Business Case - Walmart

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

Problem Statement:

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

In [182]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

In [2]:

```
df = pd.read_csv('walmart_data.txt')
df.head()
```

Out[2]:

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

In [3]:

```
df.shape
```

Out[3]:

```
(550068, 10)
```

In [4]:

```
df.info()
```

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

In [5]:

```
df.isnull().sum()/len(df)*100
```

Out[5]:

```
User_ID                       0.0
Product_ID                    0.0
Gender                        0.0
Age                           0.0
Occupation                    0.0
City_Category                 0.0
Stay_In_Current_City_Years    0.0
Marital_Status                0.0
Product_Category              0.0
Purchase                      0.0
dtype: float64
```

From the above it is clear that there are no null values in the data

In [6]:

```
df.describe()
```

Out[6]:

	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

Convert Gender and City_Category columns to numerical

In [7]:

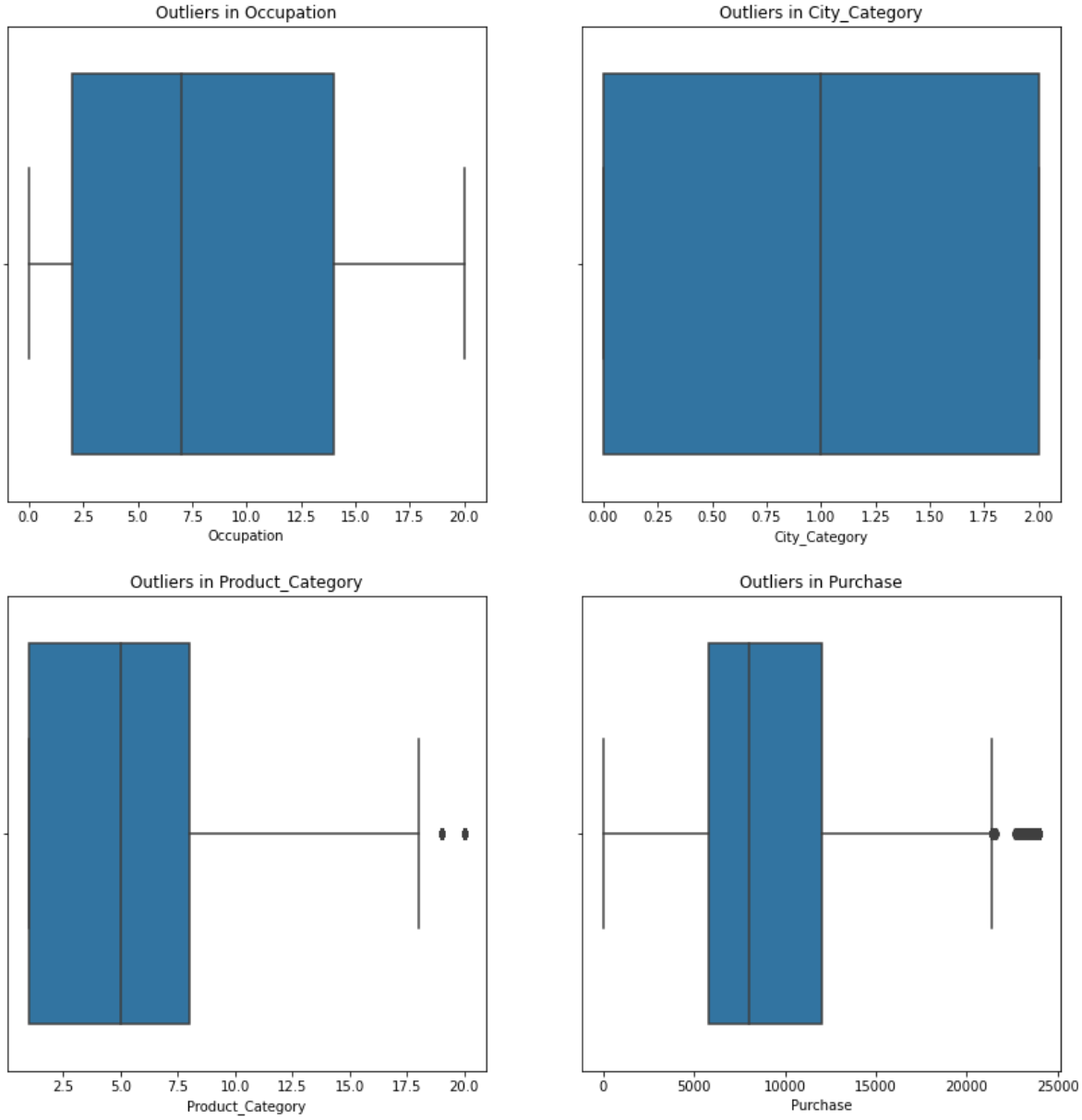
```
df['Gender'] = df['Gender'].apply(lambda x:1 if x=='M' else 0)
df['City_Category'] = df['City_Category'].apply(lambda x:0 if x=='A' else 1 if x=='B' else
```

Outlier Analysis

In [8]:

```
fig, axis = plt.subplots(nrows=2, ncols=2, figsize=(14, 10))
fig.subplots_adjust(top=1.2)
sns.boxplot(x='Occupation',data = df,ax=axis[0,0]).set_title('Outliers in Occupation')
sns.boxplot(x='City_Category',data = df,ax=axis[0,1]).set_title('Outliers in City_Category'
sns.boxplot(x='Product_Category',data = df,ax=axis[1,0]).set_title('Outliers in Product_Cat
sns.boxplot(x='Purchase',data = df,ax=axis[1,1]).set_title('Outliers in Purchase')

plt.show()
```



clearly there are outliers in product and purchase columns. We're using CLT to find out intervals, so there's no need to remove the outliers.

Gender Analysis

In [101]:

```
df.Gender.value_counts()
```

Out[101]:

```
1    414259
0    135809
Name: Gender, dtype: int64
```

The number of users that are actually present in the dataset. We need to extrapolate the information from these customers to 100million users.

In [106]:

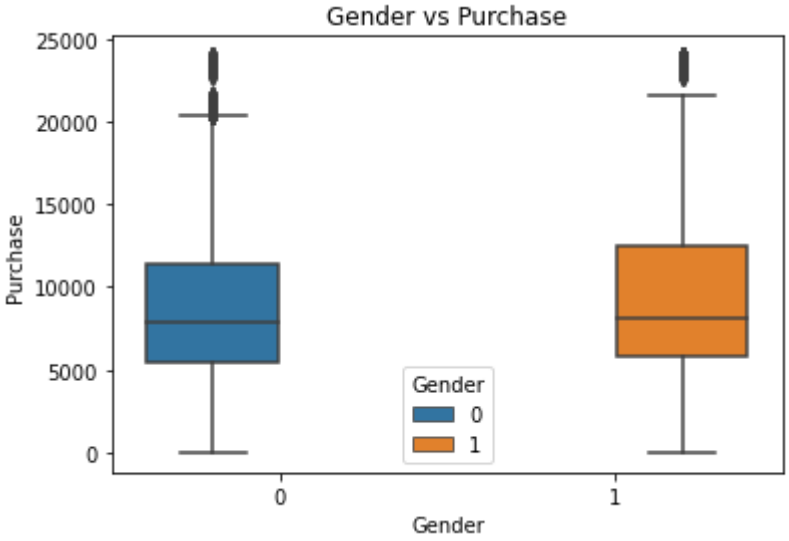
```
df.groupby('Gender')['User_ID'].nunique()
```

Out[106]:

```
Gender
0    1666
1    4225
Name: User_ID, dtype: int64
```

In [183]:

```
sns.boxplot(x='Gender', y='Purchase', data=df,hue='Gender')
plt.title('Gender vs Purchase')
plt.show()
```



We can see that the median of both males and female customers are almost in the same range.

In [110]:

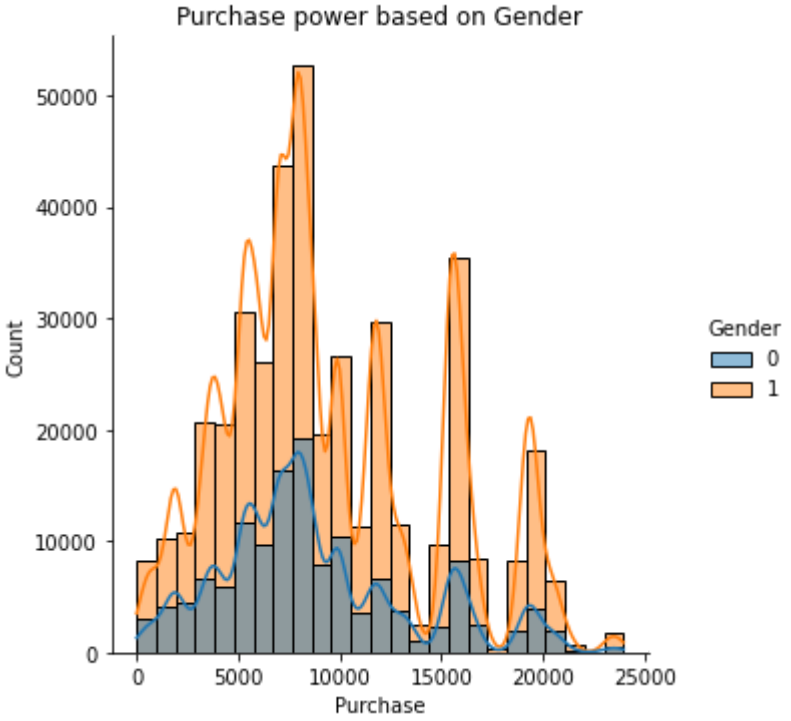
```
df.groupby('Gender')['Purchase'].describe()
```

Out[110]:

Gender	count	mean	std	min	25%	50%	75%	max
0	135809.0	8734.565765	4767.233289	12.0	5433.0	7914.0	11400.0	23959.0
1	414259.0	9437.526040	5092.186210	12.0	5863.0	8098.0	12454.0	23961.0

In [185]:

```
sns.displot(x='Purchase', bins=25, kde=True,hue='Gender', data=df)
plt.title('Purchase power based on Gender')
plt.show()
```



We can't say from here that male customers are buying more in number than females.

CLT Analysis

In [134]:

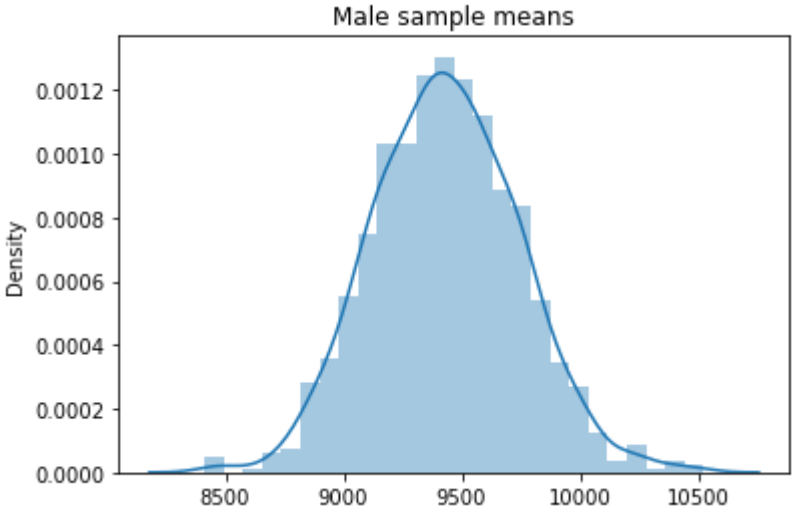
```
sample_size=300
male_sample_means=[df[df['Gender']==1].sample(sample_size, replace=True)['Purchase'].mean()
```

In [133]:

```
females_sample_means=[df[df['Gender']==0].sample(sample_size, replace=True)['Purchase'].mea
```

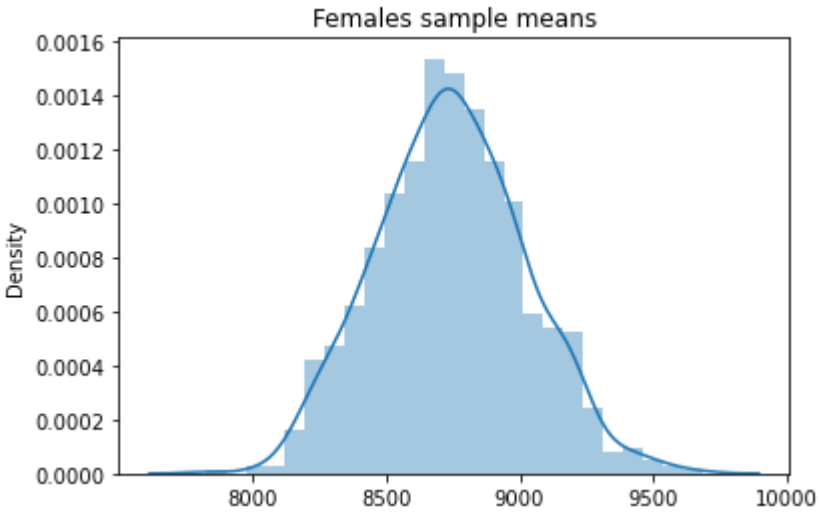
In [187]:

```
sns.distplot(male_sample_means)
plt.title('Male sample means')
plt.show()
```



In [188]:

```
sns.distplot(females_sample_means)
plt.title('Females sample means')
plt.show()
```



In [193]:

```
male_upper_limit= np.mean(male_sample_means) + (1.96 * np.std(male_sample_means))
male_lower_limit= np.mean(male_sample_means) - (1.96 * np.std(male_sample_means))
print(f'The mean of males purchasing lies between with 95% confidence :{(male_lower_limit,
female_upper_limit= np.mean(females_sample_means) + (1.96 * np.std(females_sample_means))
female_lower_limit= np.mean(females_sample_means) - (1.96 * np.std(females_sample_means))
print(f'The mean of females purchasing lies between with 95% confidence :{(female_lower_lim
```

```
The mean of males purchasing lies between with 95% confidence :(8825.4023123
32371, 10041.25884100096)
The mean of females purchasing lies between with 95% confidence :(8196.05167
684947, 9289.038843150527)
```

In [194]:

```
male_upper_limit= np.mean(male_sample_means) + (1.645 * np.std(male_sample_means))
male_lower_limit= np.mean(male_sample_means) - (1.645 * np.std(male_sample_means))
print(f'The mean of males purchasing lies between with 90% confidence :{(male_lower_limit,
female_upper_limit= np.mean(females_sample_means) + (1.645 * np.std(females_sample_means))
female_lower_limit= np.mean(females_sample_means) - (1.645 * np.std(females_sample_means))
print(f'The mean of females purchasing lies between with 90% confidence :{(female_lower_lim
```

```
The mean of males purchasing lies between with 90% confidence :(8923.1050691
00384, 9943.556084232947)
The mean of females purchasing lies between with 90% confidence :(8283.88100
2712948, 9201.20951728705)
```

In [195]:

```
male_upper_limit= np.mean(male_sample_means) + (2.58 * np.std(male_sample_means))
male_lower_limit= np.mean(male_sample_means) - (2.58 * np.std(male_sample_means))
print(f'The mean of males purchasing lies between with 99% confidence :{(male_lower_limit,
female_upper_limit= np.mean(females_sample_means) + (2.58 * np.std(females_sample_means))
female_lower_limit= np.mean(females_sample_means) - (2.58 * np.std(females_sample_means))
print(f'The mean of females purchasing lies between with 99% confidence :{(female_lower_lim
```

```
The mean of males purchasing lies between with 99% confidence :(8633.0984736
1438, 10233.56267971895)
The mean of females purchasing lies between with 99% confidence :(8023.18125
7689609, 9461.909262310388)
```

I can see the range is overlaping, so we can't decide if the males are buying more on the holiday season than females.

Married and Unmarried Analysis

In [138]:

```
df.Marital_Status.value_counts()
```

Out[138]:

```
0    324731
1    225337
Name: Marital_Status, dtype: int64
```

In [137]:

```
df.groupby('Marital_Status')['User_ID'].nunique()
```

Out[137]:

```
Marital_Status
0    3417
1    2474
Name: User_ID, dtype: int64
```

In [139]:

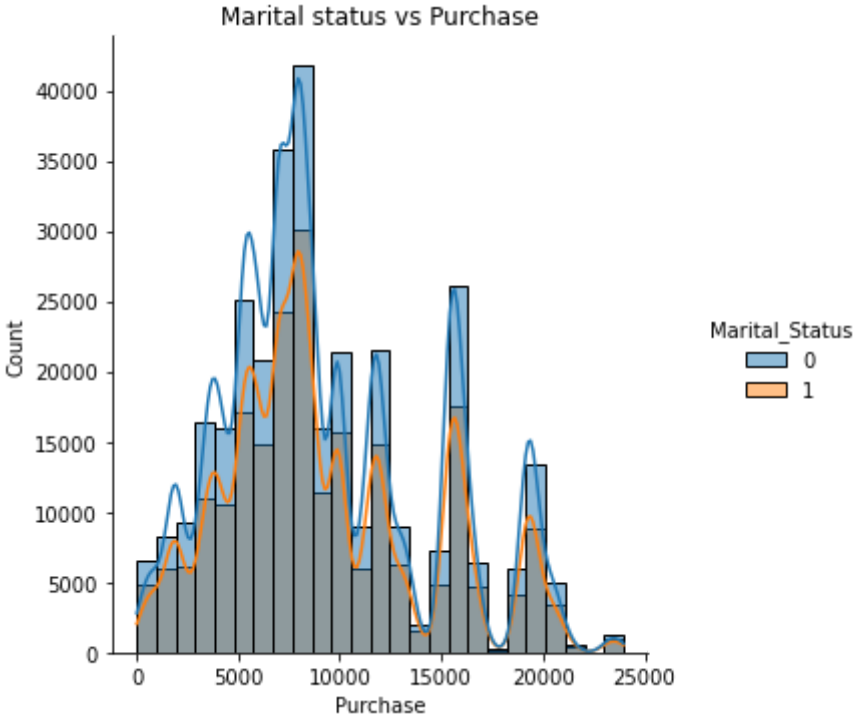
```
df.groupby('Marital_Status')['Purchase'].describe()
```

Out[139]:

Marital_Status	count	mean	std	min	25%	50%	75%	max
0	324731.0	9265.907619	5027.347859	12.0	5605.0	8044.0	12061.0	23961.0
1	225337.0	9261.174574	5016.897378	12.0	5843.0	8051.0	12042.0	23961.0

In [213]:

```
sns.displot(x='Purchase', bins=25, kde=True,hue='Marital_Status', data=df )
plt.title('Marital status vs Purchase')
plt.show()
```



We can't say that the Unmarried people are buying more in number yet.

In [147]:

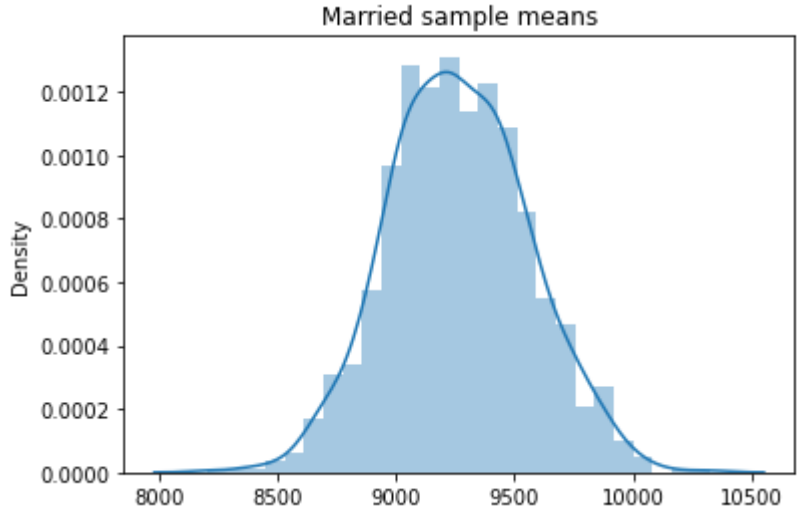
```
married_sample_means=[df[df['Marital_Status']==1].sample(sample_size, replace=True)['Purcha
```

In [144]:

```
unmarried_sample_means=[df[df['Marital_Status']==0].sample(sample_size, replace=True)['Purc
```

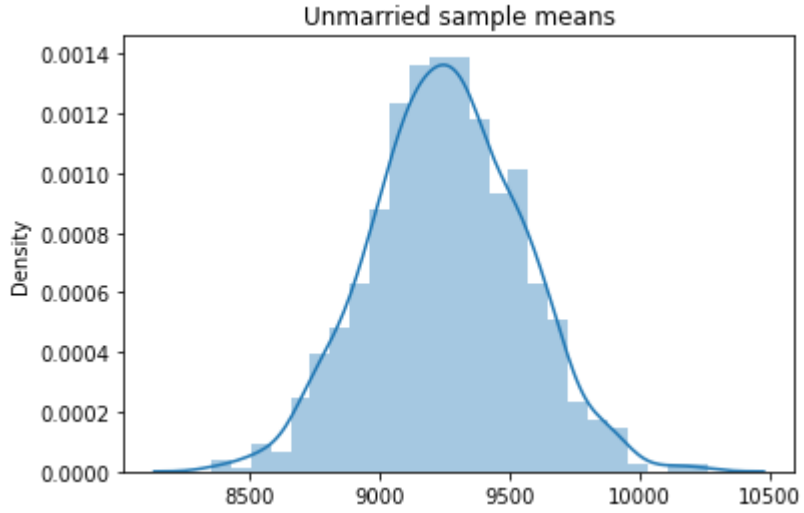
In [196]:

```
sns.distplot(married_sample_means)
plt.title('Married sample means')
plt.show()
```



In [197]:

```
sns.distplot(unmarried_sample_means)
plt.title('Unmarried sample means')
plt.show()
```



In [198]:

```
ed_upper_limit= np.mean(married_sample_means) + (1.96 * np.std(married_sample_means))
ed_lower_limit= np.mean(married_sample_means) - (1.96 * np.std(married_sample_means))
(f'The mean of married people purchasing lies between with 95% confidence is :{(married_lowe
ried_upper_limit= np.mean(unmarried_sample_means) + (1.96 * np.std(unmarried_sample_means))
ried_lower_limit= np.mean(unmarried_sample_means) - (1.96 * np.std(unmarried_sample_means))
(f'The mean of unmarried people purchasing lies between with 95% confidence is :{(unmarried_
```

```
The mean of married people purchasing lies between with 95% confidence is :
(8682.703617203531, 9845.036889463137)
The mean of unmarried people purchasing lies between with 95% confidence is
:(8690.264514727254, 9824.257878606079)
```

In [199]:

```
married_upper_limit= np.mean(married_sample_means) + (1.645 * np.std(married_sample_means))
married_lower_limit= np.mean(married_sample_means) - (1.645 * np.std(married_sample_means))
print(f'The mean of married people purchasing lies between with 90% confidence is :{(marrie
unmarried_upper_limit= np.mean(unmarried_sample_means) + (1.645 * np.std(unmarried_sample_m
unmarried_lower_limit= np.mean(unmarried_sample_means) - (1.645 * np.std(unmarried_sample_m
print(f'The mean of unmarried people purchasing lies between with 90% confidence is :{(unma
```

```
The mean of married people purchasing lies between with 90% confidence is :
(8776.105398010108, 9751.635108656561)
The mean of unmarried people purchasing lies between with 90% confidence is
:(8781.388981467517, 9733.133411865816)
```

In [200]:

```
married_upper_limit= np.mean(married_sample_means) + (2.58 * np.std(married_sample_means))
married_lower_limit= np.mean(married_sample_means) - (2.58 * np.std(married_sample_means))
print(f'The mean of married people purchasing lies between with 99% confidence is :{(marrie
unmarried_upper_limit= np.mean(unmarried_sample_means) + (2.58 * np.std(unmarried_sample_me
unmarried_lower_limit= np.mean(unmarried_sample_means) - (2.58 * np.std(unmarried_sample_me
print(f'The mean of unmarried people purchasing lies between with 99% confidence is :{(unma
```

```
The mean of married people purchasing lies between with 99% confidence is :
(8498.865191489003, 10028.875315177665)
The mean of unmarried people purchasing lies between with 99% confidence is
:(8510.908421460706, 10003.613971872626)
```

Can't say either that Married people are buying more in number than Unmarried or Vice versa

Age Analysis

In [159]:

```
df.Age.value_counts()
```

Out[159]:

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

In [166]:

```
df.groupby('Age')['User_ID'].nunique()
```

Out[166]:

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

In [167]:

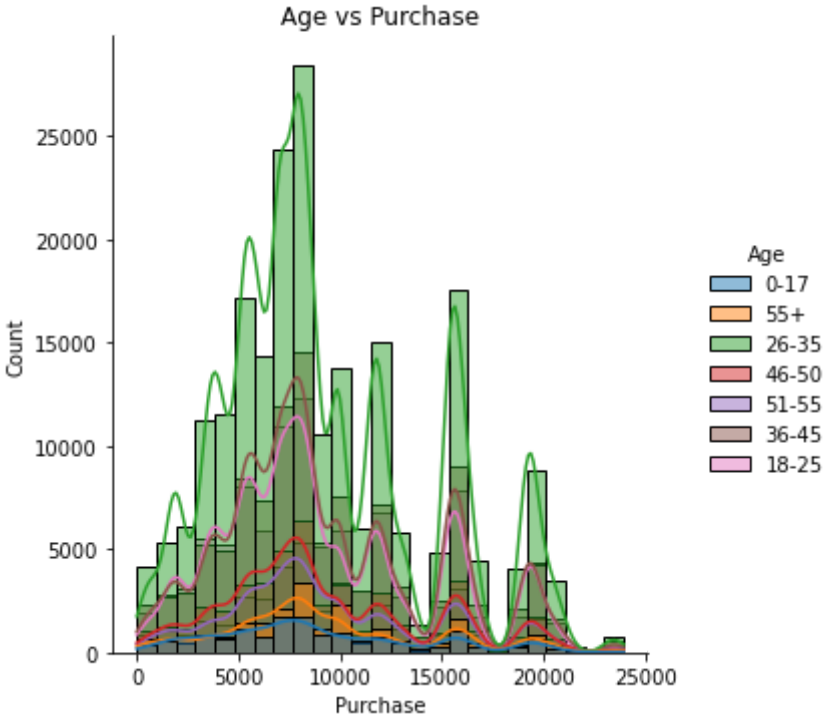
```
df.groupby('Age')['Purchase'].describe()
```

Out[167]:

Age	count	mean	std	min	25%	50%	75%	max
0-17	15102.0	8933.464640	5111.114046	12.0	5328.0	7986.0	11874.0	23955.0
18-25	99660.0	9169.663606	5034.321997	12.0	5415.0	8027.0	12028.0	23958.0
26-35	219587.0	9252.690633	5010.527303	12.0	5475.0	8030.0	12047.0	23961.0
36-45	110013.0	9331.350695	5022.923879	12.0	5876.0	8061.0	12107.0	23960.0
46-50	45701.0	9208.625697	4967.216367	12.0	5888.0	8036.0	11997.0	23960.0
51-55	38501.0	9534.808031	5087.368080	12.0	6017.0	8130.0	12462.0	23960.0
55+	21504.0	9336.280459	5011.493996	12.0	6018.0	8105.5	11932.0	23960.0

In [214]:

```
sns.displot(x='Purchase', bins=25, kde=True,hue='Age', data=df )
plt.title('Age vs Purchase')
plt.show()
```



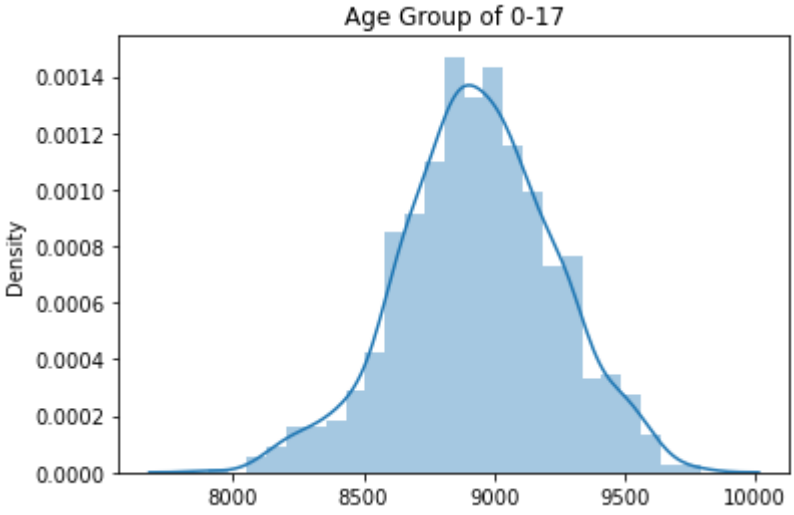
Generally we can see that the people in the range between the age group of 26-35 are buying more but we can't say that yet.

In [171]:

```
sample_means1=[df[df['Age']=='0-17'].sample(sample_size, replace=True)['Purchase'].mean() f
sample_means2=[df[df['Age']=='18-25'].sample(sample_size, replace=True)['Purchase'].mean()
sample_means3=[df[df['Age']=='26-35'].sample(sample_size, replace=True)['Purchase'].mean()
sample_means4=[df[df['Age']=='36-45'].sample(sample_size, replace=True)['Purchase'].mean()
sample_means5=[df[df['Age']=='46-50'].sample(sample_size, replace=True)['Purchase'].mean()
sample_means6=[df[df['Age']=='51-55'].sample(sample_size, replace=True)['Purchase'].mean()
sample_means7=[df[df['Age']=='55+'].sample(sample_size, replace=True)['Purchase'].mean() fo
```

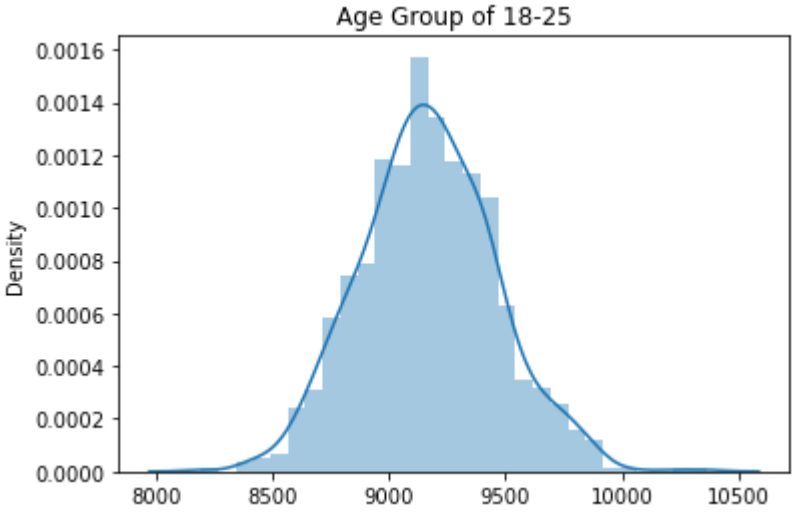
In [201]:

```
sns.distplot(sample_means1)
plt.title('Age Group of 0-17')
plt.show()
```



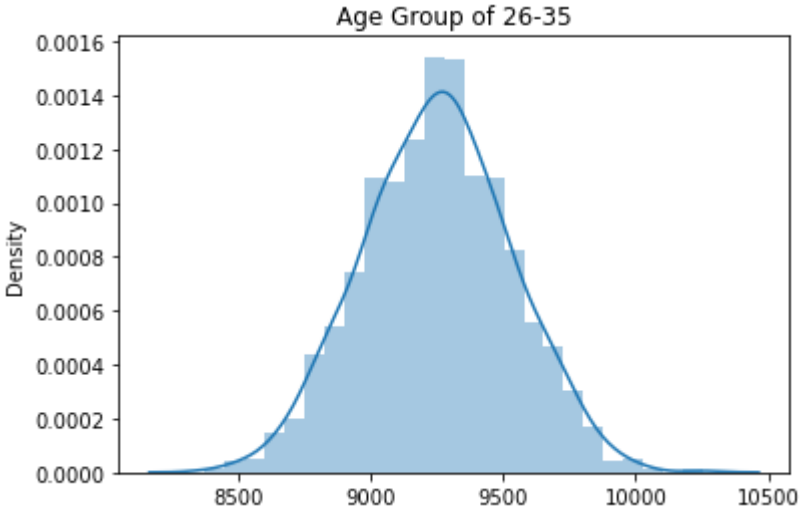
In [202]:

```
sns.distplot(sample_means2)
plt.title('Age Group of 18-25')
plt.show()
```



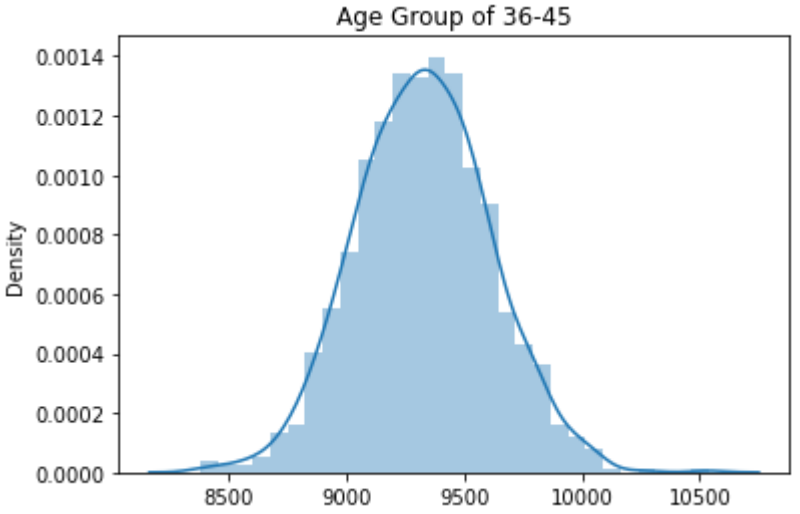
In [203]:

```
sns.distplot(sample_means3)
plt.title('Age Group of 26-35')
plt.show()
```



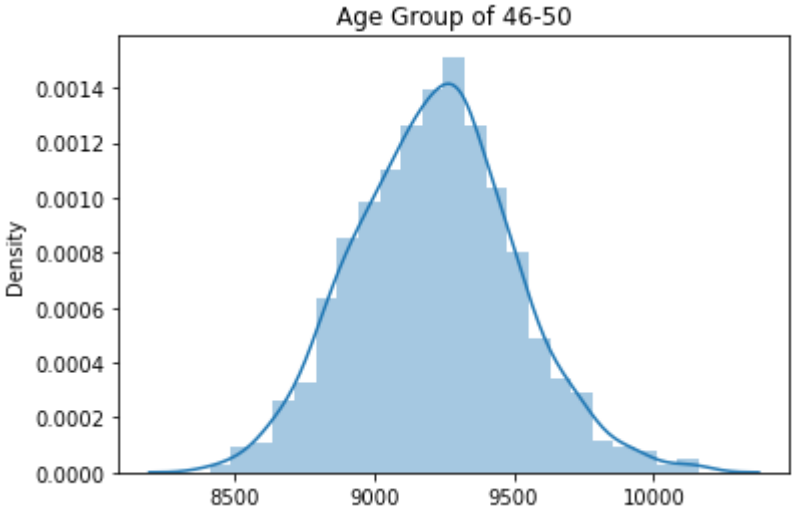
In [204]:

```
sns.distplot(sample_means4)
plt.title('Age Group of 36-45')
plt.show()
```



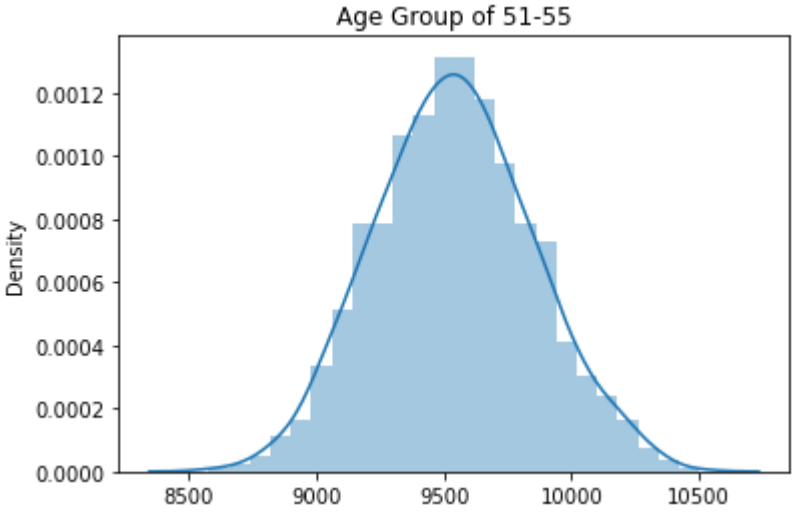
In [205]:

```
sns.distplot(sample_means5)
plt.title('Age Group of 46-50')
plt.show()
```



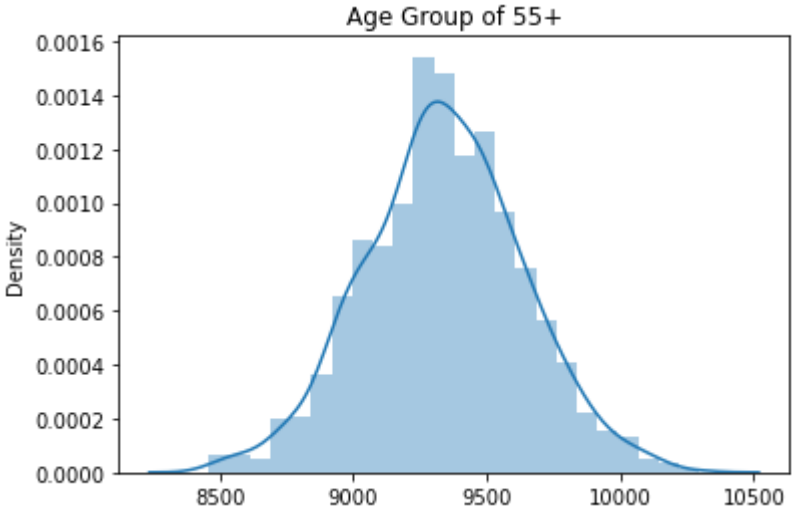
In [206]:

```
sns.distplot(sample_means6)
plt.title('Age Group of 51-55')
plt.show()
```



In [207]:

```
sns.distplot(sample_means7)
plt.title('Age Group of 55+')
plt.show()
```



In [208]:

```
upper_limit1= np.mean(sample_means1) + (1.96 * np.std(sample_means1))
lower_limit1= np.mean(sample_means1) - (1.96 * np.std(sample_means1))
print(f'confidence intervals for 0-17 years of age with 95% is:{(lower_limit1,upper_limit1)

upper_limit2= np.mean(sample_means2) + (1.96 * np.std(sample_means2))
lower_limit2= np.mean(sample_means2) - (1.96 * np.std(sample_means2))
print(f'confidence intervals for 18-25 years of age with 95% is:{(lower_limit2,upper_limit2

upper_limit3= np.mean(sample_means3) + (1.96 * np.std(sample_means3))
lower_limit3= np.mean(sample_means3) - (1.96 * np.std(sample_means3))
print(f'confidence intervals for 26-35 years of age with 95% is:{(lower_limit3,upper_limit3

upper_limit4= np.mean(sample_means4) + (1.96 * np.std(sample_means4))
lower_limit4= np.mean(sample_means4) - (1.96 * np.std(sample_means4))
print(f'confidence intervals for 36-45 years of age with 95% is:{(lower_limit4,upper_limit4

upper_limit5= np.mean(sample_means5) + (1.96 * np.std(sample_means5))
lower_limit5= np.mean(sample_means5) - (1.96 * np.std(sample_means5))
print(f'confidence intervals for 46-50 years of age with 95% is:{(lower_limit5,upper_limit5

upper_limit6= np.mean(sample_means6) + (1.96 * np.std(sample_means6))
lower_limit6= np.mean(sample_means6) - (1.96 * np.std(sample_means6))
print(f'confidence intervals for 51-55 years of age with 95% is:{(lower_limit6,upper_limit6

upper_limit7= np.mean(sample_means7) + (1.96 * np.std(sample_means7))
lower_limit7= np.mean(sample_means7) - (1.96 * np.std(sample_means7))
print(f'confidence intervals for 55+ years of age with 95% is:{(lower_limit7,upper_limit7)}
```

```
confidence intervals for 0-17 years of age with 95% is:(8354.037012163542, 9
520.18294116979)
confidence intervals for 18-25 years of age with 95% is:(8600.220575080208,
9738.384178253122)
confidence intervals for 26-35 years of age with 95% is:(8702.332584034004,
9798.782862632663)
confidence intervals for 36-45 years of age with 95% is:(8766.063685200745,
9886.657001465923)
confidence intervals for 46-50 years of age with 95% is:(8660.409754327971,
9777.93299900536)
confidence intervals for 51-55 years of age with 95% is:(8934.539103664163,
10148.03930966917)
confidence intervals for 55+ years of age with 95% is:(8757.729363776087, 99
19.827809557244)
```

In [209]:

```
upper_limit1= np.mean(sample_means1) + (1.645 * np.std(sample_means1))
lower_limit1= np.mean(sample_means1) - (1.645 * np.std(sample_means1))
print(f'confidence intervals for 0-17 years of age with 90% is:{(lower_limit1,upper_limit1)

upper_limit2= np.mean(sample_means2) + (1.645 * np.std(sample_means2))
lower_limit2= np.mean(sample_means2) - (1.645 * np.std(sample_means2))
print(f'confidence intervals for 18-25 years of age with 90% is:{(lower_limit2,upper_limit2

upper_limit3= np.mean(sample_means3) + (1.645 * np.std(sample_means3))
lower_limit3= np.mean(sample_means3) - (1.645 * np.std(sample_means3))
print(f'confidence intervals for 26-35 years of age with 90% is:{(lower_limit3,upper_limit3

upper_limit4= np.mean(sample_means4) + (1.645 * np.std(sample_means4))
lower_limit4= np.mean(sample_means4) - (1.645 * np.std(sample_means4))
print(f'confidence intervals for 36-45 years of age with 90% is:{(lower_limit4,upper_limit4

upper_limit5= np.mean(sample_means5) + (1.645 * np.std(sample_means5))
lower_limit5= np.mean(sample_means5) - (1.645 * np.std(sample_means5))
print(f'confidence intervals for 46-50 years of age with 90% is:{(lower_limit5,upper_limit5

upper_limit6= np.mean(sample_means6) + (1.645 * np.std(sample_means6))
lower_limit6= np.mean(sample_means6) - (1.645 * np.std(sample_means6))
print(f'confidence intervals for 51-55 years of age with 90% is:{(lower_limit6,upper_limit6

upper_limit7= np.mean(sample_means7) + (1.645 * np.std(sample_means7))
lower_limit7= np.mean(sample_means7) - (1.645 * np.std(sample_means7))
print(f'confidence intervals for 55+ years of age with 90% is:{(lower_limit7,upper_limit7)}
```

```
confidence intervals for 0-17 years of age with 90% is:(8447.745167172974, 9
426.47478616036)
confidence intervals for 18-25 years of age with 90% is:(8691.680150335174,
9646.924602998155)
confidence intervals for 26-35 years of age with 90% is:(8790.440195707111,
9710.675250959555)
confidence intervals for 36-45 years of age with 90% is:(8856.111362400625,
9796.609324266043)
confidence intervals for 46-50 years of age with 90% is:(8750.21072934669, 9
688.13202398664)
confidence intervals for 51-55 years of age with 90% is:(9032.05251307528, 1
0050.525900258053)
confidence intervals for 55+ years of age with 90% is:(8851.112274597786, 98
26.444898735544)
```

In [211]:

```
upper_limit1= np.mean(sample_means1) + (2.58 * np.std(sample_means1))
lower_limit1= np.mean(sample_means1) - (2.58 * np.std(sample_means1))
print(f'confidence intervals for 0-17 years of age with 99% is:{(lower_limit1,upper_limit1)

upper_limit2= np.mean(sample_means2) + (2.58 * np.std(sample_means2))
lower_limit2= np.mean(sample_means2) - (2.58 * np.std(sample_means2))
print(f'confidence intervals for 18-25 years of age with 99% is:{(lower_limit2,upper_limit2

upper_limit3= np.mean(sample_means3) + (2.58 * np.std(sample_means3))
lower_limit3= np.mean(sample_means3) - (2.58 * np.std(sample_means3))
print(f'confidence intervals for 26-35 years of age with 99% is:{(lower_limit3,upper_limit3

upper_limit4= np.mean(sample_means4) + (2.58 * np.std(sample_means4))
lower_limit4= np.mean(sample_means4) - (2.58 * np.std(sample_means4))
print(f'confidence intervals for 36-45 years of age with 99% is:{(lower_limit4,upper_limit4

upper_limit5= np.mean(sample_means5) + (2.58 * np.std(sample_means5))
lower_limit5= np.mean(sample_means5) - (2.58 * np.std(sample_means5))
print(f'confidence intervals for 46-50 years of age with 99% is:{(lower_limit5,upper_limit5

upper_limit6= np.mean(sample_means6) + (2.58 * np.std(sample_means6))
lower_limit6= np.mean(sample_means6) - (2.58 * np.std(sample_means6))
print(f'confidence intervals for 51-55 years of age with 99% is:{(lower_limit6,upper_limit6

upper_limit7= np.mean(sample_means7) + (2.58 * np.std(sample_means7))
lower_limit7= np.mean(sample_means7) - (2.58 * np.std(sample_means7))
print(f'confidence intervals for 55+ years of age with 99% is:{(lower_limit7,upper_limit7)}
```

```
confidence intervals for 0-17 years of age with 99% is:(8169.595564208474, 9
704.62438912486)
confidence intervals for 18-25 years of age with 99% is:(8420.204903149799,
9918.39985018353)
confidence intervals for 26-35 years of age with 99% is:(8528.914427725034,
9972.201018941632)
confidence intervals for 36-45 years of age with 99% is:(8588.826987220029,
10063.893699446638)
confidence intervals for 46-50 years of age with 99% is:(8483.658628894302,
9954.684124439029)
confidence intervals for 51-55 years of age with 99% is:(8742.607948632758,
10339.970464700575)
confidence intervals for 55+ years of age with 99% is:(8573.928078984169, 10
103.629094349162)
```

From all the comparisions between the intervals we can say that the purchsing factor does not depend on the Age too. As the intervals are overlapping with each other.

Conclusions

1. We can say that any factor is not effecting the purchasing capacity here as the intervals are overlapping with each other.

As we're extrapolating this data to the 100 million users we're looking at the AVERAGE buying capacity of different group of people.

2. Coming to the Gender we can see the male customers are buying more in number in usual way but that not true as the Average buying capacity intervals of both male and female customers are overlapping, so I can rule that out.

3. With Marital status, it seems Unmarried people are buying more in number but that's not the case here. The Average buying capacity of both Unmarried and married people are overlapping so we can't say that either.

4. With Age as constraint it also seems like the people in the age group of 25-36 are buying more but that is also baseless as the intervals are overlapping with other age group people Purchase capacity.

Finally we can say that while on holiday shopping there are no constraints and people are buying more no matter what the age group is, Gender and Marital status. So we can target wide range of audience here.