(30)).interval(0.95)
```

```
(8430.595337456687, 9050.804306987755)
```

```
# Calculating Confidence Interval At 99%.
```

```
norm(loc=np.mean(sample_means_female),scale=np.std(sample_means_female)/np.sqrt(30)).interval(0.99)
```

```
(8333.153446206315, 9148.246198238126)
```

- Confidence Interval of the Purchase Amounts of Female Customer Purchases :
    - 90% Confidence : [8480.45 to 9000.94]

- 95% Confidence : [8430.59 to 9050.80]
- 99% Confidence : [8333.15 to 9148.24]

- While taking 100 samples with a sample size of 30, the sampling means follows a normal distribution and 99% of the means of the samples lies between 8333.15 to 9148.24

- Only 1% of Sample means lies outside this range. This Can be similarly inferred for 90% and 95% Confidence.

Increasing the Sample size from 30 to 60 or 100 will give an even accurate estimate of Population Parameters and the Confidence Interval.

Let us take 1000 samples with a sample size of 100.

```
sample_means_male=[]
sample_means_female=[]
```

```
for i in range(1000):
    sample_m=np.random.choice(male_data['Purchase'],size=100)
    sample_means_male.append(np.mean(sample_m))
    sample_f=np.random.choice(female_data['Purchase'],size=100)
    sample_means_female.append(np.mean(sample_f))
```

```
np.mean(sample_means_male)
```

```
9443.26258
```

```
np.mean(sample_means_female)
```

```
8734.00328
```

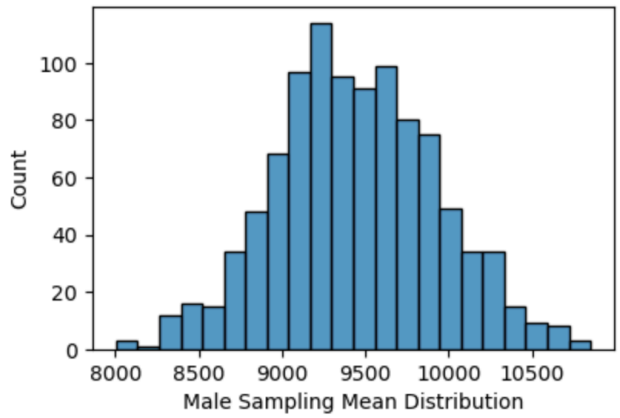
```
np.std(sample_means_male)/np.sqrt(100)
```

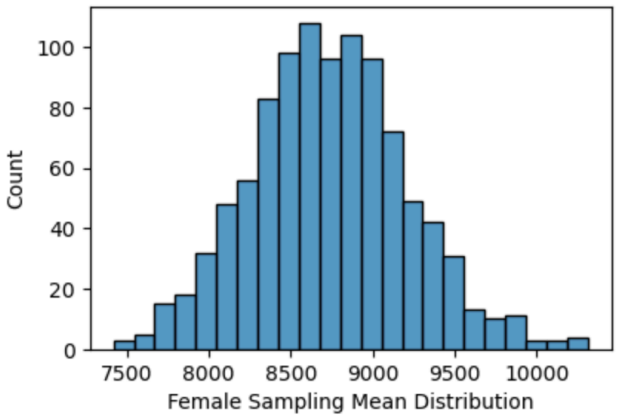
```
48.72784509921854
```

```
np.std(sample_means_female)/np.sqrt(100)
```

```
47.744720018934196
```

- As the Sample Size increases, the Standard error decreases drastically. This means we have a more uniform Normal distribution of Sampling means.






```
1 # Calculating Confidence Interval At 90%.
```

```
1 norm(loc=np.mean(sample_means_male),scale=np.std(sample_means_male)/np.sqrt(100)).interval(0.90)
```

```
(9363.112407255021, 9523.41275274498)
```

```
1 norm(loc=np.mean(sample_means_female),scale=np.std(sample_means_female)/np.sqrt(100)).interval(0.90)
```

```
(8655.470204109075, 8812.536355890927)
```

```
1 # Calculating Confidence Interval At 95%.
```

```
1 norm(loc=np.mean(sample_means_male),scale=np.std(sample_means_male)/np.sqrt(100)).interval(0.95)
```

```
(9347.757758561285, 9538.767401438716)
```

```
1 norm(loc=np.mean(sample_means_female),scale=np.std(sample_means_female)/np.sqrt(30)).interval(0.95)
```

```
(8563.154133097416, 8904.852426902586)
```

```
1 # Calculating Confidence Interval At 99%.
```

```
1 norm(loc=np.mean(sample_means_male),scale=np.std(sample_means_male)/np.sqrt(100)).interval(0.99)
```

```
(9317.747968694643, 9568.777191305358)
```

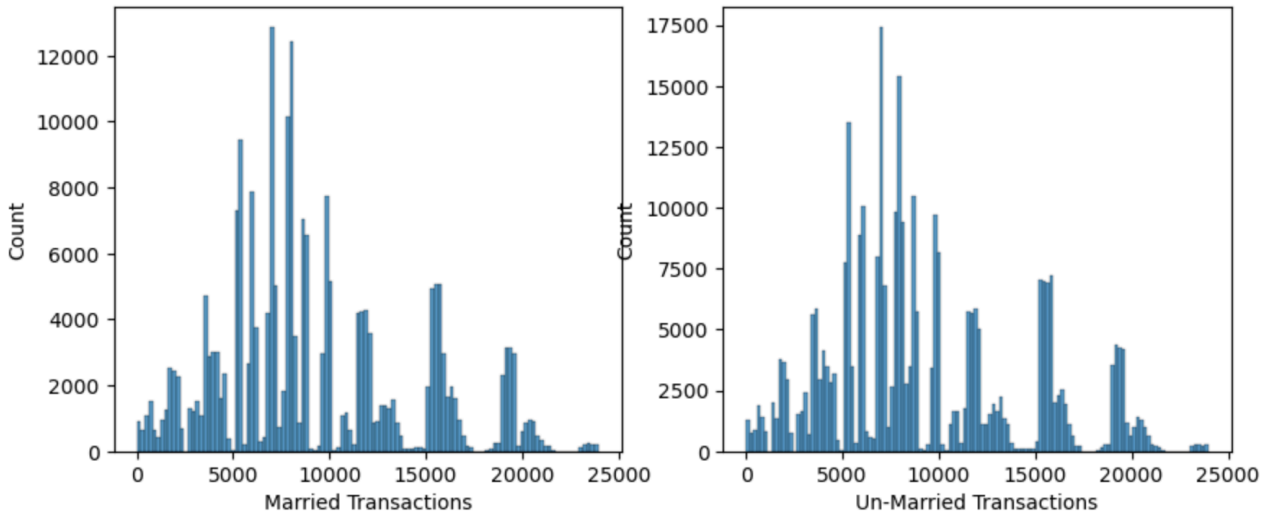
```
1 norm(loc=np.mean(sample_means_female),scale=np.std(sample_means_female)/np.sqrt(30)).interval(0.99)
```

```
(8509.469440322695, 8958.537119677307)
```

- As the Sample size increases, the standard error or the deviation of the sampling means decreases.
- This leads to the range of Confidence Interval Estimate getting reduced which is ideal.

Let us repeat the above process to find the Confidence interval at 90%, 95% and 99% for Customer Purchases who are married and Unmarried.

Sample Size : 30 & 100 Samples : 1000




```
1 married_data['Purchase'].mean()
```

```
9261.174574082374
```

```
1 unmarried_data['Purchase'].mean()
```

```
9265.907618921507
```

```
1 married_data['Purchase'].std()
```

```
5016.89737779313
```

```
1 unmarried_data['Purchase'].std()
```

```
5027.347858674457
```

- Population Mean Married= 9261.17
- Population std Married = 5016.89
- Population Mean Unmarried= 9265.9
- Population std Unmarried = 5027.34

```
sample_means_m=[]
sample_means_un=[]
```

```
for i in range(1000):
    sample_m=np.random.choice(married_data['Purchase'],size=30)
    sample_means_m.append(np.mean(sample_m))
    sample_f=np.random.choice(unmarried_data['Purchase'],size=30)
    sample_means_un.append(np.mean(sample_f))
```

```
np.mean(sample_means_m)
```

```
9245.007966666666
```

```
np.mean(sample_means_un)
```

```
9298.682933333333
```

```
np.std(sample_means_m)/np.sqrt(30)
```

```
171.07109391862213
```

```
np.std(sample_means_un)/np.sqrt(30)
```

```
167.90862056794973
```

```
for i in range(1000):
    sample_m=np.random.choice(married_data['Purchase'],size=100)
    sample_means_m.append(np.mean(sample_m))
    sample_f=np.random.choice(unmarried_data['Purchase'],size=100)
    sample_means_un.append(np.mean(sample_f))
```

```
np.mean(sample_means_m)
```

```
9271.569109999999
```

```
np.mean(sample_means_un)
```

```
9281.64449
```

```
np.std(sample_means_m)/np.sqrt(100)
```

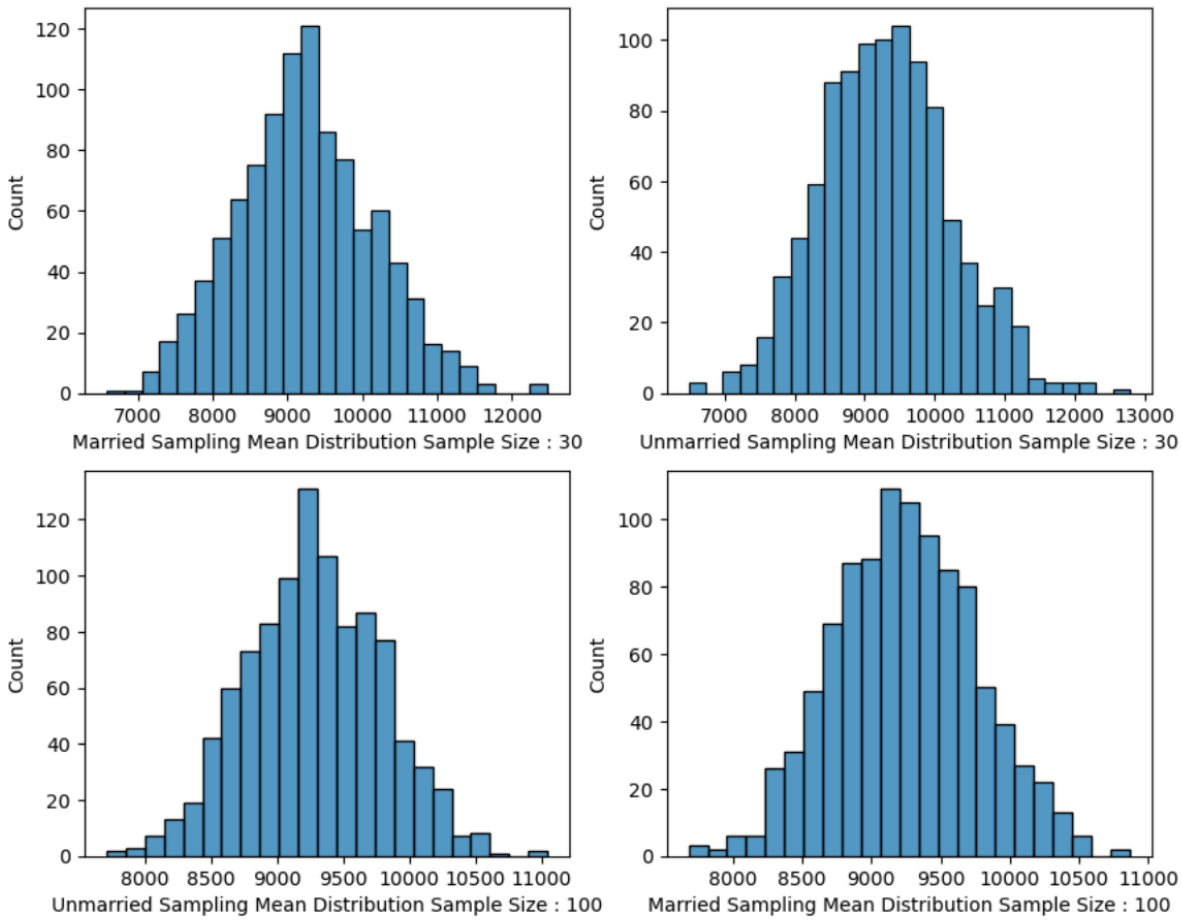
```
50.89691169958231
```

```
np.std(sample_means_un)/np.sqrt(100)
```

```
51.90436483157653
```

SAMPLE SIZE : 30

SAMPLE SIZE : 100



Clearly we can see as the sample size increases, we get a more normal distribution of the Sample Means.

Married :

```
# Calculating Confidence Interval At 90%.
```

```
norm(loc=np.mean(sample_means_m_30),scale=np.std(sample_means_m_30)/np.sqrt(30)).interval(0.90)
```

```
(8960.083130336634, 9509.386202996697)
```

```
norm(loc=np.mean(sample_means_m_100),scale=np.std(sample_means_m_100)/np.sqrt(100)).interval(0.90)
```

```
(9162.622876568023, 9331.508483431977)
```

- 90% Confidence, Size - 30 : [8960 to 9509]
- 90% Confidence, Size - 100 : [9162 to 9331]

UnMarried :

```
# Calculating Confidence Interval At 90%.
```

```
norm(loc=np.mean(sample_means_un_30),scale=np.std(sample_means_un_30)/np.sqrt(30)).interval(0.90)
```

```
(9020.846846767461, 9580.361353232543)
```

```
norm(loc=np.mean(sample_means_un_100),scale=np.std(sample_means_un_100)/np.sqrt(100)).interval(0.90)
```

```
(9196.558709050165, 9364.522150949833)
```

- 90% Confidence, Size - 30 : [9021 to 9580]
- 90% Confidence, Size - 100 : [9196 to 9364]

Married :

```
# Calculating Confidence Interval At 95%.
```

```
norm(loc=np.mean(sample_means_m_30),scale=np.std(sample_means_m_30)/np.sqrt(30)).interval(0.95)
```

```
(8907.467175696578, 9562.002157636753)
```

```
norm(loc=np.mean(sample_means_m_100),scale=np.std(sample_means_m_100)/np.sqrt(100)).interval(0.95)
```

```
(9146.445873612112, 9347.685486387887)
```

- 95% Confidence, Size - 30 : [8907 to 9562]
- 95% Confidence, Size - 100 : [9146 to 9347]

UnMarried :

```
# Calculating Confidence Interval At 95%.
```

```
norm(loc=np.mean(sample_means_un_30),scale=np.std(sample_means_un_30)/np.sqrt(30)).interval(0.95)
```

```
(8967.252772097432, 9633.955427902572)
```

```
norm(loc=np.mean(sample_means_un_100),scale=np.std(sample_means_un_100)/np.sqrt(100)).interval(0.95)
```

```
(9180.470037276706, 9380.610822723293)
```

- 95% Confidence, Size - 30 : [8967 to 9633]
- 95% Confidence, Size - 100 : [9180 to 9380]

Married :

```
# Calculating Confidence Interval At 99%.
```

```
norm(loc=np.mean(sample_means_m_30),scale=np.std(sample_means_m_30)/np.sqrt(30)).interval(0.99)
```

```
(8804.632277036531, 9664.8370562968)
```

```
norm(loc=np.mean(sample_means_m_100),scale=np.std(sample_means_m_100)/np.sqrt(100)).interval(0.99)
```

```
(9114.828838988782, 9379.302521011217)
```

- 99% Confidence, Size - 30 : [8804 to 9664]
- 99% Confidence, Size - 100 : [9114 to 9379]

Unmarried :

```
# Calculating Confidence Interval At 99%.
```

```
norm(loc=np.mean(sample_means_un_30),scale=np.std(sample_means_un_30)/np.sqrt(30)).interval(0.99)
```

```
(8862.50619332348, 9738.702006676525)
```

```
norm(loc=np.mean(sample_means_un_100),scale=np.std(sample_means_un_100)/np.sqrt(100)).interval(0.99)
```

```
(9149.025640938815, 9412.055219061183)
```

- 99% Confidence, Size - 30 : [8862 to 9738]
- 99% Confidence, Size - 100 : [9149 to 9412]

- As the Sample size increases, the standard error or the deviation of the sampling means decreases.
- This leads to the range of Confidence Interval Estimate getting reduced which is ideal.

Let us divide the transactions based on Age Category.

0-17 -> Teen

18-25 -> Young Adult

26-35 -> Adult

36-45 -> L_Middle_Age

46-50 -> Middle Age

51_55 -> U_Middle_Age

55+ -> Old Age

Let our Sample Size be 30.

Population Means

```
1 teen['Purchase'].mean()
```
8933.464640444974

```
1 young_adult['Purchase'].mean()
```
9169.663606261289

```
1 adult['Purchase'].mean()
```
9252.690632869888

```
1 l_middle_age['Purchase'].mean()
```
9331.350694917874

```
1 middle_age['Purchase'].mean()
```
9208.625697468327

```
1 u_middle_age['Purchase'].mean()
```
9534.808030960236

```
1 old_age['Purchase'].mean()
```
9336.280459449405

Sampling Means

```
1 np.mean(sm_teen)
```
8925.312533333332

```
1 np.mean(sm_young_adult)
```
9188.872

```
1 np.mean(sm_adult)
```
9234.196600000001

```
1 np.mean(sm_lm)
```
9345.003633333334

```
1 np.mean(sm_m)
```
9476.610866666668

```
1 np.mean(sm_um)
2
```
9195.4629

```
1 np.mean(sm_o)
```
9356.598833333333

Let us find the Confidence Interval at 90%, 95%, 99%.

Teen : 0-17

```
1 # Calculating Confidence Interval At 90%,95%,99% - Teen
```

```
1 norm(loc=np.mean(sm_teen),scale=np.std(sm_teen)/np.sqrt(30)).interval(0.90)
```
(8641.697430893555, 9208.927635773109)

```
1 norm(loc=np.mean(sm_teen),scale=np.std(sm_teen)/np.sqrt(30)).interval(0.95)
```
(8587.364294568863, 9263.260772097801)

```
1 norm(loc=np.mean(sm_teen),scale=np.std(sm_teen)/np.sqrt(30)).interval(0.99)
```
(8481.173261714755, 9369.451804951908)

Young Adult : 18-25

```
# Calculating Confidence Interval At 90%,95%,99% - Young Adult
```

```
norm(loc=np.mean(sm_young_adult),scale=np.std(sm_young_adult)/np.sqrt(30)).interval(0.90)
```

```
(8912.095955949359, 9465.648044050638)
```

```
norm(loc=np.mean(sm_young_adult),scale=np.std(sm_young_adult)/np.sqrt(30)).interval(0.95)
```

```
(8859.073001940746, 9518.670998059253)
```

```
norm(loc=np.mean(sm_young_adult),scale=np.std(sm_young_adult)/np.sqrt(30)).interval(0.99)
```

```
(8755.44264610223, 9622.301353897768)
```

Adult : 26-35

```
# Calculating Confidence Interval At 90%,95%,99% - Adult
```

```
norm(loc=np.mean(sm_adult),scale=np.std(sm_adult)/np.sqrt(30)).interval(0.90)
```

```
(8962.781157460147, 9505.612042539855)
```

```
norm(loc=np.mean(sm_adult),scale=np.std(sm_adult)/np.sqrt(30)).interval(0.95)
```

```
(8910.78515261666, 9557.608047383343)
```

```
norm(loc=np.mean(sm_adult),scale=np.std(sm_adult)/np.sqrt(30)).interval(0.99)
```

```
(8809.161910666835, 9659.231289333167)
```

L_Middle_Age : 36-45

```
# Calculating Confidence Interval At 90%,95%,99% - L_Middle_Age
```

```
norm(loc=np.mean(sm_lm),scale=np.std(sm_lm)/np.sqrt(30)).interval(0.90)
```

```
(9075.071384288625, 9614.935882378042)
```

```
norm(loc=np.mean(sm_lm),scale=np.std(sm_lm)/np.sqrt(30)).interval(0.95)
```

```
(9023.359520006323, 9666.647746660345)
```

```
norm(loc=np.mean(sm_lm),scale=np.std(sm_lm)/np.sqrt(30)).interval(0.99)
```

```
(8922.291614665222, 9767.715652001445)
```

Middle Age : 46-50

```
# Calculating Confidence Interval At 90%,95%,99% - U_Middle_Age
```

```
norm(loc=np.mean(sm_um),scale=np.std(sm_um)/np.sqrt(30)).interval(0.90)
```

```
(8927.56696229127, 9463.35883770873)
```

```
norm(loc=np.mean(sm_um),scale=np.std(sm_um)/np.sqrt(30)).interval(0.95)
```

```
(8876.245201287904, 9514.680598712097)
```

```
norm(loc=np.mean(sm_um),scale=np.std(sm_um)/np.sqrt(30)).interval(0.99)
```

```
(8775.93973067655, 9614.986069323451)
```

U_Middle_Age : 51-55

```
# Calculating Confidence Interval At 90%,95%,99% - Middle_Age
```

```
norm(loc=np.mean(sm_m),scale=np.std(sm_m)/np.sqrt(30)).interval(0.90)
```

```
(9193.67100503351, 9759.550728299826)
```

```
norm(loc=np.mean(sm_m),scale=np.std(sm_m)/np.sqrt(30)).interval(0.95)
```

```
(9139.467226949906, 9813.75450638343)
```

```
norm(loc=np.mean(sm_m),scale=np.std(sm_m)/np.sqrt(30)).interval(0.99)
```

```
(9033.529017440054, 9919.692715893281)
```

Old_Age : 55+

```
# Calculating Confidence Interval At 90%,95%,99% - Old_Age
```

```
norm(loc=np.mean(sm_o),scale=np.std(sm_o)/np.sqrt(30)).interval(0.90)
```

```
(9081.40957201267, 9631.788094653997)
```

```
norm(loc=np.mean(sm_o),scale=np.std(sm_o)/np.sqrt(30)).interval(0.95)
```

```
(9028.69060351697, 9684.507063149696)
```

```
norm(loc=np.mean(sm_o),scale=np.std(sm_o)/np.sqrt(30)).interval(0.99)
```

```
(8925.654370119615, 9787.543296547052)
```

INSIGHTS & RECOMMENDATIONS

- **414259** purchases have been made by **4225** Male customers which is **98** purchases on an average by a Male Customer.
- **135809** purchases have been made by **1666** Female Customers which is **82** purchases on an average by a Female Customer.
- Average Transaction value from a Male Purchase : 9438
- Average Transaction value from a Female Purchase : 8734
- Confidence Interval of the Purchase Amounts of Male Customer Purchases :
    - 90% Confidence : [9151.93 to 9712.94]
    - 95% Confidence : [9098.19 to 9766.69]
    - 99% Confidence : [8993.16 to 9871.71]
- Confidence Interval of the Purchase Amounts of Female Customer Purchases :
    - 90% Confidence : [8480.45 to 9000.94]
    - 95% Confidence : [8430.59 to 9050.80]
    - 99% Confidence : [8333.15 to 9148.24]
- This Clearly shows the Confidence Intervals do not overlap which means the difference between the sample statistics is statistically significant.
- The Company Should strategise to reduce the difference in the transaction amounts between Male and Female transactions.
- Male Customer transactions are more when compared to female transaction on black friday sale.
- Company Should Target Age group 26 to 45 as they contribute heavily in the day transactions on Black Friday.
- Product Categories which are more preferred by the age groups can be increased during sale to attract more customers.

- Average Transaction Value from a Married Customer : 9261
- Average Transaction Value from an Unmarried Customer : 9266
- Married or Unmarried doesn't seem to have a major impact in the transaction value.
- Confidence Interval of the Purchase Amounts of Married Customer Purchases :
    - 90% Confidence, Size - 100 : [9162 to 9331]
    - 95% Confidence, Size - 100 : [9146 to 9347]
    - 99% Confidence, Size - 100 : [9114 to 9379]

- Confidence Interval of the Purchase Amounts of Unmarried Customer Purchases:
    - 90% Confidence, Size - 100 : [9196 to 9364]
    - 95% Confidence, Size - 100 : [9180 to 9380]
    - 99% Confidence, Size - 100 : [9149 to 9412]

- The Confidence Intervals overlap indicating the difference between the two groups may not be statistically significant. There is a possibility that the population parameters are equal, as the intervals encompass each other.
- The Company should focus on other parameters like gender, Age instead of Marital Status as the Population parameters are almost equal for both Married and Unmarried customers.
- Customers belonging to age group 26-45 have made significant transactions on black friday sale.
- Product Categories attracting specific age group can be increased during black friday Sale.