nxwbwo7q9

September 8, 2023

1 Walmart - Confidence Interval and CLT

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.

1.1 Business Problem

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

1.1.1 Understanding the Dataset

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

User_ID: User ID

Product_ID: Product ID

Gender: Sex of User

Age: Age in bins

Occupation: Occupation(Masked)

City_Category: Category of the City (A,B,C)

StayInCurrentCityYears: Number of years stay in current city

Marital_Status: Marital Status

ProductCategory: Product Category (Masked)

Purchase: Purchase Amount

Importing the Necessary Libraries

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```python
import seaborn as sns
from scipy.stats import norm
from scipy import stats
import warnings
warnings.filterwarnings('ignore')
```

Reading the data and performing basic checks

[2]:
```python
df = pd.read_csv("walmart_data.csv")
df.head()
```

[2]:
```
   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

  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
```

[3]:
```python
print(f"Number of rows: {df.shape[0]:,} \nNumber of columns: {df.shape[1]}")
```

```
Number of rows: 550,068
Number of columns: 10
```

Checking for null values

[4]:
```python
df.isna().sum()
```

[4]:
```
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
```

- No Null values

Checking the unique values in every column

[5]:
```
df.nunique().sort_values(ascending=False)
```

[5]:
```
Purchase                      18105
User_ID                        5891
Product_ID                     3631
Occupation                       21
Product_Category                 20
Age                               7
Stay_In_Current_City_Years        5
City_Category                     3
Gender                            2
Marital_Status                    2
dtype: int64
```

Checking for duplicates

[6]:
```
df.duplicated().sum()
```

[6]:
```
0
```

- No Duplicates

[7]:
```
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
```

'User_ID','Product_ID','Gender', 'Age','City_Category','Marital_Status' have categorical values.So we need to change the datatype from int and object to category.

[8]:
```
col = ['User_ID','Product_ID','Gender', 'Age','City_Category','Marital_Status']
df[col] = df[col].astype('category')
```

[9]:
```
df.dtypes
```

[9]:
```
User_ID                       category
Product_ID                    category
Gender                        category
Age                           category
Occupation                       int64
City_Category                 category
Stay_In_Current_City_Years      object
Marital_Status                category
Product_Category                 int64
Purchase                         int64
dtype: object
```

- We can confirm the data types have changed.

[10]:
```
df.describe().T
```

[10]:
```
                     count         mean          std   min     25%     50%  \
Occupation        550068.0     8.076707     6.522660   0.0     2.0     7.0
Product_Category  550068.0     5.404270     3.936211   1.0     1.0     5.0
Purchase          550068.0  9263.968713  5023.065394  12.0  5823.0  8047.0

                      75%      max
Occupation           14.0     20.0
Product_Category      8.0     20.0
Purchase          12054.0  23961.0
```

[11]:
```
df.describe(include=['object','category']).T
```

[11]:
```
                             count  unique        top    freq
User_ID                     550068    5891    1001680    1026
Product_ID                  550068    3631  P00265242    1880
Gender                      550068       2          M  414259
Age                         550068       7      26-35  219587
City_Category               550068       3          B  231173
Stay_In_Current_City_Years  550068       5          1  193821
Marital_Status              550068       2          0  324731
```

- There are 5891 unique users. User ID 1001680 has shopped the most frequent from Walmart.
- There are 3631 unique products. Product ID P00265242 is the most frequent sold item.
- Men are more frequent buyers than Females.
- There are 7 unique age categories. The most frequent buyers fall under the age group of 26-35.
- There are 3 different city categories. Most frequent buyers fal under category B.

- Most people are in the current city since 1 year.
- Most customerd are unmarried.

1.1.2 Univariate Analysis

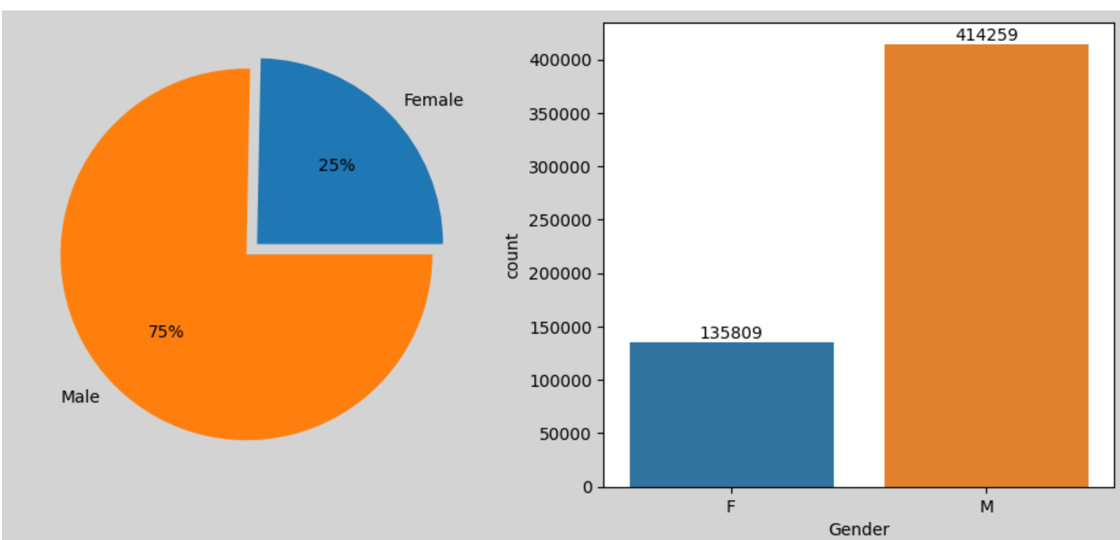
[12]:
```
df['User_ID'].nunique()
```

[12]: 5891

[13]:
```
df['Product_ID'].nunique()
```

[13]: 3631

[14]:
```
plt.figure(figsize = (12,5)).set_facecolor("lightgrey")

plt.subplot(1,2,1)
labels = ['Female','Male']
plt.pie(df.groupby('Gender')['Gender'].count(), labels = labels, explode = (0.
 ↪08,0), autopct = '%0.0f%%')

plt.subplot(1,2,2)
label = sns.countplot(data = df, x='Gender')
for i in label.containers:
    label.bar_label(i)

plt.show()
```

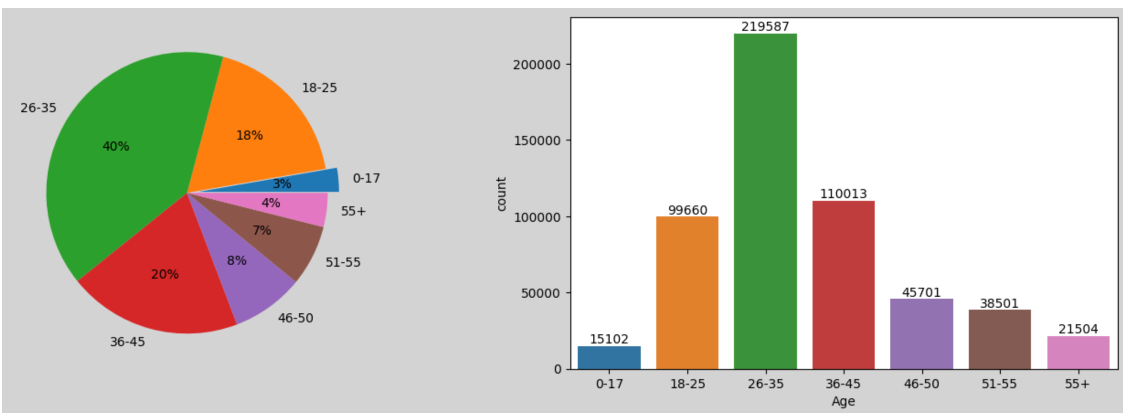


- Out of 0.54 million entries, 75% records are of men and 25% of women.
- Approximately there are 0.41 million records for men and 0.13 for Females.

[15]:
```
df['Age'].unique()
```

[15]:
```
['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25']
Categories (7, object): ['0-17', '18-25', '26-35', '36-45', '46-50', '51-55',
'55+']
```

[16]:
```
plt.figure(figsize = (17,5)).set_facecolor("lightgrey")

plt.subplot(1,2,1)
labels = ['0-17','18-25','26-35','36-45','46-50','51-55','55+']
plt.pie(df.groupby('Age')['Age'].count(), labels = labels, explode = (0.
 ↪08,0,0,0,0,0,0), autopct = '%0.0f%%')

plt.subplot(1,2,2)
label = sns.countplot(data = df, x='Age')
for i in label.containers:
    label.bar_label(i)

plt.show()
```

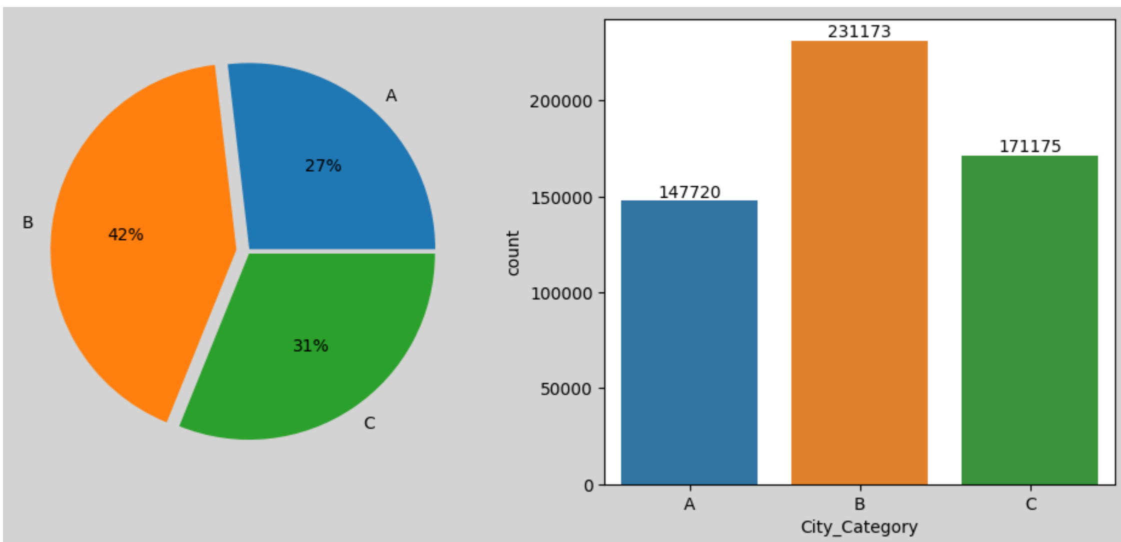


- 40% of the buyers fall under the age group of 26-35 which is the highest amongst all age groups.
- Approximately 0.21 million records are present for age group 26-35 followed by 0.11 million records for group 36-45.
- Age group 0-17 and 55+ are the least frequent buyers which is only 3% and 4% of the data respectively.
- Approximately only 15k and 21k records are there for age group 0-17 and group 55+.
- We can observe that most buyers are in within the age of 18-45 before and after this range we can see less buyers.

[17]:
```
df['City_Category'].unique()
```

[17]: ['A', 'C', 'B']
Categories (3, object): ['A', 'B', 'C']

[18]:
```
plt.figure(figsize = (12,5)).set_facecolor("lightgrey")

plt.subplot(1,2,1)
labels = ['A','B','C']
plt.pie(df.groupby('City_Category')['City_Category'].count(), labels = labels,␣
↪explode = (0.015,0.06,0.015), autopct = '%0.0f%%')

plt.subplot(1,2,2)
label = sns.countplot(data = df, x='City_Category')
for i in label.containers:
    label.bar_label(i)

plt.show()
```

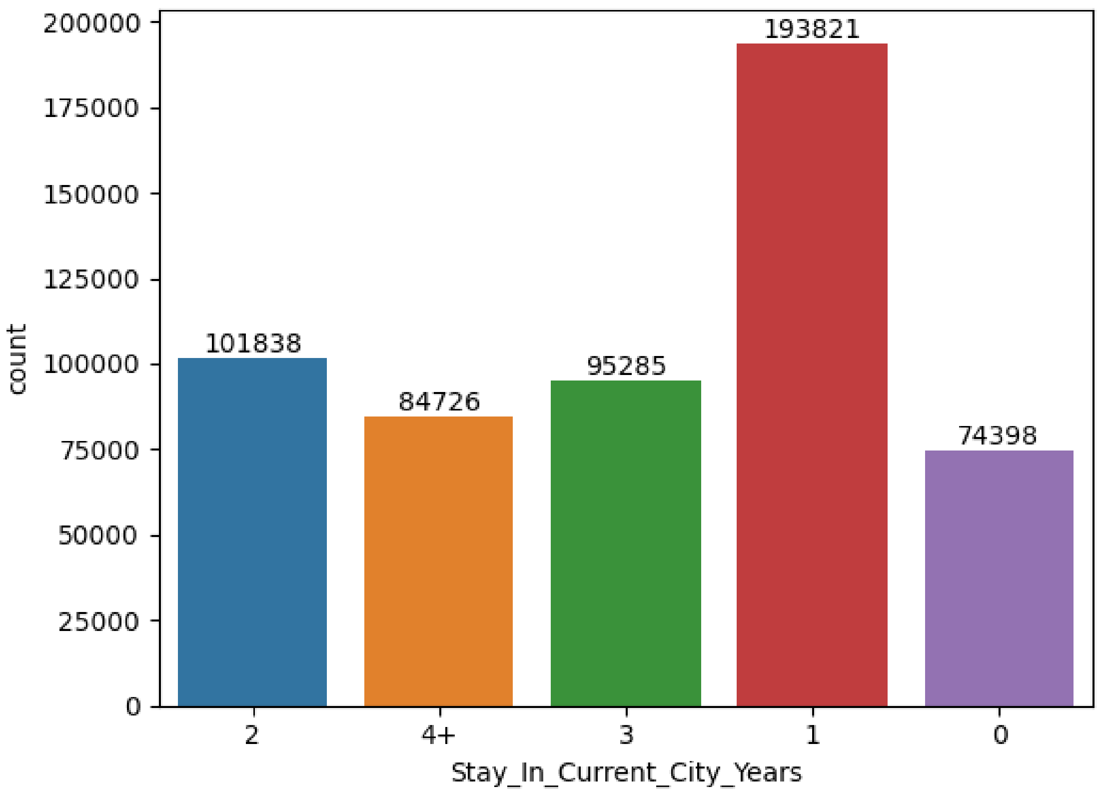


- There are 42% buyers from City Category B, 31% from Category C and 27% from Category A
- Approximately 0.23 million records are present for Category B, 0.17 million for Category C and 0.14 million for category A.

[19]:
```
df['Stay_In_Current_City_Years'].unique()
```

[19]: array(['2', '4+', '3', '1', '0'], dtype=object)

[20]:
```
label = sns.countplot(data = df, x='Stay_In_Current_City_Years')
for i in label.containers:
    label.bar_label(i)
```



- Most buyers are in their current cities since 1 year followed by 2 years and 3 years.

[21]:
```
df['Marital_Status'].unique()
```

[21]:
```
[0, 1]
Categories (2, int64): [0, 1]
```

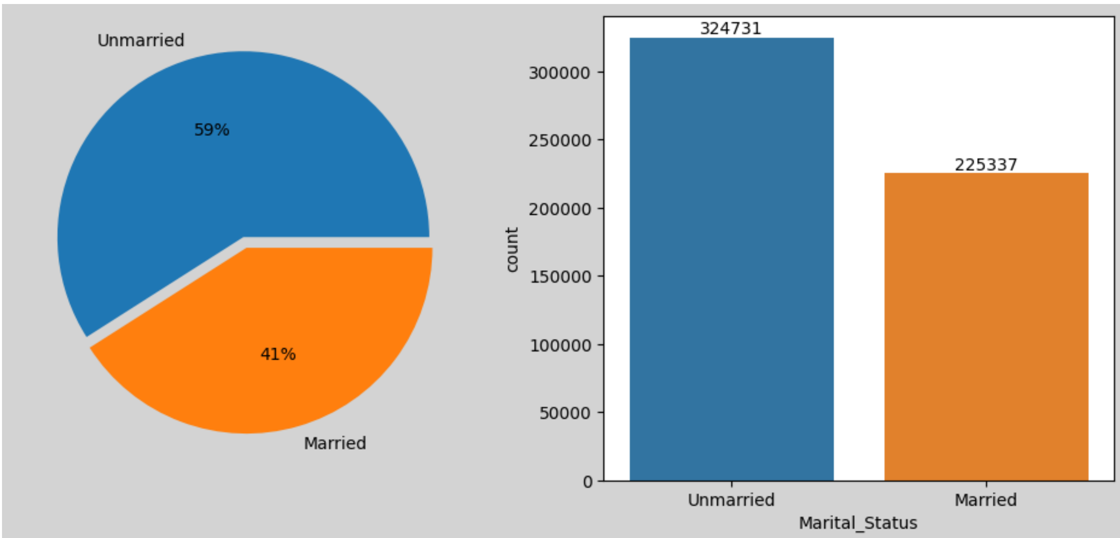
We can observe that in dataset for marital_status column there values 0 and 1.

0 means Unmarried and 1 means Married. So lets replace these values in the dataset.

[22]:
```
df['Marital_Status'].replace(to_replace = 0, value = 'Unmarried', inplace =␣
↪True)
df['Marital_Status'].replace(to_replace = 1, value = 'Married', inplace = True)
```

[23]:
```
plt.figure(figsize = (12,5)).set_facecolor("lightgrey")

plt.subplot(1,2,1)
labels = ['Unmarried','Married']
plt.pie(df.groupby('Marital_Status')['Marital_Status'].count(), labels =␣
↪labels, explode = (0.06,0), autopct = '%0.0f%%')
```

```
plt.subplot(1,2,2)
label = sns.countplot(data = df, x='Marital_Status')
for i in label.containers:
    label.bar_label(i)

plt.show()
```



- We can observe that 59% of the frequent buyers are of unmarried people, while 41% of married.
- There are an approximate of 0.32 million entries for unmarried people and 0.22 million for married people.

[24]:
```
round(df['Purchase'].describe(),2)
```

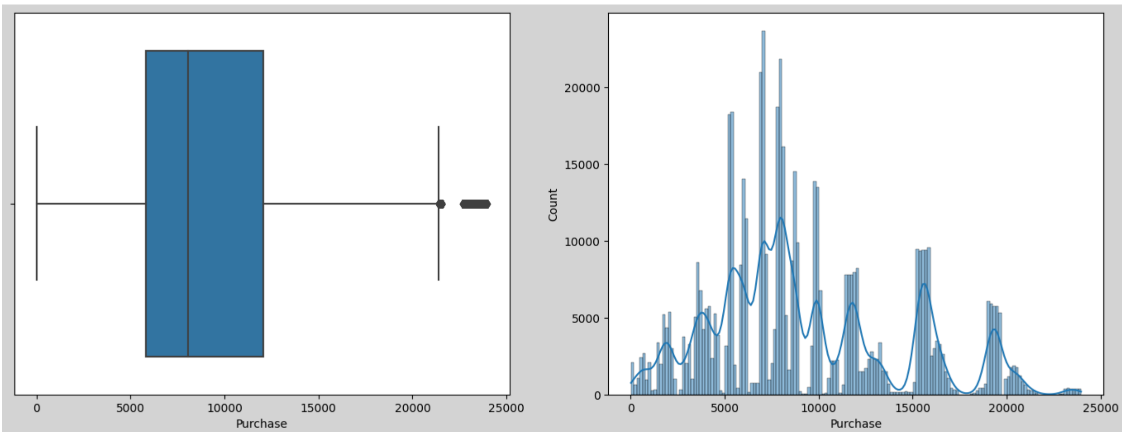
[24]:
```
count    550068.00
mean       9263.97
std        5023.07
min          12.00
25%        5823.00
50%        8047.00
75%       12054.00
max       23961.00
Name: Purchase, dtype: float64
```

While observing their spending habits of all buyers

- The average order value is 9263.97
- While 50% of the buyers spend an approximate of 8047.
- The lowest order value is as low as 12.

- While, the highest order value is of 23961.

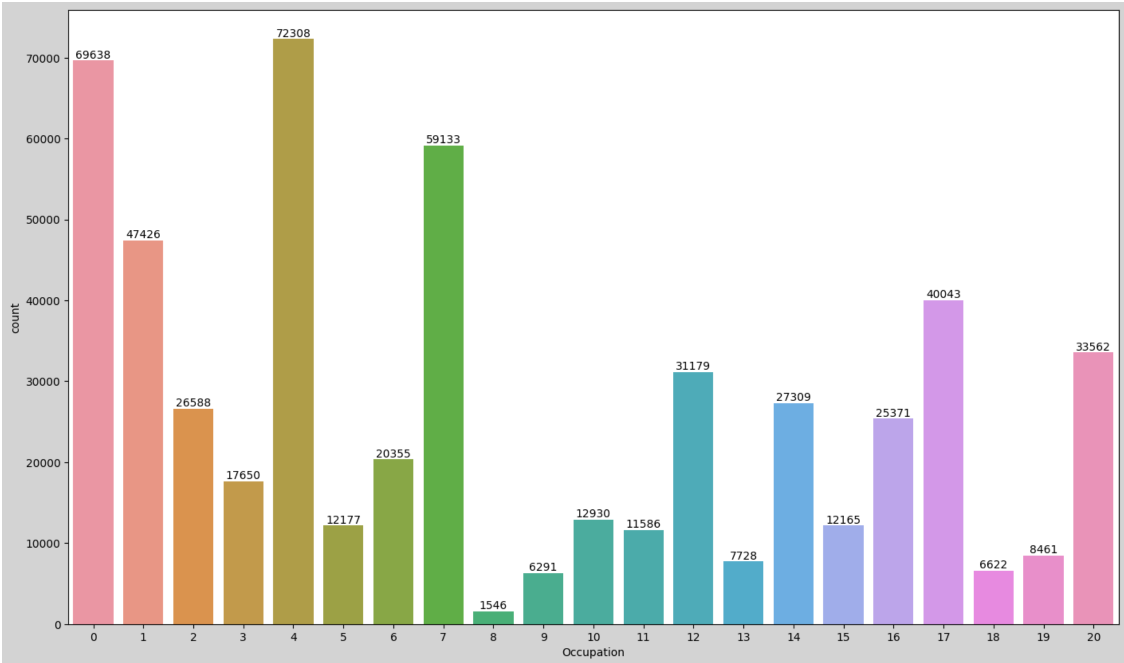
[25]:
```
plt.figure(figsize=(17, 6)).set_facecolor("lightgrey")

plt.subplot(1,2,1)
sns.boxplot(data=df, x='Purchase', orient='h')

plt.subplot(1,2,2)
sns.histplot(data=df, x='Purchase', kde=True)
plt.show()
```



While observing the purchase values of the orders we can infer that

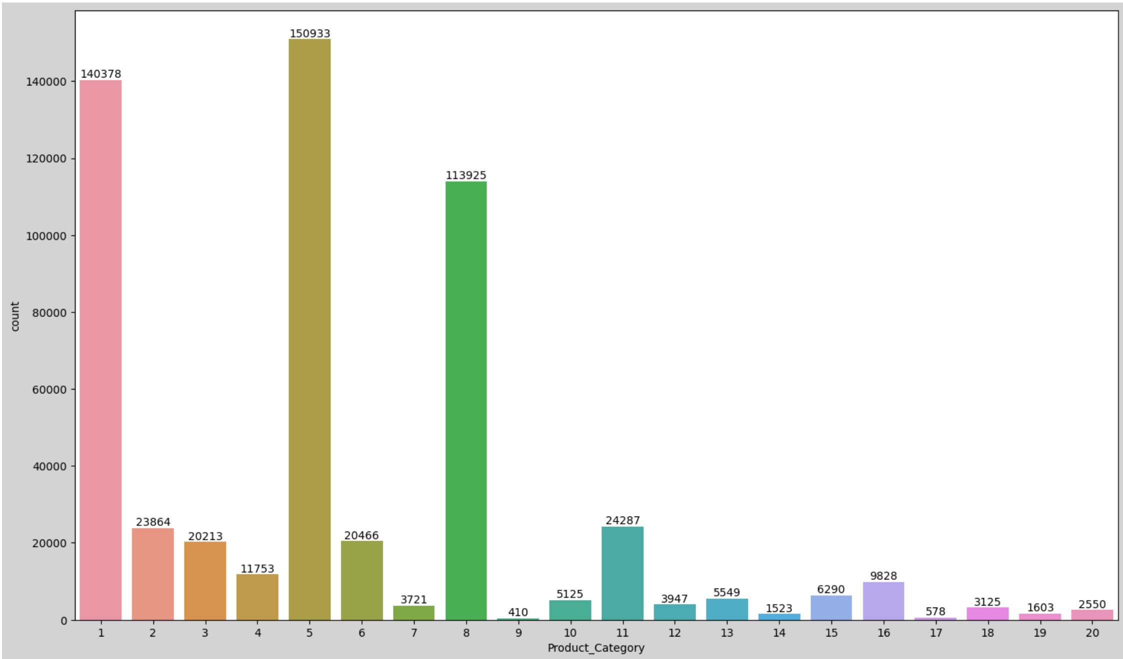
- Most of the values lies between 6000 and 12000.
- Most order values lies in the range of 5000 - 10000
- There are more orders in the range 15000 - 16000 followed by 11000 - 11500 range and a few also in the 19000 - 20000 range.

[26]:
```
plt.figure(figsize=(17, 10)).set_facecolor("lightgrey")
label = sns.countplot(data = df, x='Occupation')
for i in label.containers:
    label.bar_label(i)
```



- People having occupation 4 are the most frequent buyers followed by occupation 0 and 7.
- People having occupation 8 are the least frequent buyers followed by occupation 9 and 18.

[27]:
```
plt.figure(figsize=(17, 10)).set_facecolor("lightgrey")
label = sns.countplot(data = df, x='Product_Category')
for i in label.containers:
    label.bar_label(i)
```



- The most frequent bought product category is 5 followed by 1 and 8.
- All the other categories are not much touched.
- The least frequent bought are category 9 followed by 17 and 14.

1.1.3 Bi-variate Analysis

Lets observe gender while purchase habits.

[28]:
```
plt.figure(figsize = (10,6)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Gender', palette = 'Set3')
plt.title('Purchase vs Gender')
plt.show()
```

- We can observe Males spend more than Females.

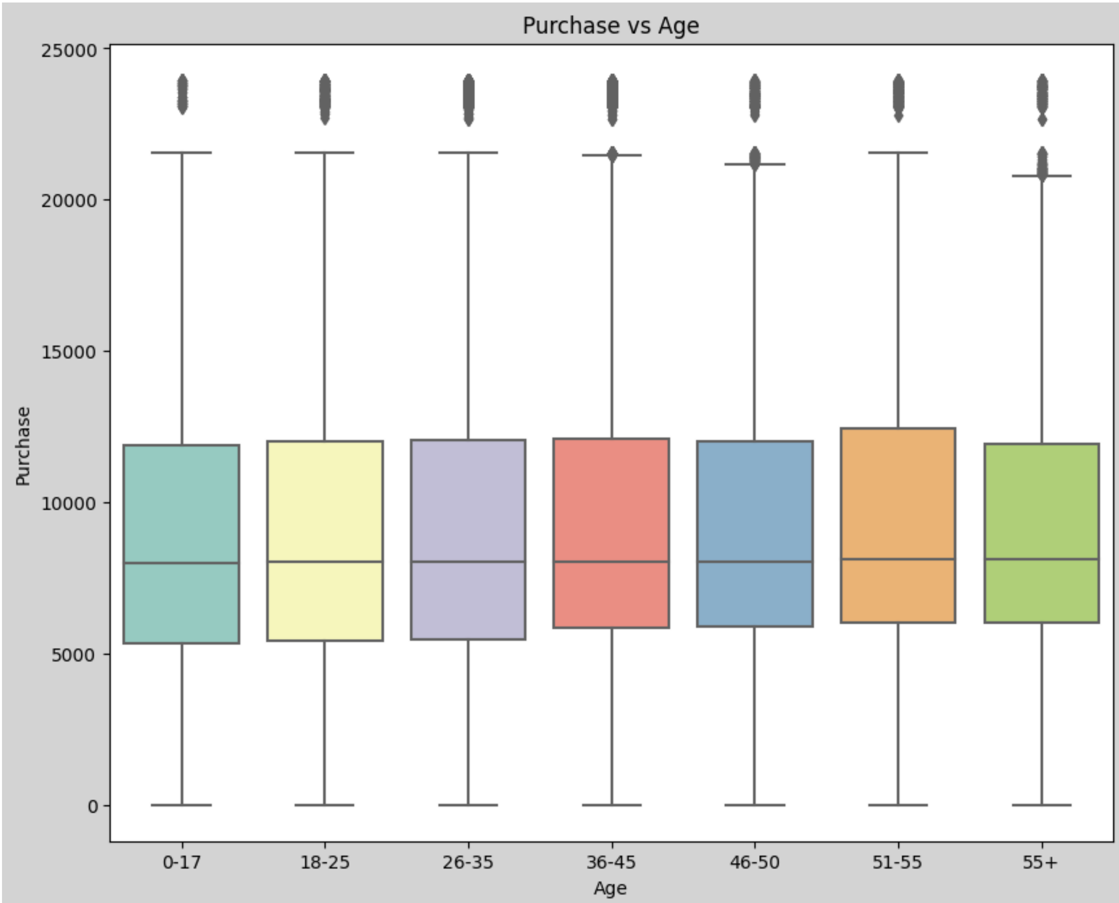
[29]:
```
df.groupby(['Gender'])['Purchase'].describe()
```

[29]:
```
           count         mean          std   min     25%     50%      75%  \
Gender
F       135809.0  8734.565765  4767.233289  12.0  5433.0  7914.0  11400.0
M       414259.0  9437.526040  5092.186210  12.0  5863.0  8098.0  12454.0

            max
Gender
F       23959.0
M       23961.0
```

- The average order value for a male is 9437.
- While for a female it is 8734.
- Most of the purchases for men is around 8098 and for females it is around 7914.

Now, lets see the Purchase habits age group wise

[30]:
```
plt.figure(figsize = (10,8)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Age', palette = 'Set3')
plt.title('Purchase vs Age')
plt.show()
```



- We can not see much difference in the median purchase values for different age groups.

Lets check the mean values

[31]:
```
df.groupby(['Age'])['Purchase'].describe()
```

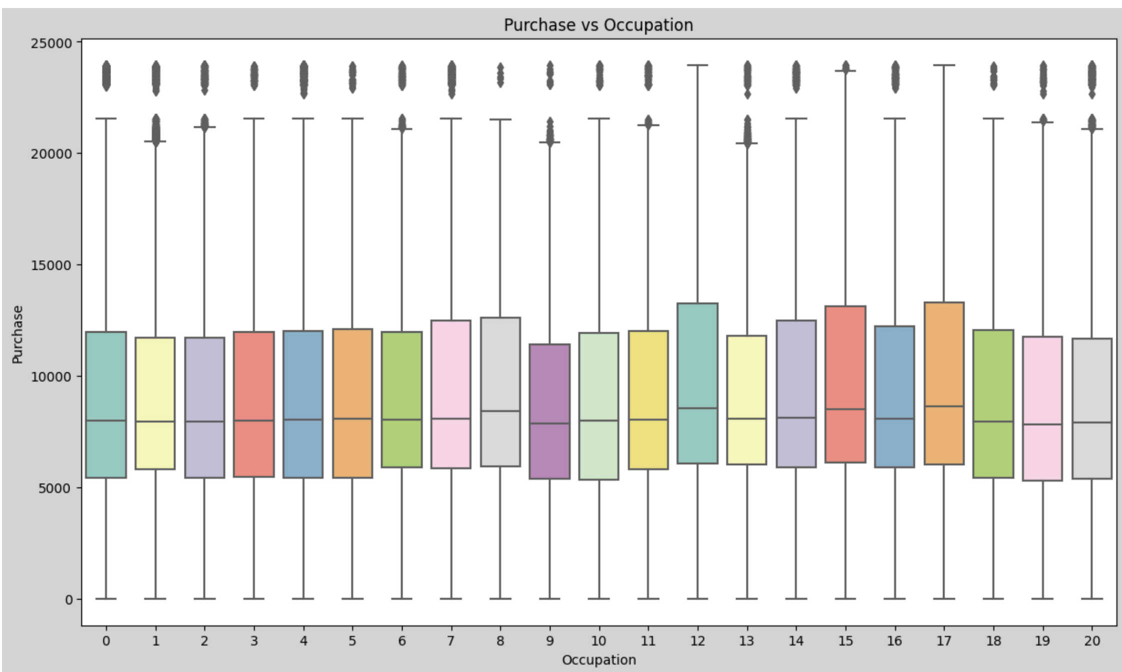
[31]:
```
          count         mean          std   min     25%     50%      75%  \
Age
0-17    15102.0  8933.464640  5111.114046  12.0  5328.0  7986.0  11874.0
18-25   99660.0  9169.663606  5034.321997  12.0  5415.0  8027.0  12028.0
26-35  219587.0  9252.690633  5010.527303  12.0  5475.0  8030.0  12047.0
36-45  110013.0  9331.350695  5022.923879  12.0  5876.0  8061.0  12107.0
46-50   45701.0  9208.625697  4967.216367  12.0  5888.0  8036.0  11997.0
51-55   38501.0  9534.808031  5087.368080  12.0  6017.0  8130.0  12462.0
55+     21504.0  9336.280459  5011.493996  12.0  6018.0  8105.5  11932.0

           max
Age
0-17   23955.0
18-25  23958.0
```

```
26-35   23961.0
36-45   23960.0
46-50   23960.0
51-55   23960.0
55+     23960.0
```

- The average order value is highest for age group 51-55 which is around 9534.
- While, the average amount is lowest for age group 0-17 which is arouns 8933.
- The highest order value for all the groups is around 23960.
- The losest order value is 12 for all the groups.

Lets see purchase habits according to Occupation

[32]:
```
plt.figure(figsize = (14,8)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Occupation', palette = 'Set3')
plt.title('Purchase vs Occupation')
plt.show()
```



- There are many outliers in the data.
- We can not see much difference in the median values.

[33]:
```
df.groupby(['Occupation'])['Purchase'].describe()
```

[33]:

Occupation	count	mean	std	min	25%	50%
0	69638.0	9124.428588	4971.757402	12.0	5445.00	8001.0
1	47426.0	8953.193270	4838.482159	12.0	5825.00	7966.0
2	26588.0	8952.481683	4939.418663	12.0	5419.00	7952.0
3	17650.0	9178.593088	5000.942719	12.0	5478.00	8008.0
4	72308.0	9213.980251	5043.674855	12.0	5441.75	8043.0
5	12177.0	9333.149298	5025.616603	12.0	5452.00	8080.0
6	20355.0	9256.535691	4989.216005	12.0	5888.00	8050.0
7	59133.0	9425.728223	5086.097089	12.0	5878.00	8069.0
8	1546.0	9532.592497	4916.641374	14.0	5961.75	8419.5
9	6291.0	8637.743761	4653.290986	13.0	5403.00	7886.0
10	12930.0	8959.355375	5124.339999	12.0	5326.25	8012.5
11	11586.0	9213.845848	5103.802992	12.0	5835.75	8041.5
12	31179.0	9796.640239	5140.437446	12.0	6054.00	8569.0
13	7728.0	9306.351061	4940.156591	12.0	6038.00	8090.5
14	27309.0	9500.702772	5069.600234	12.0	5922.00	8122.0
15	12165.0	9778.891163	5088.424301	12.0	6109.00	8513.0
16	25371.0	9394.464349	4995.918117	12.0	5917.00	8070.0
17	40043.0	9821.478236	5137.024383	12.0	6012.00	8635.0
18	6622.0	9169.655844	4987.697451	12.0	5420.00	7955.0
19	8461.0	8710.627231	5024.181000	12.0	5292.00	7840.0
20	33562.0	8836.494905	4919.662409	12.0	5389.00	7903.5

Occupation	75%	max
0	11957.00	23961.0
1	11702.75	23960.0
2	11718.00	23955.0
3	11961.00	23914.0
4	12034.00	23961.0
5	12091.00	23924.0
6	11971.50	23951.0
7	12486.00	23948.0
8	12607.00	23869.0
9	11436.00	23943.0
10	11931.75	23955.0
11	12010.00	23946.0
12	13239.00	23960.0
13	11798.50	23959.0
14	12508.00	23941.0
15	13150.00	23949.0
16	12218.50	23947.0
17	13292.50	23961.0
18	12062.75	23894.0
19	11745.00	23939.0
20	11677.00	23960.0

- But, here we can observe that the highest median value is for occupation 17
- The lowest median value is for occupation 19.
- Occupation 17 have the high average order values compared to other occupations which is 9821.
- Occupation 9 have the lowest average order value which is 8637.

Now, lets see city wise purchase habits.

[34]:
```
plt.figure(figsize = (10,6)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'City_Category', palette = 'Set3')
plt.title('Purchase vs City_Category')
plt.show()
```

- City Category c has the highest median value followed by city B and city A.
- There are a few outliers fro city A and B.

[35]:
```
df.groupby(['City_Category'])['Purchase'].describe()
```

[35]:
```
                  count         mean          std   min     25%     50%  \
City_Category
A              147720.0  8911.939216  4892.115238  12.0  5403.0  7931.0
B              231173.0  9151.300563  4955.496566  12.0  5460.0  8005.0
C              171175.0  9719.920993  5189.465121  12.0  6031.5  8585.0
```

```
                  75%      max
City_Category
A             11786.0  23961.0
B             11986.0  23960.0
C             13197.0  23961.0
```

- We can also observe that the mean value for a order is highest for city C followed by B and A.

Lets see if stay years of a person in a city affects his/her purchase habits or not.

[36]:
```python
plt.figure(figsize = (10,6)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Stay_In_Current_City_Years', palette
 ↪= 'Set3')
plt.title('Purchase vs Stay_In_Current_City_Years')
plt.show()
```

- We can see that the median value is almost the same for all the years.

[37]:
```python
df.groupby(['Stay_In_Current_City_Years'])['Purchase'].describe()
```

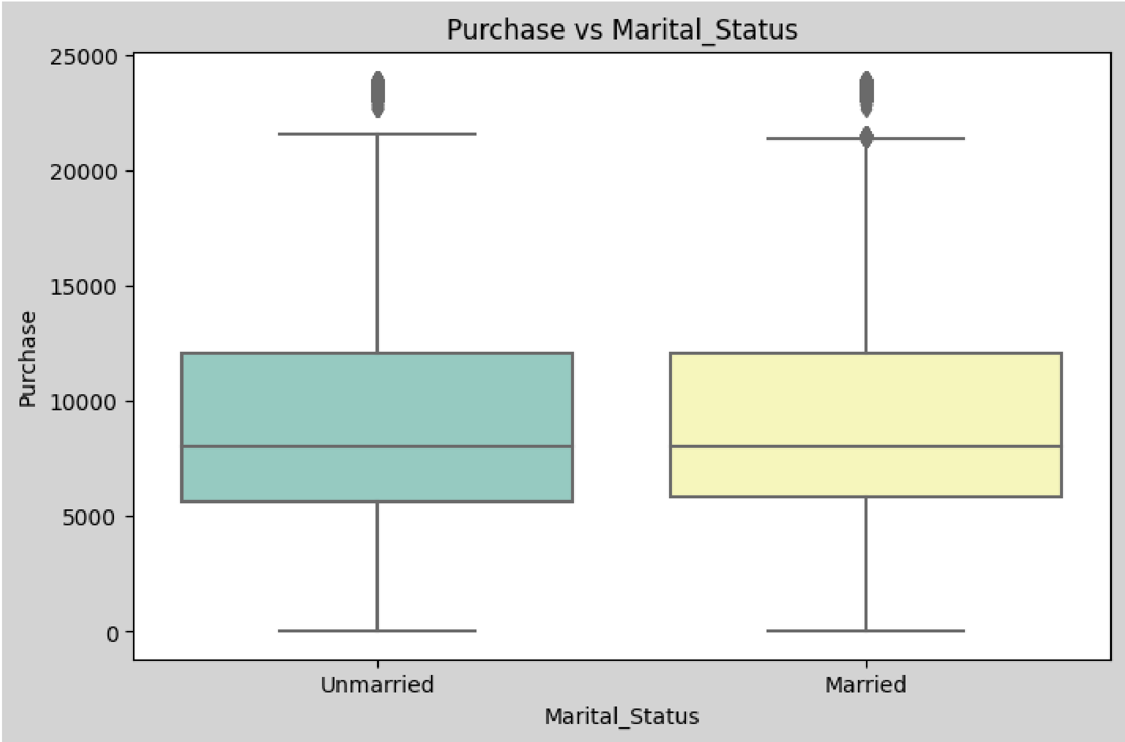
[37]:
```
                               count         mean          std   min     25%  \
Stay_In_Current_City_Years
0                            74398.0  9180.075123  4990.479940  12.0  5480.0
1                           193821.0  9250.145923  5027.476933  12.0  5500.0
```

```
2                            101838.0  9320.429810  5044.588224  12.0  5846.0
3                             95285.0  9286.904119  5020.343541  12.0  5832.0
4+                            84726.0  9275.598872  5017.627594  12.0  5844.0

                                50%      75%      max
Stay_In_Current_City_Years
0                            8025.0  11990.0  23960.0
1                            8041.0  12042.0  23961.0
2                            8072.0  12117.0  23961.0
3                            8047.0  12075.0  23961.0
4+                           8052.0  12038.0  23958.0
```

- We can also see that the average order value is also almost the same which lies in the range of 9180 to 9286.
- One more thing we can observe here is that the highest order value is also the same for all the years.

Lets see if Marital Status affects the spending habits of a person

[38]:
```
plt.figure(figsize = (8,5)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Marital_Status', palette = 'Set3')
plt.title('Purchase vs Marital_Status')
plt.show()
```



- We can observe that the median value is almost the same.

Lets check the minimum, maximum and average order value.

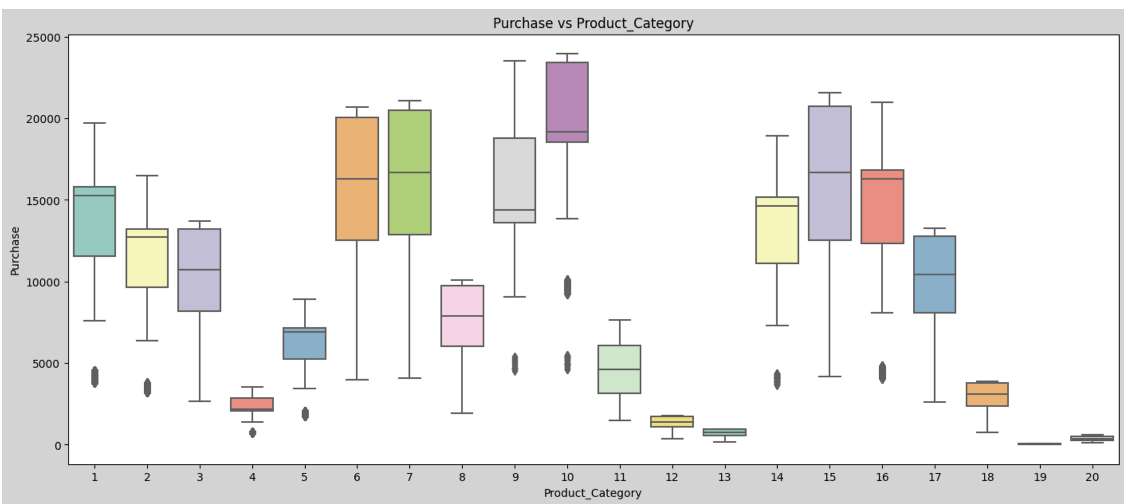
[39]:
```
df.groupby(['Marital_Status'])['Purchase'].describe()
```

[39]:
```
                   count         mean          std   min     25%     50%  \
Marital_Status
Unmarried       324731.0  9265.907619  5027.347859  12.0  5605.0  8044.0
Married         225337.0  9261.174574  5016.897378  12.0  5843.0  8051.0

                    75%      max
Marital_Status
Unmarried       12061.0  23961.0
Married         12042.0  23961.0
```

- The minimum and maximum order value is same for both types of people.
- We can observe that the average is also almost the same for both.

Lets see on which product category people spend more or less.

[40]:
```
plt.figure(figsize = (17,7)).set_facecolor("lightgrey")
sns.boxplot(data = df, y ='Purchase', x = 'Product_Category', palette = 'Set3')
plt.title('Purchase vs Product_Category')
plt.show()
```



- We can clearly observe hige differences in the median values for all the product categories.

[41]:
```
df.groupby(['Product_Category'])['Purchase'].describe()
```

[41]:
```
                       count          mean           std      min       25%  \
Product_Category
```

```
1                   140378.0  13606.218596  4298.834894  3790.0  11546.00
2                    23864.0  11251.935384  3570.642713  3176.0   9645.75
3                    20213.0  10096.705734  2824.626957  2638.0   8198.00
4                    11753.0   2329.659491   812.540292   684.0   2058.00
5                   150933.0   6240.088178  1909.091687  1713.0   5242.00
6                    20466.0  15838.478550  4011.233690  3981.0  12505.00
7                     3721.0  16365.689600  4174.554105  4061.0  12848.00
8                   113925.0   7498.958078  2013.015062  1939.0   6036.00
9                      410.0  15537.375610  5330.847116  4528.0  13583.50
10                    5125.0  19675.570927  4225.721898  4624.0  18546.00
11                   24287.0   4685.268456  1834.901184  1472.0   3131.00
12                    3947.0   1350.859894   362.510258   342.0   1071.00
13                    5549.0    722.400613   183.493126   185.0    578.00
14                    1523.0  13141.625739  4069.009293  3657.0  11097.00
15                    6290.0  14780.451828  5175.465852  4148.0  12523.25
16                    9828.0  14766.037037  4360.213198  4036.0  12354.00
17                     578.0  10170.759516  2333.993073  2616.0   8063.50
18                    3125.0   2972.864320   727.051652   754.0   2359.00
19                    1603.0     37.041797    16.869148    12.0     24.00
20                    2550.0    370.481176   167.116975   118.0    242.00

                      50%       75%      max
Product_Category
1                 15245.0  15812.00  19708.0
2                 12728.5  13212.00  16504.0
3                 10742.0  13211.00  13717.0
4                  2175.0   2837.00   3556.0
5                  6912.0   7156.00   8907.0
6                 16312.0  20051.00  20690.0
7                 16700.0  20486.00  21080.0
8                  7905.0   9722.00  10082.0
9                 14388.5  18764.00  23531.0
10                19197.0  23438.00  23961.0
11                 4611.0   6058.00   7654.0
12                 1401.0   1723.00   1778.0
13                  755.0    927.00    962.0
14                14654.0  15176.50  18931.0
15                16660.0  20745.75  21569.0
16                16292.5  16831.00  20971.0
17                10435.5  12776.75  13264.0
18                 3071.0   3769.00   3900.0
19                   37.0     50.00     62.0
20                  368.0    490.00    613.0
```

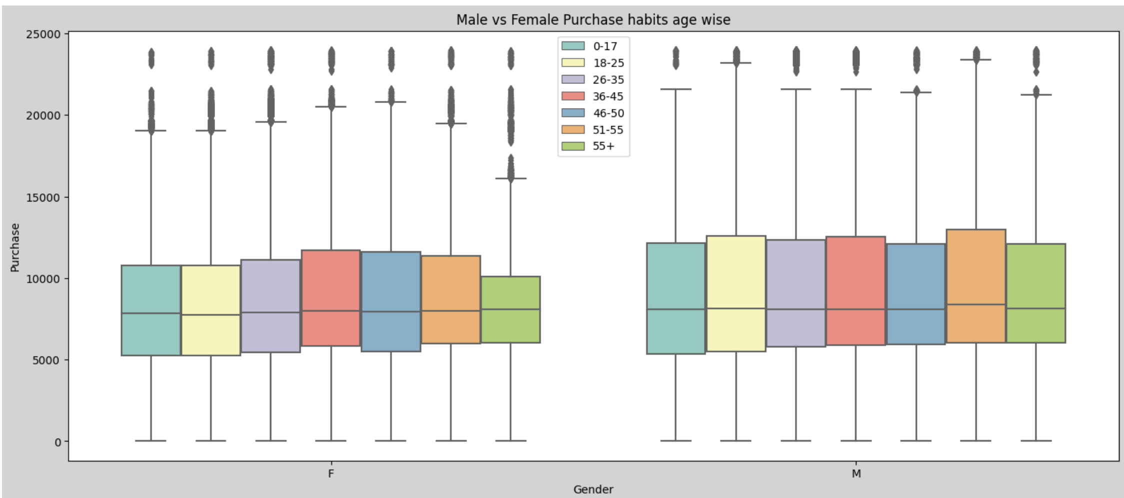
- The median value for product category 10 is the highest which is 19197.
- The median value for product category 19 is the lowest which is only 37.

- The average order value for category 10 is the highest which is 19675.
- The average order value for category 19 is also the lowest which is 37.
- Clearly, category 19 is the least preferred or least frequent bought product category.

1.1.4 Multi-variate Analysis

Lets see Male vs Female Purchase habits age wise.

[42]:
```
plt.figure(figsize = (17,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Age', palette='Set3')
plt.legend(loc=9)
plt.title('Male vs Female Purchase habits age wise')

plt.show()
```



- The median values for 18-25 age females is the lowest and almost same for the rest.
- The median values for all age categories is almost the same and is highest for 51-55 age group.

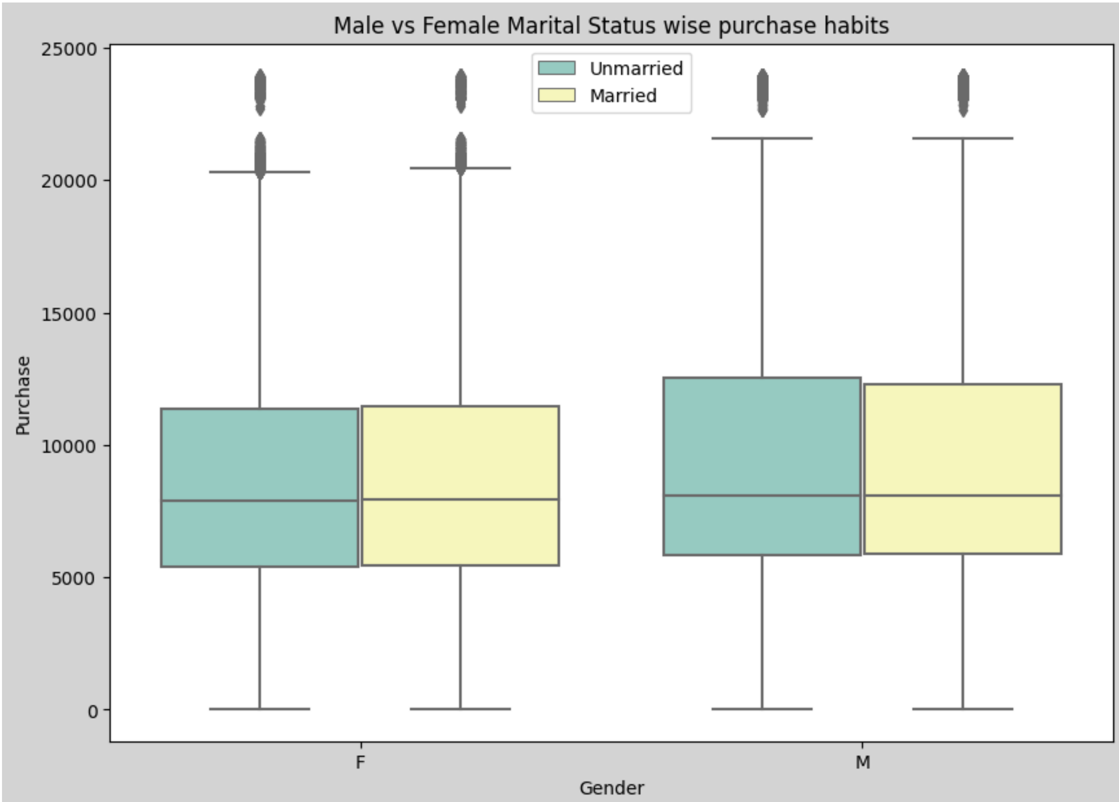
Lets see Male vs Female City wise purchase habits.

[43]:
```
plt.figure(figsize = (12,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender', hue='City_Category',␣
↪palette='Set3')
plt.legend(loc=9)
plt.title("Male vs Female City Category wise Purchase habits")
plt.show()
```

- The median value for females in city category C is highest compared to city A and B.
- The median value for males in city category C is also highest compared to city A and B.

Lets see Male vs Female Marital Status wise purchase habits.

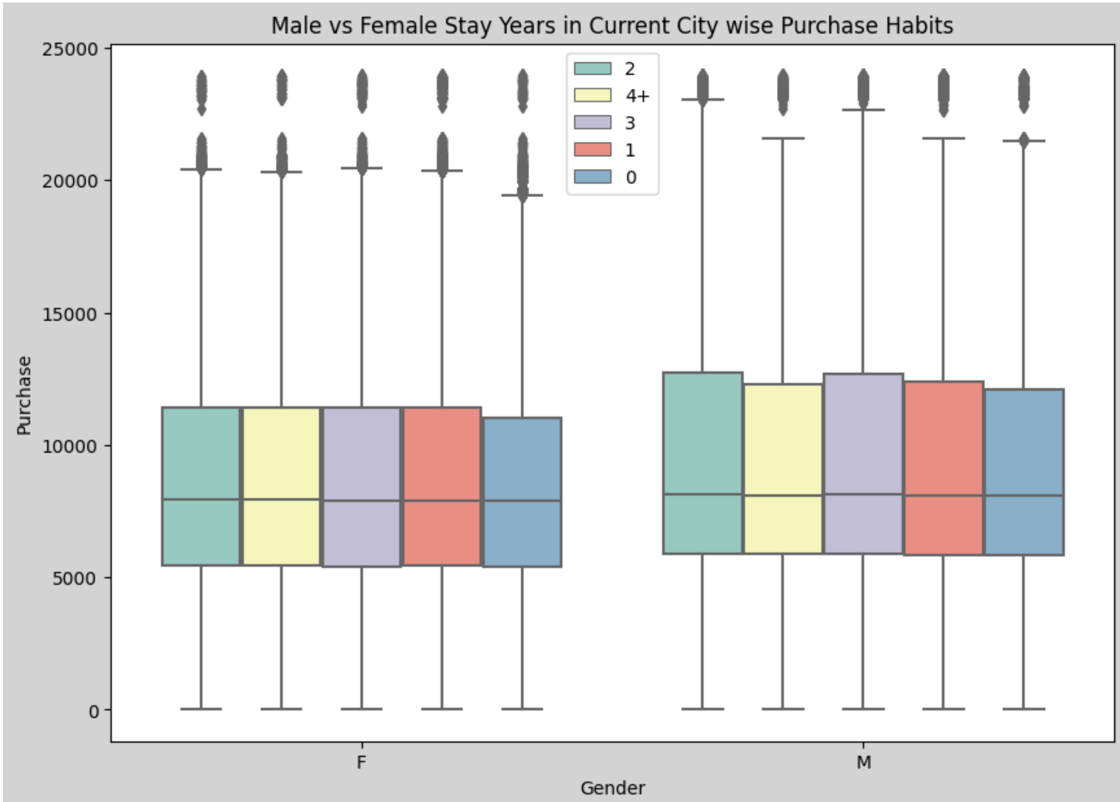
[44]:
```
plt.figure(figsize = (10,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Marital_Status',␣
 ↪palette='Set3')
plt.legend(loc=9)
plt.title('Male vs Female Marital Status wise purchase habits')
plt.show()
```



- There is no effect of marital status on the spending habits of both the genders.
- While we can observe that the median values for Male is higher comapred to Females.

Lets see Male vs Female Stay Years in Current City wise Purchase Habits

[45]:
```
plt.figure(figsize = (10,7)).set_facecolor("lightgrey")
sns.boxplot(data=df, y='Purchase', x='Gender',␣
 ↪hue='Stay_In_Current_City_Years', palette='Set3')
plt.legend(loc=9)
plt.title('Male vs Female Stay Years in Current City wise Purchase Habits')
plt.show()
```



- We can observe for females the median values for purchase amount is a little lower for women staying for 3 and 0 years as compared to others.
- For men, there is no much difference.

Lets check the Correlation in the numerical values of the dataset.

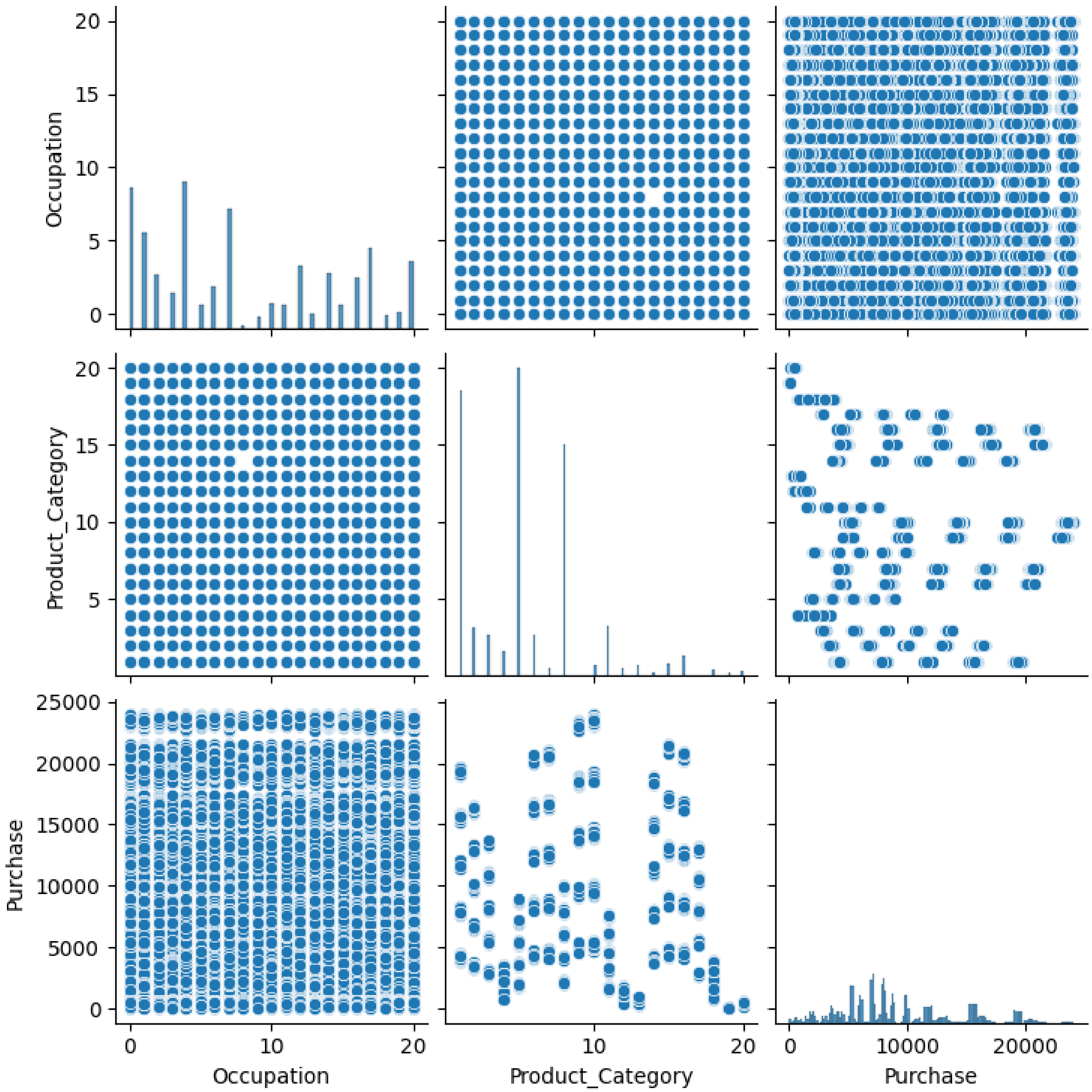
[46]:
```
sns.heatmap(df.corr(), annot = True)
plt.show()
```

We can observe that there is

- High Negative Correlation(-0.0076) between Product Category and Occupation.
- Slight Positive Correlation(0.021) between Purchase and Occupation.
- Negative Correlation(-0.34) between Product Category and Purchase.

Lets plot the pairplot and see relations between the columns.

[47]:
```
sns.pairplot(df)
plt.show()
```



1.2 Central Limit Theorom

[48]:
```
def bootstrap(sample1,sample2,sample_size,itr_size=1000,ci=90):
    ci = ci/100

    plt.figure(figsize=(16,8))
    sample1_n = [np.mean(sample1.sample(sample_size)) for i in range(itr_size)]
    sample2_n = [np.mean(sample2.sample(sample_size)) for i in range(itr_size)]

    # For Sample1's means
    mean1 = np.mean(sample1_n)
    sigma1 = np.std(sample1_n)
    sem1 = stats.sem(sample1_n)
```

```
    lower_limit_1 = norm.ppf((1-ci)/2) * sigma1 + mean1
    upper_limit_1 = norm.ppf(ci+(1-ci)/2) * sigma1 + mean1

    # For Sample2's means
    mean2 = np.mean(sample2_n)
    sigma2 = np.std(sample2_n)
    sem2 = stats.sem(sample2_n)

    lower_limit_2 = norm.ppf((1-ci)/2) * sigma2 + mean2
    upper_limit_2 = norm.ppf(ci + (1-ci)/2) * sigma2 + mean2

    sns.kdeplot(data = sample1_n, color="#F2D2BD", fill = True, linewidth = 2)
    label_mean1=("  (Males) :  {:.2f}".format(mean1))
    plt.axvline(mean1, color = '#FF00FF', linestyle = 'solid', linewidth = 2, label=label_mean1)
    label_limits1=("Lower Limit(M):  {:.2f}\nUpper Limit(M):   {:.2f}".format(lower_limit_1,upper_limit_1))
    plt.axvline(lower_limit_1, color = '#FF69B4', linestyle = 'dashdot', linewidth = 2, label=label_limits1)
    plt.axvline(upper_limit_1, color = '#FF69B4', linestyle = 'dashdot', linewidth = 2)

    sns.kdeplot(data = sample2_n ,color='#ADD8E6', fill = True, linewidth = 2)
    label_mean2=("  (Females):  {:.2f}".format(mean2))
    plt.axvline(mean2, color = '#1434A4', linestyle = 'solid', linewidth = 2, label=label_mean2)
    label_limits2=("Lower Limit(F):  {:.2f}\nUpper Limit(F):   {:.2f}".format(lower_limit_2,upper_limit_2))
    plt.axvline(lower_limit_2, color = '#4682B4', linestyle = 'dashdot', linewidth = 2, label=label_limits2)
    plt.axvline(upper_limit_2, color = '#4682B4', linestyle = 'dashdot', linewidth = 2)

    plt.title(f"Sample Size: {sample_size}, Male Avg: {np.round(mean1, 2)}, Male SME: {np.round(sem1,2)}, Female Avg:{np.round(mean2, 2)}, Female SME: {np.round(sem2,2)}")
    plt.legend(loc = 'upper right')
    plt.xlabel('Purchase')
    plt.ylabel('Density')


    return round(mean1,2), round(mean2,2), round(lower_limit_1,2), round(upper_limit_1,2), round(lower_limit_2,2), round(upper_limit_2,2)
```

[49]:

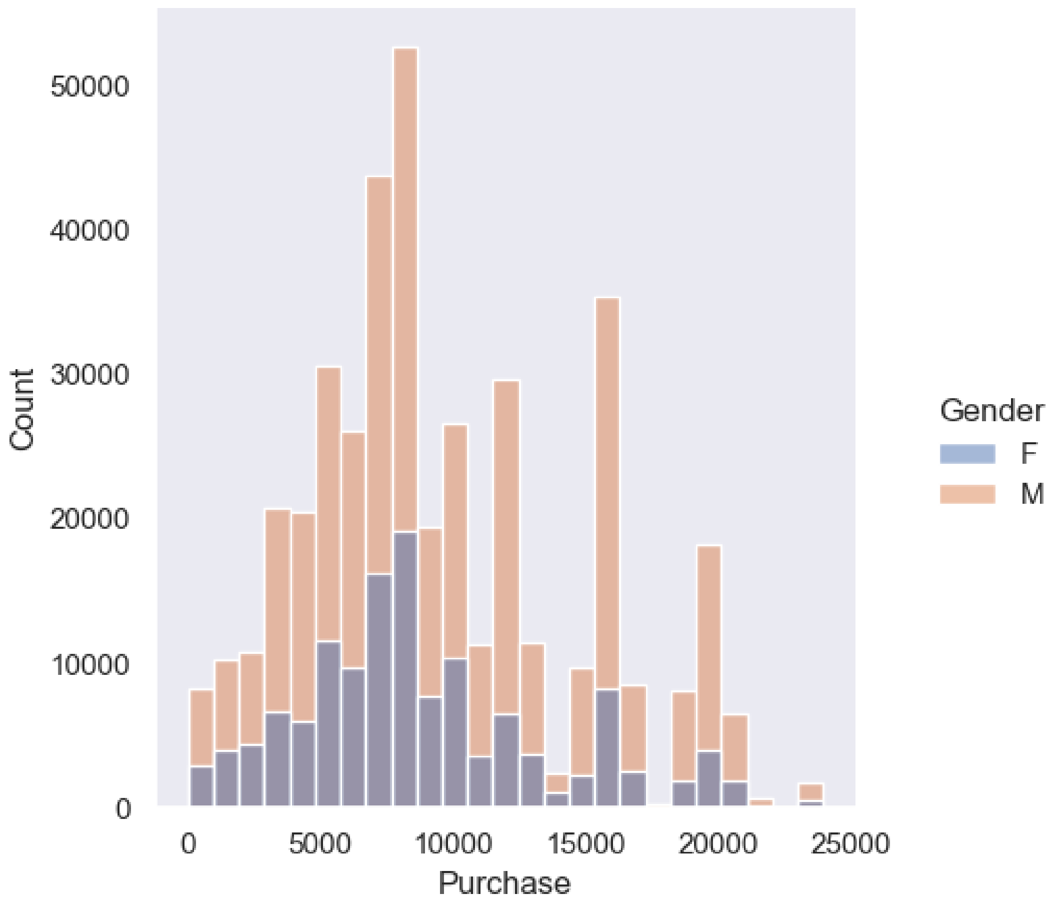
```
df_male = df[df['Gender']=='M']
df_female = df[df['Gender']=='F']
```

1.2.1 Male Vs Female Purchase Values

[50]:

```
plt.figure(figsize=(12,8))
sns.set(style='dark')
sns.displot(x= 'Purchase',data=df,hue='Gender',bins=25)
plt.show()
```

```
<Figure size 1200x800 with 0 Axes>
```



- We can observe that Male spend more than Female.

[51]:

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

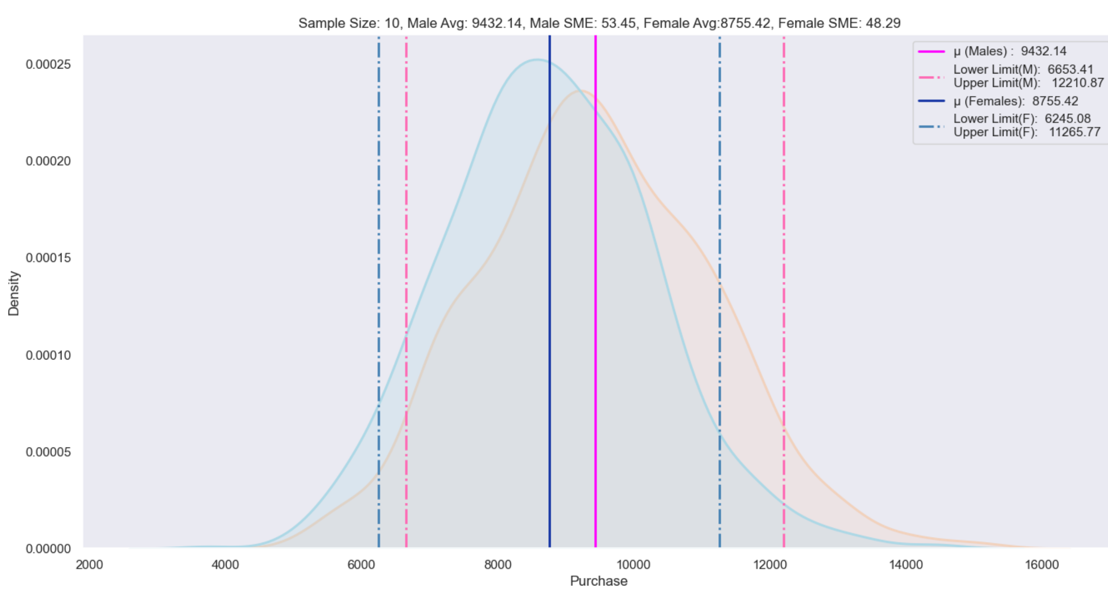
[51]:

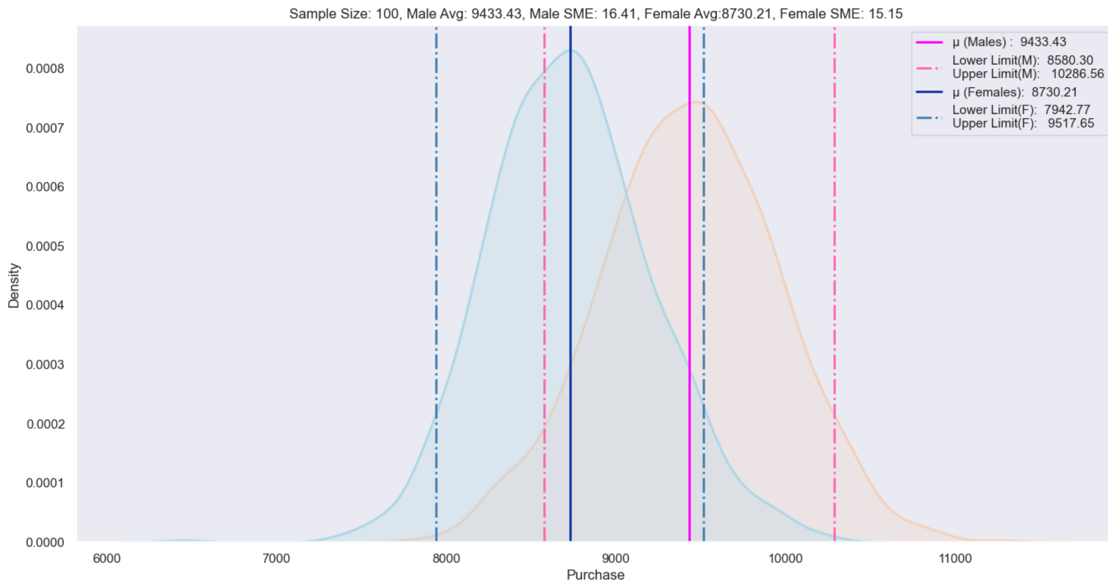
```
           count         mean          std  min     25%     50%      75%  \
Gender
F       135809.0  8734.565765  4767.233289  12.0  5433.0  7914.0  11400.0
```

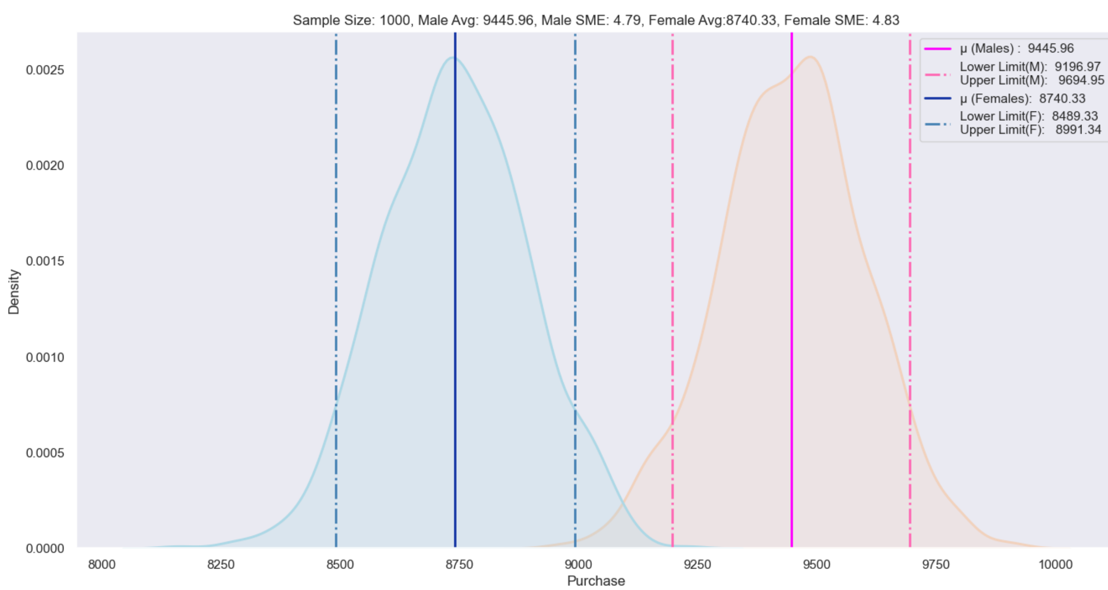
```
M       414259.0  9437.526040  5092.186210  12.0  5863.0  8098.0  12454.0

           max
Gender
F       23959.0
M       23961.0
```

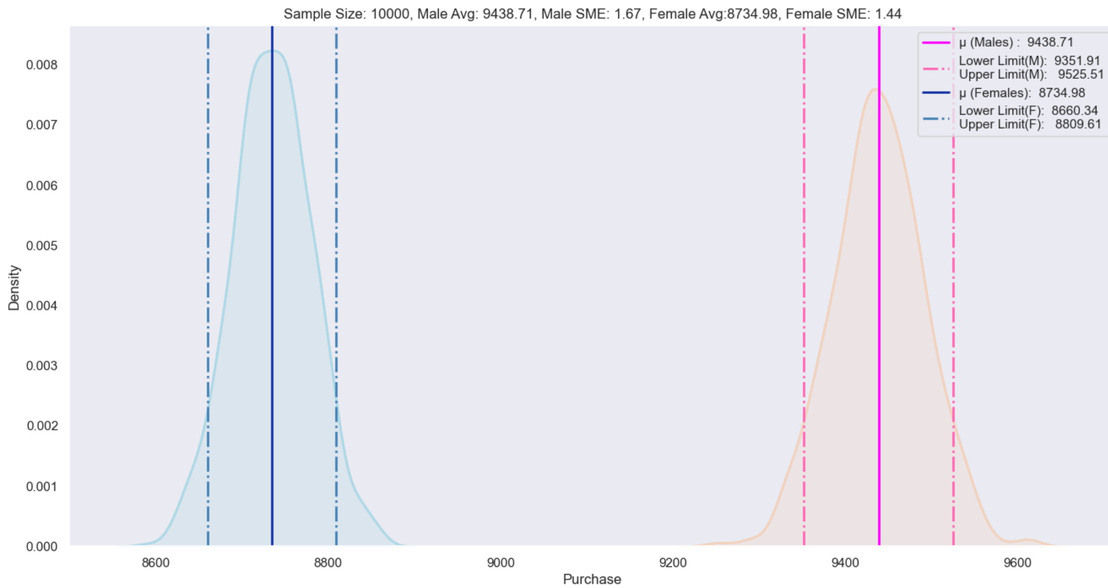
Lets plot the mean of 1000 Random Samples of sizes 10,100,1000,10000 and 100000 with 90% Confidence Interval

[52]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 90
itr_size = 1000

res = pd.DataFrame(columns = ['Gender','Sample Size','Lower Limit','Upper␣
↪Limit','Sample Mean','Confidence Interval','Interval Range','Range'])

for i in sample_sizes:
    m_avg, f_avg, ll_m, ul_m, ll_f, ul_f =␣
↪bootstrap(df_male['Purchase'],df_female['Purchase'],i,itr_size,ci)

    res = res.append({'Gender':'M','Sample Size':i,'Lower Limit':ll_m,'Upper␣
↪Limit':ul_m,'Sample Mean':m_avg,'Confidence Interval':ci,'Interval Range':
↪[ll_m,ul_m],'Range': ul_m-ll_m}, ignore_index = True)
    res = res.append({'Gender':'F','Sample Size':i,'Lower Limit':ll_f,'Upper␣
↪Limit':ul_f,'Sample Mean':f_avg,'Confidence Interval':ci,'Interval Range':
↪[ll_f,ul_f],'Range': ul_f-ll_f}, ignore_index = True)
```




Sample Size: 100, Male Avg: 9433.43, Male SME: 16.41, Female Avg:8730.21, Female SME: 15.15
μ (Males) : 9433.43
Lower Limit(M): 8580.30
Upper Limit(M): 10286.56
μ (Females): 8730.21
Lower Limit(F): 7942.77
Upper Limit(F): 9517.65
Density
Purchase


Sample Size: 1000, Male Avg: 9445.96, Male SME: 4.79, Female Avg:8740.33, Female SME: 4.83
μ (Males) : 9445.96
Lower Limit(M): 9196.97
Upper Limit(M): 9694.95
μ (Females): 8740.33
Lower Limit(F): 8489.33
Upper Limit(F): 8991.34
Density
Purchase

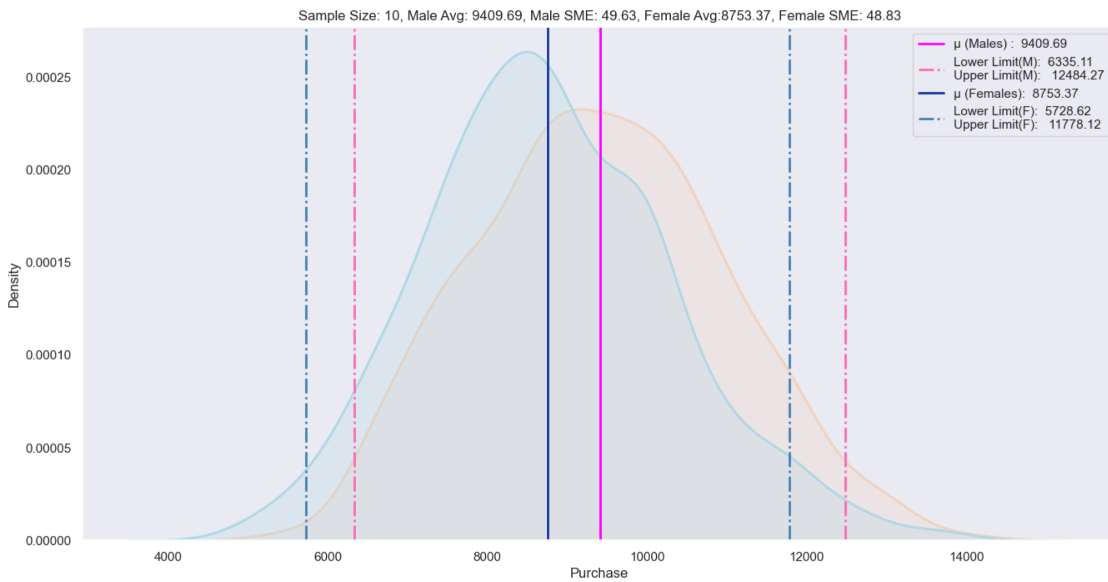


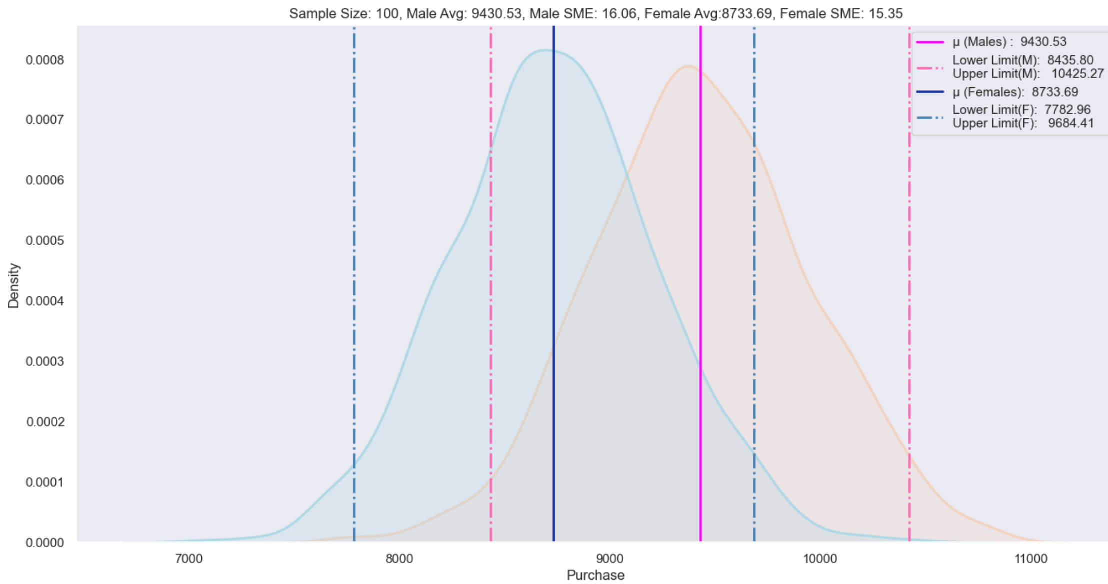
We can observe that as the sample size increases,

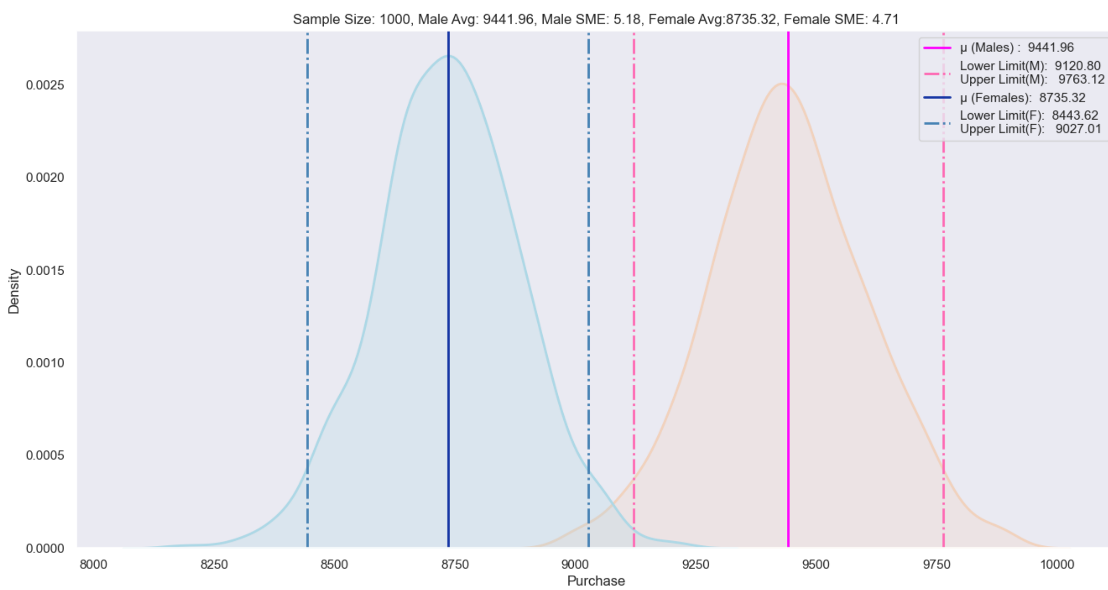
- The average for both of them change significantly.
- Both the plots start to seperate and become distinct.

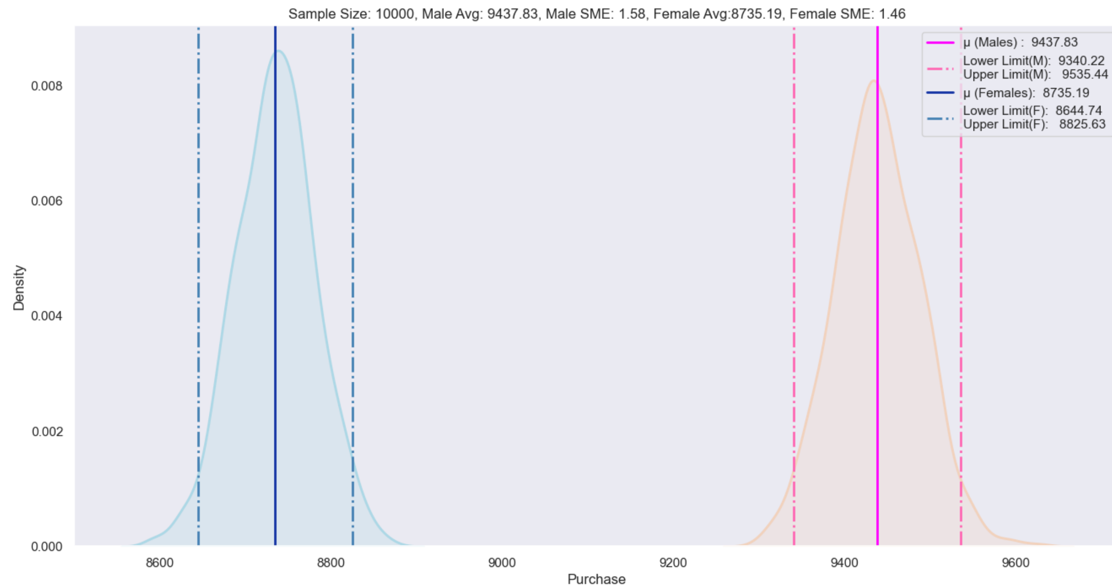
Lets plot the mean of 1000 Random Samples of sizes 10,100,1000,10000 and 100000 with 95% Confidence Interval

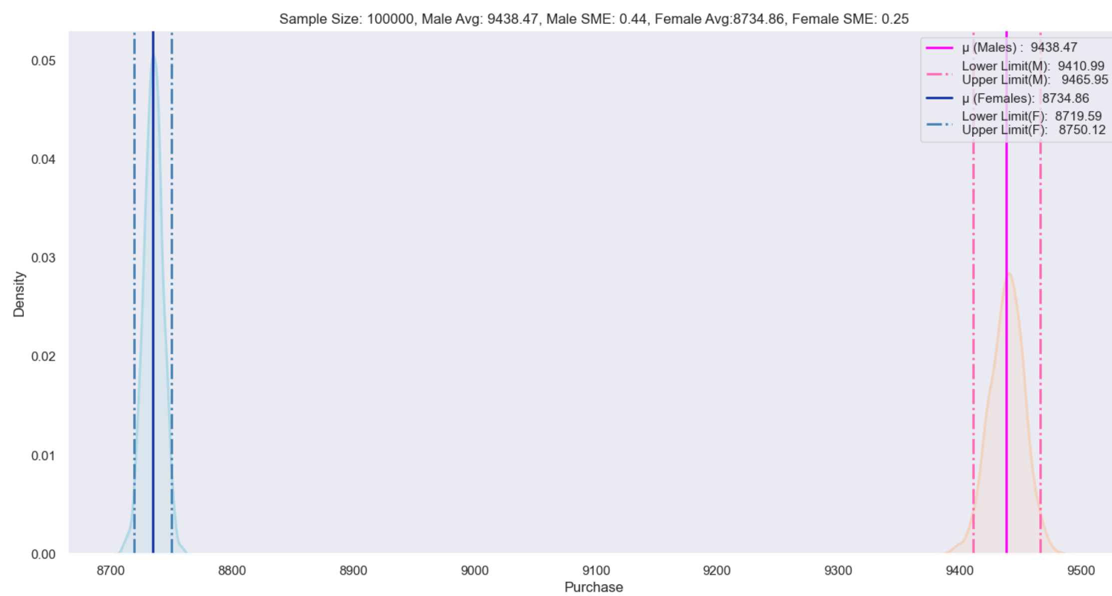
[53]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 95
itr_size = 1000

for i in sample_sizes:
    m_avg, f_avg, ll_m, ul_m, ll_f, ul_f = bootstrap(df_male['Purchase'],df_female['Purchase'],i,itr_size,ci)

    res = res.append({'Gender':'M','Sample Size':i,'Lower Limit':ll_m,'Upper Limit':ul_m,'Sample Mean':m_avg,'Confidence Interval':ci,'Interval Range':[ll_m,ul_m],'Range': ul_m-ll_m}, ignore_index = True)
    res = res.append({'Gender':'F','Sample Size':i,'Lower Limit':ll_f,'Upper Limit':ul_f,'Sample Mean':f_avg,'Confidence Interval':ci,'Interval Range':[ll_f,ul_f],'Range': ul_f-ll_f}, ignore_index = True)
```




Sample Size: 100, Male Avg: 9430.53, Male SME: 16.06, Female Avg:8733.69, Female SME: 15.35
μ (Males) : 9430.53
Lower Limit(M): 8435.80
Upper Limit(M): 10425.27
μ (Females): 8733.69
Lower Limit(F): 7782.96
Upper Limit(F): 9684.41
Density
Purchase


Sample Size: 1000, Male Avg: 9441.96, Male SME: 5.18, Female Avg:8735.32, Female SME: 4.71
μ (Males) : 9441.96
Lower Limit(M): 9120.80
Upper Limit(M): 9763.12
μ (Females): 8735.32
Lower Limit(F): 8443.62
Upper Limit(F): 9027.01
Density
Purchase





We understood the graph, lets understand it a bit deeper with the values.

[54]:
```
res
```

[54]:
```
   Gender  Sample Size  Lower Limit  Upper Limit  Sample Mean  \
0       M           10      6653.41     12210.87      9432.14
1       F           10      6245.08     11265.77      8755.42
2       M          100      8580.30     10286.56      9433.43
```

3	F	100	7942.77	9517.65	8730.21
4	M	1000	9196.97	9694.95	9445.96
5	F	1000	8489.33	8991.34	8740.33
6	M	10000	9351.91	9525.51	9438.71
7	F	10000	8660.34	8809.61	8734.98
8	M	100000	9415.08	9460.27	9437.68
9	F	100000	8721.97	8747.07	8734.52
10	M	10	6335.11	12484.27	9409.69
11	F	10	5728.62	11778.12	8753.37
12	M	100	8435.80	10425.27	9430.53
13	F	100	7782.96	9684.41	8733.69
14	M	1000	9120.80	9763.12	9441.96
15	F	1000	8443.62	9027.01	8735.32
16	M	10000	9340.22	9535.44	9437.83
17	F	10000	8644.74	8825.63	8735.19
18	M	100000	9410.99	9465.95	9438.47
19	F	100000	8719.59	8750.12	8734.86

	Confidence Interval	Interval Range	Range
0	90	[6653.41, 12210.87]	5557.46
1	90	[6245.08, 11265.77]	5020.69
2	90	[8580.3, 10286.56]	1706.26
3	90	[7942.77, 9517.65]	1574.88
4	90	[9196.97, 9694.95]	497.98
5	90	[8489.33, 8991.34]	502.01
6	90	[9351.91, 9525.51]	173.60
7	90	[8660.34, 8809.61]	149.27
8	90	[9415.08, 9460.27]	45.19
9	90	[8721.97, 8747.07]	25.10
10	95	[6335.11, 12484.27]	6149.16
11	95	[5728.62, 11778.12]	6049.50
12	95	[8435.8, 10425.27]	1989.47
13	95	[7782.96, 9684.41]	1901.45
14	95	[9120.8, 9763.12]	642.32
15	95	[8443.62, 9027.01]	583.39
16	95	[9340.22, 9535.44]	195.22
17	95	[8644.74, 8825.63]	180.89
18	95	[9410.99, 9465.95]	54.96
19	95	[8719.59, 8750.12]	30.53

We can observe that

- The CI with 90% confidence for sample size 10 for Males is [6653.41, 12210.87]
- The CI with 90% confidence for sample size 10 for Females is [6245.08, 11265.77]
- For Sample size 10 The confidence interval for both Male and Female is overlapping

and as the sample size increases, we can see the interval ranges seperating and then finally they both dont overalap.

- The CI with 90% confidence for sample size 100000 for Males is [9415.08, 9460.27]
- The CI with 90% confidence for sample size 100000 for Females is [8721.97, 8747.07]
- For Sample size 100000 The confidence interval for both Male and Female is now not overlapping.

We can also observe the same with 95% Confidence.

- The CI with 95% confidence for sample size 10 for Males is [6335.11, 12484.27]
- The CI with 95% confidence for sample size 10 for Females is [5728.62, 11778.12]
- For Sample size 10 The confidence interval for both Male and Female is overlapping

and as the sample size increases, we can see the interval ranges seperating and then finally they both dont overalap.

- The CI with 95% confidence for sample size 100000 for Males is [9410.99, 9465.95]
- The CI with 95% confidence for sample size 100000 for Females is [8719.59, 8750.12]
- For Sample size 100000 The confidence interval for both Male and Female is now not overlapping.

1.2.2 Married Vs Unmarried Purchase Values

[55]:
```
def bootstrap_m_vs_um(sample1,sample2,sample_size,itr_size=1000,ci=90):
    ci = ci/100

    plt.figure(figsize=(16,8))
    sample1_n = [np.mean(sample1.sample(sample_size)) for i in range(itr_size)]
    sample2_n = [np.mean(sample2.sample(sample_size)) for i in range(itr_size)]

    # For Sample1's means
    mean1 = np.mean(sample1_n)
    sigma1 = np.std(sample1_n)
    sem1 = stats.sem(sample1_n)

    lower_limit_1 = norm.ppf((1-ci)/2) * sigma1 + mean1
    upper_limit_1 = norm.ppf(ci+(1-ci)/2) * sigma1 + mean1

    # For Sample2's means
    mean2 = np.mean(sample2_n)
    sigma2 = np.std(sample2_n)
    sem2 = stats.sem(sample2_n)

    lower_limit_2 = norm.ppf((1-ci)/2) * sigma2 + mean2
    upper_limit_2 = norm.ppf(ci + (1-ci)/2) * sigma2 + mean2

    sns.kdeplot(data = sample1_n, color="#F2D2BD", fill = True, linewidth = 2)
```

```
    label_mean1=("  (Married) :  {:.2f}".format(mean1))
    plt.axvline(mean1, color = '#FF00FF', linestyle = 'solid', linewidth = 2,␣
↪label=label_mean1)
    label_limits1=("Lower Limit(M):  {:.2f}\nUpper Limit(M):   {:.2f}".
↪format(lower_limit_1,upper_limit_1))
    plt.axvline(lower_limit_1, color = '#FF69B4', linestyle = 'dashdot',␣
↪linewidth = 2, label=label_limits1)
    plt.axvline(upper_limit_1, color = '#FF69B4', linestyle = 'dashdot',␣
↪linewidth = 2)

    sns.kdeplot(data = sample2_n ,color='#ADD8E6', fill = True, linewidth = 2)
    label_mean2=("  (Unmarried):  {:.2f}".format(mean2))
    plt.axvline(mean2, color = '#1434A4', linestyle = 'solid', linewidth = 2,␣
↪label=label_mean2)
    label_limits2=("Lower Limit(F):  {:.2f}\nUpper Limit(F):   {:.2f}".
↪format(lower_limit_2,upper_limit_2))
    plt.axvline(lower_limit_2, color = '#4682B4', linestyle = 'dashdot',␣
↪linewidth = 2, label=label_limits2)
    plt.axvline(upper_limit_2, color = '#4682B4', linestyle = 'dashdot',␣
↪linewidth = 2)

    plt.title(f"Sample Size: {sample_size}, Married Avg: {np.round(mean1, 2)},␣
↪Married SME: {np.round(sem1,2)}, Unmarried Avg:{np.round(mean2, 2)},␣
↪Unmarried SME: {np.round(sem2,2)}")
    plt.legend(loc = 'upper right')
    plt.xlabel('Purchase')
    plt.ylabel('Density')

    return round(mean1,2), round(mean2,2), round(lower_limit_1,2),␣
↪round(upper_limit_1,2), round(lower_limit_2,2), round(upper_limit_2,2)
```

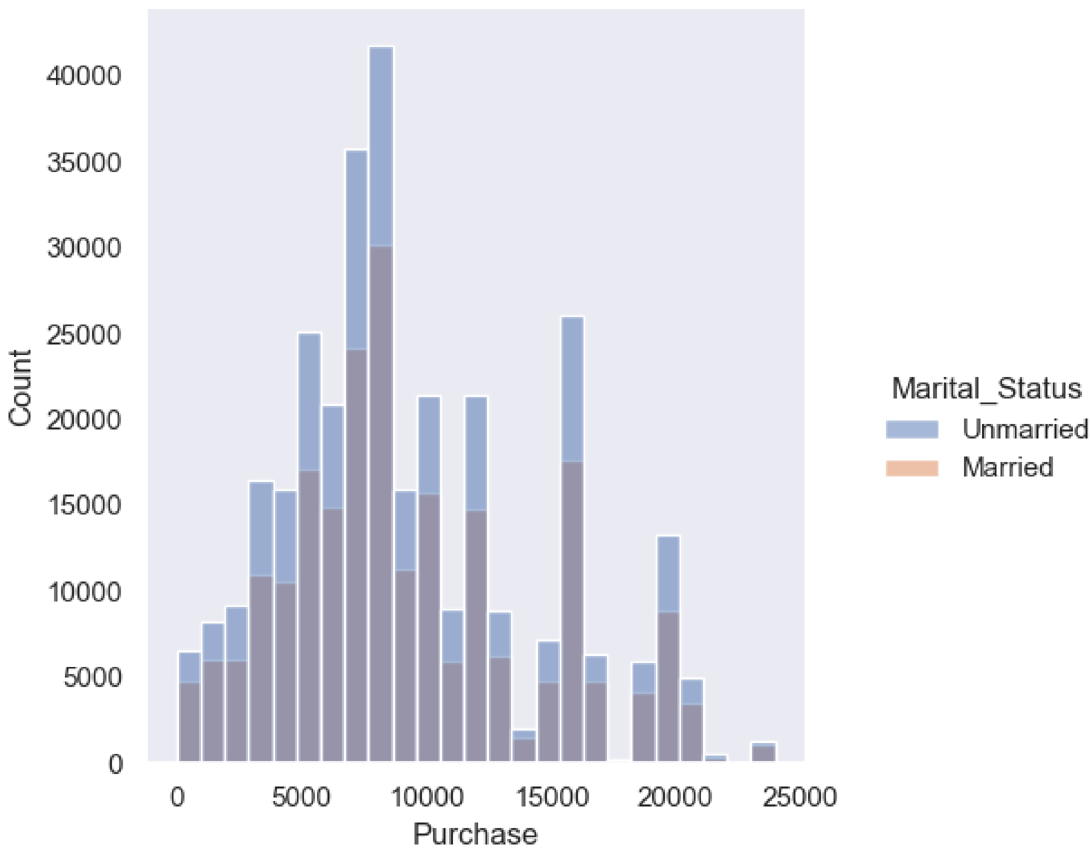
[56]:

```
df_married = df[df['Marital_Status'] == 'Married']
df_unmarried = df[df['Marital_Status'] == 'Unmarried']
```

[57]:

```
plt.figure(figsize = (16,8))
sns.displot(data = df, x = 'Purchase', hue = 'Marital_Status',bins = 25)
plt.show()
```

```
<Figure size 1600x800 with 0 Axes>
```



- The count of orders of unmarried customers is more than Married customers.

[58]:
```
df.groupby(['Marital_Status'])['Purchase'].describe()
```

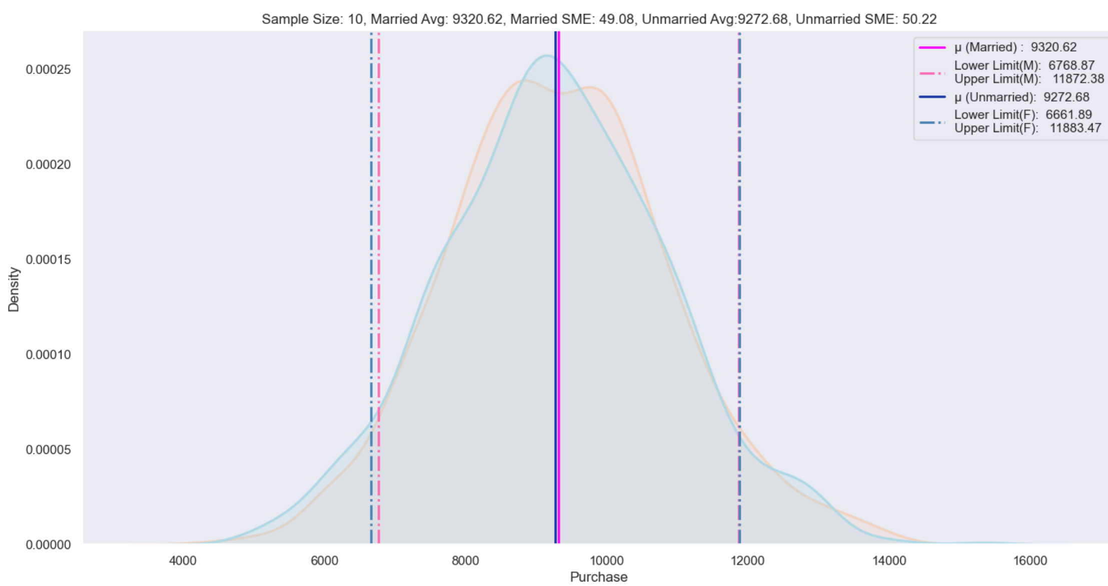
[58]:
```
                   count         mean          std   min     25%     50%  \
Marital_Status
Unmarried       324731.0  9265.907619  5027.347859  12.0  5605.0  8044.0
Married         225337.0  9261.174574  5016.897378  12.0  5843.0  8051.0

                    75%      max
Marital_Status
Unmarried       12061.0  23961.0
Married         12042.0  23961.0
```

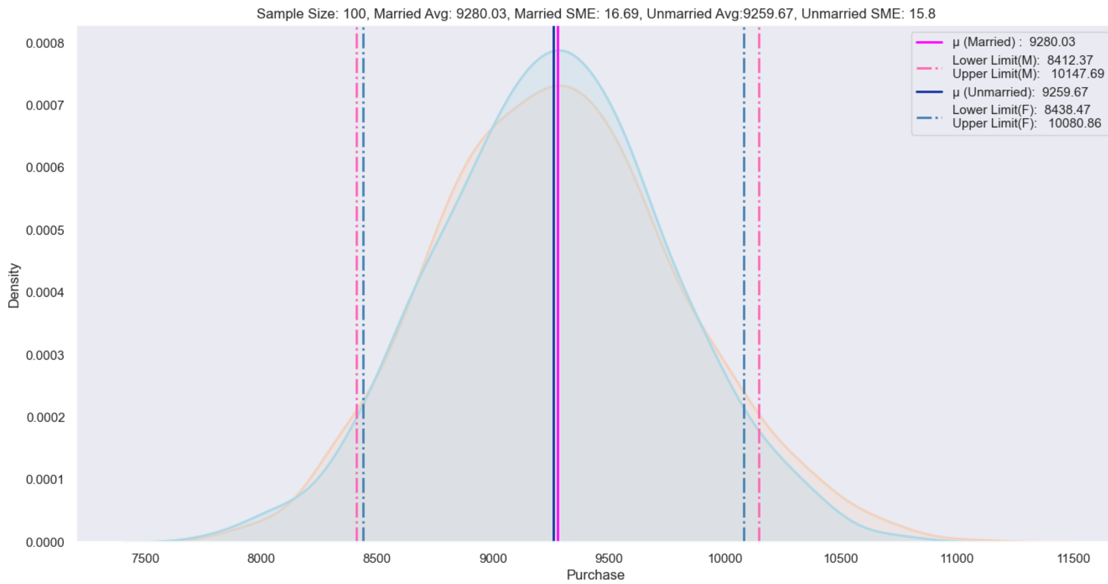
- There is no difference in the mean or median values for both of them.

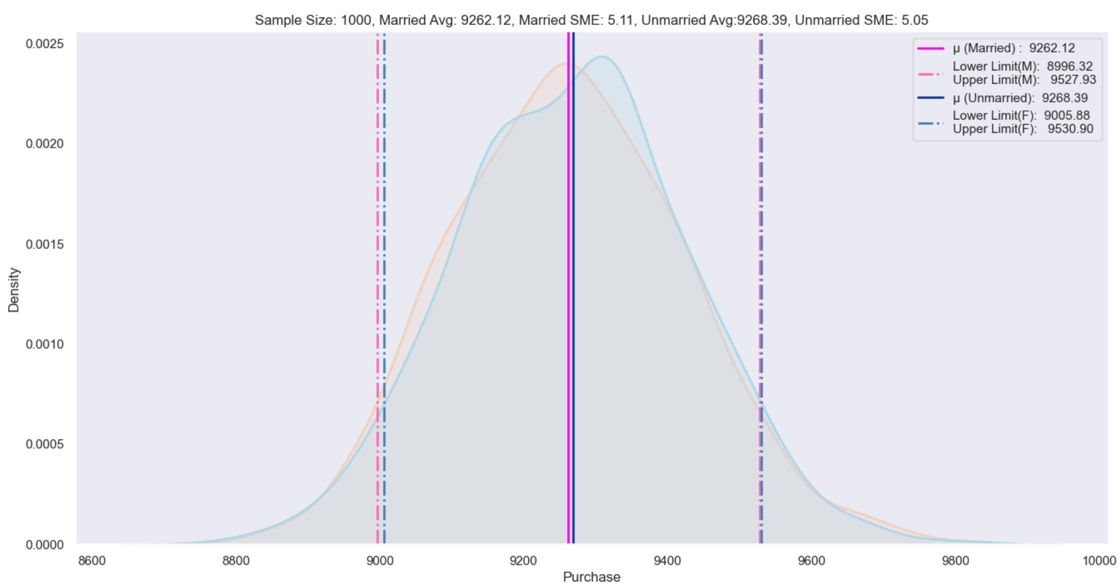
Lets dive deeper using bootstrapping and verify.

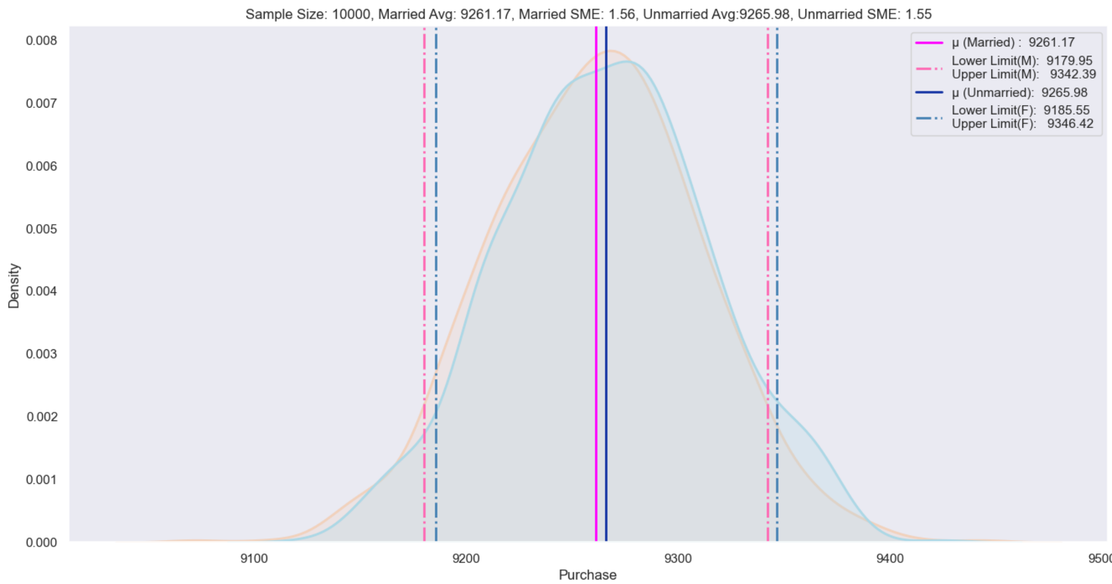
Lets plot the mean of 1000 Random Samples of sizes 10,100,1000,10000 and 100000 with 90% Confidence Interval

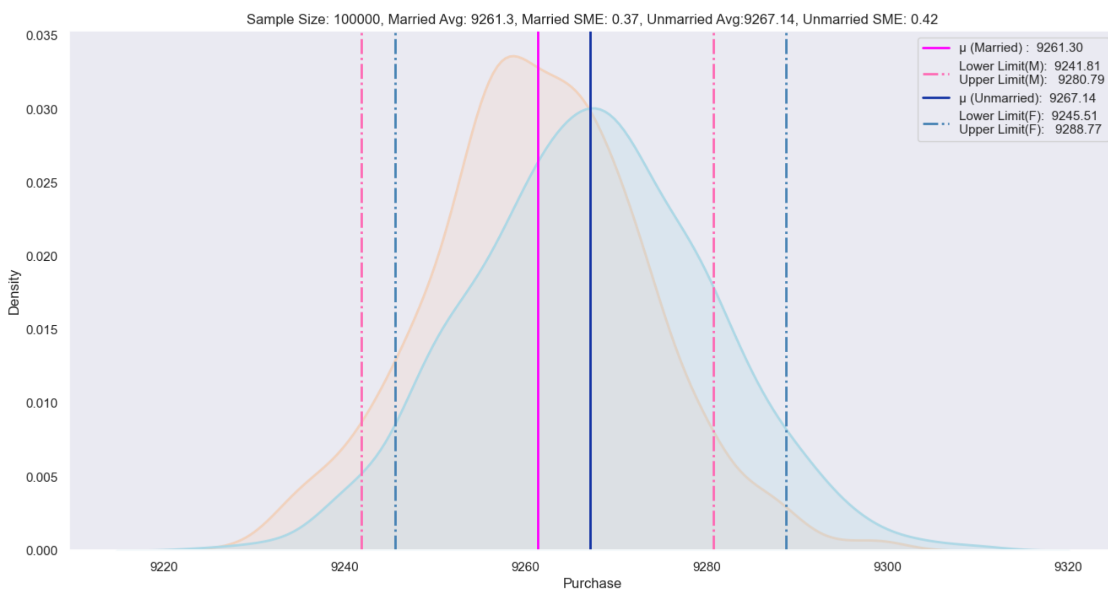
[59]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 90
itr_size = 1000

res = pd.DataFrame(columns = ['Marital_Status','Sample Size','Lower␣
 ↪Limit','Upper Limit','Sample Mean','Confidence Interval','Interval␣
 ↪Range','Range'])

for i in sample_sizes:
    m_avg, un_avg, ll_m, ul_m, ll_un, ul_un =␣
 ↪bootstrap_m_vs_um(df_married['Purchase'],df_unmarried['Purchase'],i,itr_size,ci)

    res = res.append({'Marital_Status':'Married','Sample Size':i,'Lower Limit':
 ↪ll_m,'Upper Limit':ul_m,'Sample Mean':m_avg,'Confidence Interval':
 ↪ci,'Interval Range':[ll_m,ul_m],'Range': ul_m-ll_m}, ignore_index = True)
    res = res.append({'Marital_Status':'Unmarried','Sample Size':i,'Lower␣
 ↪Limit':ll_un,'Upper Limit':ul_un,'Sample Mean':un_avg,'Confidence Interval':
 ↪ci,'Interval Range':[ll_un,ul_un],'Range': ul_un-ll_un}, ignore_index = True)
```




Sample Size: 100, Married Avg: 9280.03, Married SME: 16.69, Unmarried Avg:9259.67, Unmarried SME: 15.8
μ (Married) : 9280.03
Lower Limit(M): 8412.37
Upper Limit(M): 10147.69
μ (Unmarried): 9259.67
Lower Limit(F): 8438.47
Upper Limit(F): 10080.86
Density
Purchase


Sample Size: 1000, Married Avg: 9262.12, Married SME: 5.11, Unmarried Avg:9268.39, Unmarried SME: 5.05
μ (Married) : 9262.12
Lower Limit(M): 8996.32
Upper Limit(M): 9527.93
μ (Unmarried): 9268.39
Lower Limit(F): 9005.88
Upper Limit(F): 9530.90
Density
Purchase

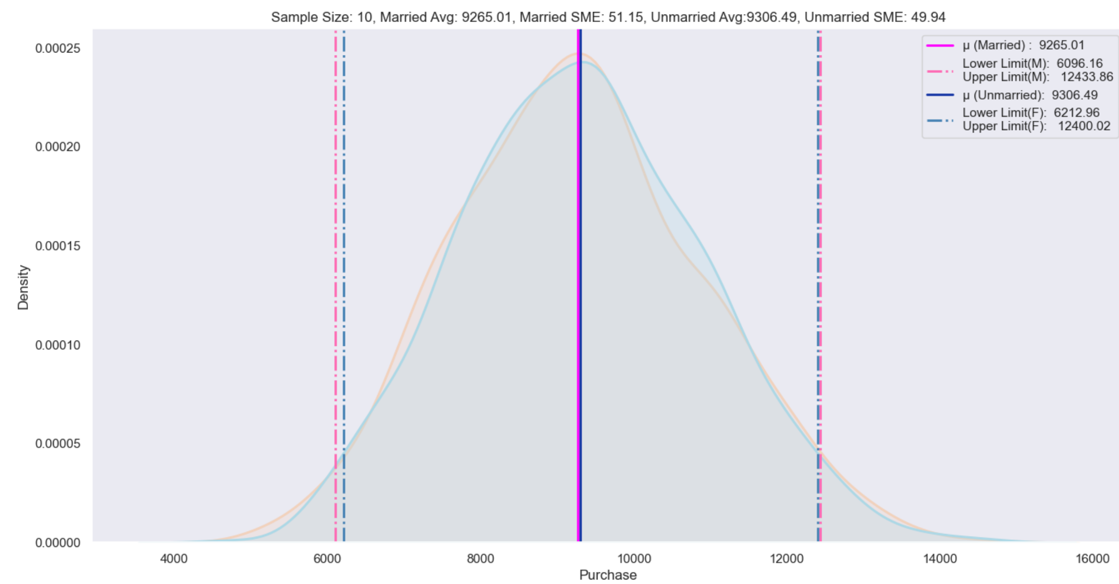


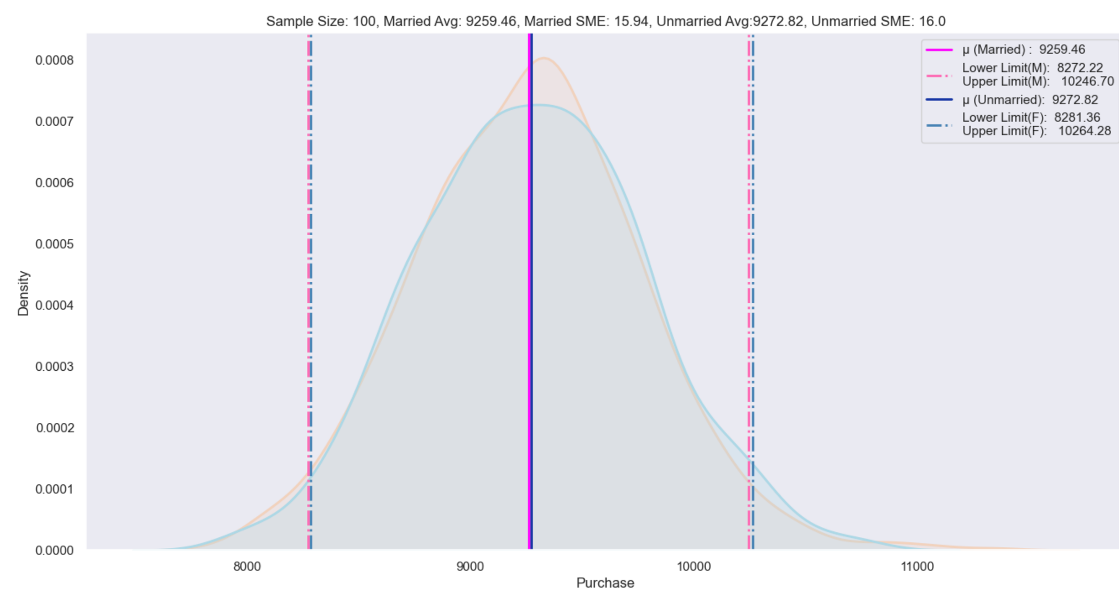


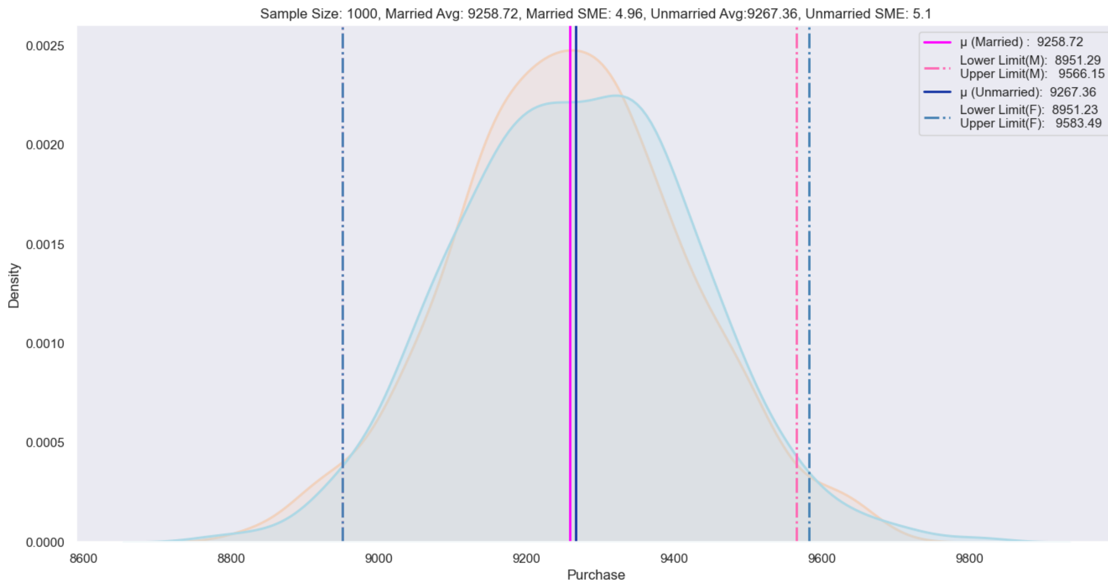
Lets plot the mean of 1000 Random Samples of sizes 10,100,1000,10000 and 100000 with 95% Confidence Interval

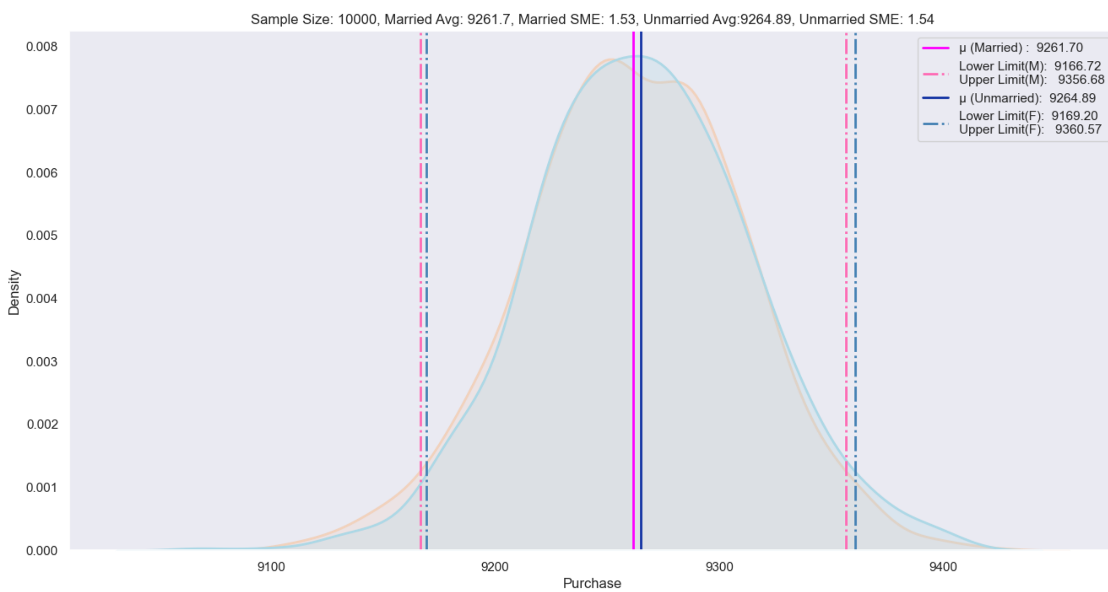
[60]:
```
sample_sizes = [10,100,1000,10000,100000]
ci = 95
itr_size = 1000

for i in sample_sizes:
```

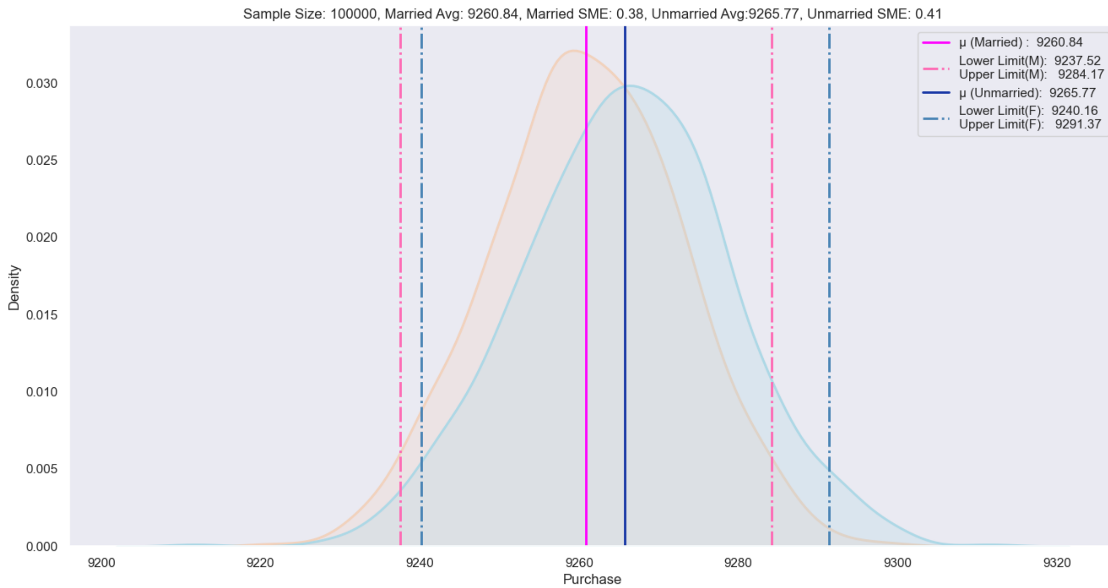
```
    m_avg, un_avg, ll_m, ul_m, ll_un, ul_un = bootstrap_m_vs_um(df_married['Purchase'],df_unmarried['Purchase'],i,itr_size,ci)

    res = res.append({'Marital_Status':'Married','Sample Size':i,'Lower Limit':ll_m,'Upper Limit':ul_m,'Sample Mean':m_avg,'Confidence Interval':ci,'Interval Range':[ll_m,ul_m],'Range': ul_m-ll_m}, ignore_index = True)
    res = res.append({'Marital_Status':'Unmarried','Sample Size':i,'Lower Limit':ll_un,'Upper Limit':ul_un,'Sample Mean':un_avg,'Confidence Interval':ci,'Interval Range':[ll_un,ul_un],'Range': ul_un-ll_un}, ignore_index = True)
```






Sample Size: 1000, Married Avg: 9258.72, Married SME: 4.96, Unmarried Avg:9267.36, Unmarried SME: 5.1
μ (Married) : 9258.72
Lower Limit(M): 8951.29
Upper Limit(M): 9566.15
μ (Unmarried): 9267.36
Lower Limit(F): 8951.23
Upper Limit(F): 9583.49
Density
Purchase
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800


Sample Size: 10000, Married Avg: 9261.7, Married SME: 1.53, Unmarried Avg:9264.89, Unmarried SME: 1.54
μ (Married) : 9261.70
Lower Limit(M): 9166.72
Upper Limit(M): 9356.68
μ (Unmarried): 9264.89
Lower Limit(F): 9169.20
Upper Limit(F): 9360.57
Density
Purchase
0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0.000
9100
9200
9300
9400



We can observe that

- There is overlapping even if we increase the sample size.
- There is no effect of their marital status on their purchases.

[61]:
```
res
```

[61]:

	Marital_Status	Sample Size	Lower Limit	Upper Limit	Sample Mean	\
0	Married	10	6768.87	11872.38	9320.62	
1	Unmarried	10	6661.89	11883.47	9272.68	
2	Married	100	8412.37	10147.69	9280.03	
3	Unmarried	100	8438.47	10080.86	9259.67	
4	Married	1000	8996.32	9527.93	9262.12	
5	Unmarried	1000	9005.88	9530.90	9268.39	
6	Married	10000	9179.95	9342.39	9261.17	
7	Unmarried	10000	9185.55	9346.42	9265.98	
8	Married	100000	9241.81	9280.79	9261.30	
9	Unmarried	100000	9245.51	9288.77	9267.14	
10	Married	10	6096.16	12433.86	9265.01	
11	Unmarried	10	6212.96	12400.02	9306.49	
12	Married	100	8272.22	10246.70	9259.46	
13	Unmarried	100	8281.36	10264.28	9272.82	
14	Married	1000	8951.29	9566.15	9258.72	
15	Unmarried	1000	8951.23	9583.49	9267.36	
16	Married	10000	9166.72	9356.68	9261.70	
17	Unmarried	10000	9169.20	9360.57	9264.89	
18	Married	100000	9237.52	9284.17	9260.84	
19	Unmarried	100000	9240.16	9291.37	9265.77	

```
   Confidence Interval       Interval Range    Range
0                   90  [6768.87, 11872.38]  5103.51
1                   90  [6661.89, 11883.47]  5221.58
2                   90  [8412.37, 10147.69]  1735.32
3                   90  [8438.47, 10080.86]  1642.39
4                   90   [8996.32, 9527.93]   531.61
5                   90    [9005.88, 9530.9]   525.02
6                   90   [9179.95, 9342.39]   162.44
7                   90   [9185.55, 9346.42]   160.87
8                   90   [9241.81, 9280.79]    38.98
9                   90   [9245.51, 9288.77]    43.26
10                  95  [6096.16, 12433.86]  6337.70
11                  95  [6212.96, 12400.02]  6187.06
12                  95   [8272.22, 10246.7]  1974.48
13                  95  [8281.36, 10264.28]  1982.92
14                  95   [8951.29, 9566.15]   614.86
15                  95   [8951.23, 9583.49]   632.26
16                  95   [9166.72, 9356.68]   189.96
17                  95    [9169.2, 9360.57]   191.37
18                  95   [9237.52, 9284.17]    46.65
19                  95   [9240.16, 9291.37]    51.21
```

- For married and unmarried customers, sample size 10, confidence interval 90 we can observe that the interval range is overlapping
- For married and unmarried customers, sample size 100000, confidence interval 90 we can observe that the interval range is still overlapping
- This means there is no effect of marital status on purchase habits of customers

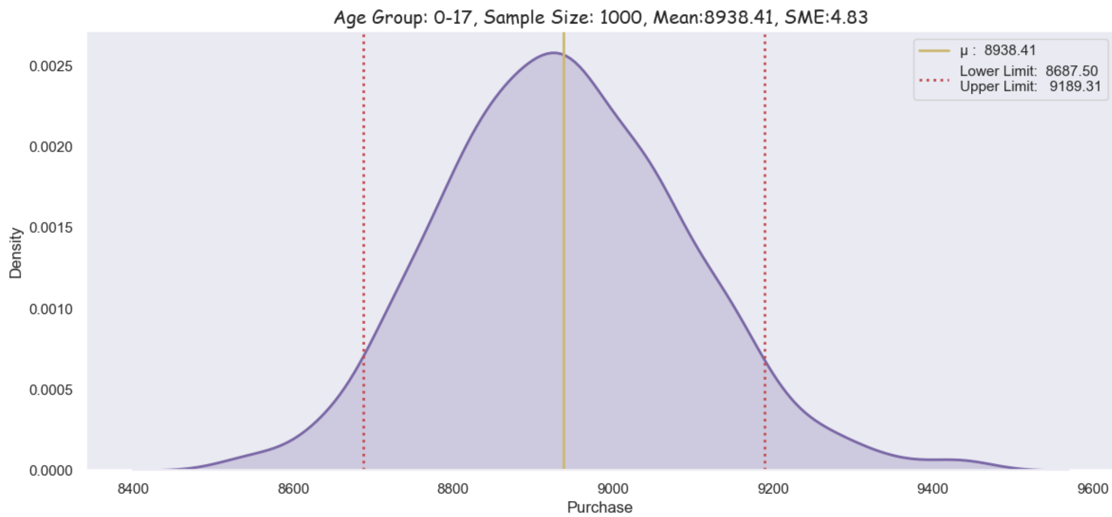
1.2.3 Age groups wise purchase habits

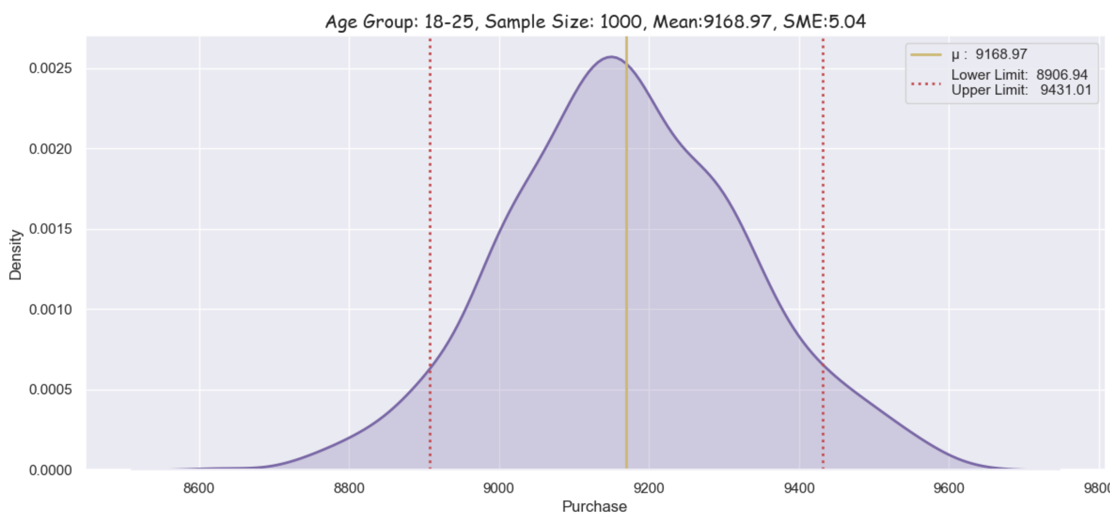
[62]:
```python
def bootstrap_age(sample, sample_size, itr_size=1000, ci = 90):
    ci = ci/100

    global flag

    sample_n = [np.mean(sample.sample(sample_size)) for i in range(itr_size)]

    mean  = np.mean(sample_n)
    sigma = np.std(sample_n)
    sem   = stats.sem(sample_n)

    lower_limit = norm.ppf((1-ci)/2) * sigma + mean
    upper_limit = norm.ppf(ci + (1-ci)/2) * sigma + mean

    fig, ax = plt.subplots(figsize=(14,6))
```

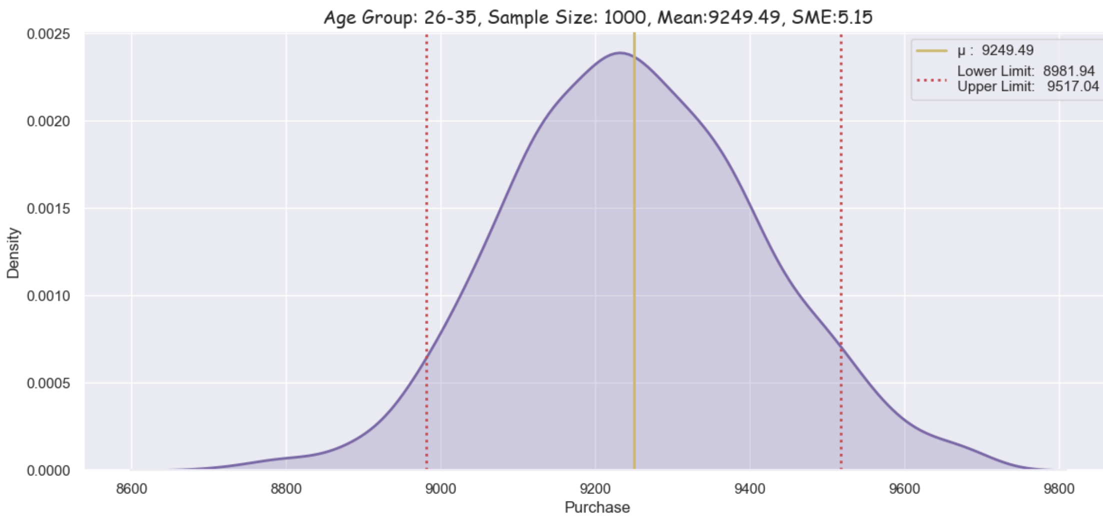
```
    sns.set_style("darkgrid")

    sns.kdeplot(data=sample_n,color="#7A68A6",fill=True,linewidth=2)

    label_mean=("  :  {:.2f}".format(mean))
    label_ult=("Lower Limit:  {:.2f}\nUpper Limit:   {:.2f}".
 ↪format(lower_limit,upper_limit))

    plt.title(f"Age Group: {age_group[flag]}, Sample Size: {sample_size}, Mean:
 ↪{np.round(mean,2)}, SME:{np.round(sem,2)}",fontsize=14,family="Comic Sans␣
 ↪MS")
    plt.xlabel('Purchase')
    plt.axvline(mean, color = 'y', linestyle = 'solid', linewidth =␣
 ↪2,label=label_mean)
    plt.axvline(upper_limit, color = 'r', linestyle = 'dotted', linewidth =␣
 ↪2,label=label_ult)
    plt.axvline(lower_limit, color = 'r', linestyle = 'dotted', linewidth = 2)
    plt.legend(loc='upper right')

    plt.show()
    flag += 1

    return sample_n ,np.round(lower_limit,2),np.round(upper_limit,2),␣
 ↪round(mean,2)
```

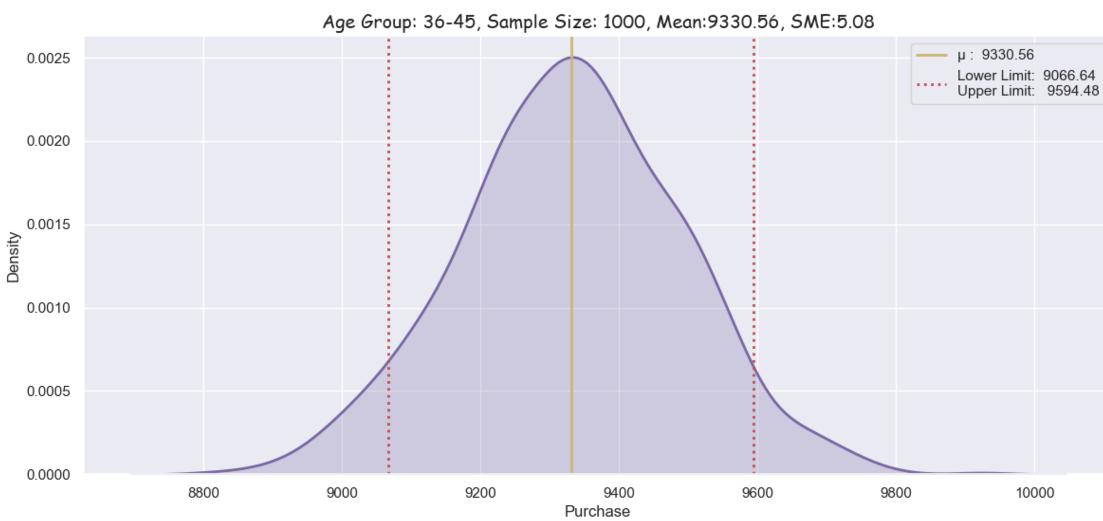
Lets visualise the graphs of 1000 mean values of purchase samples for sample size of 1000 for all the age groups with 90% confidence interval.

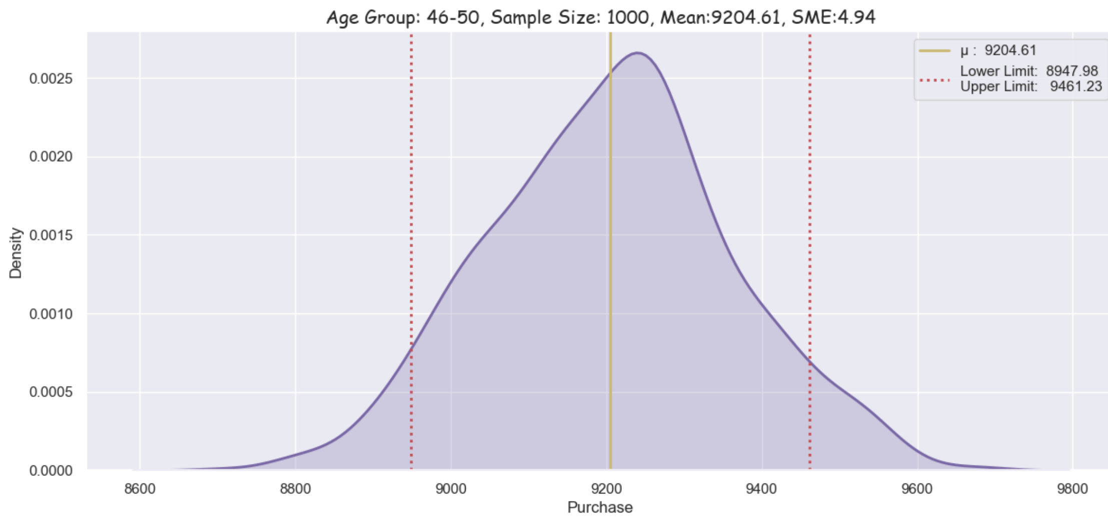
[63]:
```
ci = 90
itr_size = 1000
sample_size = 1000
flag = 0
global age_group
age_group = ['0-17', '18-25', '26-35', '36-45', '46-50', '51-55', '55+']

res = pd.DataFrame(columns = ['Age_Group','Sample Size','Lower Limit','Upper␣
 ↪Limit','Sample Mean','Confidence Interval','Interval Range','Range'])

for i in age_group:
    m_avg, ll, ul, mean =␣
 ↪bootstrap_age(df[df['Age']==i]['Purchase'],sample_size,itr_size,ci)

    res = res.append({'Age_Group':i,'Sample Size':sample_size,'Lower Limit':
 ↪ll,'Upper Limit':ul,'Sample Mean':mean,'Confidence Interval':ci,'Interval␣
 ↪Range':[ll,ul],'Range': ul-ll}, ignore_index = True)
```

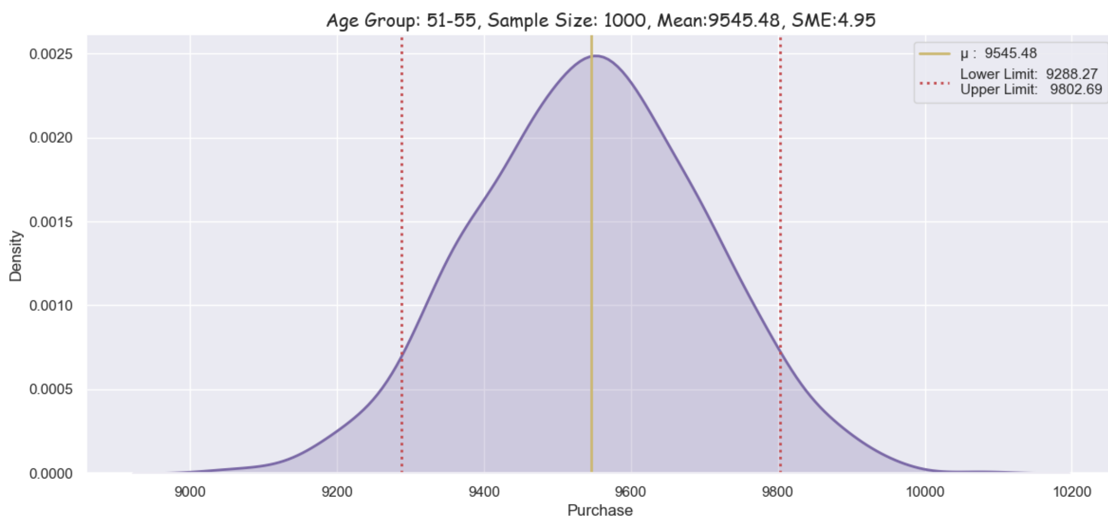

Age Group: 0-17, Sample Size: 1000, Mean:8938.41, SME:4.83
μ : 8938.41
Lower Limit: 8687.50
Upper Limit: 9189.31
Density
Purchase
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
8400
8600
8800
9000
9200
9400
9600

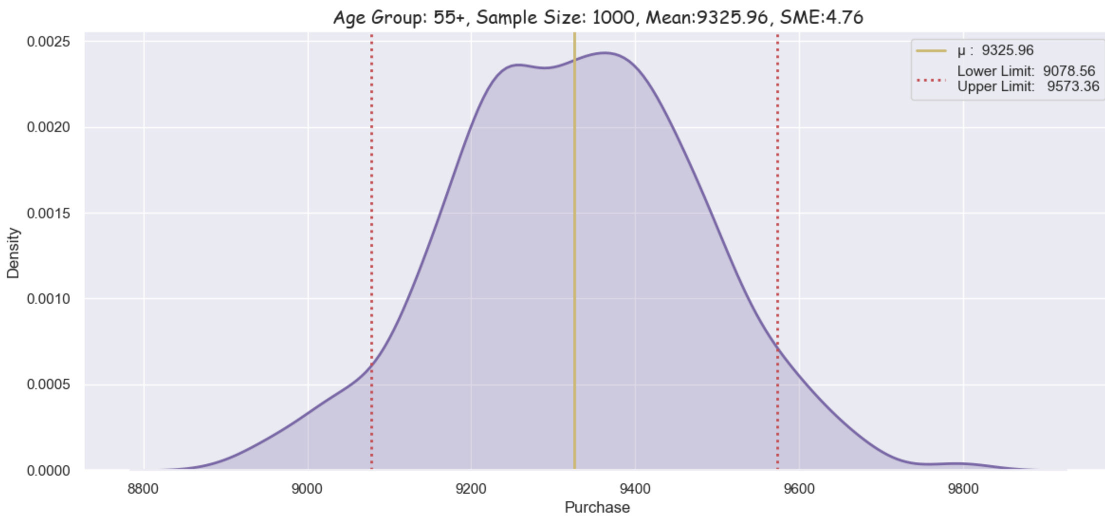

Age Group: 18-25, Sample Size: 1000, Mean:9168.97, SME:5.04
μ : 9168.97
Lower Limit: 8906.94
Upper Limit: 9431.01
Density
Purchase
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
8600
8800
9000
9200
9400
9600
9800


Age Group: 26-35, Sample Size: 1000, Mean:9249.49, SME:5.15
μ : 9249.49
Lower Limit: 8981.94
Upper Limit: 9517.04
Density
Purchase
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800


Age Group: 36-45, Sample Size: 1000, Mean:9330.56, SME:5.08
μ : 9330.56
Lower Limit: 9066.64
Upper Limit: 9594.48
Density
Purchase
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
10000

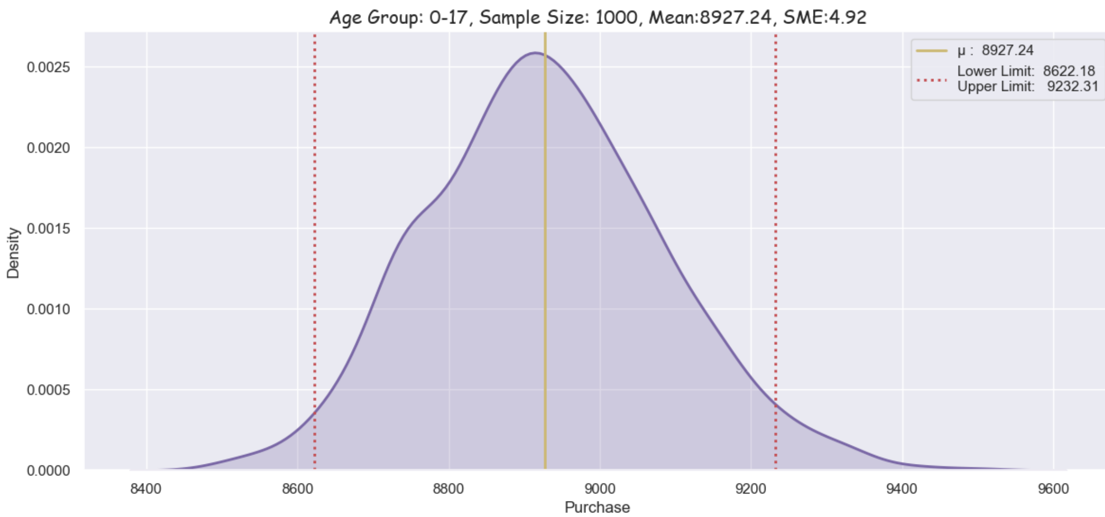

Age Group: 46-50, Sample Size: 1000, Mean:9204.61, SME:4.94
μ : 9204.61
Lower Limit: 8947.98
Upper Limit: 9461.23
Density
Purchase

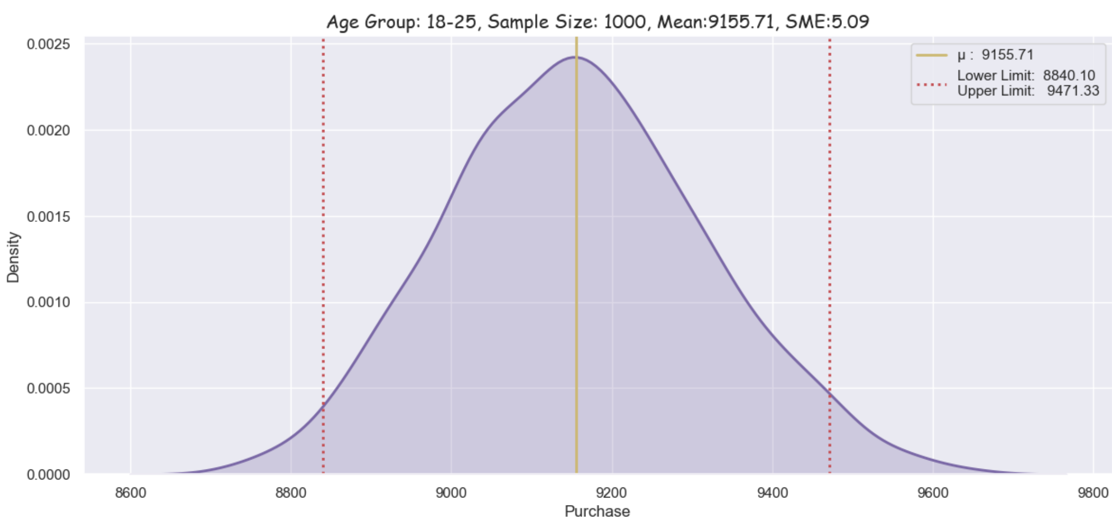

Age Group: 51-55, Sample Size: 1000, Mean:9545.48, SME:4.95
μ : 9545.48
Lower Limit: 9288.27
Upper Limit: 9802.69
Density
Purchase



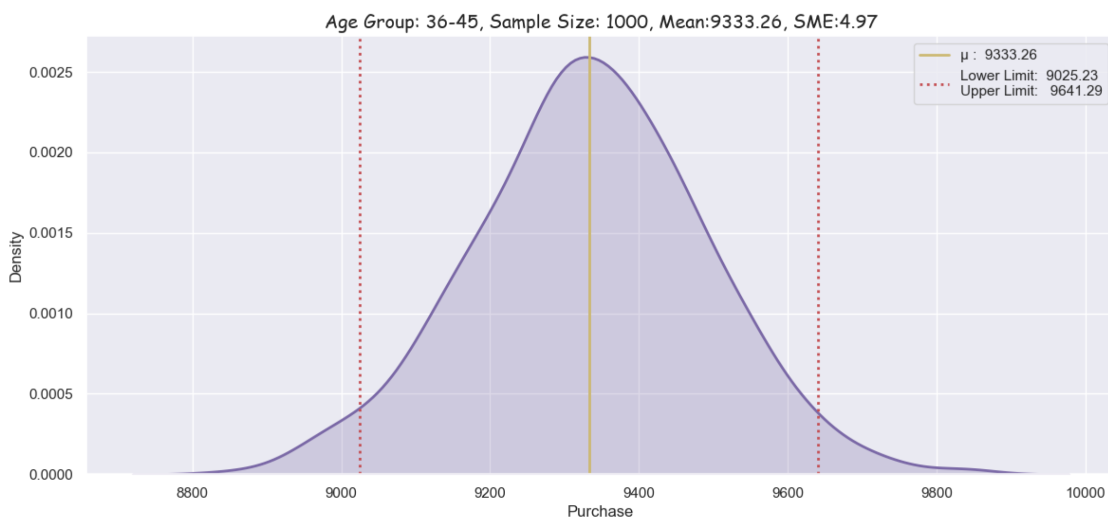
Lets visualise the graphs of 1000 mean values of purchase samples for sample size of 1000 for all the age groups with 95% confidence interval.

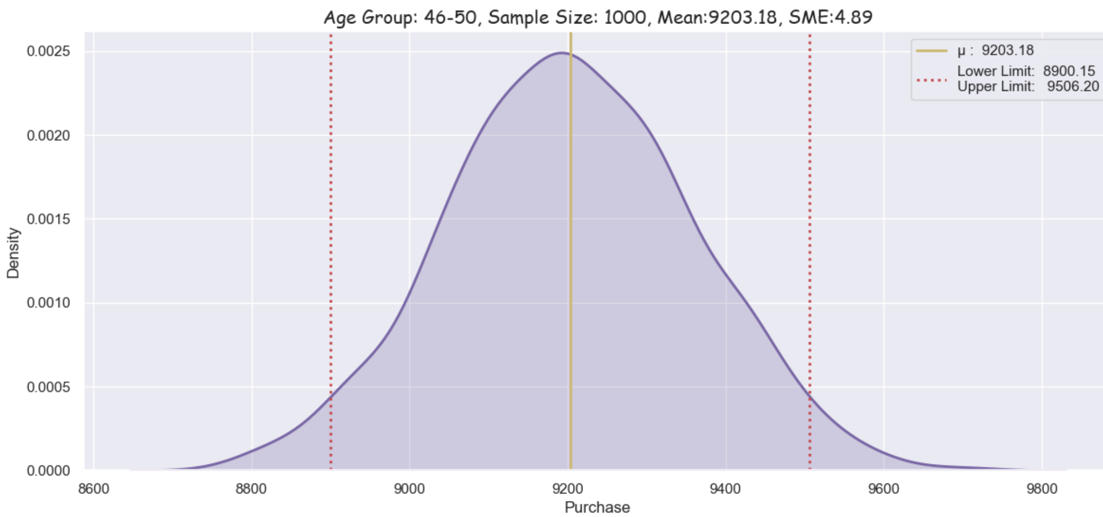
[64]:
```
ci = 95
itr_size = 1000
sample_size = 1000
flag = 0

for i in age_group:
    m_avg, ll, ul, mean =␣
 ↪bootstrap_age(df[df['Age']==i]['Purchase'],sample_size,itr_size,ci)

    res = res.append({'Age_Group':i,'Sample Size':sample_size,'Lower Limit':
 ↪ll,'Upper Limit':ul,'Sample Mean':mean,'Confidence Interval':ci,'Interval␣
 ↪Range':[ll,ul],'Range': ul-ll}, ignore_index = True)
```

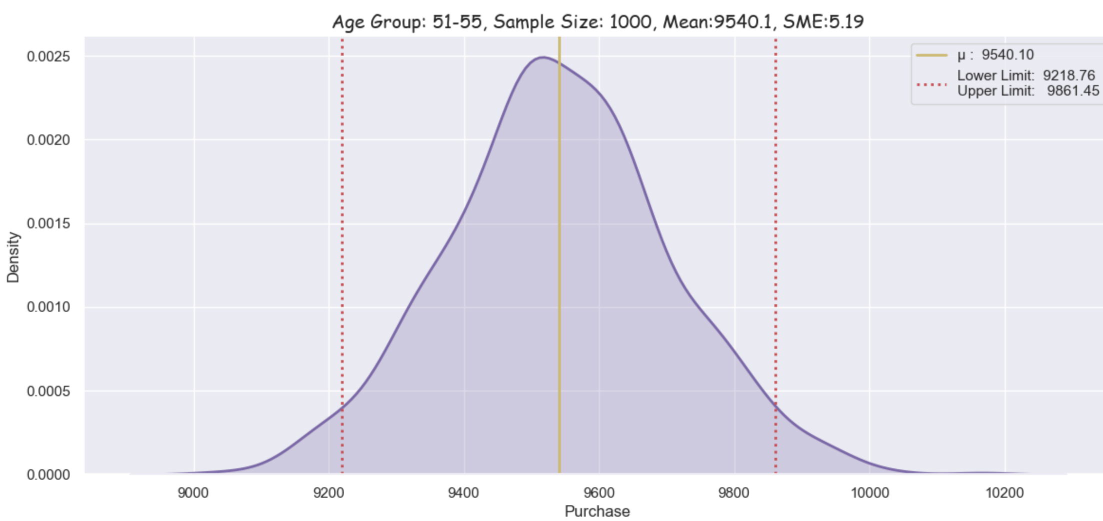

Age Group: 0-17, Sample Size: 1000, Mean:8927.24, SME:4.92
μ : 8927.24
Lower Limit: 8622.18
Upper Limit: 9232.31
Density
Purchase
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
8400
8600
8800
9000
9200
9400
9600

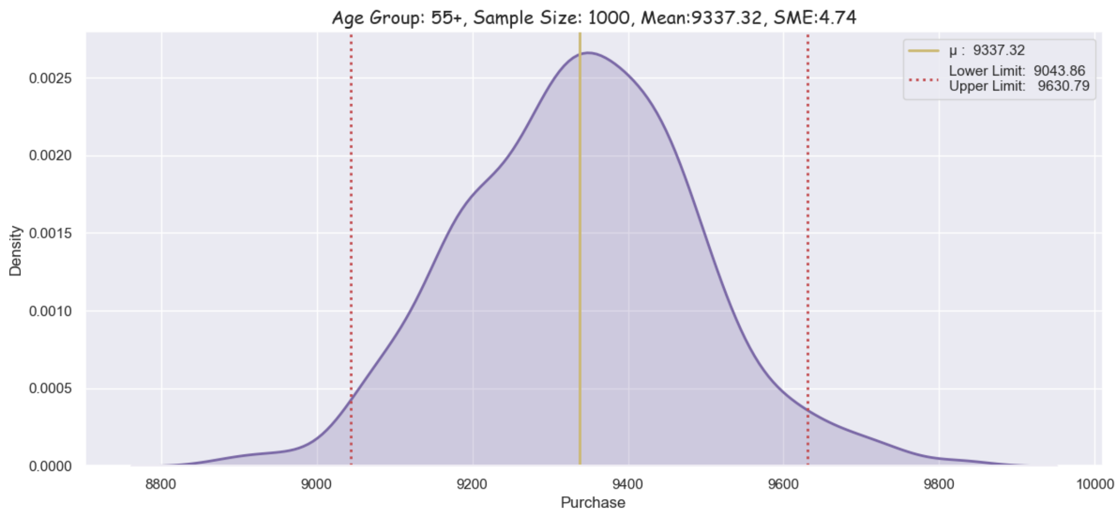

Age Group: 18-25, Sample Size: 1000, Mean:9155.71, SME:5.09
μ : 9155.71
Lower Limit: 8840.10
Upper Limit: 9471.33
Density
Purchase
0.0000
0.0005
0.0010
0.0015
0.0020
0.0025
8600
8800
9000
9200
9400
9600
9800


Age Group: 26-35, Sample Size: 1000, Mean:9256.53, SME:4.94
μ : 9256.53
Lower Limit: 8950.47
Upper Limit: 9562.60
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase


Age Group: 36-45, Sample Size: 1000, Mean:9333.26, SME:4.97
μ : 9333.26
Lower Limit: 9025.23
Upper Limit: 9641.29
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
10000
Purchase


Age Group: 46-50, Sample Size: 1000, Mean:9203.18, SME:4.89
μ : 9203.18
Lower Limit: 8900.15
Upper Limit: 9506.20
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase


Age Group: 51-55, Sample Size: 1000, Mean:9540.1, SME:5.19
μ : 9540.10
Lower Limit: 9218.76
Upper Limit: 9861.45
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
9000
9200
9400
9600
9800
10000
10200
Purchase



[65]:
```
res
```

[65]:
```
   Age_Group Sample Size  Lower Limit  Upper Limit  Sample Mean  \
0       0-17        1000      8687.50      9189.31      8938.41
1      18-25        1000      8906.94      9431.01      9168.97
2      26-35        1000      8981.94      9517.04      9249.49
3      36-45        1000      9066.64      9594.48      9330.56
4      46-50        1000      8947.98      9461.23      9204.61
5      51-55        1000      9288.27      9802.69      9545.48
6        55+        1000      9078.56      9573.36      9325.96
7       0-17        1000      8622.18      9232.31      8927.24
8      18-25        1000      8840.10      9471.33      9155.71
9      26-35        1000      8950.47      9562.60      9256.53
10     36-45        1000      9025.23      9641.29      9333.26
11     46-50        1000      8900.15      9506.20      9203.18
12     51-55        1000      9218.76      9861.45      9540.10
13       55+        1000      9043.86      9630.79      9337.32

   Confidence Interval      Interval Range   Range
0                   90   [8687.5, 9189.31]  501.81
1                   90  [8906.94, 9431.01]  524.07
2                   90  [8981.94, 9517.04]  535.10
3                   90  [9066.64, 9594.48]  527.84
4                   90  [8947.98, 9461.23]  513.25
5                   90  [9288.27, 9802.69]  514.42
6                   90  [9078.56, 9573.36]  494.80
7                   95  [8622.18, 9232.31]  610.13
8                   95   [8840.1, 9471.33]  631.23
9                   95   [8950.47, 9562.6]  612.13
10                  95  [9025.23, 9641.29]  616.06
```

```
11                    95    [8900.15, 9506.2]   606.05
12                    95   [9218.76, 9861.45]   642.69
13                    95   [9043.86, 9630.79]   586.93
```

We can observe with 90% confidence that

- Age group 0-17 has the least purchase value range of [8719.59, 8750.12].
- Age group 51-55 has highest purchase value range of [9288.27, 9802.69].

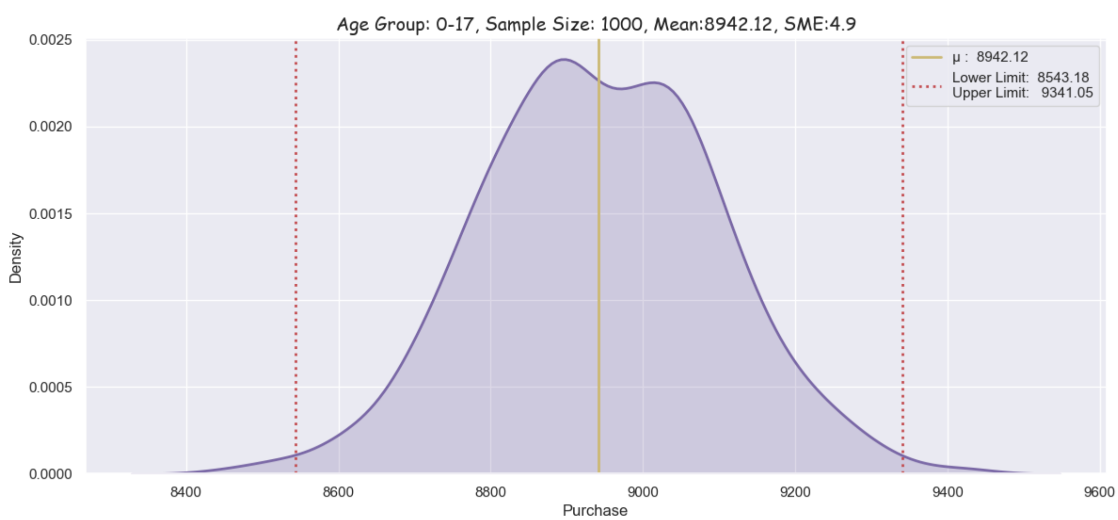
We can observe with 95% confidence that

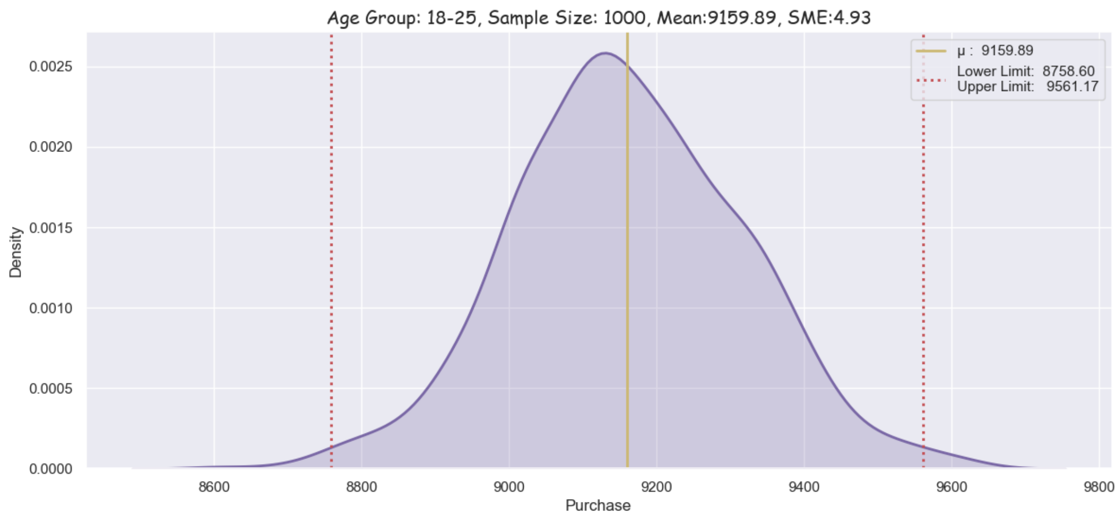
- Age group 0-17 has the least purchase value range of [9288.27, 9802.69].
- Age group 51-55 has highest purchase value range of [9218.76, 9861.45].

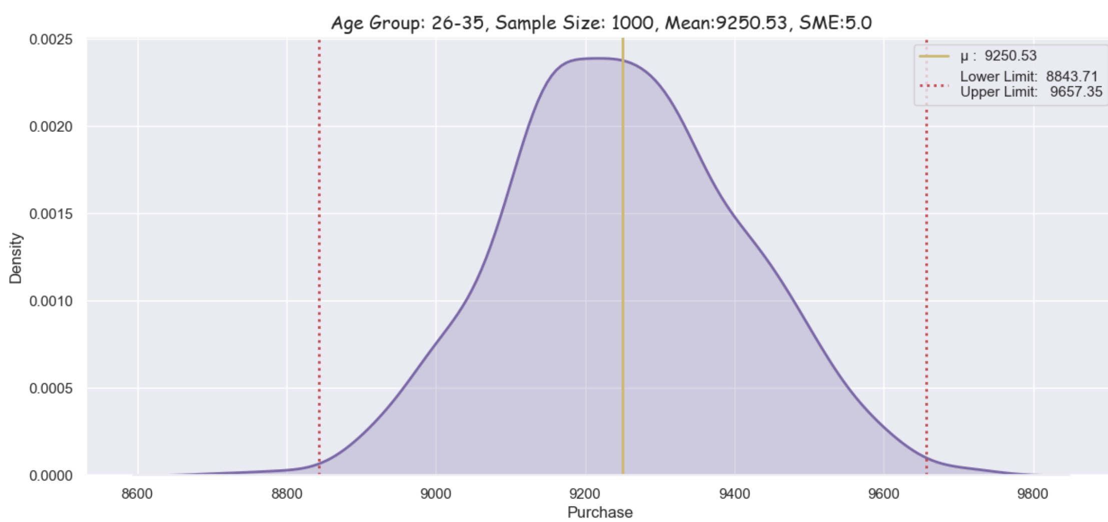
All the age groups still have overlap which makes it difficult to interpret the ranges.

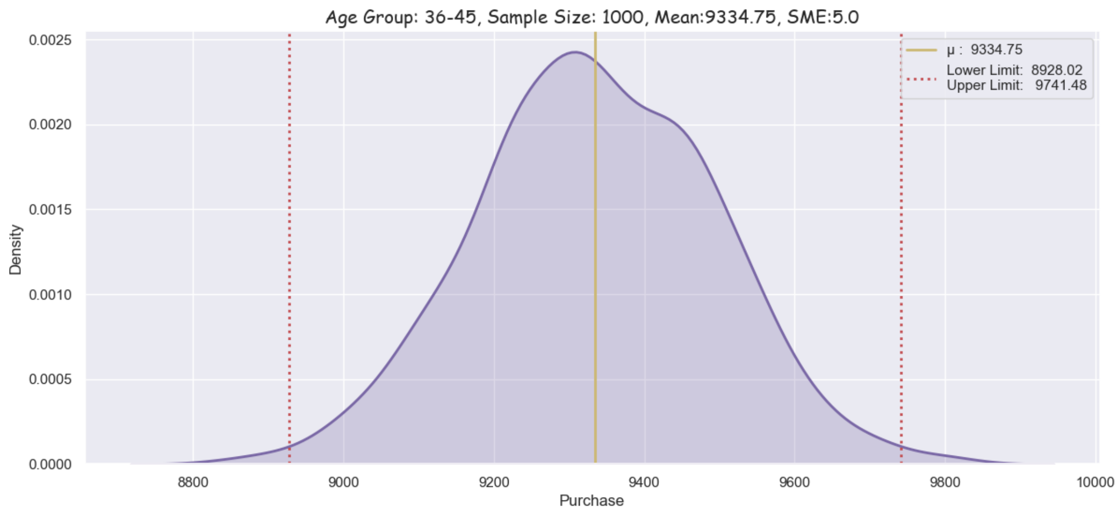
So now, Lets visualise the graphs of 1000 mean values of purchase samples for sample size of 1000 for all the age groups with 99% confidence interval.

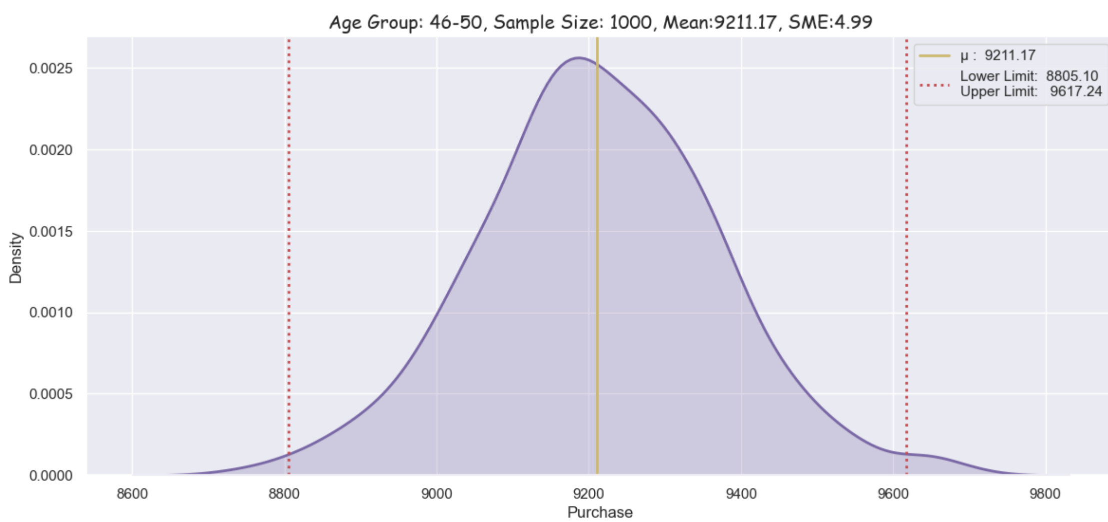
[66]:
```
ci = 99
itr_size = 1000
sample_size = 1000
flag = 0

for i in age_group:
     m_avg, ll, ul, mean =␣
 ↪bootstrap_age(df[df['Age']==i]['Purchase'],sample_size,itr_size,ci)

     res = res.append({'Age_Group':i,'Sample Size':sample_size,'Lower Limit':
 ↪ll,'Upper Limit':ul,'Sample Mean':mean,'Confidence Interval':ci,'Interval␣
 ↪Range':[ll,ul],'Range': ul-ll}, ignore_index = True)
```

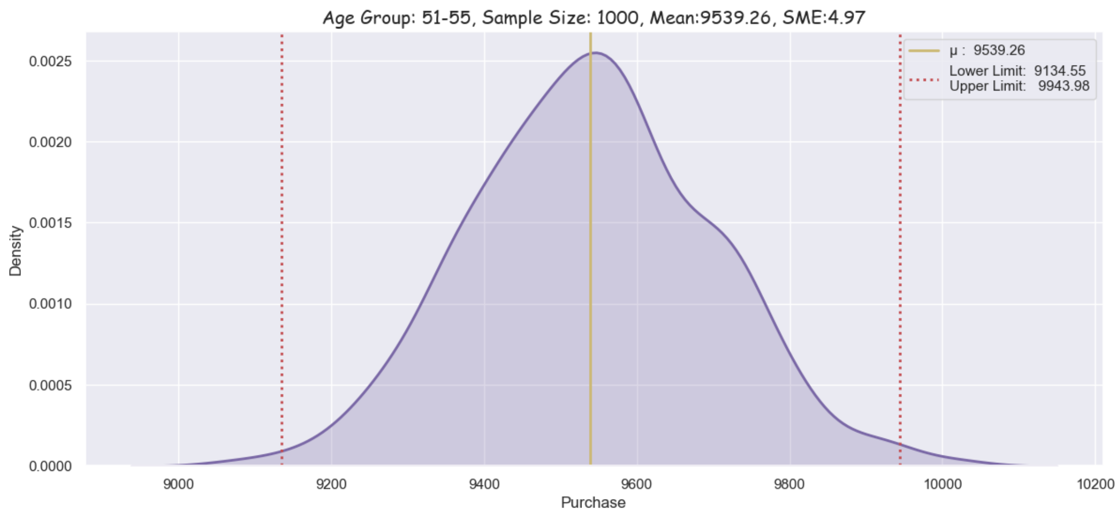


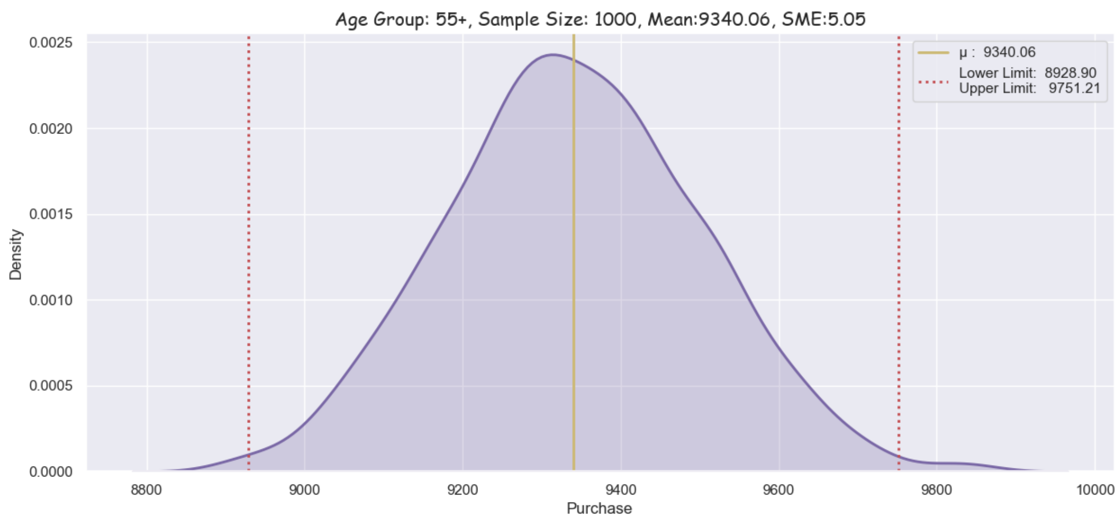

Age Group: 18-25, Sample Size: 1000, Mean:9159.89, SME:4.93
μ : 9159.89
Lower Limit: 8758.60
Upper Limit: 9561.17
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase


Age Group: 26-35, Sample Size: 1000, Mean:9250.53, SME:5.0
μ : 9250.53
Lower Limit: 8843.71
Upper Limit: 9657.35
Density
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800
Purchase


Age Group: 36-45, Sample Size: 1000, Mean:9334.75, SME:5.0
μ : 9334.75
Lower Limit: 8928.02
Upper Limit: 9741.48
Density
Purchase
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8800
9000
9200
9400
9600
9800
10000


Age Group: 46-50, Sample Size: 1000, Mean:9211.17, SME:4.99
μ : 9211.17
Lower Limit: 8805.10
Upper Limit: 9617.24
Density
Purchase
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
8600
8800
9000
9200
9400
9600
9800





[67]:
```
res
```

[67]:
```
  Age_Group Sample Size  Lower Limit  Upper Limit  Sample Mean  \
0      0-17         1000      8687.50      9189.31      8938.41
1     18-25         1000      8906.94      9431.01      9168.97
2     26-35         1000      8981.94      9517.04      9249.49
3     36-45         1000      9066.64      9594.48      9330.56
4     46-50         1000      8947.98      9461.23      9204.61
5     51-55         1000      9288.27      9802.69      9545.48
6       55+         1000      9078.56      9573.36      9325.96
7      0-17         1000      8622.18      9232.31      8927.24
8     18-25         1000      8840.10      9471.33      9155.71
```

9	26-35	1000	8950.47	9562.60	9256.53
10	36-45	1000	9025.23	9641.29	9333.26
11	46-50	1000	8900.15	9506.20	9203.18
12	51-55	1000	9218.76	9861.45	9540.10
13	55+	1000	9043.86	9630.79	9337.32
14	0-17	1000	8543.18	9341.05	8942.12
15	18-25	1000	8758.60	9561.17	9159.89
16	26-35	1000	8843.71	9657.35	9250.53
17	36-45	1000	8928.02	9741.48	9334.75
18	46-50	1000	8805.10	9617.24	9211.17
19	51-55	1000	9134.55	9943.98	9539.26
20	55+	1000	8928.90	9751.21	9340.06

	Confidence Interval	Interval Range	Range
0	90	[8687.5, 9189.31]	501.81
1	90	[8906.94, 9431.01]	524.07
2	90	[8981.94, 9517.04]	535.10
3	90	[9066.64, 9594.48]	527.84
4	90	[8947.98, 9461.23]	513.25
5	90	[9288.27, 9802.69]	514.42
6	90	[9078.56, 9573.36]	494.80
7	95	[8622.18, 9232.31]	610.13
8	95	[8840.1, 9471.33]	631.23
9	95	[8950.47, 9562.6]	612.13
10	95	[9025.23, 9641.29]	616.06
11	95	[8900.15, 9506.2]	606.05
12	95	[9218.76, 9861.45]	642.69
13	95	[9043.86, 9630.79]	586.93
14	99	[8543.18, 9341.05]	797.87
15	99	[8758.6, 9561.17]	802.57
16	99	[8843.71, 9657.35]	813.64
17	99	[8928.02, 9741.48]	813.46
18	99	[8805.1, 9617.24]	812.14
19	99	[9134.55, 9943.98]	809.43
20	99	[8928.9, 9751.21]	822.31

We can observe with 99% confidence that

- Age group 0-17 has the least purchase value range of [8543.18, 9341.05].
- Age group 51-55 has highest purchase value range of [9134.55, 9943.98].
- We can say that age group does not have much effect on the spending of customers as their interval range is overalpping with 90%, 95% and 99% confidence intervals.

1.2.4 Inferences

- 80% of the users are between the age 18-50 (40%: 26-35, 18%: 18-25, 20%: 36-45)
- 75% of the users are Male and 25% are Female. Males clearly purchase more than females.

- 59% Single, 41% Married
- 35% Staying in the city from 1 year, 18% from 2 years, 17% from 3 years
- The majority of our customers come from city category B but customers come from City category C spent more as mean is 9719.
- The majority of users come from City Category C, but more people from City Category B tend to purchase, which suggests the same users visit the mall multiple times in City Category B.
- Majority of Customers purchase within the 5,000 - 20,000 range.
- Most mall customers are between the ages of 26 and 35.60% of purchases are made by people between the ages of 26 and 45
- City Category B accounts for 42%, City Category C 31%, and City Category A represents 27% of all customer purchases.Purchases are high in city category C
- Most mall customers are between the ages of 26 and 35. City category C has more customers between the ages of 18 and 45.
- In City Category C, there are slightly more female customers.
- Product 5 and 8 is common among females.

1.2.5 Recommendations

- Men spent more money than women, So company should focus on retaining the male customers and getting more male customers.
- Product_Category - 1, 5, 8, & 11 have highest purchasing frequency. it means these are the products in these categories are liked more by customers. Company can focus on * selling more of these products or selling more of the products which are purchased less.
- Unmarried customers spend more money than married customers, So company should focus on acquisition of Unmarried customers.
- Customers in the age 18-45 spend more money than the others, So company should focus on acquisition of customers who are in the age 18-45.
- Male customers living in City_Category C spend more money than other male customers living in B or C, Selling more products in the City_Category C will help the company increase the revenue.
- In light of the fact that females spend less than males on average, management needs to focus on their specific needs differently. Adding some additional offers for women can increase their spending on Black Friday.
- Management should come-up with some games in the mall to attract more younger generation will can help them to increase the sale.
- The management should have some offers on kids (0-17 years) in order to increase sales.
- In order to attract more young shoppers, they can offer some games for the younger generation.