walmart

March 25, 2024

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm, binom, poisson, expon,geom
```

[3]:
```
path = "./walmart_data.csv"
df = pd.read_csv(path)
```

[4]:
```
df.head()
```

[4]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

[5]:
```
df.shape
```

[5]:
```
(550068, 10)
```

[6]:
```
df.dtypes
```

[6]:
```
User_ID                      int64
Product_ID                  object
Gender                      object
Age                         object
Occupation                   int64
City_Category               object
```

```
Stay_In_Current_City_Years    object
Marital_Status                 int64
Product_Category               int64
Purchase                       int64
dtype: object
```

Is there any missing value in the dataset?

[8]:
```
df.isnull().sum()
```

[8]:
```
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
```

Is there any duplicate value in the dataset ?

[11]:
```
np.any(df.duplicated())
```

[11]:
```
False
```

[12]:
```
df.describe()
```

[12]:
```
            User_ID     Occupation  Marital_Status  Product_Category  \
count  5.500680e+05  550068.000000   550068.000000     550068.000000
mean   1.003029e+06       8.076707        0.409653          5.404270
std    1.727592e+03       6.522660        0.491770          3.936211
min    1.000001e+06       0.000000        0.000000          1.000000
25%    1.001516e+06       2.000000        0.000000          1.000000
50%    1.003077e+06       7.000000        0.000000          5.000000
75%    1.004478e+06      14.000000        1.000000          8.000000
max    1.006040e+06      20.000000        1.000000         20.000000

            Purchase
count  550068.000000
mean     9263.968713
std      5023.065394
min        12.000000
25%      5823.000000
50%      8047.000000
75%     12054.000000
```

```
max     23961.000000
```

Observations: - There are no missing values in the dataset. - Purchase amount might have outliers.

[20]:
```
df.describe(include='object')
```

[20]:

	Product_ID	Gender	Age	City_Category	Stay_In_Current_City_Years
count	550068	550068	550068	550068	550068
unique	3631	2	7	3	5
top	P00265242	M	26-35	B	1
freq	1880	414259	219587	231173	193821

1 value_counts and unique attributes

[14]:
```
df["User_ID"].nunique()
```

[14]:
```
5891
```

We have 5891 unique customers in the dataset.

[19]:
```
df["Product_ID"].nunique()
```

[19]:
```
3631
```

we have 3631 unique products in the dataset.

[25]:
```
# Total number of transactions made by each gender
np.round(df['Gender'].value_counts(normalize = True) * 100, 0)
```

[25]:
```
Gender
M    75.0
F    25.0
Name: proportion, dtype: float64
```

It is clear from the above that out of every four transactions, three are made by males.

[26]:
```
np.round(df['Occupation'].value_counts(normalize = True) * 100, 2).cumsum()
```

[26]:
```
Occupation
4     13.15
0     25.81
7     36.56
1     45.18
17    52.46
20    58.56
12    64.23
14    69.19
```

```
2     74.02
16    78.63
6     82.33
3     85.54
10    87.89
5     90.10
15    92.31
11    94.42
19    95.96
13    97.36
18    98.56
9     99.70
8     99.98
Name: proportion, dtype: float64
```

It can be inferred from the above that 82.33 % of the total transactions are made by the customers belonging to 11 occupations. These are 4, 0, 7, 1, 17, 20, 12, 14, 2, 16, 6 (Ordered in descending order of the total transactions' share.)

[27]:
```
np.round(df['Stay_In_Current_City_Years'].value_counts(normalize = True) * 100,␣
↪2)
```

[27]:
```
Stay_In_Current_City_Years
1     35.24
2     18.51
3     17.32
4+    15.40
0     13.53
Name: proportion, dtype: float64
```

From the above result, it is clear that majority of the transactions (53.75 % of total transactions) are made by the customers having 1 or 2 years of stay in the current city.

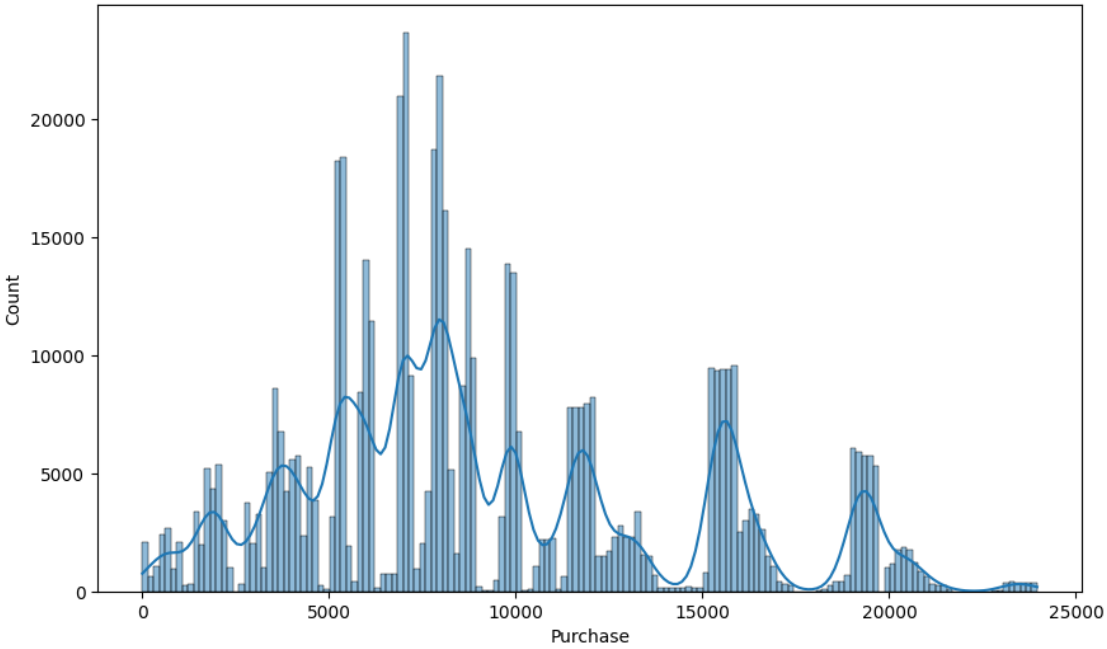
[32]:
```
np.round(df['Product_Category'].value_counts(normalize = True).head(10) * 100,␣
↪2).cumsum()
```

[32]:
```
Product_Category
5     27.44
1     52.96
8     73.67
11    78.09
2     82.43
6     86.15
3     89.82
4     91.96
16    93.75
15    94.89
Name: proportion, dtype: float64
```

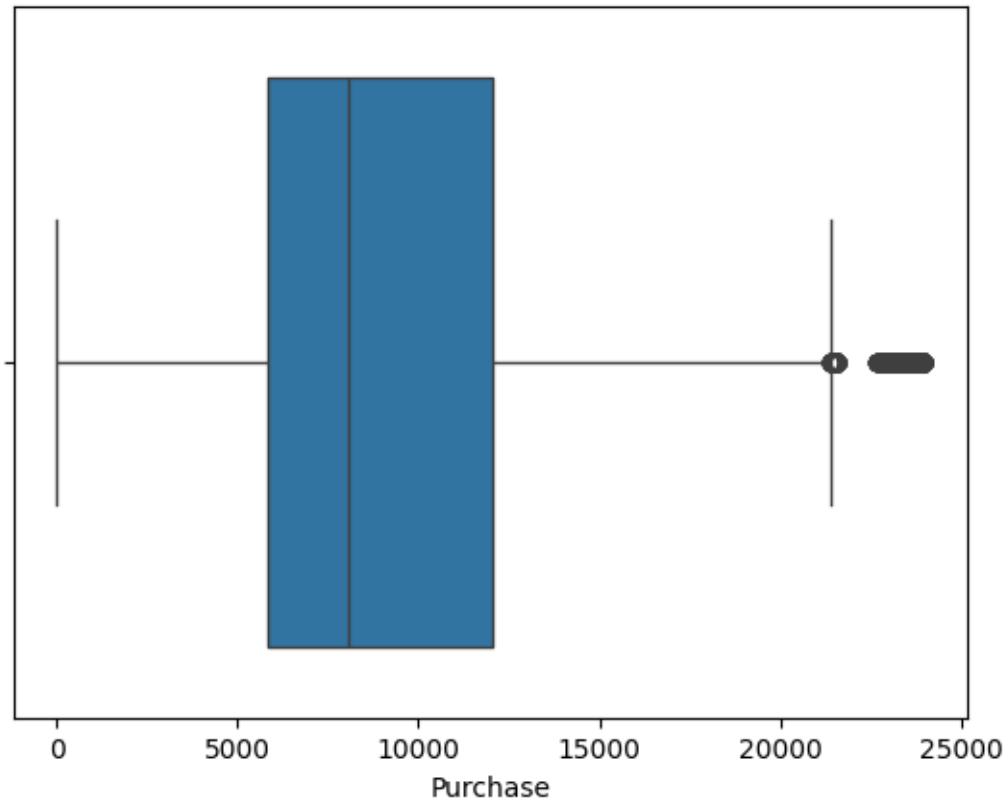
It can be inferred from the above result that 82.43% of the total transactions are made for only 5 Product Categories. These are, 5, 1, 8, 11 and 2.

2 Univariate Analysis

[36]:
```
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='Purchase', kde=True)
plt.show()
```



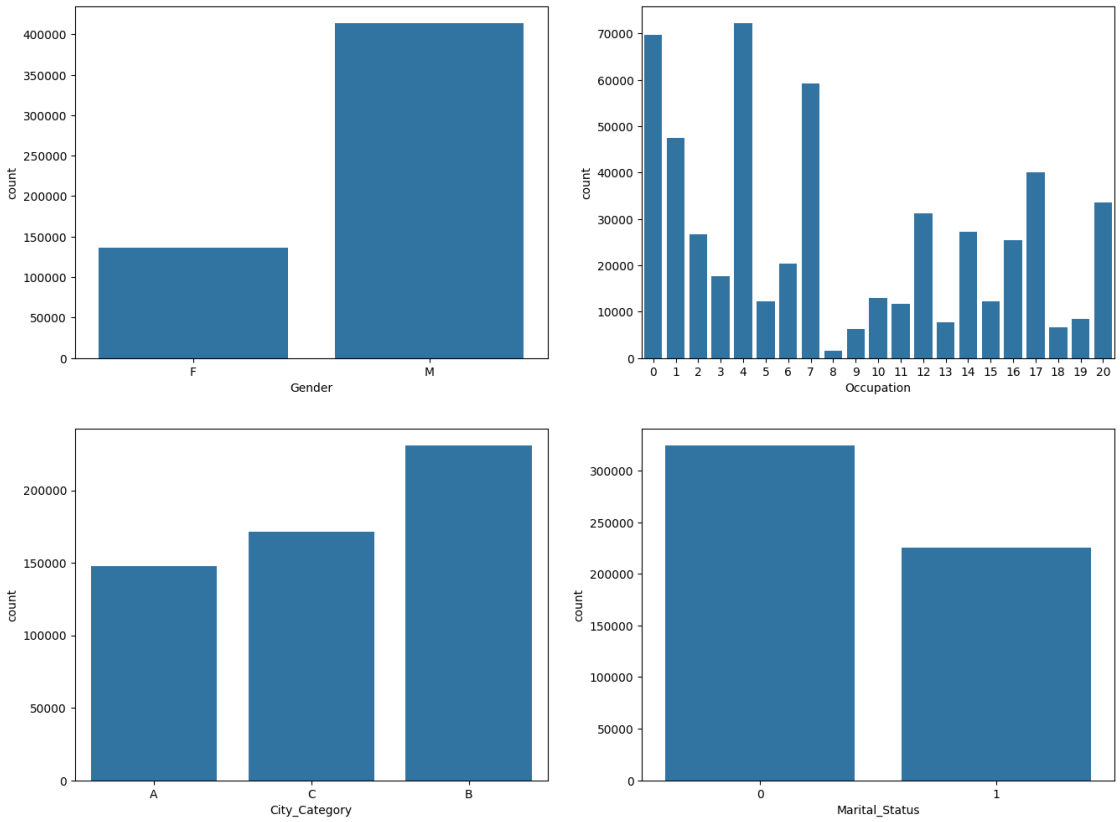
[39]:
```
sns.boxplot(data=df, x='Purchase', orient='h')
plt.show()
```



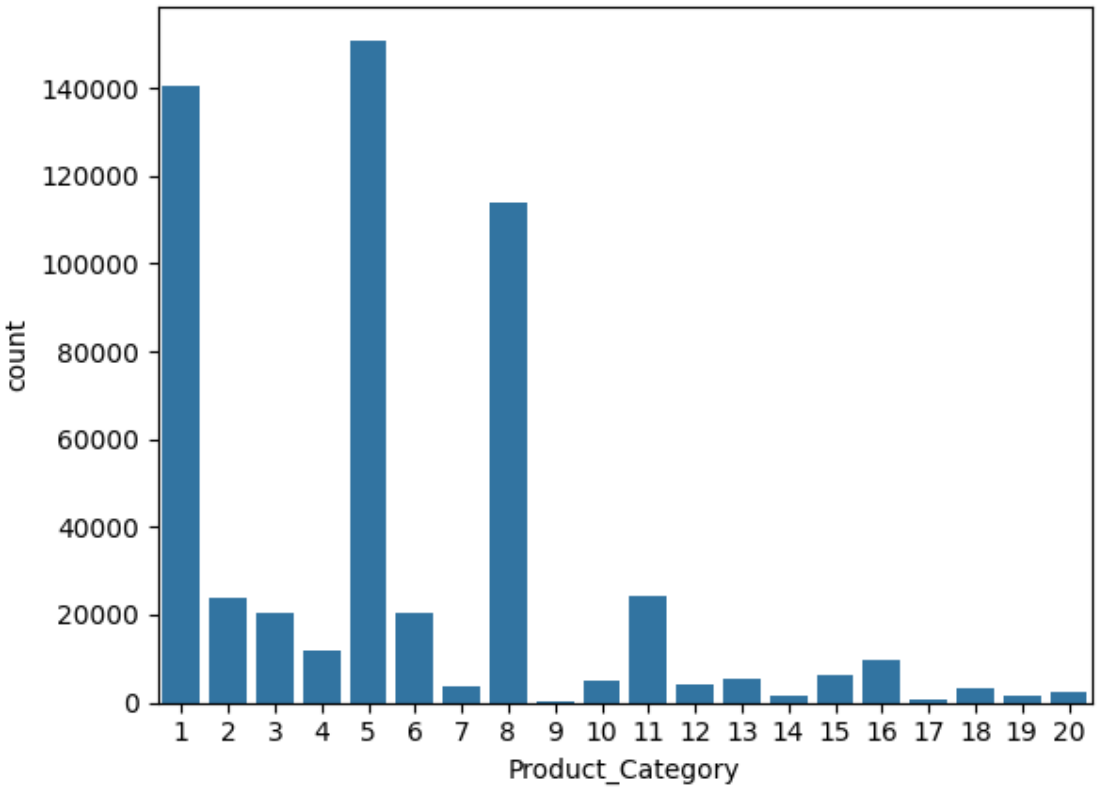
Observation:

Purchase is having outliers

[42]:
```
categorical_cols = ['Gender',␣
 ↪'Occupation','City_Category','Marital_Status','Product_Category']

fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(16, 12))
sns.countplot(data=df, x='Gender', ax=axs[0,0])
sns.countplot(data=df, x='Occupation', ax=axs[0,1])
sns.countplot(data=df, x='City_Category', ax=axs[1,0])
sns.countplot(data=df, x='Marital_Status', ax=axs[1,1])
plt.show()
```

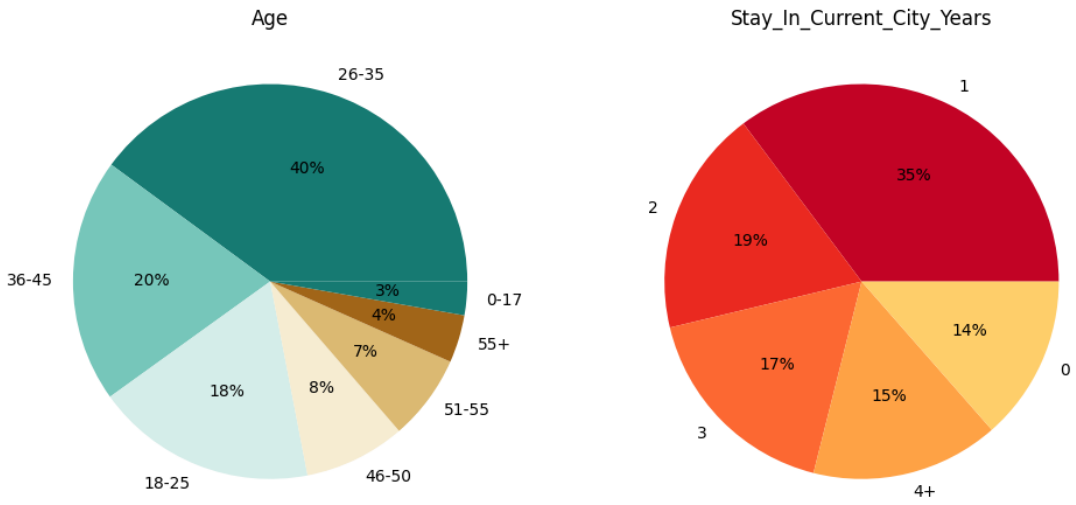


[43]:
```
sns.countplot(data=df, x='Product_Category')
plt.show()
```



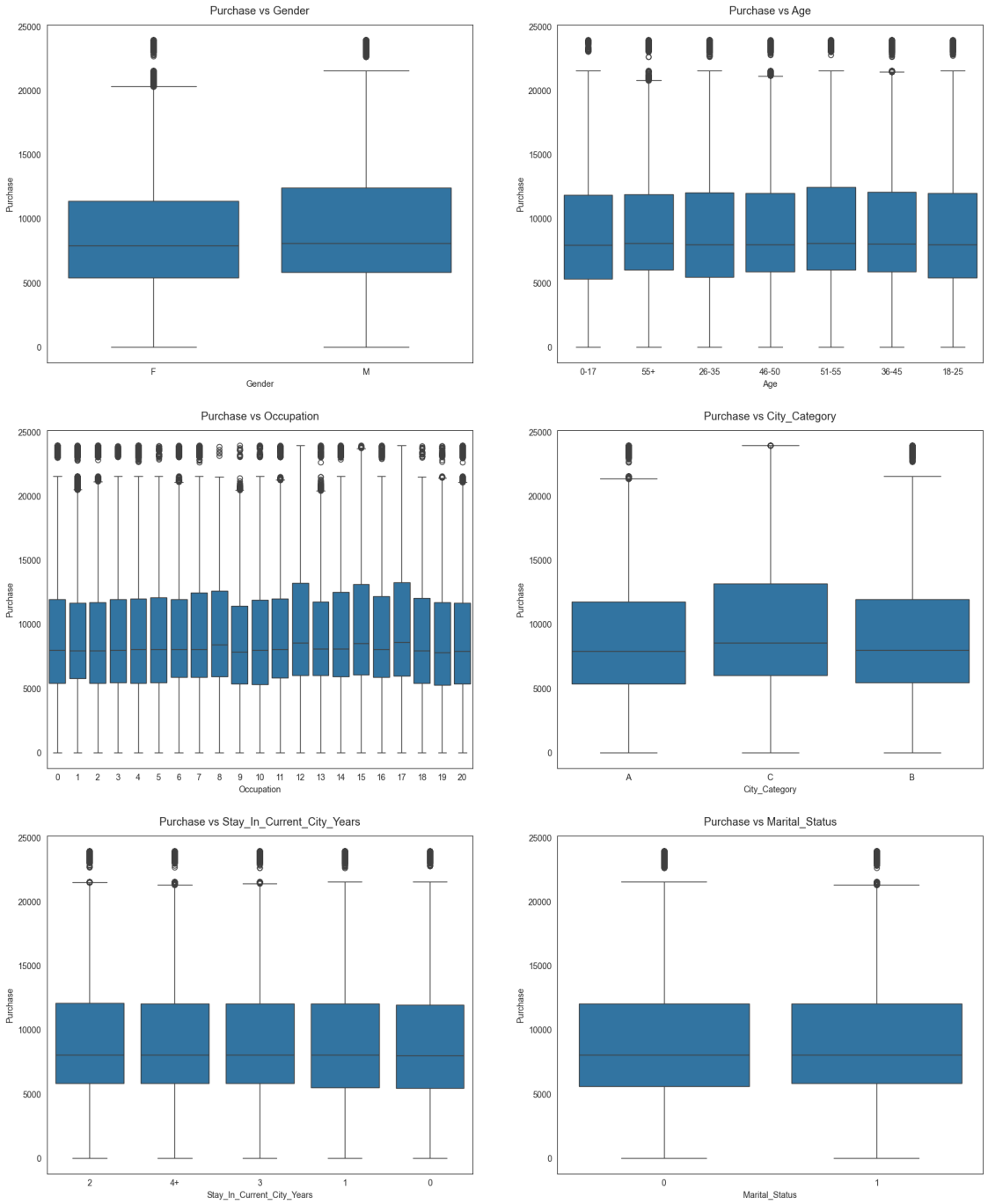
Observations: - Most of the users are Male - There are 20 different types of Occupation and Product_Category - More users belong to B City_Category - More users are Single as compare to Married - Product_Category - 1, 5, 8, & 11 have highest purchasing frequency.

[44]:
```
fig, axs = plt.subplots(nrows=1, ncols=2, figsize=(12, 8))

data = df['Age'].value_counts(normalize=True)*100
palette_color = sns.color_palette('BrBG_r')
axs[0].pie(x=data.values, labels=data.index, autopct='%.0f%%',␣
 ↪colors=palette_color)
axs[0].set_title("Age")

data = df['Stay_In_Current_City_Years'].value_counts(normalize=True)*100
palette_color = sns.color_palette('YlOrRd_r')
axs[1].pie(x=data.values, labels=data.index, autopct='%.0f%%',␣
 ↪colors=palette_color)
axs[1].set_title("Stay_In_Current_City_Years")


plt.show()
```

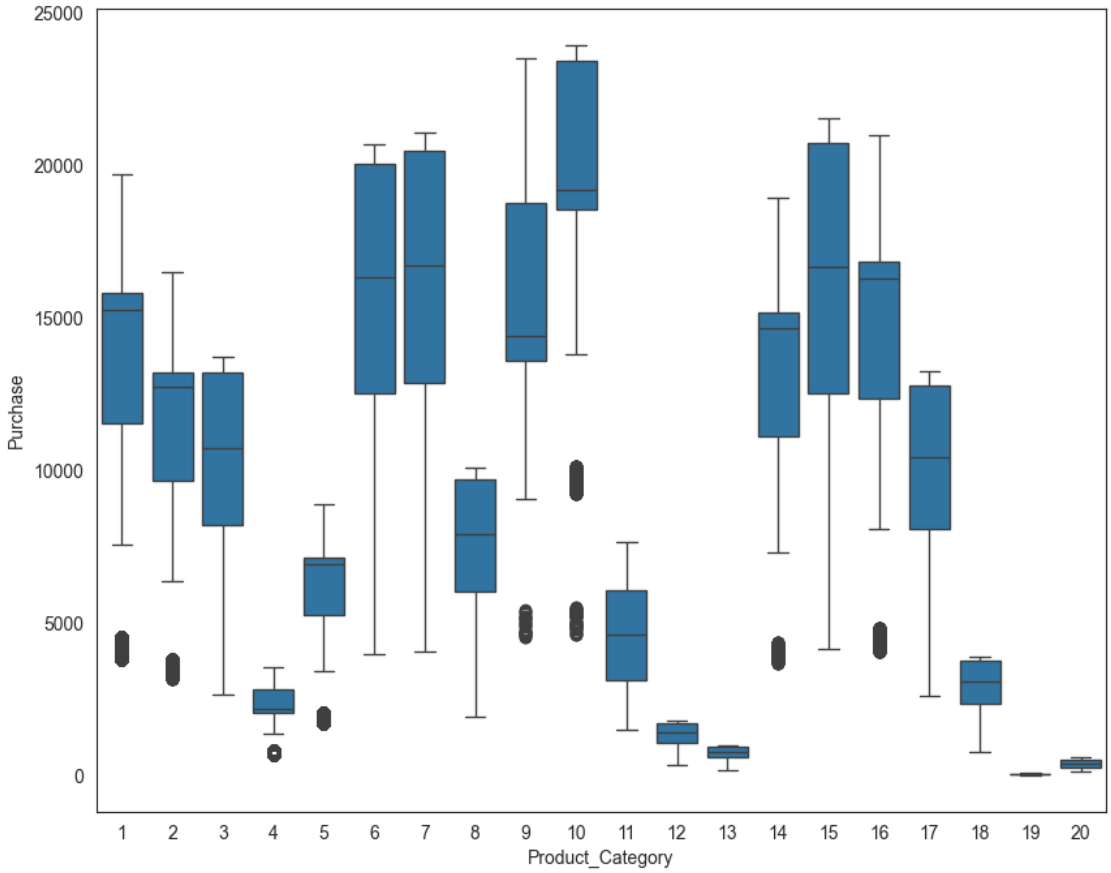


3 Bi-variate Analysis

[50]:
```
attrs = ['Gender', 'Age', 'Occupation', 'City_Category',␣
 ↪'Stay_In_Current_City_Years', 'Marital_Status', 'Product_Category']
sns.set_style("white")

fig, axs = plt.subplots(nrows=3, ncols=2, figsize=(20, 16))
fig.subplots_adjust(top=1.3)
count = 0
for row in range(3):
    for col in range(2):
        sns.boxplot(data=df, y='Purchase', x=attrs[count], ax=axs[row, col])
        axs[row,col].set_title(f"Purchase vs {attrs[count]}", pad=12,␣
 ↪fontsize=13)
        count += 1
plt.show()
```

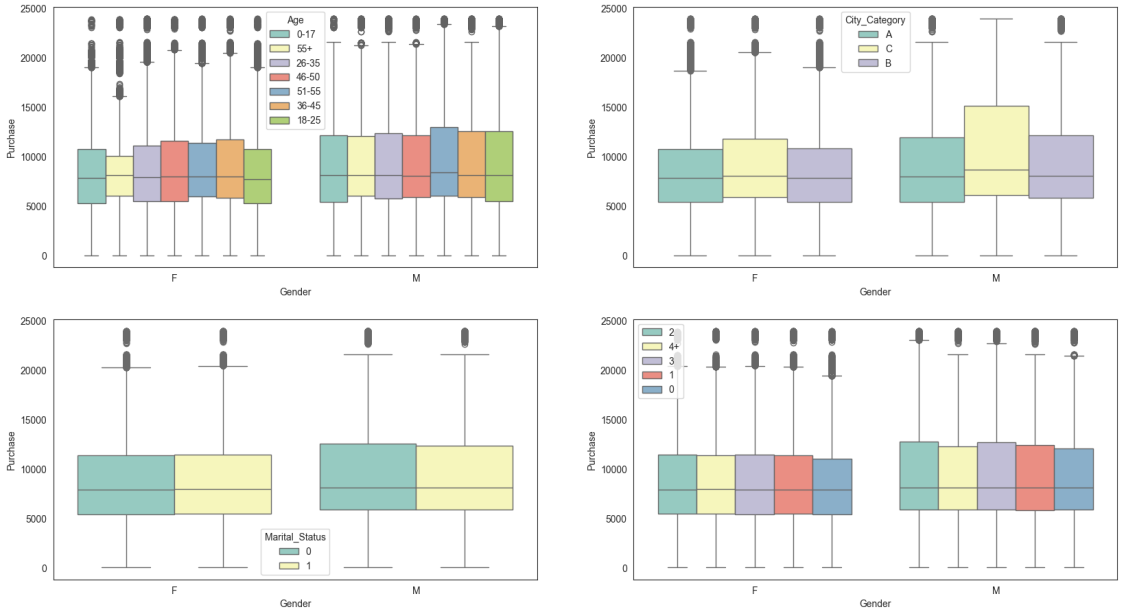


[51]:
```
plt.figure(figsize=(10, 8))
sns.boxplot(data=df, y='Purchase', x=attrs[-1])
plt.show()
```



4 Multivariate Analysis

[49]:
```
fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(20, 6))
fig.subplots_adjust(top=1.5)
sns.boxplot(data=df, y='Purchase', x='Gender', hue='Age', palette='Set3',␣
 ↪ax=axs[0,0])
sns.boxplot(data=df, y='Purchase', x='Gender', hue='City_Category',␣
 ↪palette='Set3', ax=axs[0,1])

sns.boxplot(data=df, y='Purchase', x='Gender', hue='Marital_Status',␣
 ↪palette='Set3', ax=axs[1,0])
sns.boxplot(data=df, y='Purchase', x='Gender',␣
 ↪hue='Stay_In_Current_City_Years', palette='Set3', ax=axs[1,1])
axs[1,1].legend(loc='upper left')

plt.show()
```



4.0.1 How many unique customers are there for each gender

[53]:
```
df_gender_dist = pd.DataFrame(df.groupby(by = ['Gender'])['User_ID'].nunique()).
↪reset_index().rename(columns = {'User_ID' : 'unique_customers'})
df_gender_dist['percent_share'] = np.round(df_gender_dist['unique_customers'] /␣
↪df_gender_dist['unique_customers'].sum() * 100, 2)
df_gender_dist
```

[53]:

	Gender	unique_customers	percent_share
0	F	1666	28.28
1	M	4225	71.72

[54]:
```
df.groupby(by = ['Gender'])['User_ID'].count()
print('Average number of transactions made by each Male on Black Friday is',␣
↪round(414259 / 4225))
print('Average number of transactions made by each Female on Black Friday is',␣
↪round(135809 / 1666))
```

```
Average number of transactions made by each Male on Black Friday is 98
Average number of transactions made by each Female on Black Friday is 82
```

4.0.2 Average amount spend per customer for Male and Female

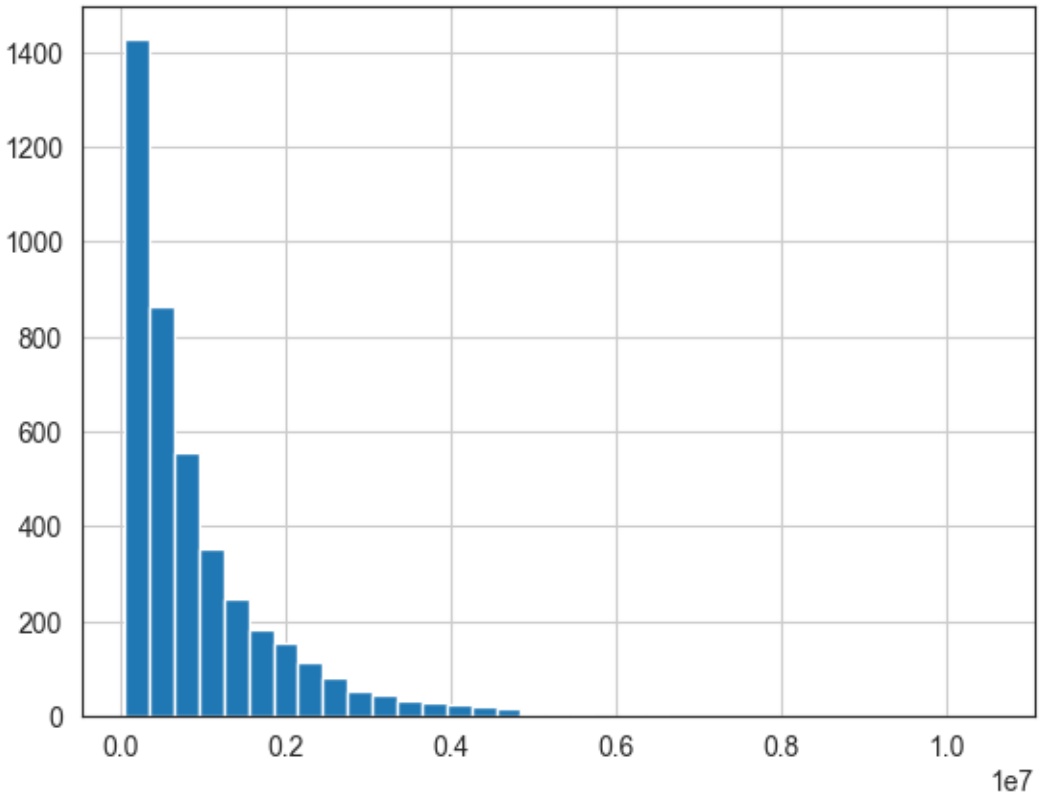
[55]:
```
amt_df = df.groupby(['User_ID', 'Gender'])[['Purchase']].sum()
amt_df = amt_df.reset_index()
amt_df
```

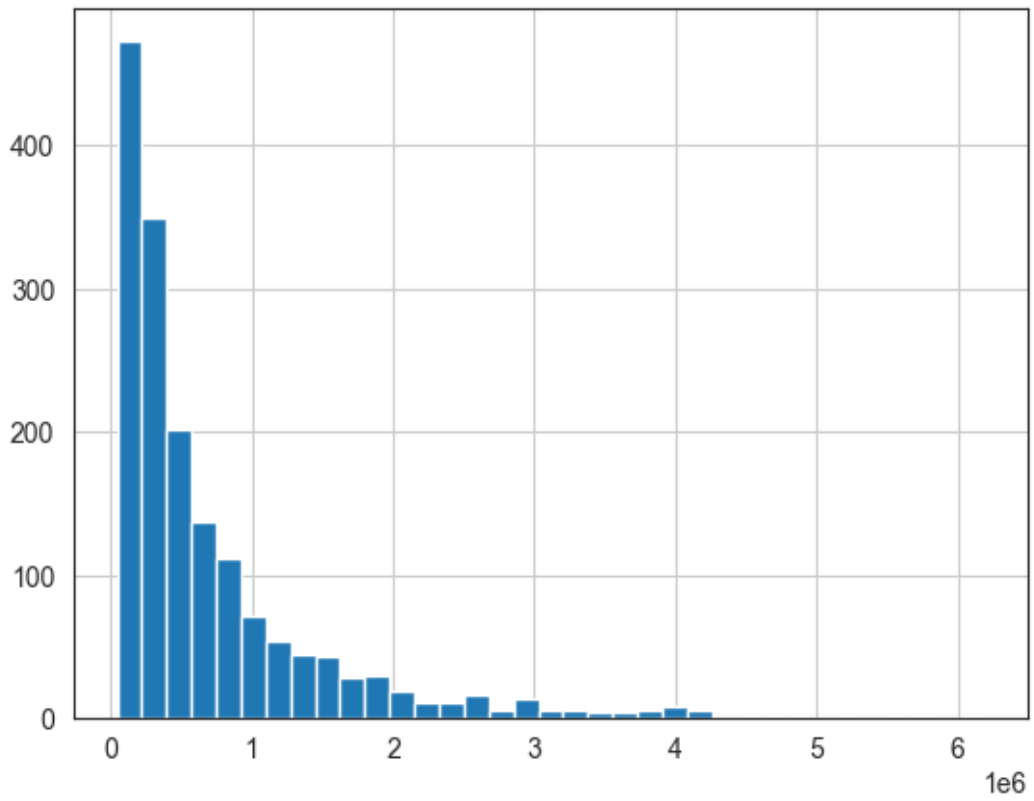
[55]:

	User_ID	Gender	Purchase
0	1000001	F	334093
1	1000002	M	810472
2	1000003	M	341635
3	1000004	M	206468
4	1000005	M	821001
…	…	…	…
5886	1006036	F	4116058
5887	1006037	F	1119538
5888	1006038	F	90034
5889	1006039	F	590319
5890	1006040	M	1653299

[5891 rows x 3 columns]

[58]:
```
# histogram of average amount spend for each customer - Male & Female
amt_df[amt_df['Gender']=='M']['Purchase'].hist(bins=35)
plt.show()

amt_df[amt_df['Gender']=='F']['Purchase'].hist(bins=35)
plt.show()
```





[59]:
```
male_avg = amt_df[amt_df['Gender']=='M']['Purchase'].mean()
female_avg = amt_df[amt_df['Gender']=='F']['Purchase'].mean()

print("Average amount spend by Male customers: {:.2f}".format(male_avg))
print("Average amount spend by Female customers: {:.2f}".format(female_avg))
```

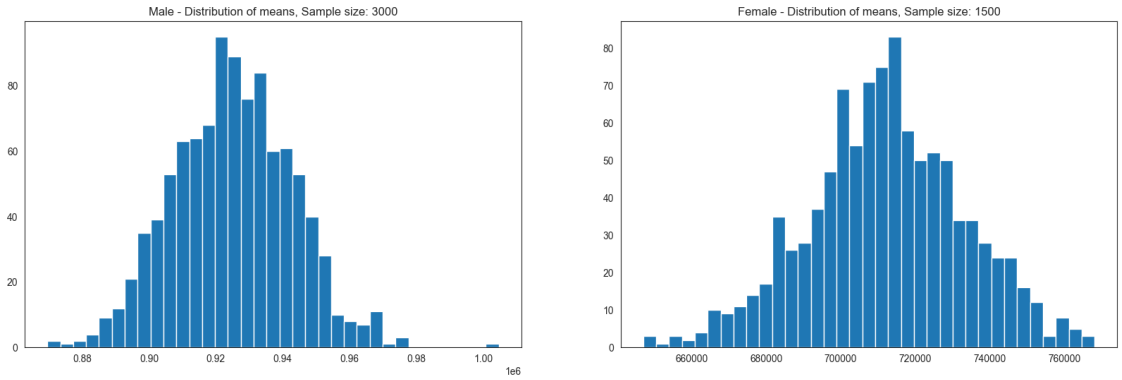
```
Average amount spend by Male customers: 925344.40
Average amount spend by Female customers: 712024.39
```

4.0.3 Observations

[60]:
```
male_df = amt_df[amt_df['Gender']=='M']
female_df = amt_df[amt_df['Gender']=='F']

genders = ["M", "F"]

male_sample_size = 3000
female_sample_size = 1500
num_repitions = 1000
male_means = []
female_means = []
```

```
for _ in range(num_repitions):
    male_mean = male_df.sample(male_sample_size, replace=True)['Purchase'].
 ↪mean()
    female_mean = female_df.sample(female_sample_size,␣
 ↪replace=True)['Purchase'].mean()

    male_means.append(male_mean)
    female_means.append(female_mean)
```

[61]:
```
fig, axis = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))

axis[0].hist(male_means, bins=35)
axis[1].hist(female_means, bins=35)
axis[0].set_title("Male - Distribution of means, Sample size: 3000")
axis[1].set_title("Female - Distribution of means, Sample size: 1500")

plt.show()
```



[62]:
```
print("Population mean - Mean of sample means of amount spend for Male: {:.2f}".
 ↪format(np.mean(male_means)))
print("Population mean - Mean of sample means of amount spend for Female: {:.
 ↪2f}".format(np.mean(female_means)))

print("\nMale - Sample mean: {:.2f} Sample std: {:.2f}".
 ↪format(male_df['Purchase'].mean(), male_df['Purchase'].std()))
print("Female - Sample mean: {:.2f} Sample std: {:.2f}".
 ↪format(female_df['Purchase'].mean(), female_df['Purchase'].std()))
```

```
Population mean - Mean of sample means of amount spend for Male: 925115.50
Population mean - Mean of sample means of amount spend for Female: 712021.10

Male - Sample mean: 925344.40 Sample std: 985830.10
```

```
Female - Sample mean: 712024.39 Sample std: 807370.73
```

Observation Now using the Central Limit Theorem for the population we can say that:

1. Average amount spend by male customers is 9,26,341.86
2. Average amount spend by female customers is 7,11,704.09

[63]:
```
male_margin_of_error_clt = 1.96*male_df['Purchase'].std()/np.sqrt(len(male_df))
male_sample_mean = male_df['Purchase'].mean()
male_lower_lim = male_sample_mean - male_margin_of_error_clt
male_upper_lim = male_sample_mean + male_margin_of_error_clt

female_margin_of_error_clt = 1.96*female_df['Purchase'].std()/np.
 ↪sqrt(len(female_df))
female_sample_mean = female_df['Purchase'].mean()
female_lower_lim = female_sample_mean - female_margin_of_error_clt
female_upper_lim = female_sample_mean + female_margin_of_error_clt

print("Male confidence interval of means: ({:.2f}, {:.2f})".
 ↪format(male_lower_lim, male_upper_lim))
print("Female confidence interval of means: ({:.2f}, {:.2f})".
 ↪format(female_lower_lim, female_upper_lim))
```

```
Male confidence interval of means: (895617.83, 955070.97)
Female confidence interval of means: (673254.77, 750794.02)
```

Doing the same for married vs unmarried

[64]:
```
amt_df = df.groupby(['User_ID', 'Marital_Status'])[['Purchase']].sum()
amt_df = amt_df.reset_index()
amt_df
```

[64]:

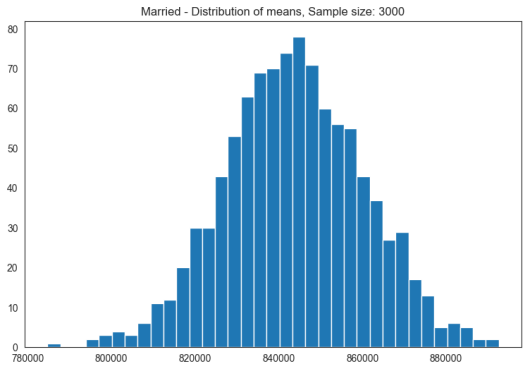
	User_ID	Marital_Status	Purchase
0	1000001	0	334093
1	1000002	0	810472
2	1000003	0	341635
3	1000004	1	206468
4	1000005	1	821001
…	…	…	…
5886	1006036	1	4116058
5887	1006037	0	1119538
5888	1006038	0	90034
5889	1006039	1	590319
5890	1006040	0	1653299

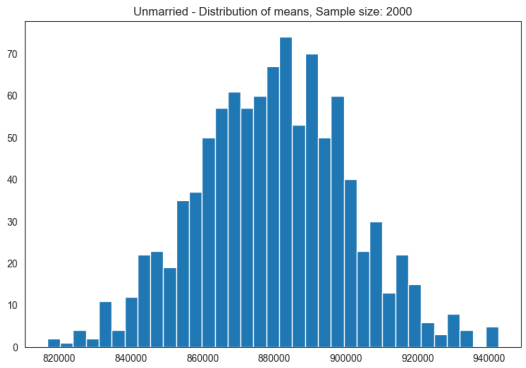
[5891 rows x 3 columns]

[65]:
```
amt_df['Marital_Status'].value_counts()
```

[65]:
```
Marital_Status
0    3417
1    2474
Name: count, dtype: int64
```

[66]:
```
marid_samp_size = 3000
unmarid_sample_size = 2000
num_repitions = 1000
marid_means = []
unmarid_means = []

for _ in range(num_repitions):
    marid_mean = amt_df[amt_df['Marital_Status']==1].sample(marid_samp_size,␣
 ↪replace=True)['Purchase'].mean()
    unmarid_mean = amt_df[amt_df['Marital_Status']==0].
 ↪sample(unmarid_sample_size, replace=True)['Purchase'].mean()

    marid_means.append(marid_mean)
    unmarid_means.append(unmarid_mean)


fig, axis = plt.subplots(nrows=1, ncols=2, figsize=(20, 6))

axis[0].hist(marid_means, bins=35)
axis[1].hist(unmarid_means, bins=35)
axis[0].set_title("Married - Distribution of means, Sample size: 3000")
axis[1].set_title("Unmarried - Distribution of means, Sample size: 2000")

plt.show()

print("Population mean - Mean of sample means of amount spend for Married: {:.
 ↪2f}".format(np.mean(marid_means)))
print("Population mean - Mean of sample means of amount spend for Unmarried: {:.
 ↪2f}".format(np.mean(unmarid_means)))

print("\nMarried - Sample mean: {:.2f} Sample std: {:.2f}".
 ↪format(amt_df[amt_df['Marital_Status']==1]['Purchase'].mean(),␣
 ↪amt_df[amt_df['Marital_Status']==1]['Purchase'].std()))
print("Unmarried - Sample mean: {:.2f} Sample std: {:.2f}".
 ↪format(amt_df[amt_df['Marital_Status']==0]['Purchase'].mean(),␣
 ↪amt_df[amt_df['Marital_Status']==0]['Purchase'].std()))
```






```
Population mean - Mean of sample means of amount spend for Married: 843744.55
Population mean - Mean of sample means of amount spend for Unmarried: 879834.09

Married - Sample mean: 843526.80 Sample std: 935352.12
Unmarried - Sample mean: 880575.78 Sample std: 949436.25
```

[67]:
```
for val in ["Married", "Unmarried"]:

    new_val = 1 if val == "Married" else 0

    new_df = amt_df[amt_df['Marital_Status']==new_val]

    margin_of_error_clt = 1.96*new_df['Purchase'].std()/np.sqrt(len(new_df))
    sample_mean = new_df['Purchase'].mean()
    lower_lim = sample_mean - margin_of_error_clt
    upper_lim = sample_mean + margin_of_error_clt

    print("{} confidence interval of means: ({:.2f}, {:.2f})".format(val,␣
 ↪lower_lim, upper_lim))
```

```
Married confidence interval of means: (806668.83, 880384.76)
Unmarried confidence interval of means: (848741.18, 912410.38)
```

Calculating the average amount spent by Age

[68]:
```
amt_df = df.groupby(['User_ID', 'Age'])[['Purchase']].sum()
amt_df = amt_df.reset_index()
amt_df
```

[68]:

	User_ID	Age	Purchase
0	1000001	0-17	334093
1	1000002	55+	810472
2	1000003	26-35	341635
3	1000004	46-50	206468
4	1000005	26-35	821001

```
…       …     …        …
5886  1006036  26-35   4116058
5887  1006037  46-50   1119538
5888  1006038    55+     90034
5889  1006039  46-50    590319
5890  1006040  26-35   1653299

[5891 rows x 3 columns]
```

[69]:
```
amt_df['Age'].value_counts()
```

[69]:
```
Age
26-35    2053
36-45    1167
18-25    1069
46-50     531
51-55     481
55+       372
0-17      218
Name: count, dtype: int64
```

[70]:
```
sample_size = 200
num_repitions = 1000

all_means = {}

age_intervals = ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']
for age_interval in age_intervals:
    all_means[age_interval] = []

for age_interval in age_intervals:
    for _ in range(num_repitions):
        mean = amt_df[amt_df['Age']==age_interval].sample(sample_size,␣
 ↪replace=True)['Purchase'].mean()
        all_means[age_interval].append(mean)
```

[71]:
```
for val in ['26-35', '36-45', '18-25', '46-50', '51-55', '55+', '0-17']:

    new_df = amt_df[amt_df['Age']==val]

    margin_of_error_clt = 1.96*new_df['Purchase'].std()/np.sqrt(len(new_df))
    sample_mean = new_df['Purchase'].mean()
    lower_lim = sample_mean - margin_of_error_clt
    upper_lim = sample_mean + margin_of_error_clt

    print("For age {} --> confidence interval of means: ({:.2f}, {:.2f})".
 ↪format(val, lower_lim, upper_lim))
```

```
For age 26-35 --> confidence interval of means: (945034.42, 1034284.21)
For age 36-45 --> confidence interval of means: (823347.80, 935983.62)
For age 18-25 --> confidence interval of means: (801632.78, 908093.46)
For age 46-50 --> confidence interval of means: (713505.63, 871591.93)
For age 51-55 --> confidence interval of means: (692392.43, 834009.42)
For age 55+ --> confidence interval of means: (476948.26, 602446.23)
For age 0-17 --> confidence interval of means: (527662.46, 710073.17)
```

5 Insights

- ~ 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
- 60% Single, 40% Married
- 35% Staying in the city from 1 year, 18% from 2 years, 17% from 3 years
- Total of 20 product categories are there
- There are 20 different types of occupations in the city
- Most of the users are Male
- There are 20 different types of Occupation and Product_Category
- More users belong to B City_Category
- More users are Single as compare to Married
- Product_Category - 1, 5, 8, & 11 have highest purchasing frequency.

Confidence Interval by Gender

Now using the Central Limit Theorem for the population:

- Average amount spend by male customers is 9,26,341.86
- Average amount spend by female customers is 7,11,704.09

Now we can infer about the population that, 95% of the times:

- Average amount spend by male customer will lie in between: (895617.83, 955070.97)
- Average amount spend by female customer will lie in between: (673254.77, 750794.02)

Confidence Interval by Marital_Status - Married confidence interval of means: (806668.83, 880384.76) - Unmarried confidence interval of means: (848741.18, 912410.38)

Confidence Interval by Age - For age 26-35 –> confidence interval of means: (945034.42, 1034284.21) - For age 36-45 –> confidence interval of means: (823347.80, 935983.62) - For age 18-25 –> confidence interval of means: (801632.78, 908093.46) - For age 46-50 –> confidence interval of means: (713505.63, 871591.93) - For age 51-55 –> confidence interval of means: (692392.43, 834009.42) - For age 55+ –> confidence interval of means: (476948.26, 602446.23) - For age 0-17 –> confidence interval of means: (527662.46, 710073.17)

6 Recommendations

1. Men spent more money than women, So company should focus on retaining the male customers and getting more male customers.
2. 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.

3. Unmarried customers spend more money than married customers, So company should focus on acquisition of Unmarried customers.
4. 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
5. 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.