Business Case: Walmart - Confidence Interval and CLT

Colab link - https://colab.research.google.com/drive/1-OSyZ4bv9_w0jYfHc0NyJd8VvLGSUrL_#scrollTo=Ou8eafgUPHKH

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
df = pd.read_csv('/content/walmart_data.csv') # Loading the dataset
df
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Ca
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	
...	...	...	...	...	...	...	...	...	
550063	1006033	P00372445	M	51-55	13	B	1	1	
550064	1006035	P00375436	F	26-35	1	C	3	0	

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

```
df.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Catego
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	

```
df.info()
```

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

```
df.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

```
df.sample(10)
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Ca
231947	1005770	P00335242	M	26-35	0	C	3	1	
450242	1003391	P00303942	M	18-25	4	A	0	0	
73931	1005387	P00118942	F	26-35	1	B	4+	1	
268379	1005361	P00255942	F	51-55	16	A	3	0	
451325	1003526	P00331342	M	36-45	2	B	1	1	
282941	1001599	P00114942	M	26-35	0	A	1	0	

2. Detect Null values & Outliers (using boxplot, “describe” method by checking the difference between mean and median, isnull etc.)

```
print(df.isnull()) #to detect the null/missing values
```

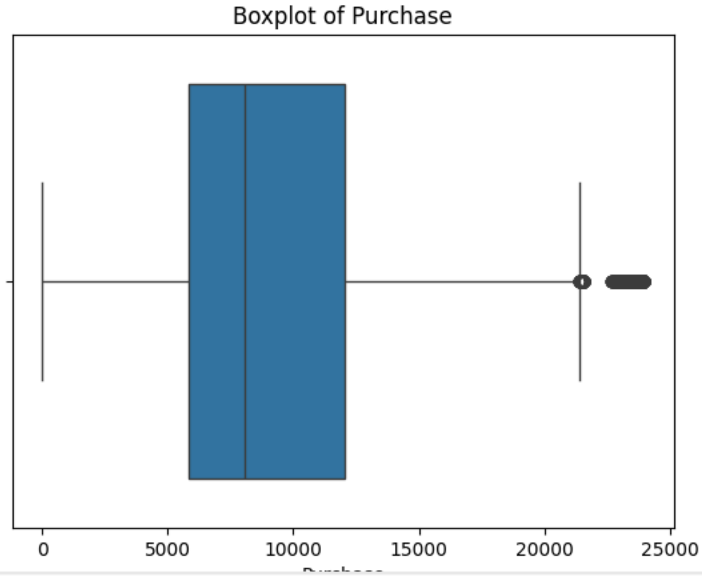
```
        User_ID  Product_ID  Gender    Age  Occupation  City_Category  \
0         False       False   False  False       False          False
1         False       False   False  False       False          False
2         False       False   False  False       False          False
3         False       False   False  False       False          False
4         False       False   False  False       False          False
...         ...         ...     ...    ...         ...            ...
550063    False       False   False  False       False          False
550064    False       False   False  False       False          False
550065    False       False   False  False       False          False
550066    False       False   False  False       False          False
550067    False       False   False  False       False          False

        Stay_In_Current_City_Years  Marital_Status  Product_Category  Purchase
0                            False           False             False     False
1                            False           False             False     False
2                            False           False             False     False
3                            False           False             False     False
4                            False           False             False     False
...                            ...             ...               ...       ...
550063                       False           False             False     False
550064                       False           False             False     False
550065                       False           False             False     False
550066                       False           False             False     False
550067                       False           False             False     False

[550068 rows x 10 columns]
```

```
print(df.isnull().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
```

```
sns.boxplot(x=df['Purchase']) #plotting boxplot to check for outliers
plt.title('Boxplot of Purchase')
plt.show()
```



3. Do some data exploration steps

```
avg_spending_m=df[df['Gender']=='M']['Purchase'].mean()
print("Average amount spent by male customers - ", avg_spending_m)
```

```
Average amount spent by male customers -  9437.526040472265
```

```
avg_spending_f=df[df['Gender']=='F']['Purchase'].mean()
print("Average amount spent by female customers - ", avg_spending_f)
```

```
Average amount spent by female customers -  8734.565765155476
```

Inference : There some difference in average spending between male and female customers based on the sample data.As we can see for male it is higher than female.

4. Confidence Interval Using Central Limit Theorem

For female customers

```
# Function to compute confidence interval
def confidence_interval(data, confidence=0.95):
  n = len(data)
  mean = np.mean(data)
  std_dev = np.std(data, ddof=1) # Sample standard deviation
  z_score = stats.norm.ppf((1 + confidence) / 2)
  margin_of_error = z_score * (std_dev / np.sqrt(n))
  lower_bound = mean - margin_of_error
  upper_bound = mean + margin_of_error
  return lower_bound, upper_bound
```

For female customers

```
# Sample female customers
female_data = df[df['Gender'] == 'F']['Purchase'].sample(1000, random_state=0)
lower_female, upper_female = confidence_interval(female_data)
print("Confidence Interval for Female Customers:")
print("Lower Bound:", lower_female)
print("Upper Bound:", upper_female)
```

```
Confidence Interval for Female Customers:
Lower Bound: 8606.534231090494
Upper Bound: 9174.429768909506
```

For male customers

```
# Sample male customers
male_data = df[df['Gender'] == 'M']['Purchase'].sample(1000, random_state=0)
lower_male, upper_male = confidence_interval(male_data)
print("Confidence Interval for Male Customers:")
print("Lower Bound:", lower_male)
print("Upper Bound:", upper_male)
```

```
Confidence Interval for Male Customers:
Lower Bound: 9061.306880523503
Upper Bound: 9708.321119476497
```

Observe Distribution with Different Sample Sizes

```
def observe_dist(data, sample_sizes, confidence=0.95):
  for size in sample_sizes:
    sample = data.sample(size, random_state=0)
    conf_interval = confidence_interval(sample, confidence)
    print(f'Sample size: {size}, Confidence interval: {conf_interval}')


sample_sizes = [100, 500, 1000, 5000, 10000]

print('Distribution for female customers:')
observe_dist(df[df['Gender'] == 'F']['Purchase'], sample_sizes)

print('\nDistribution for male customers:')
observe_dist(df[df['Gender'] == 'M']['Purchase'], sample_sizes)
```

```
Distribution for female customers:
Sample size: 100, Confidence interval: (7857.3658096880445, 9656.714190311957)
Sample size: 500, Confidence interval: (8414.246110785865, 9213.341889214134)
Sample size: 1000, Confidence interval: (8606.534231090494, 9174.429768909506)
Sample size: 5000, Confidence interval: (8752.016630637601, 9017.112169362397)
Sample size: 10000, Confidence interval: (8725.774471035325, 8912.164328964674)

Distribution for male customers:
Sample size: 100, Confidence interval: (8164.370879075117, 10361.029120924884)
Sample size: 500, Confidence interval: (9193.63988782697, 10127.812112173031)
Sample size: 1000, Confidence interval: (9061.306880523503, 9708.321119476497)
Sample size: 5000, Confidence interval: (9340.990105572078, 9626.265894427923)
Sample size: 10000, Confidence interval: (9339.559692448147, 9539.054307551854)
```

Different confidnce levels

```
levels = [0.90, 0.95, 0.99]


print('Confidence intervals for female customers at different levels:')
for level in levels:
  conf_interval = confidence_interval(female_data, level)
  print(f'Confidence level: {level}, Confidence interval: {conf_interval}')
```

```
Confidence intervals for female customers at different levels:
Confidence level: 0.9, Confidence interval: (8652.185520452605, 9128.778479547394)
Confidence level: 0.95, Confidence interval: (8606.534231090494, 9174.429768909506)
Confidence level: 0.99, Confidence interval: (8517.31137563974, 9263.65262436026)
```

```
print('Confidence intervals for male customers at different levels:')
for level in levels:
  conf_interval = confidence_interval(male_data, level)
  print(f'Confidence level: {level}, Confidence interval: {conf_interval}')
```

```
Confidence intervals for male customers at different levels:
Confidence level: 0.9, Confidence interval: (9113.318266908553, 9656.309733091448)
Confidence level: 0.95, Confidence interval: (9061.306880523503, 9708.321119476497)
Confidence level: 0.99, Confidence interval: (8959.65357622323, 9809.97442377677)
```

5: Conclusions

Insight -

Overlap - The confidence intervals for female customers at the 95% confidence level - (8606.534, 9174.430). The confidence intervals for male customers at the 95% confidence level - (9061.307, 9708.321).

The confidence intervals for male and female customers can be seen overlapping at the intervals (8606.534, 9174.430) for females and (9061.307, 9708.321) for males, in the range (9061.307, 9174.430).

Thus, we can see there isn't a significant difference in the average spending of male and female customers during Black Friday.

Recommendations -

1.Walmart should consider other marketting strategies and campaigns upper in the priority list rather than gender based once,

2.It should focus on better collections of items and bigger offers during festive seasons,

3.The inventory at the store should be focused to both the genders equally, rather can have specific sections for both for the gender specific items.

6: Analysis for Marital Status and Age

Analysis for Marital Status

```
# Married vs Unmarried: Average spending
avg_married_spending = df[df['Marital_Status'] == 1]['Purchase'].mean()
avg_unmarried_spending = df[df['Marital_Status'] == 0]['Purchase'].mean()


print(f'Average spending done by married customers - {avg_married_spending}')
print(f'Average spending done by unmarried customers - {avg_unmarried_spending}')
```

```
Average spending done by married customers - 9261.174574082374
Average spending done by unmarried customers - 9265.907618921507
```

```
# Confidence intervals
spending_sample_married= df[df['Marital_Status'] == 1]['Purchase'].sample(1000, random_state=0)
spending_sample_unmarried = df[df['Marital_Status'] == 0]['Purchase'].sample(1000, random_state=0)
conf_interval_married = confidence_interval(spending_sample_married)
conf_interval_unmarried = confidence_interval(spending_sample_unmarried)
print(f'Confidence interval of married customers average spenidng -  {conf_interval_married}')
print(f'Confidence interval of unmarried customers average spenidng -  {conf_interval_unmarried}')
```

```
Confidence interval of married customers average spenidng -  (8947.047594219928, 9557.436405780072)
Confidence interval of unmarried customers average spenidng -  (8983.080551233248, 9599.943448766753)
```

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

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

Analysis of age

```
bins_age = ['0-17', '18-25', '26-35', '36-45', '46-50', '51-55', '55+']
```

```
# Average spending for each age bin
for age_bin in bins_age:
  avg_spending = df[df['Age'] == age_bin]['Purchase'].mean()
  print(f'Average spending for each age bin {age_bin} - {avg_spending}')
```

```
Average spending for each age bin 0-17 - 8933.464640444974
Average spending for each age bin 18-25 - 9169.663606261289
Average spending for each age bin 26-35 - 9252.690632869888
Average spending for each age bin 36-45 - 9331.350694917874
Average spending for each age bin 46-50 - 9208.625697468327
Average spending for each age bin 51-55 - 9534.808030960236
Average spending for each age bin 55+ - 9336.280459449405
```

```
# Confidence interval
age_bin_sample = df[df['Age'] == age_bin]['Purchase'].sample(1000, random_state=0)
conf_interval_age_bin = confidence_interval(age_bin_sample)
print(f'Confidence interval of average spending of the age bin {age_bin} - {conf_interval_age_bin}')
```

```
Confidence interval of average spending of the age bin 55+ - (9138.014312279633, 9734.955687720369)
```

7: Recommendations and Action Items

Recommendations -

1.As per the average spending and confidence intervals seen above, Walmart should now run campaigns according to the spent per age group,

2.Campaigns should target age groups with age specsfic items, such as gadegts for the youth age group,

3.The inventory should be comprised of items specefic to each age group, in order to cater all of their purchase needs efficiently,

4.High selling items per age group should also be stocked well in the inventory so as not to let any customer divert to any other platform,

5.Such items should be provided with promotional offers on regular basis, specially during festive seasons,

6.As per the overall analysis, the market strategy of Walmart to focus less on gender based and more on age group based marketing strategies.

Actions Needed -

1.The company should get the data analysed regularly based on the sales data,

2.Proper feedback of the campaigns and all should be taken to get proper data through analysis.

Double-click (or enter) to edit