```
''' Importing the required modules '''

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

''' Reading the data from the csv file to the variable data '''

data = pd.read_csv('walmart_data.csv')
```

1. Importing the dataset and did usual data analysis steps like checking the structure & characteristics of the dataset.

```
data

        User_ID Product_ID Gender   Age  Occupation City_Category  \
0       1000001  P00069042      F  0-17          10             A
1       1000001  P00248942      F  0-17          10             A
2       1000001  P00087842      F  0-17          10             A
3       1000001  P00085442      F  0-17          10             A
4       1000002  P00285442      M   55+          16             C
...         ...        ...    ...   ...         ...           ...
550063  1006033  P00372445      M 51-55          13             B
550064  1006035  P00375436      F 26-35           1             C
550065  1006036  P00375436      F 26-35          15             B
550066  1006038  P00375436      F   55+           1             C
550067  1006039  P00371644      F 46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
4                              4+               0                 8
7969
...                           ...             ...               ...
...
550063                          1               1                20
```

```
368
550064                         3                0                 20
371
550065                        4+                1                 20
137
550066                         2                0                 20
365
550067                        4+                1                 20
490

[550068 rows x 10 columns]
```

```python
''' Getting the whole information about the dataset '''

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
```

```python
''' Checking for the null values in the data set for data cleaning 
process '''

data.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
```

```python
''' Checking the data tyes of the dataframe for consistency '''

data.dtypes
```

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

```python
''' Checking the null values for eac column in the data frame '''

for i in data.columns:
  print(i ,':',data[i].nunique() )
```

```
User_ID : 5891
Product_ID : 3631
Gender : 2
Age : 7
Occupation : 21
City_Category : 3
Stay_In_Current_City_Years : 5
Marital_Status : 2
Product_Category : 20
Purchase : 18105
```

2. Detecting Null values and outliers.

```python
''' Checking for the null values in the data set for data cleaning
process '''

data.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
```

```python
''' getting the number of values for Gender '''

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

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

```python
''' getting the number of values for Age '''

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

```
26-35    219587
36-45    110013
18-25     99660
46-50     45701
51-55     38501
55+       21504
0-17      15102
Name: Age, dtype: int64
```

```python
''' getting the number of values for Occupation grade '''

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

```
4     72308
0     69638
7     59133
1     47426
17    40043
20    33562
12    31179
14    27309
2     26588
16    25371
6     20355
3     17650
10    12930
5     12177
15    12165
11    11586
19     8461
13     7728
18     6622
9      6291
8      1546
Name: Occupation, dtype: int64
```

```python
''' getting the number of values for Product category '''

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
```

```python
''' getting the number of values for Product_ID '''


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

```
P00265242    1880
P00025442    1615
P00110742    1612
P00112142    1562
P00057642    1470
             ... 
P00314842       1
P00298842       1
P00231642       1
P00204442       1
P00066342       1
Name: Product_ID, Length: 3631, dtype: int64
```

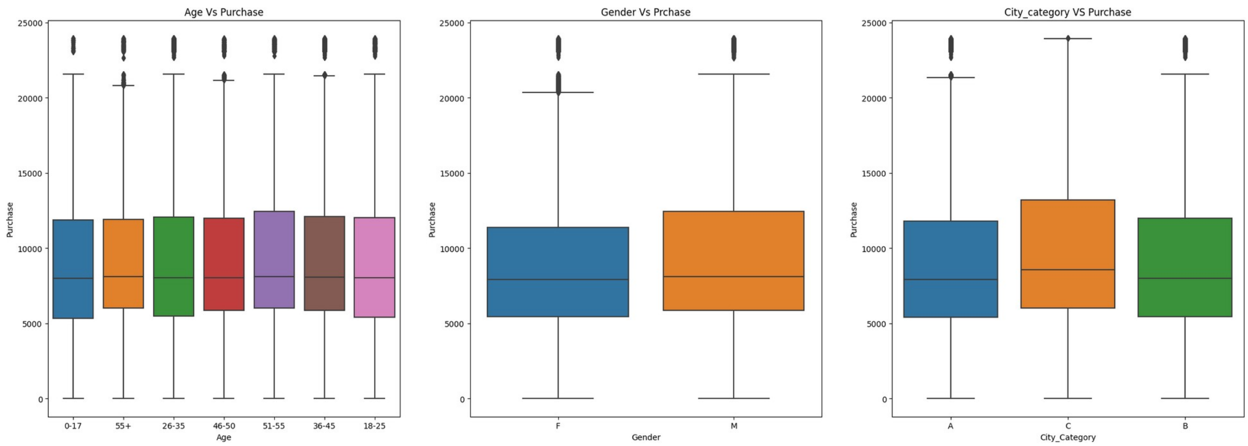
```python
''' getting the number of values for city_category '''


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

```
B    231173
C    171175
```

```
A    147720
Name: City_Category, dtype: int64
```

2(a).Finding the outliers for every continuous variable in the dataset.

```
''' Plotting the boxplot for the different componenets of the 
dataframe for getting percentiles(20,50,75) and outliers '''


plt.figure(figsize = (27,9))
plt.subplot(1,3,1)
sns.boxplot(
    x = 'Age',
    y = 'Purchase',
    data =  data
)
plt.title('Age Vs Purchase')
plt.subplot(1,3,2)
sns.boxplot(
    x = 'Gender',
    y = 'Purchase',
    data = data
)
plt.title('Gender Vs Prchase')
plt.subplot(1,3,3)
sns.boxplot(
    x = 'City_Category',
    y = 'Purchase',
    data = data

)
plt.title('City_category VS Purchase')
plt.show()
```



'Age vs Purchase':

1. We can clearly see from the above boxoplots in the 'Age vs Purchase', The 25 percentile for the different ages are almost same with some deviation in the values.
2. the median or the 50 percentile for different ages are same how ever ther may be small deviation that cant be shown in the grpahs.
3. The outliers are more in the age range -- 55+ and has the minimum outliers for the age 0-17.

'Gender vs Purchase':

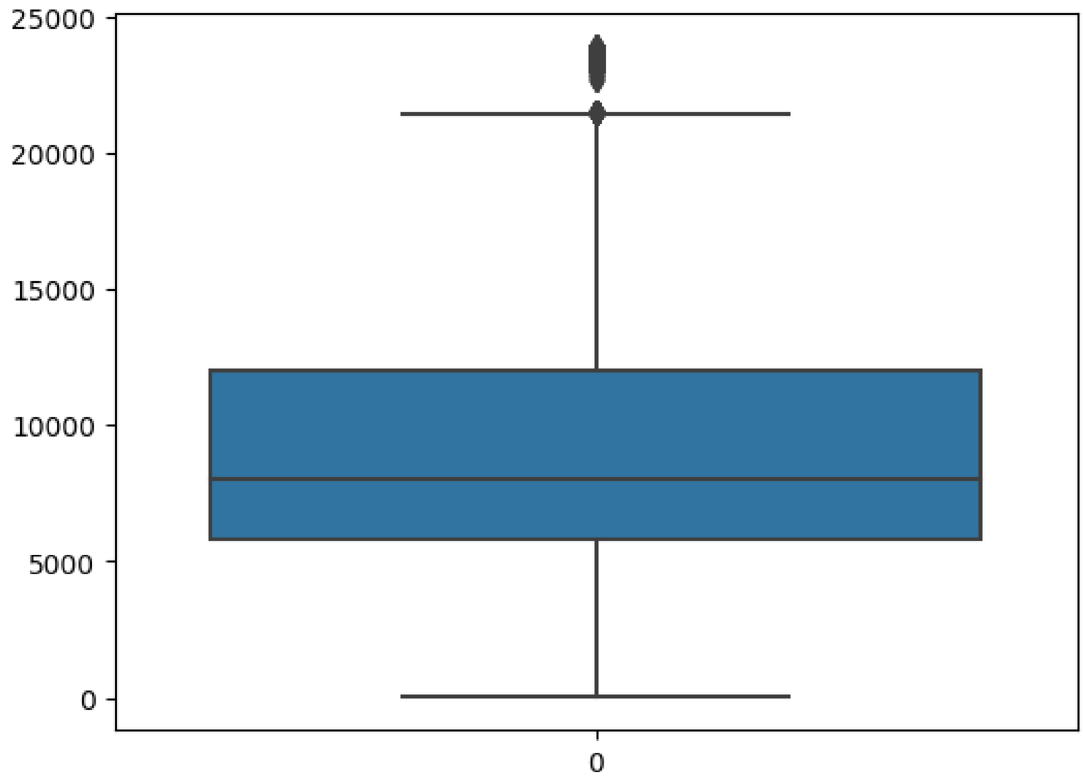
1. From the "Gender vs Purchase" box lot we can observe that the male haa more expansion thn the female indicating the purchase range.
2. Even though purchase power for men is greater the 50 ercentile is almost same for both the Gender.
3. the 75 percentile for the male are sligtly more than the Females.
4. The outliers count are most in Female category than the Male.

'City Category vs Purchase':

1. City category -- C has the more spread over in the graph indicating the more number of purchases made.
2. The other two categories -- A,B are almost similar in the sense of purchase power.
3. The median values for the City Category -- C is more then comapred to other two Categories
4. the lower quartile of Categories A,B are almost similar.
5. The upper Quartile of City CAtegory -- C also higher than other A,B.

2(b). Removing/clipping the data between the 5 percentile and 95 percentile

```
''' Filtering the data only for the purchase column '''

data_purchase = data['Purchase']

''' Boxplot for purchase'''

sns.boxplot(data_purchase)

<Axes: >
```




```
''' Calculating the 5 percentile for the purchase'''

dp_5_percetile = np.percentile(data_purchase,5)

dp_5_percetile

1984.0

''' Calculating the 5 percentile for the purchase'''

dp_95_percetile = np.percentile(data_purchase,95)
```

```
dp_95_percetile
```

```
19336.0
```

```
''' Getting all the information about like mean,mediam etc for 
purchase '''

round(data_purchase.describe(),2)
```

```
count    550068.00
mean       9263.97
std        5023.07
min          12.00
25%        5823.00
50%        8047.00
75%       12054.00
max       23961.00
Name: Purchase, dtype: float64
```

```
''' clipping the data  for 5 and 95 percentile '''

dp_clip_final = np.clip(data_purchase,dp_5_percetile,dp_95_percetile)
```

'Clipping':

In the clip() function,when we pass the interval(combination of minimum value and maximum value -- here 5 percentile and 95 percentile), values outside the interval are clipped to the interval edges.

```
dp_clip_final
```

```
0          8370
1         15200
2          1984
3          1984
4          7969
          ...  
550063     1984
550064     1984
550065     1984
550066     1984
550067     1984
Name: Purchase, Length: 550068, dtype: int64
```

```
''' Checking for the values other than  clipping values '''

dp_clip_final > 19336
```

```
0         False
1         False
2         False
```

```
3         False
4         False
          ...
550063    False
550064    False
550065    False
550066    False
550067    False
Name: Purchase, Length: 550068, dtype: bool
```

```
''' Checking for the values other than  clipping values '''

dp_clip_final < 1984
```

```
0         False
1         False
2         False
3         False
4         False
          ...
550063    False
550064    False
550065    False
550066    False
550067    False
Name: Purchase, Length: 550068, dtype: bool
```

3. Data Exploration

3.1 Tracking the amount spent per transaction of all the 50 million female customers, and all the 50 million male customers, calculating the average, and conclude the results.

```
''' Filtering the data for gender 'Male' '''

data_men = data[data['Gender']== 'M']

data_men
```

```
        User_ID Product_ID Gender    Age  Occupation City_Category  \
4       1000002  P00285442      M    55+          16             C
5       1000003  P00193542      M  26-35          15             A
6       1000004  P00184942      M  46-50           7             B
7       1000004  P00346142      M  46-50           7             B
8       1000004   P0097242      M  46-50           7             B
...         ...        ...    ...    ...         ...           ...
550057  1006023  P00370853      M  26-35           0             C
550058  1006024  P00372445      M  26-35          12             A
550060  1006026  P00371644      M  36-45           6             C
```

```
550062  1006032  P00372445      M  46-50           7               A
550063  1006033  P00372445      M  51-55          13               B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
4                              4+               0                 8
7969
5                               3               0                 1
15227
6                               2               1                 1
19215
7                               2               1                 1
15854
8                               2               1                 1
15686
...                           ...             ...               ...
...
550057                          2               1                19
61
550058                          0               1                20
121
550060                          1               1                20
494
550062                          3               0                20
473
550063                          1               1                20
368

[414259 rows x 10 columns]

''' Filtering the purchase column for Gender Male '''

data_purchase_men = data_men['Purchase']

''' caluculating the mean for gender MAle '''

data_men_mean = data_purchase_men.mean()

''' Calculating the median for gender male '''

data_men_median = data_purchase_men.median()

''' Filtering the data for gender female '''

data_women = data[data['Gender']== 'F']

data_women

        User_ID Product_ID Gender    Age  Occupation City_Category  \
0       1000001  P00069042      F   0-17          10             A
1       1000001  P00248942      F   0-17          10             A
```

```
2       1000001  P00087842      F   0-17          10             A
3       1000001  P00085442      F   0-17          10             A
14      1000006  P00231342      F  51-55           9             A
...         ...        ...    ...    ...         ...           ...
550061  1006029  P00372445      F  26-35           1             C
550064  1006035  P00375436      F  26-35           1             C
550065  1006036  P00375436      F  26-35          15             B
550066  1006038  P00375436      F    55+           1             C
550067  1006039  P00371644      F  46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
14                              1               0                 5
5378
...                           ...             ...               ...
...
550061                          1               1                20
599
550064                          3               0                20
371
550065                         4+               1                20
137
550066                          2               0                20
365
550067                         4+               1                20
490

[135809 rows x 10 columns]

''' Filtering the purchase column for gender female '''

data_purchase_women = data_women['Purchase']

''' Calculating the mean for gender female '''

data_women_mean = data_purchase_women.mean()

''' Calculating the median for gender female '''

data_women_median  = data_purchase_women.median()
```

```
''' difference between mean and median for Male customers : '''

round(data_men_mean - data_men_median,2)
```

```
1339.53
```

```
''' difference between mean and median for female customers : '''

data_women_mean  - data_women_median
```

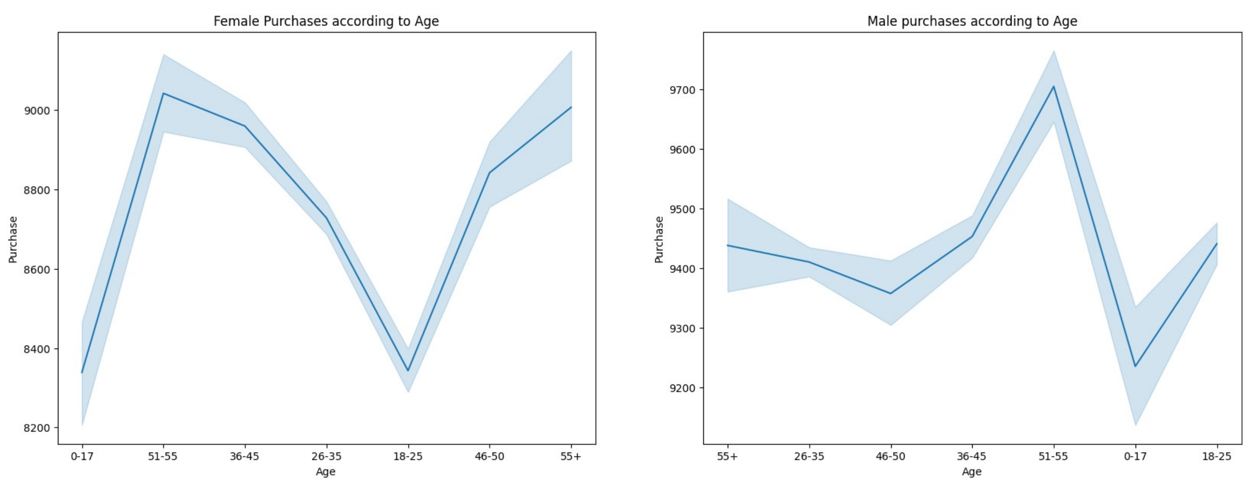
```
820.5657651554757
```

3.1 What products are different age groups buying?

```
data_men
```

```
        User_ID Product_ID Gender    Age  Occupation City_Category  \
4       1000002  P00285442      M    55+          16             C
5       1000003  P00193542      M  26-35          15             A
6       1000004  P00184942      M  46-50           7             B
7       1000004  P00346142      M  46-50           7             B
8       1000004   P0097242      M  46-50           7             B
...         ...        ...    ...    ...         ...           ...
550057  1006023  P00370853      M  26-35           0             C
550058  1006024  P00372445      M  26-35          12             A
550060  1006026  P00371644      M  36-45           6             C
550062  1006032  P00372445      M  46-50           7             A
550063  1006033  P00372445      M  51-55          13             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
4                              4+               0                 8
7969
5                               3               0                 1
15227
6                               2               1                 1
19215
7                               2               1                 1
15854
8                               2               1                 1
15686
...                           ...             ...               ...
...
550057                          2               1                19
61
550058                          0               1                20
121
550060                          1               1                20
494
550062                          3               0                20
```

```
473
550063                                   1                1                20
368

[414259 rows x 10 columns]

''' Plotting the purchase power according the gender of MAle and
Female according to age '''

plt.figure(figsize = (20,7))
plt.subplot(1,2,1)
sns.lineplot(
    x = 'Age',
    y = 'Purchase',
    data = data_women
)
plt.title('Female Purchases according to Age')
plt.subplot(1,2,2)
sns.lineplot(
    x = 'Age',
    y = 'Purchase',
    data = data_men
)
plt.title('Male purchases according to Age')
plt.show()
```



1. We can observe from the two graphs for feamle the purchasing is less inthe age between 0- 25 tahn the males. 2.The purchase power of females increased more at the ages later than the male customers.
2. Purchasing power of male customer peaked at the age 51-55 and same as for female customers at same 51-55
3. There is some fluctuation in the purchase power of male customers for ages between 26-50.
4. The female customers wins the competition by dramatically increasing the purchase power later the age from 26-55+.

3.2Checking for relationship between age, marital status, and the amount spent?

```
data
```

```
        User_ID Product_ID Gender    Age  Occupation City_Category  \
0       1000001  P00069042      F   0-17          10             A
1       1000001  P00248942      F   0-17          10             A
2       1000001  P00087842      F   0-17          10             A
3       1000001  P00085442      F   0-17          10             A
4       1000002  P00285442      M    55+          16             C
...         ...        ...    ...    ...         ...           ...
550063  1006033  P00372445      M  51-55          13             B
550064  1006035  P00375436      F  26-35           1             C
550065  1006036  P00375436      F  26-35          15             B
550066  1006038  P00375436      F    55+           1             C
550067  1006039  P00371644      F  46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
4                              4+               0                 8
7969
...                           ...             ...               ...
...
550063                          1               1                20
368
550064                          3               0                20
371
550065                         4+               1                20
137
550066                          2               0                20
365
550067                         4+               1                20
490

[550068 rows x 10 columns]
```

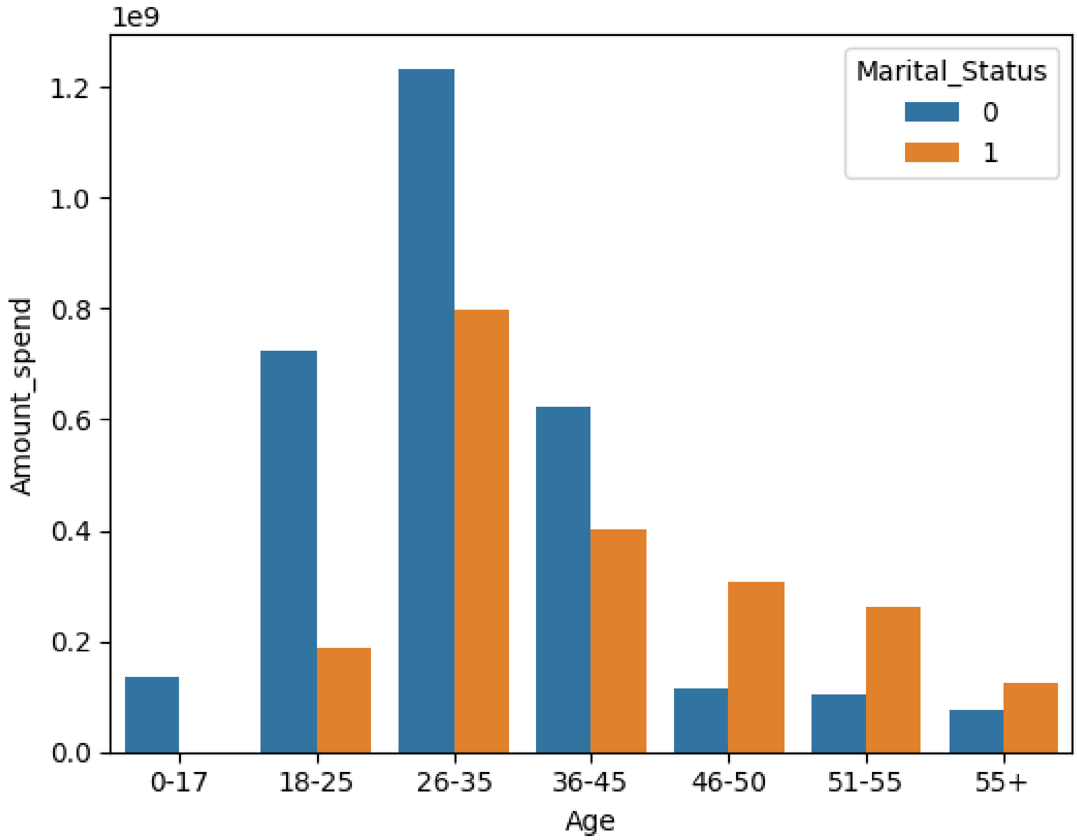
```
''' grouping the data of age and Marital Status and calculating the
total amount '''
```

```python
data_mrg = data.groupby(['Age','Marital_Status']).agg(total_amount =
('Purchase','sum')).reset_index()

data_mrg
```

	Age	Marital_Status	total_amount
0	0-17	0	134913183
1	18-25	0	723920602
2	18-25	1	189928073
3	26-35	0	1233330102
4	26-35	1	798440476
5	36-45	0	624110760
6	36-45	1	402459124
7	46-50	0	113658360
8	46-50	1	307185043
9	51-55	0	103792394
10	51-55	1	263307250
11	55+	0	75202046
12	55+	1	125565329

```python
''' plotting the graph between the componenets of Age,Marital status
and the amount spend '''

sns.barplot(
    x = 'Age',
    y = 'total_amount',
    hue = 'Marital_Status',
    data = data_mrg
)
plt.ylabel('Amount_spend')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



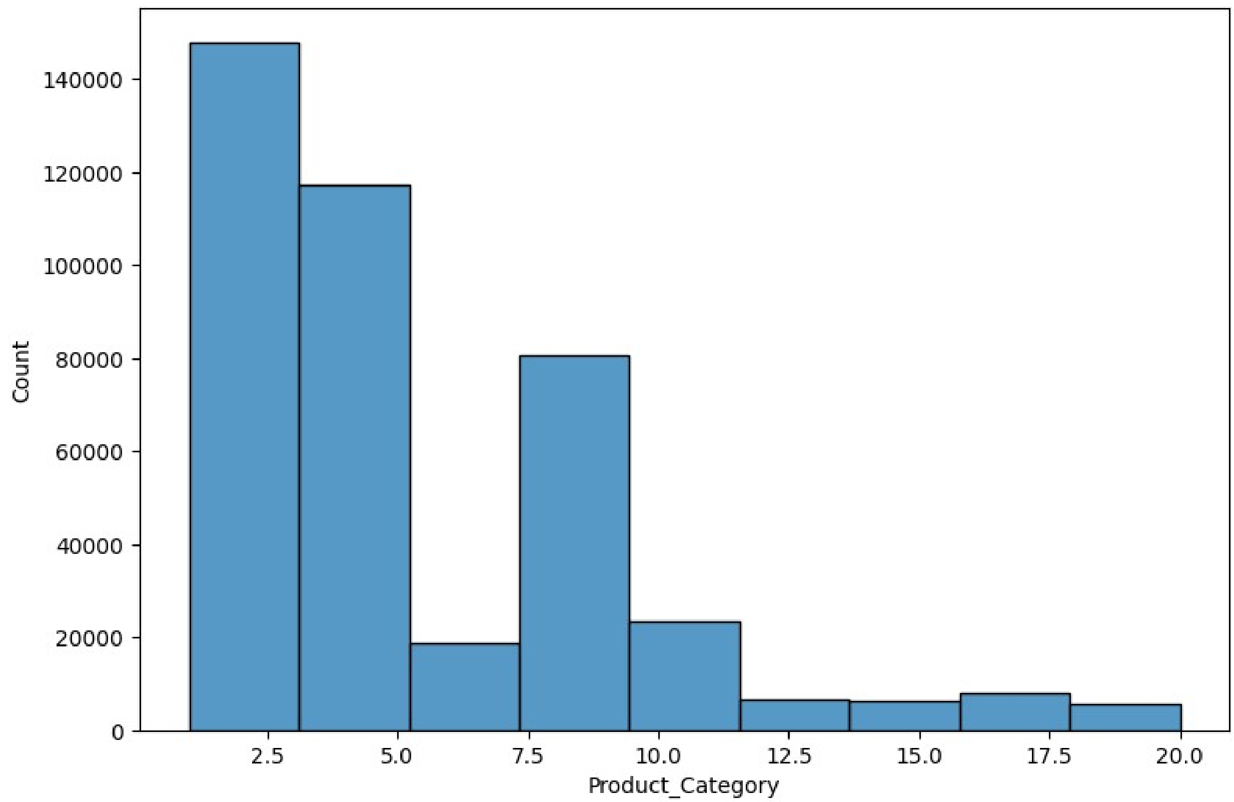
3.2 Checking for any preferred product categories for different genders?

```
''' Dropping the duplicates form the data to maintain consistency '''

data_men.drop_duplicates(inplace = True)

<ipython-input-407-300c8c1fc163>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  data_men.drop_duplicates(inplace = True)

''' Resetting the index for dataframe data_men '''

data_men.reset_index(inplace = True)

''' Dropping the unwanted columns from the dataframe '''

data_men.drop(columns = ['index'],inplace = True)

<ipython-input-409-3bf219ce1a23>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  data_men.drop(columns = ['index'],inplace = True)

data_men

        User_ID Product_ID Gender    Age  Occupation City_Category  \
0       1000002  P00285442      M    55+          16             C
1       1000003  P00193542      M  26-35          15             A
2       1000004  P00184942      M  46-50           7             B
3       1000004  P00346142      M  46-50           7             B
4       1000004   P0097242      M  46-50           7             B
...         ...        ...    ...    ...         ...           ...
414254  1006023  P00370853      M  26-35           0             C
414255  1006024  P00372445      M  26-35          12             A
414256  1006026  P00371644      M  36-45           6             C
414257  1006032  P00372445      M  46-50           7             A
414258  1006033  P00372445      M  51-55          13             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                              4+               0                 8
7969
1                               3               0                 1
15227
2                               2               1                 1
19215
3                               2               1                 1
15854
4                               2               1                 1
15686
...                           ...             ...               ...
...
414254                          2               1                19
61
414255                          0               1                20
121
414256                          1               1                20
494
414257                          3               0                20
473
414258                          1               1                20
368

[414259 rows x 10 columns]

'''' Plotting the histogram of Product_category for gender male '''
```

```
plt.figure(figsize = (9,6))
sns.histplot(data_men['Product_Category'],bins = 9)
```

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

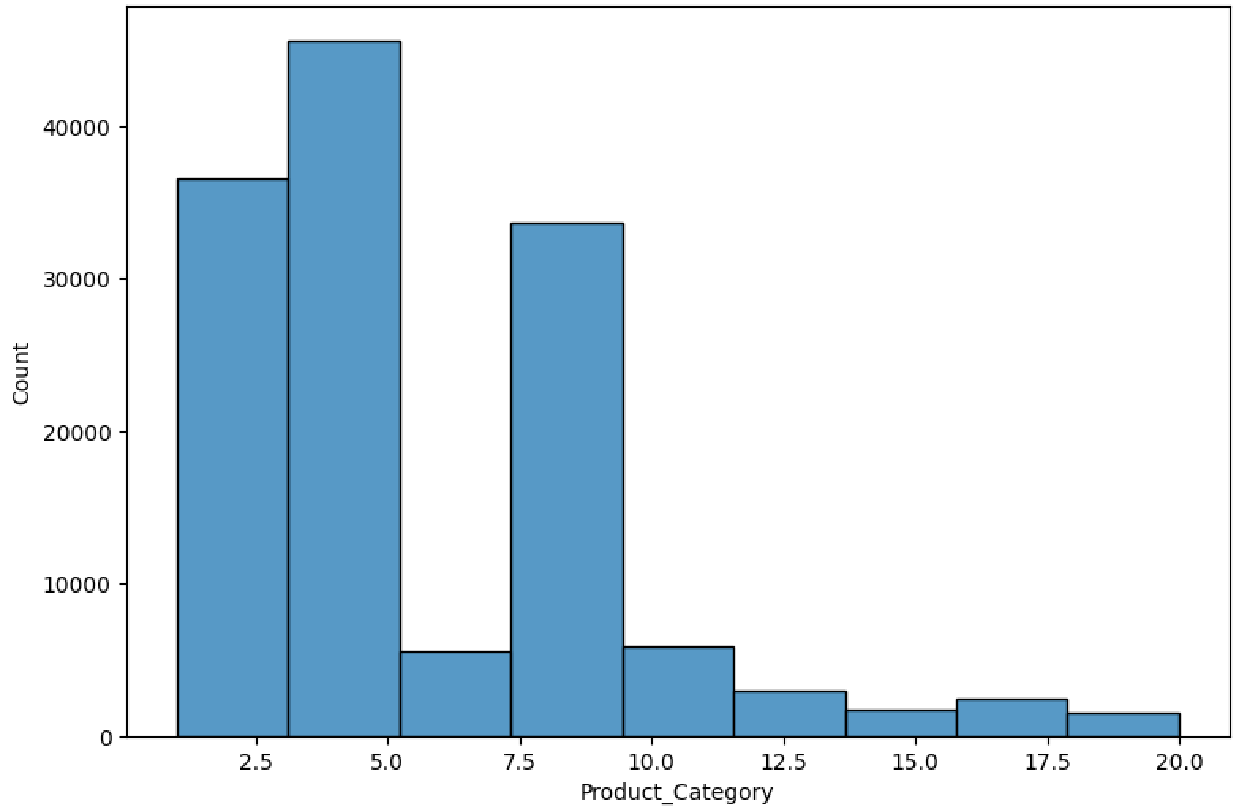


" Product Categories for male Customers "

1. We can clearly observe the product categories between 1 and 2 has the highest customers.
2. The categories from 2-5 stands the second highest for the male customers.
3. the Product_category from 17-20 has recorded the lowest customers count.

```
'''' Plotting the histogram of Product_category for gender Female '''

plt.figure(figsize = (9,6))
sns.histplot(data_women['Product_Category'],bins = 9)
```

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



" Product Categories for Female Customers "

1. We can clearly observe the product categories between 3 - 5 has the highest customers.
2. The categories from 1 and 2 stands the second highest for the male customers.
3. the Product_category from 17-20 has recorded the lowest customers count.
4. The product categories 7 -10 also recorded good number of customers.

4. How does gender affect the amount spent?

```
data_men
```

```
        User_ID Product_ID Gender   Age  Occupation City_Category  \
0       1000002  P00285442      M   55+          16             C
1       1000003  P00193542      M 26-35          15             A
2       1000004  P00184942      M 46-50           7             B
3       1000004  P00346142      M 46-50           7             B
4       1000004   P0097242      M 46-50           7             B
...         ...        ...    ...   ...         ...           ...
414254  1006023  P00370853      M 26-35           0             C
414255  1006024  P00372445      M 26-35          12             A
414256  1006026  P00371644      M 36-45           6             C
414257  1006032  P00372445      M 46-50           7             A
414258  1006033  P00372445      M 51-55          13             B
```

```
       Stay_In_Current_City_Years  Marital_Status  Product_Category 
Purchase
0                              4+               0                 8 
7969
1                               3               0                 1 
15227
2                               2               1                 1 
19215
3                               2               1                 1 
15854
4                               2               1                 1 
15686
...                           ...             ...               ... 
...
414254                          2               1                19 
61
414255                          0               1                20 
121
414256                          1               1                20 
494
414257                          3               0                20 
473
414258                          1               1                20 
368

[414259 rows x 10 columns]
```

```
''' Filtering the data for purchases made by Male customers '''

men_amount  = data_men['Purchase']

men_amount
```

```
0          7969
1         15227
2         19215
3         15854
4         15686
          ...  
414254       61
414255      121
414256      494
414257      473
414258      368
Name: Purchase, Length: 414259, dtype: int64
```

```
''' Calculating the mean for the total population of gender Male '''

mu_men_population = men_amount.mean()
```

```
''' Round the mean value '''

round(mu_men_population,2)

9437.53

''' Getting the number of records in the dataframe for grnder male '''

len(men_amount)

414259

''' Calculating the standard deviation for the total population '''

sigma_men_population = men_amount.std()

''' Round the satandard deviation upto 2 decimals '''

round(sigma_men_population,2)

5092.19

''' Now we are calculating the 95 % confidence interval for whole data
of gender 'Male' '''

mu = mu_men_population

sigma = sigma_men_population

n = 414259        # male candidate population

std_error = sigma / n

# confidence interval = 95%

std_error

0.012292276594541025

''' 95 confidence Interval for total population of Male '''

CI_m = np.round(norm.interval(0.95,loc = mu,scale = std_error),5)

CI_m

array([9437.50195, 9437.55013])
```

We can observe that the 95% confidence interval has the variance of very low as the populationn size is huge.

.

```
''' now we are calculating the 95 CI for 300 sample of MAle: '''

men_sample_300  = [np.mean(men_amount.sample(300)) for i in
range(1000)]

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference

n1 = 300

mu_men_300 = np.mean(men_sample_300)

sigma_men_300 = np.std(men_sample_300)

men_sample_300_std = sigma_men_300 / n1

CI_m300 = norm.interval(0.95,loc = mu_men_300,scale =
men_sample_300_std)

CI_m300

(9427.66669892937, 9431.477081070632)
```

We can observe thet the Confidence interval is larger here for the sample size of population customers.

.

bootstrapping method for calculating the 95% CI for sample of 300 Male customers

```
boot_300_men = []

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference

for i in range(1000):
  bootstrap_survey = np.random.choice(men_amount,300)
  boot_mean = np.mean(bootstrap_survey)
  boot_300_men.append(boot_mean)

np.mean(boot_300_men)

9442.52139

x1 = np.percentile(boot_300_men,95)   #  95 percentile

x2 = np.percentile(boot_300_men,5)    # 5 percentile

np.round((x1,x2),2)

array([9931.98, 8937.69])
```

We can observe thet the Confidence interval is larger here for the sample size of population customers.

```
```

.

```
''' now we are calculating the 95 CI for 3000 sample of MAle: '''

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference

men_sample_3000  = [np.mean(men_amount.sample(3000)) for i in
range(1000)]

n2 = 3000

mu_men_3000 = np.mean(men_sample_3000)

sigma_men_3000 = np.mean(men_sample_3000)

men_sample_3000_std = sigma_men_3000 / n2

CI_m3000 = norm.interval(0.95,loc = mu_men_3000,scale =
men_sample_3000_std)

CI_m3000

(9436.213434558947, 9448.551254107715)
```

We can observe that the Confidence Interval for has brrn decreased for the sample size of 3000 than comapsred to 300 sample.:

.

.Bootstrapping method for 3000 sample customers of Gender male:

```
boot_3000_men = []

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference


for i in range(1000):
  bootstrap_sample = np.random.choice(men_amount,3000)
  bootstrap_mean = np.mean(bootstrap_sample)
  boot_3000_men.append(bootstrap_mean)

np.mean(boot_3000_men)

9440.131455333334
```

```
x3 = np.percentile(boot_3000_men,95)

x4 = np.percentile(boot_3000_men,5)

np.round((x3,x4),3)

array([9592.32 , 9290.707])
```

We can observe that the Confidence Interval for has been decreased for the sample size of 3000 than comapsred to 300 sample.

.

```
'''  now we are calculating the 95 CI for 30000 sample of MAle: '''

men_sample_30000  = [np.mean(men_amount.sample(30000)) for i in range(1000)]

# calculating the sample of 30000 customers with repitition of 1000 sets to get the varinace diference


n3 = 30000

mu_men_30000 = np.mean(men_sample_30000)

sigma_men_30000 = np.mean(men_sample_30000)

men_sample_30000_std = sigma_men_30000 / n3

CI_m30000 = norm.interval(0.95,loc = mu_men_30000,scale = men_sample_30000_std)

CI_m30000

(9437.53719561504, 9438.770425051627)
```

1. We can observe that the Confidence Interval for has been decreased for the sample size of 3000 than comapsred to 3000 sample.
2. almost has the near Ci as population.

Conclusion:

Hence we can conclude from the above three sample tests of 300,3000 and 30000 that increase in the sample size will decrease the variance in the data.

Boot Strapping method for sample 30000 gender male:

```
boot_men_30000 = []
for i in range(1000):
  bootstrap_sample = np.random.choice(men_amount,30000)
  bootstrap_mean = np.mean(bootstrap_sample)
  boot_men_30000.append(bootstrap_mean)

x5 = np.percentile(boot_men_30000,95)

x6 = np.percentile(boot_men_30000,5)

np.round((x5,x6),2)
```

```
array([9485.72, 9386.33])
```

We can observe from the above the Confidence Interval for population is similar to the Confidence Interval for sample 30000.

```
print('CI for population:',CI_m,
      'CI for sample - 300 :',
      CI_m300,'CI for sample - 3000 :',CI_m3000,
      'Ci for sample 30000 :',CI_m30000,sep = '\n ')
```

```
CI for population:
 [9437.50195 9437.55013]
 CI for sample - 300 :
 (9427.66669892937, 9431.477081070632)
 CI for sample - 3000 :
 (9436.213434558947, 9448.551254107715)
 Ci for sample 30000 :
 (9437.53719561504, 9438.770425051627)
```

Hence we can say that sample size increases,variance decreases.

1. We can observe that the Confidence Interval for has been decreased for the sample size of 3000 than comapsred to 3000 sample.
2. almost has the near CI as population.

Conclusion:

Hence we can conclude from the above three sample tests of 300,3000 and 30000 that increase in the sample size will decrease the variance in the data.

.

```
data_women
```

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

```
14       1000006  P00231342      F  51-55          9           A
...          ...        ...    ...    ...        ...         ...
550061  1006029  P00372445      F  26-35          1           C
550064  1006035  P00375436      F  26-35          1           C
550065  1006036  P00375436      F  26-35         15           B
550066  1006038  P00375436      F    55+          1           C
550067  1006039  P00371644      F  46-50          0           B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
14                              1               0                 5
5378
...                           ...             ...               ...
...
550061                          1               1                20
599
550064                          3               0                20
371
550065                         4+               1                20
137
550066                          2               0                20
365
550067                         4+               1                20
490

[135809 rows x 10 columns]

''' filtering the data for gender Female of purchase column '''

women_amount = data_women['Purchase']

''' Checking the length of the dataframe '''

len(women_amount)

135809

''' Calculating the mean for the population of gender female '''

mu_women_population = np.mean(women_amount)

nw = 135809
```

```python
''' Calculating the standard deviation of population for gender female
'''

sigma_women_population = np.std(women_amount)

''' Calculating the standard error for the population for gnder female
'''

std_women_population = sigma_women_population/nw

''' Calculating the Confidence Interval for whole poulation of Gender
Female '''

CI_w = norm.interval(0.95,loc = mu_women_population,scale =
std_women_population)

CI_w
```

```
(8734.49696580379, 8734.634564507161)
```

We can observe that the 95% Confidence interval is very low for the population as the size of the data is huge.

.

```python
''' Calculating the 95% CI for 300 sample of Gender Women: '''

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference

data_300_women = [np.mean(women_amount.sample(300)) for i in
range(1000)]


n4 = 300

mu_women_300  = np.mean(data_300_women)

sigma_women_300 = np.std(data_300_women)

std_women_300 = sigma_women_300 / n4

CI_w300 = norm.interval(0.95,loc = mu_women_300,scale = std_women_300)

CI_w300
```

```
(8735.590221977598, 8739.164704689063)
```

From the above observation the Confidence interval is larger than the population Confidence interval for the sample size of 300.

.

Bootstrapping method for 300 samples of gender Female:

```
boot_300_women = []

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference

for i in range(1000):
  bootstrap_sample = np.random.choice(women_amount,300)
  bootstrap_mean = np.mean(bootstrap_sample)
  boot_300_women.append(bootstrap_mean)

y1 = np.percentile(boot_300_women,95)

y2 = np.percentile(boot_300_women,5)

np.round((y1,y2),2)
```

```
array([9198.15, 8271.76])
```

From the above observation the Confidence interval is larger than the population Confidence interval for the sample size of 300.

.

```
''' Calculating the CI for 3000 sample of Gender Women: '''

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference


data_3000_women = [np.mean(women_amount.sample(3000)) for i in
range(1000)]


n5 = 3000

mu_women_3000  = np.mean(data_3000_women)

sigma_women_3000 = np.std(data_3000_women)

std_women_3000 = sigma_women_3000 / n5

CI_w3000 = norm.interval(0.95,loc = mu_women_3000,scale =
std_women_3000)

CI_w3000
```

```
(8735.63463085667, 8735.744587809995)
```

Here from the above observation the confidence interval for the sample size of 3000 has beeen decreased than the sample size of 300.

.

Bootstrapping method for sample 3000 for gender female:

```
boot_3000_women = []

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference

for i  in range(1000):
  bootstarp_sample = np.random.choice(women_amount,3000)
  bootstarp_mean = np.mean(bootstrap_sample)
  boot_3000_women.append(bootstrap_mean)

y3 = np.percentile(boot_3000_women,95)

y4 = np.percentile(boot_3000_women,5)

np.round((y3,y4),4)

array([8960.95, 8960.95])
```

Here from the above observation the confidence interval for the sample size of 3000 has beeen decreased than the sample size of 300.

.

```
''' Calculating the CI for 30000 sample of Gender Women: '''

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference

data_30000_women = [np.mean(women_amount.sample(30000)) for i in
range(1000)]

n5 = 30000

mu_women_30000  = np.mean(data_30000_women)

sigma_women_30000 = np.std(data_30000_women)

std_women_30000 = sigma_women_3000 / n5

CI_w30000 = norm.interval(0.95,loc = mu_women_30000,scale =
std_women_30000)

CI_w30000

(8733.089552585669, 8733.100548281)

print('CI for population:',CI_w,
      'CI for sample - 300 :',
```

```
      CI_w300,'CI for sample - 3000 :',CI_w3000,
      'Ci for sample 30000 :',CI_w30000,sep = '\n ')

CI for population:
 (8734.49696580379, 8734.634564507161)
 CI for sample - 300 :
 (8735.590221977598, 8739.164704689063)
 CI for sample - 3000 :
 (8735.63463085667, 8735.744587809995)
 Ci for sample 30000 :
 (8733.089552585669, 8733.100548281)
```

1. We can observe that the Confidence Interval for has been decreased for the sample size of 3000 than comapsred to 3000 sample.
2. almost has the near CI as population.

Conclusion:

Hence we can conclude from the above three sample tests of 300,3000 and 30000 that increase in the sample size will decrease the variance in the data.

.

5. How does Marital_Status affect the amount spent?

```
data

        User_ID Product_ID Gender   Age  Occupation City_Category  \
0       1000001  P00069042      F  0-17          10             A
1       1000001  P00248942      F  0-17          10             A
2       1000001  P00087842      F  0-17          10             A
3       1000001  P00085442      F  0-17          10             A
4       1000002  P00285442      M   55+          16             C
...         ...        ...    ...   ...         ...           ...
550063  1006033  P00372445      M 51-55          13             B
550064  1006035  P00375436      F 26-35           1             C
550065  1006036  P00375436      F 26-35          15             B
550066  1006038  P00375436      F   55+           1             C
550067  1006039  P00371644      F 46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
```

```
1422
3                              2               0                12
1057
4                             4+               0                 8
7969
...                          ...             ...               ...
...
550063                         1               1                20
368
550064                         3               0                20
371
550065                        4+               1                20
137
550066                         2               0                20
365
550067                        4+               1                20
490

[550068 rows x 10 columns]

''' Filtering data on the column Marital status == 1 (Married) '''

data_mrg = data[data['Marital_Status'] == 1]

data_mrg

        User_ID Product_ID Gender    Age  Occupation City_Category  \
6       1000004  P00184942      M  46-50           7             B
7       1000004  P00346142      M  46-50           7             B
8       1000004   P0097242      M  46-50           7             B
9       1000005  P00274942      M  26-35          20             A
10      1000005  P00251242      M  26-35          20             A
...         ...        ...    ...    ...         ...           ...
550060  1006026  P00371644      M  36-45           6             C
550061  1006029  P00372445      F  26-35           1             C
550063  1006033  P00372445      M  51-55          13             B
550065  1006036  P00375436      F  26-35          15             B
550067  1006039  P00371644      F  46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
6                              2               1                 1
19215
7                              2               1                 1
15854
8                              2               1                 1
15686
9                              1               1                 8
7871
10                             1               1                 5
```

```
5254
...                           ...             ...               ...
...
550060                          1               1                20
494
550061                          1               1                20
599
550063                          1               1                20
368
550065                         4+               1                20
137
550067                         4+               1                20
490

[225337 rows x 10 columns]
```

```python
''' Filtering the data on the column Marital Status == 0 (Single) '''


data_single = data[data['Marital_Status'] == 0]

data_single
```

```
        User_ID Product_ID Gender    Age  Occupation City_Category  \
0       1000001  P00069042      F   0-17          10             A
1       1000001  P00248942      F   0-17          10             A
2       1000001  P00087842      F   0-17          10             A
3       1000001  P00085442      F   0-17          10             A
4       1000002  P00285442      M    55+          16             C
...         ...        ...    ...    ...         ...           ...
550056  1006022  P00375436      M  26-35          17             C
550059  1006025  P00370853      F  26-35           1             B
550062  1006032  P00372445      M  46-50           7             A
550064  1006035  P00375436      F  26-35           1             C
550066  1006038  P00375436      F    55+           1             C

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
4                              4+               0                 8
7969
...                           ...             ...               ...
...
```

```
550056                                          4+                0                 20
254
550059                                           1                0                 19
48
550062                                           3                0                 20
473
550064                                           3                0                 20
371
550066                                           2                0                 20
365

[324731 rows x 10 columns]
```

```
''' Filtering data of Purchase for Married customers '''

data_mrg1 = data_mrg['Purchase']

data_mrg1.info()
```

```
<class 'pandas.core.series.Series'>
Int64Index: 225337 entries, 6 to 550067
Series name: Purchase
Non-Null Count   Dtype
--------------   -----
225337 non-null  int64
dtypes: int64(1)
memory usage: 3.4 MB
```

```
mu_mrg1_population = np.mean(data_mrg1)

n0 = 225337

sigma_mrg1_population = np.std(data_mrg1)

std_mrg1_population = sigma_mrg1_population/ n0

data_mrg1.index.size
```

```
225337
```

```
CI_m = norm.interval(0.95,loc = mu_mrg1_population,scale =
std_mrg1_population )

CI_m
```

```
(9261.13093758967, 9261.218210575076)
```

Here the Variance between the Coinfidence Interval of population for Married people is very low as the size of the data is huge.

.

```python
'''  calculating the 95 CI for marriage status 1(YES) for 300 samples:
'''

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference


mrg_300 = [np.mean(data_mrg1.sample(300)) for i in range(1000)]


mrg_300_mean = np.mean(mrg_300)

mrg_300_sigma = np.std(mrg_300)

mrg_300_std = mrg_300_sigma/300

CI_m300 = norm.interval(0.95,loc = mrg_300_mean,scale = mrg_300_std)

CI_m300
```

```
(9255.96167564408, 9259.586171022589)
```

Here we can observe that the Confidence Interval for sample size of 300 is larger than the confidence interval for whole population

.

```python
''' calculating the 95 CI for marriage status 1(YES) for 3000 samples:
'''

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference


mrg_3000 = [np.mean(data_mrg1.sample(3000)) for i in range(1000)]

mrg_3000_mean = np.mean(mrg_3000)

mrg_3000_sigma = np.std(mrg_3000)

mrg_3000_std = mrg_3000_sigma/3000

CI_m3000 = norm.interval(0.95,loc = mrg_3000_mean,scale =
mrg_3000_std)

CI_m3000
```

```
(9258.8940011005, 9259.013300232837)
```

Here the Confidence Interval has been decreased for the sample size of 3000 than compared to sample size of 300

.

```
''' calculating the 95 CI for marriage status 1(YES) for 30000
samples: '''

# calculating the sample of 30000 customers with repitition of 1000
sets to get the varinace diference


mrg_30000 = [np.mean(data_mrg1.sample(30000)) for i in range(1000)]

mrg_30000_mean = np.mean(mrg_30000)

mrg_30000_sigma = np.std(mrg_30000)

mrg_30000_std = mrg_30000_sigma/30000

CI_m30000  = norm.interval(0.95,loc = mrg_30000_mean,scale =
mrg_30000_std)

CI_m30000
```

```
(9262.35123522168, 9262.354753778322)
```

Here the Confidence interval for the sample size of 30000 is very low as like the Confidence interval for the Population.

Hence we can observe that the increase in sample size , decreases the variance.

.

```
''' bootstrap for marraiage 1(yes) for 300 samples '''


boot_300_mrg = []

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference


for i in range(1000):
  bootstrap_sample = np.random.choice(data_mrg1,300)
  bootstarp_mean = np.mean(bootstarp_sample)
  boot_300_mrg.append(bootstrap_mean)


x1 = np.percentile(boot_300_mrg,95)

x2 = np.percentile(boot_300_mrg,5)

np.round((x1,x2),2)
```

```
array([8960.95, 8960.95])
```

```
''' bootstrap for maraaiage 1(yes) for 3000 samples '''

boot_3000_mrg = []

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference


for i in range(1000):
  bootstrap_sample = np.random.choice(data_mrg1,3000)
  bootstarp_mean = np.mean(bootstarp_sample)
  boot_3000_mrg.append(bootstrap_mean)


x3 = np.percentile(boot_3000_mrg,95)

x4 = np.percentile(boot_3000_mrg,5)

(x3,x4)
```

```
(8960.95, 8960.95)
```

```
''' bootstrap for maraaiage 1(yes) for 30000 samples '''

boot_30000_mrg = []

# calculating the sample of 30000 customers with repitition of 1000
sets to get the varinace diference


for i in range(1000):
  bootstrap_sample = np.random.choice(data_mrg1,30000)
  bootstarp_mean = np.mean(bootstarp_sample)
  boot_30000_mrg.append(bootstrap_mean)



x5 = np.percentile(boot_30000_mrg,95)

x6 = np.percentile(boot_30000_mrg,5)

(x5,x6)
```

```
(8960.95, 8960.95)
```

```
print('CI for population:',CI_m,
      'CI for sample - 300 :',
```

```
      CI_m300,'CI for sample - 3000 :',CI_m3000,
      'Ci for sample 30000 :',CI_m30000,sep = '\n ')
```

```
CI for population:
 (9261.13093758967, 9261.218210575076)
 CI for sample - 300 :
 (9255.96167564408, 9259.586171022589)
 CI for sample - 3000 :
 (9258.8940011005, 9259.013300232837)
 Ci for sample 30000 :
 (9262.35123522168, 9262.354753778322)
```

1. We can observe that the Confidence Interval for has been decreased for the sample size of 3000 than comapsred to 3000 sample.
2. almost has the near CI as population.

Conclusion:

Hence we can conclude from the above three sample tests of 300,3000 and 30000 that increase in the sample size will decrease the variance in the data.

.

```
data_single
```

```
        User_ID Product_ID Gender    Age  Occupation City_Category  \
0       1000001  P00069042      F   0-17          10             A
1       1000001  P00248942      F   0-17          10             A
2       1000001  P00087842      F   0-17          10             A
3       1000001  P00085442      F   0-17          10             A
4       1000002  P00285442      M    55+          16             C
...         ...        ...    ...    ...         ...           ...
550056  1006022  P00375436      M  26-35          17             C
550059  1006025  P00370853      F  26-35           1             B
550062  1006032  P00372445      M  46-50           7             A
550064  1006035  P00375436      F  26-35           1             C
550066  1006038  P00375436      F    55+           1             C

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
4                              4+               0                 8
7969
```

```
...                              ...              ...                  ...
...
550056                            4+                0                   20
254
550059                             1                0                   19
48
550062                             3                0                   20
473
550064                             3                0                   20
371
550066                             2                0                   20
365

[324731 rows x 10 columns]

''' Filtering the data for the marrital stats == 0 (Single) for
purchase column '''

data_single1 = data['Purchase']

data_single.index.size

324731

''' Claculating the Confidence Interval for whole population of Single
customers '''

mu_single_population = data_single1.mean()

sigma_single_population = data_single1.std()

std_single_population = sigma_single_population / 324731

CI_s  = norm.interval(0.95,loc = mu_single_population,scale =
std_single_population)

CI_s

(9263.938395473995, 9263.999030444256)
```

Here we can observe that the Confidence Interval for the population of Single customers is with very low variance as the data size is huge.

.

```
''' calculating the 95 CI for single status 0(NO) for 300 samples: '''

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference
```

```
single_300 = [np.mean(data_single1.sample(300)) for i in range(1000)]


single_300_mean = np.mean(single_300)

single_300_sigma = np.std(single_300)

single_300_std = mrg_300_sigma/300

CI_s300  = norm.interval(0.95,loc = single_300_mean,scale =
single_300_std)

CI_s300

(9263.232042310745, 9266.856537689255)
```

Here the Confidence interval is larger than the population of the single customers as the sample size is loww of 300.

.

```
''' calculating the 95 CI for single status 0(NO) for 3000 samples:
'''

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference


single_3000 = [np.mean(data_single1.sample(3000)) for i in
range(1000)]

single_3000_mean = np.mean(single_3000)

single_3000_sigma = np.std(single_3000)

single_3000_std = mrg_3000_sigma/3000

CI_s3000 = norm.interval(0.95,loc = single_3000_mean,scale =
single_3000_std)

CI_s3000

(9264.980225767165, 9265.099524899502)
```

From the above observation the Confidence interval is much samller for the sample size of 3000 than comapsred to the sample size of 300

.

```
''' calculating the 95 CI for single status 0(NO) for 30000 samples:
'''
```

```
# calculating the sample of 30000 customers with repitition of 1000
sets to get the varinace diference

single_30000 = [np.mean(data_single1.sample(30000)) for i in
range(1000)]

single_30000_mean = np.mean(single_30000)

single_30000_sigma = np.std(single_30000)

single_30000_std = mrg_30000_sigma/30000

CI_s30000 = norm.interval(0.95,loc = single_30000_mean,scale =
single_30000_std)

CI_s30000
```

```
(9264.918324255013, 9264.921842811655)
```

Here the Confidence interval for sample size of 30000 is much smaller than the other sample sizes of 300,3000 and much similar to the population confidence Interval

.

```
''' bootstrap for single 0(NO) for 300 samples '''


boot_300_single = []

# calculating the sample of 300 customers with repitition of 1000 sets
to get the varinace diference


for i in range(1000):
  bootstrap_sample = np.random.choice(data_single1,300)
  bootstarp_mean = np.mean(bootstarp_sample)
  boot_300_single.append(bootstrap_mean)


y1 = np.percentile(boot_300_single,95)

y2 = np.percentile(boot_300_single,5)

np.round((y1,y2),2)
```

```
array([8960.95, 8960.95])
```

```
''' bootstrap for single 0(NO) for 3000 samples '''
```

```
boot_3000_single = []

# calculating the sample of 3000 customers with repitition of 1000
sets to get the varinace diference

for i in range(1000):
  bootstrap_sample = np.random.choice(data_single1,3000)
  bootstarp_mean = np.mean(bootstarp_sample)
  boot_3000_single.append(bootstrap_mean)


y3 = np.percentile(boot_3000_single,95)

y4 = np.percentile(boot_3000_single,5)

np.round((y3,y4),2)
```

```
array([8960.95, 8960.95])
```

```
''' bootstrap for single 0(NO) for 30000 samples '''


boot_30000_single = []

# calculating the sample of 30000 customers with repitition of 1000
sets to get the varinace diference


for i in range(1000):
  bootstrap_sample = np.random.choice(data_single1,30000)
  bootstarp_mean = np.mean(bootstarp_sample)
  boot_30000_single.append(bootstrap_mean)


y5 = np.percentile(boot_30000_single,95)

y6 = np.percentile(boot_30000_single,5)

np.round((y5,y6),2)
```

```
array([8960.95, 8960.95])
```

```
print('CI for population:',CI_s,
      'CI for sample - 300 :',
      CI_s300,'CI for sample - 3000 :',CI_s3000,
      'Ci for sample 30000 :',CI_s30000,sep = '\n ')
```

```
CI for population:
 (9263.938395473995, 9263.999030444256)
```

```
CI for sample - 300 :
(9263.232042310745, 9266.856537689255)
CI for sample - 3000 :
(9264.980225767165, 9265.099524899502)
Ci for sample 30000 :
(9264.918324255013, 9264.921842811655)
```

1. We can observe that the Confidence Interval for has been decreased for the sample size of 3000 than comapsred to 3000 sample.
2. almost has the near CI as population.

Conclusion:

Hence we can conclude from the above three sample tests of 300,3000 and 30000 that increase in the sample size will decrease the variance in the data.

.

6. How does Age affect the amount spent?

```
data

        User_ID Product_ID Gender   Age  Occupation City_Category  \
0       1000001  P00069042      F  0-17          10             A
1       1000001  P00248942      F  0-17          10             A
2       1000001  P00087842      F  0-17          10             A
3       1000001  P00085442      F  0-17          10             A
4       1000002  P00285442      M   55+          16             C
...         ...        ...    ...   ...         ...           ...
550063  1006033  P00372445      M 51-55          13             B
550064  1006035  P00375436      F 26-35           1             C
550065  1006036  P00375436      F 26-35          15             B
550066  1006038  P00375436      F   55+           1             C
550067  1006039  P00371644      F 46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
4                              4+               0                 8
7969
...                           ...             ...               ...
...
```

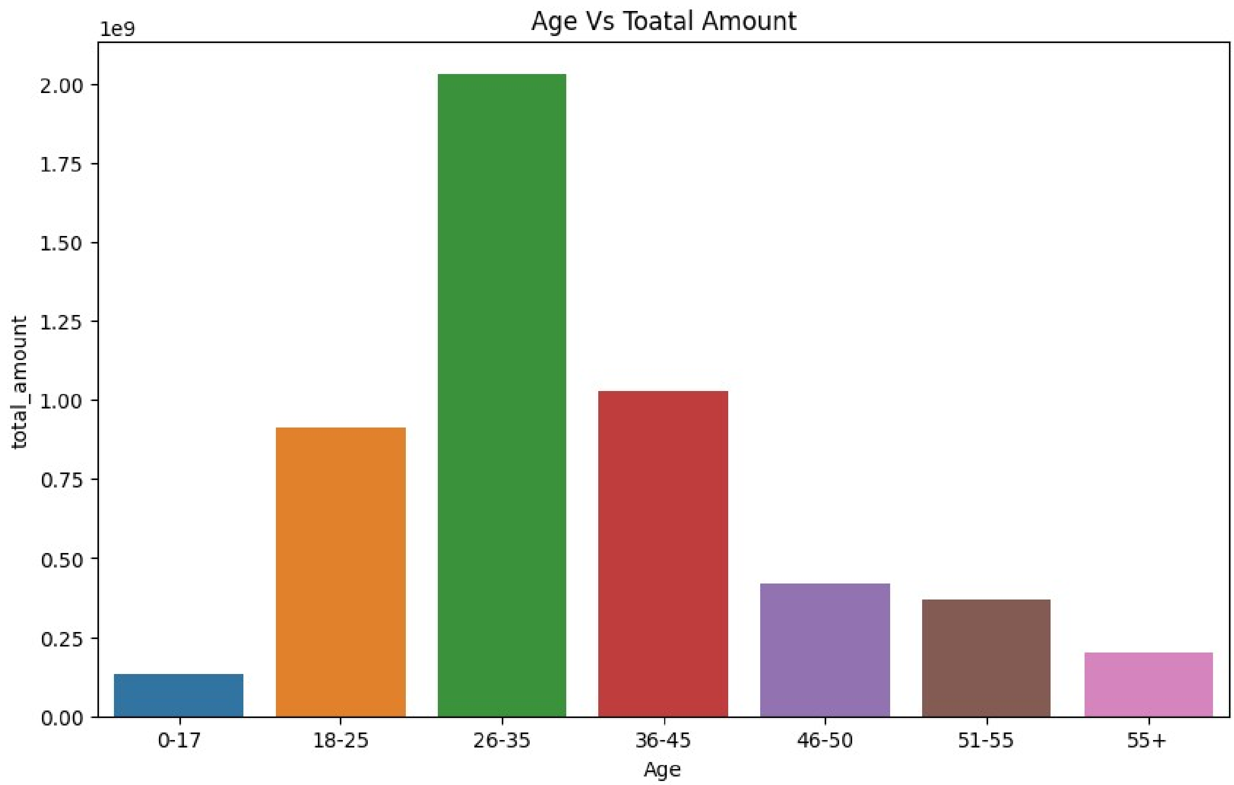
```
550063                                      1               1                20
368
550064                                      3               0                20
371
550065                                     4+               1                20
137
550066                                      2               0                20
365
550067                                     4+               1                20
490

[550068 rows x 10 columns]
```

```python
''' Grouping the data by Age column and calculating the total amount
'''

c = data.groupby(['Age']).agg(total_amount =
('Purchase','sum')).reset_index()

c
```

```
     Age  total_amount
0   0-17     134913183
1  18-25     913848675
2  26-35    2031770578
3  36-45    1026569884
4  46-50     420843403
5  51-55     367099644
6    55+     200767375
```

```python
''' plotting the graph for age vs total amount '''

plt.figure(figsize = (10,6))
sns.barplot(
    x = 'Age',
    y = 'total_amount',
    data = c

)
plt.title('Age Vs Toatal Amount')
```

```
Text(0.5, 1.0, 'Age Vs Toatal Amount')
```




```
''' filtering the dataframe for the age between 46-50 '''

data_46_50 = data[data['Age'] == '46-50']

''' filtering data for column Purchase '''

data_46_501 = data_46_50['Purchase']

data_46_501
```

```
6         19215
7         15854
8         15686
52         5839
53        15912
          ...
550041      488
550043       48
550052      239
550062      473
550067      490
Name: Purchase, Length: 45701, dtype: int64
```

```
data_46_501.index.size
```

```
45701
```

```python
''' Calculating the Confidence Interval for the age 46-50 '''
d_46_50_mean  = data_46_501.mean()
d_46_50_sigma = data_46_501.std()
d_46_50_std = d_46_50_sigma/45701

CI_46_50 = norm.interval(0.95,loc = d_46_50_mean,scale = d_46_50_std)
CI_46_50
```

```
(9208.412670068861, 9208.838724867794)
```

```python
''' Calculating the Confidence Interval for the age 0-17 '''

data_0_17 = data[data['Age'] == '0-17']
data_0_17_ = data_0_17['Purchase']
d_0_17_mean  = data_0_17_.mean()
d_0_17_sigma = data_0_17_.std()

data_0_17_.index.size
```

```
15102
```

```python
d_0_17_std = d_0_17_sigma/15102
CI_0_17 = norm.interval(0.95,loc = d_0_17_mean,scale = d_0_17_std)
CI_0_17
```

```
(8932.801311120975, 8934.127969768973)
```

```python
''' Calculating the Confidence Interval for the age 18-25 '''

data_18_25 = data[data['Age'] == '18-25']
data_18_25_ = data_18_25['Purchase']
d_18_25_mean  = data_18_25_.mean()
d_18_25_sigma = data_18_25_.std()
data_18_25_.index.size
```

```
99660
```

```
d_18_25_std = d_18_25_sigma/99660

CI_18_25 = norm.interval(0.95,loc = d_18_25_mean,scale = d_18_25_std)

CI_18_25

(9169.564598737697, 9169.76261378488)

''' Calculating the Confidence Interval for the age 26-35 '''

data_26_35 = data[data['Age'] == '26-35']

data_26_35_ = data_26_35['Purchase']

d_26_35_mean  = data_26_35_.mean()

d_26_35_sigma = data_26_35_.std()

data_26_35_.index.size

219587

d_26_35_std = d_26_35_sigma/219587

CI_26_35 = norm.interval(0.95,loc = d_26_35_mean,scale = d_26_35_std)

CI_26_35

(9252.645910490797, 9252.735355248979)

''' Calculating the Confidence Interval for the age 36-45 '''


data_36_45 = data[data['Age'] == '36-45']

data_36_45_ = data_36_45['Purchase']

d_36_45_mean  = data_36_45_.mean()

d_36_45_sigma = data_36_45_.std()

data_36_45_.index.size

110013

d_36_45_std = d_36_45_sigma / 110013

CI_36_45  = norm.interval(0.95,loc = d_36_45_mean,scale = d_36_45_std)

CI_36_45

(9331.261207767262, 9331.440182068485)
```

```
''' Calculating the Confidence Interval for the age 51-55 '''

data_51_55 = data[data['Age'] == '51-55']

data_51_55_ = data_51_55['Purchase']

d_51_55_mean  = data_51_55_.mean()

d_51_55_sigma = data_51_55_.std()

data_51_55_.index.size

38501

d_51_55_std = d_51_55_sigma / 38501

CI_51_55 = norm.interval(0.95,loc = d_51_55_mean,scale = d_51_55_std)

CI_51_55

(9534.549049162046, 9535.067012758425)
```

.

```
''' Calculating the Confidence Interval for the age 55+ '''

data_55 = data[data['Age'] > '55']

data_55_ = data_55['Purchase']

d_55_mean  = data_55_.mean()

d_55_sigma = data_55_.std()

data_55_.index.size

21504

d_55_std = d_55_sigma / 21504

CI_55_ = norm.interval(0.95,loc = d_55_mean,scale = d_55_std)

CI_55_

(9335.823691046311, 9336.737227852498)
```

.

```
mu_dt = data['Purchase'].mean()
```

```
sigma_dt = data['Purchase'].std()


data['Purchase'].index.size

550068

std_dt = sigma_dt / 550068

CI_dt = norm.interval(0.95,loc = mu_dt,scale = std_dt)

CI_dt

(9263.950815122378, 9263.986610795873)

print('Confidence_interval for purchase  whole population :',CI_dt,
    'Confidence interval for age  0-17 :',CI_0_17,
      'Confidence_interval for age  17-25 :',CI_18_25,
      'Confidence_interval for age  26-35 :',CI_26_35,
      'Confidence_interval for age  36-45 :',CI_36_45,
      'Confidence_interval for age  46-50 :',CI_46_50,
      'Confidence_interval for age  51-55 :',CI_51_55,
      'Confidence_interval for age  55+ :',CI_55_,
      sep = ' \n \n')

Confidence_interval for purchase  whole population :

(9263.950815122378, 9263.986610795873)

Confidence interval for age  0-17 :

(8932.801311120975, 8934.127969768973)

Confidence_interval for age  17-25 :

(9169.564598737697, 9169.76261378488)

Confidence_interval for age  26-35 :

(9252.645910490797, 9252.735355248979)

Confidence_interval for age  36-45 :

(9331.261207767262, 9331.440182068485)

Confidence_interval for age  46-50 :

(9208.412670068861, 9208.838724867794)

Confidence_interval for age  51-55 :
```

```
(9534.549049162046, 9535.067012758425)

Confidence_interval for age  55+ :

(9335.823691046311, 9336.737227852498)
```

.

1. From the above oabservations of the confidence Intervals for the Populations and the ages 0-17,18-25,26-35,36-45.46-50,51-55,55+, it is clear that the variance in the data for the confidence interval lies in the size of the data.
2. More the size of the data less the varince in the Confidence interval.

.

```
data

        User_ID Product_ID Gender    Age  Occupation City_Category  \
0       1000001  P00069042      F   0-17          10             A
1       1000001  P00248942      F   0-17          10             A
2       1000001  P00087842      F   0-17          10             A
3       1000001  P00085442      F   0-17          10             A
4       1000002  P00285442      M    55+          16             C
...         ...        ...    ...    ...         ...           ...
550063  1006033  P00372445      M  51-55          13             B
550064  1006035  P00375436      F  26-35           1             C
550065  1006036  P00375436      F  26-35          15             B
550066  1006038  P00375436      F    55+           1             C
550067  1006039  P00371644      F  46-50           0             B

       Stay_In_Current_City_Years  Marital_Status  Product_Category
Purchase
0                               2               0                 3
8370
1                               2               0                 1
15200
2                               2               0                12
1422
3                               2               0                12
1057
4                              4+               0                 8
7969
...                           ...             ...               ...
...
550063                          1               1                20
368
550064                          3               0                20
371
550065                         4+               1                20
```

```
137
550066                          2               0                20
365
550067                         4+               1                20
490

[550068 rows x 10 columns]
```

Recommendations:

1. The data is of customers shopping at Walmart of different categories.
2. The Gender of MAle and FEmale has different purchasing power to each other when compared to total purchasing power.
3. Female customers predominantly incerased in purchasing power with later of age like from 25+.
4. Male customers are purchasing well but not in increasing of age, there are lot more ups and down in purchasing capacity over increasing ages.
5. As for the Confidence intervals calu;ating the male and female customers are almost equally purchasing.
6. The age plays a key role in the purchasing power of an customer, so it is recommended to concentrate on tha the ages where the purchasing decreses for both the genders Male and female.
7. The status like Married and Single also predominantly effecst the purchasinh power of both male and female customers, So the actions must be taken in order to attract the customers of both the categories by offering discounts and add-on,coupons and other Bussiness Strategies.
8. The city Category also plays important role to analyze the purchase power, City_Ctegory -- C has been leading for both amle and female customers than the othe rtwo city Categories A,B. Necessary steps must be taken to attarct the customers of different forms of age,marital status etc.
9. The male and female customers has the Product catrgory favrouite of nearly from 1-5 categories for both. the added discounts should applied to other Product Categories in order to increase the sales.
10. Walmrt is a Giant in the market, So it should be more careful according to needs of the customers in order to amintain their grip in the Bussiness and the market.
11. These are some of my recommendations for the given data for customers at Walmart.