In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import scipy.stats as stats
```

The problem statement for the Walmart project, based on the available data, can be defined as follows:

"The goal of the Walmart project is to analyze customer spending patterns during Black Friday sales and make informed decisions based on the data. By examining the dataset, including gender, marital status, and age, the project aims to understand if there are differences in spending between various customer segments. Additionally, using statistical methods such as bootstrapping, the project seeks to estimate population parameters, such as average spending per transaction, and determine confidence intervals for these estimates. The insights gained from the analysis will help Walmart make strategic improvements and cater to the needs of its customers more effectively."

Some potential analyzing metrics for this project could include:

1. Average Spending: Calculating the average purchase amount per transaction for different customer segments such as gender, marital status, and age groups. This metric provides insights into the general spending behavior and patterns of customers.
2. Comparison of Spending: Comparing the average spending between different customer segments, such as male and female customers, married and unmarried customers, or different age groups. This analysis helps identify any significant differences in spending behavior among these segments.
3. Confidence Intervals: Estimating confidence intervals for average spending to determine the range within which the true population average spending is likely to lie. This metric helps in understanding the uncertainty associated with the average spending estimates and enables more robust decision-making.
4. Distribution Analysis: Analyzing the distribution of spending amounts using methods like histograms, density plots, or box plots. This analysis provides insights into the spread, skewness, or any outliers in the spending data.
5. Hypothesis Testing: Conducting statistical hypothesis tests, such as t-tests or ANOVA, to assess whether there are significant differences in spending between different customer segments. This analysis helps identify if the observed differences in spending are statistically significant or due to random chance.
6. Visualization: Creating visual representations, such as bar charts, pie charts, or scatter plots, to present the spending patterns and comparisons between different customer segments. Visualizations aid in conveying information effectively and identifying any trends or patterns in the data.

These analyzing metrics provide a framework for understanding customer spending patterns

In [2]:
```
df = pd.read_csv("walmart.csv")
df
```

Out[2]:

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

550068 rows × 10 columns

In [3]:
```
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 [29]:
```
# Convert columns to category data type
columns_to_convert = ["Gender", "Occupation", "Product_Category", "Marital_Sta
df[columns_to_convert] = df[columns_to_convert].astype("category")
# datatype of a few column having categorical attributes have been changed so
# more efficient storage and potential performance benefits when working with
```

In [8]:
```
# Check for null values
null_counts = df.isnull().sum()
print("Null Value Counts:")
print(null_counts)
# Here we observe that the given dataset doesn't have any null values.
```

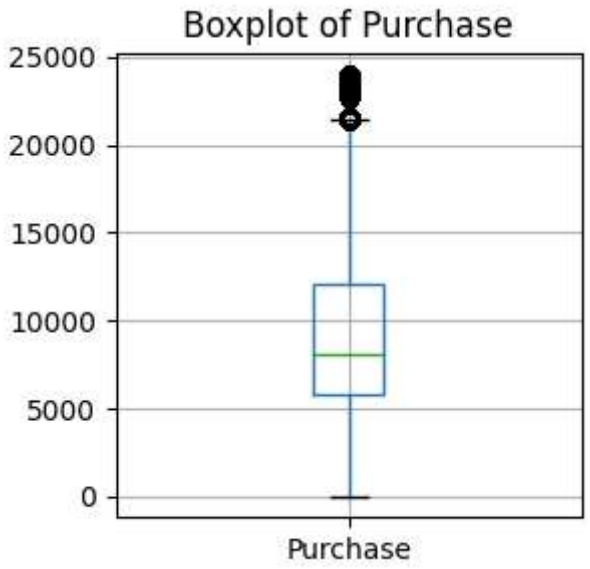
```
Null Value Counts:
User_ID                       0
Product_ID                    0
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category              0
Purchase                      0
dtype: int64
```

In the above line of code, we observe that the given dataset doesn't have any null values.

In [30]:
```
df["Purchase"].describe()
```

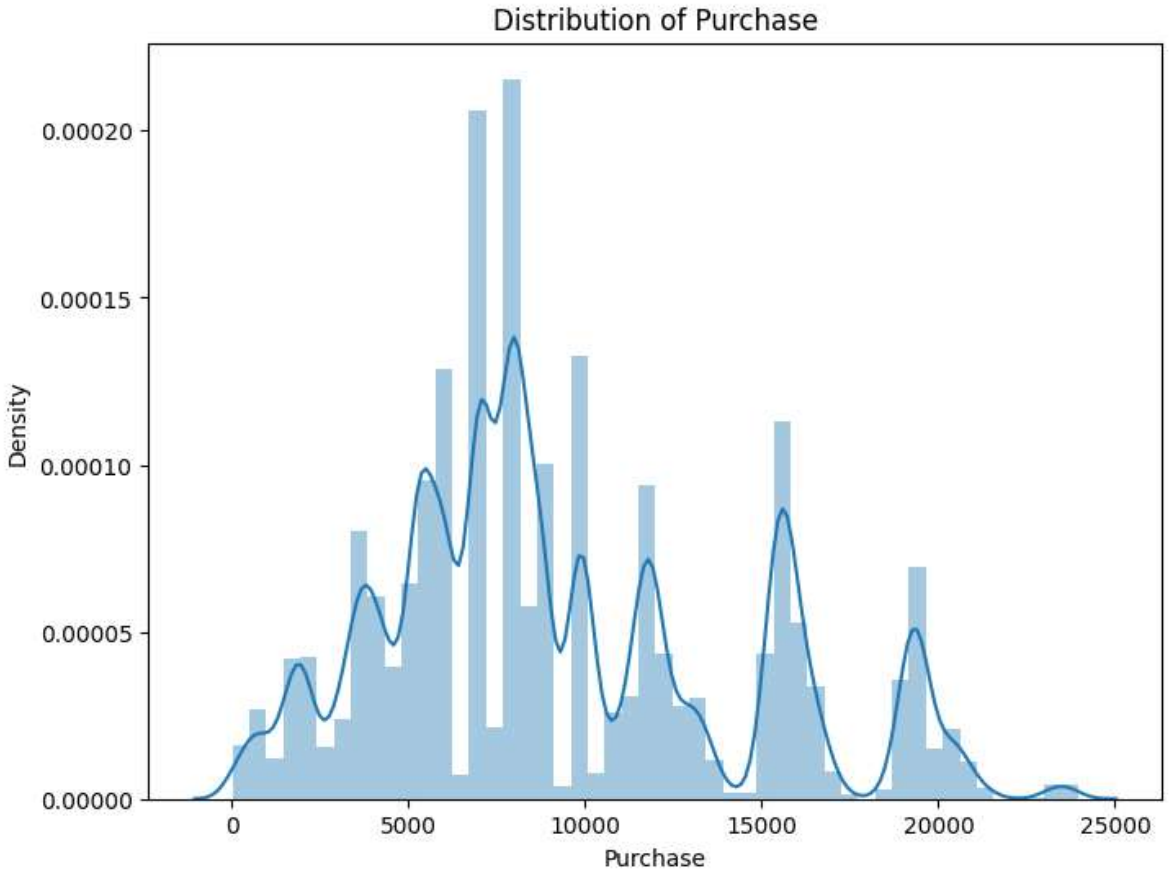
Out[30]:
```
count    550068.000000
mean       9263.968713
std        5023.065394
min          12.000000
25%        5823.000000
50%        8047.000000
75%       12054.000000
max       23961.000000
Name: Purchase, dtype: float64
```

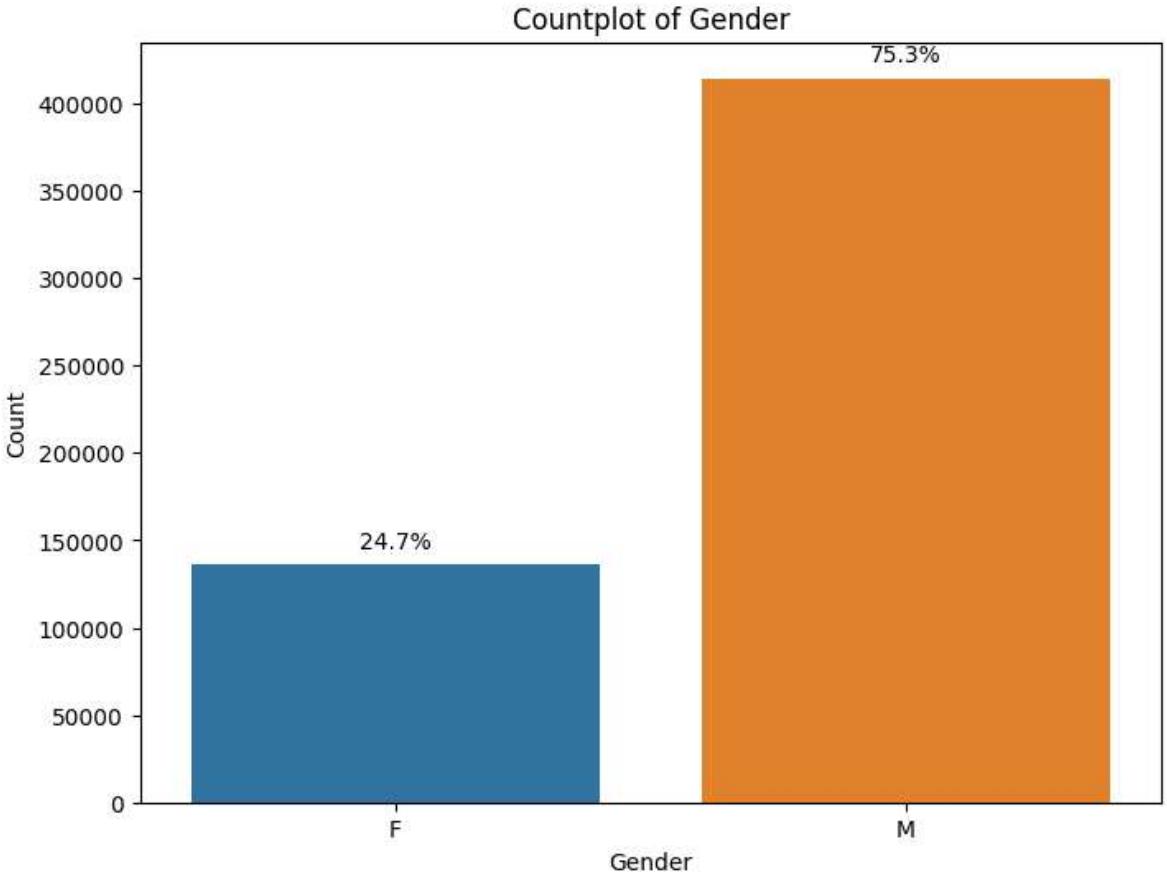
In [11]:
```
# Detect outliers using boxplot
plt.figure(figsize=(3,3))
df.boxplot(column=["Purchase"])
plt.title("Boxplot of Purchase")
plt.show()
```

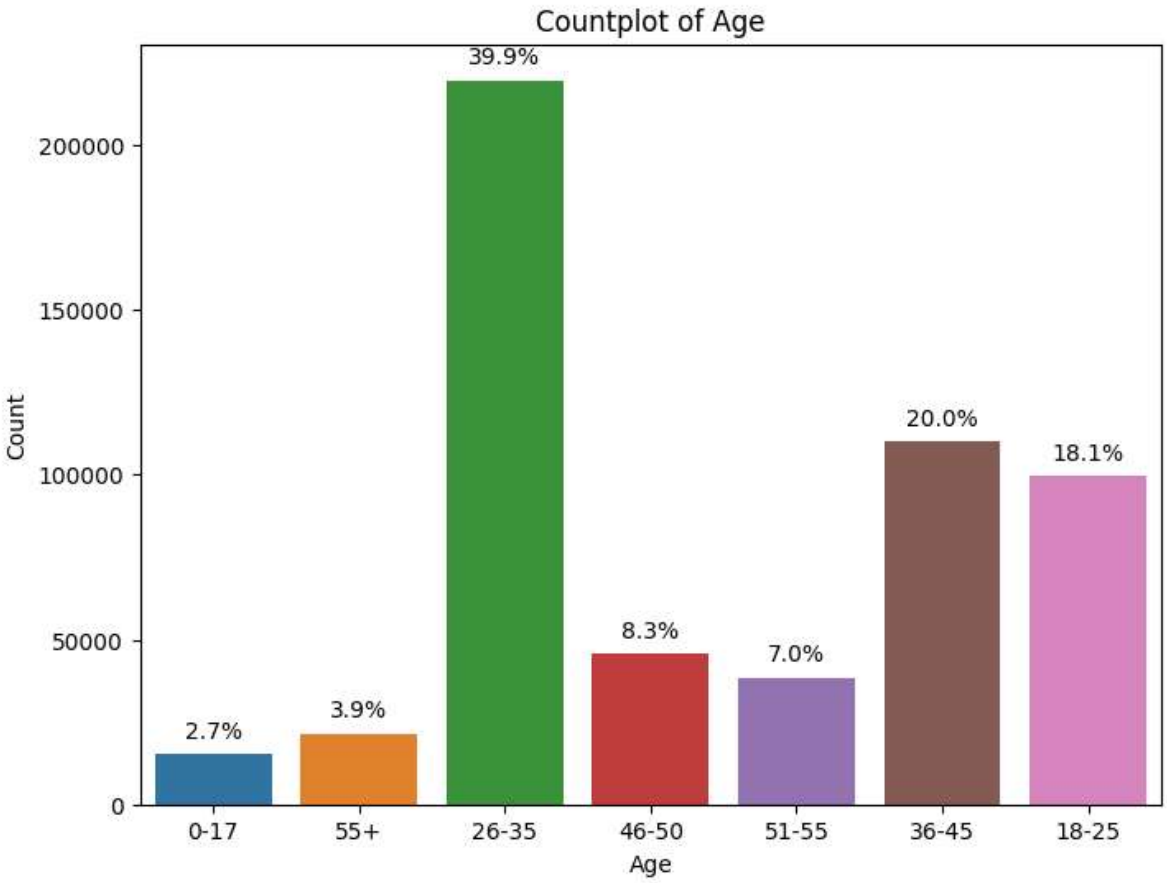


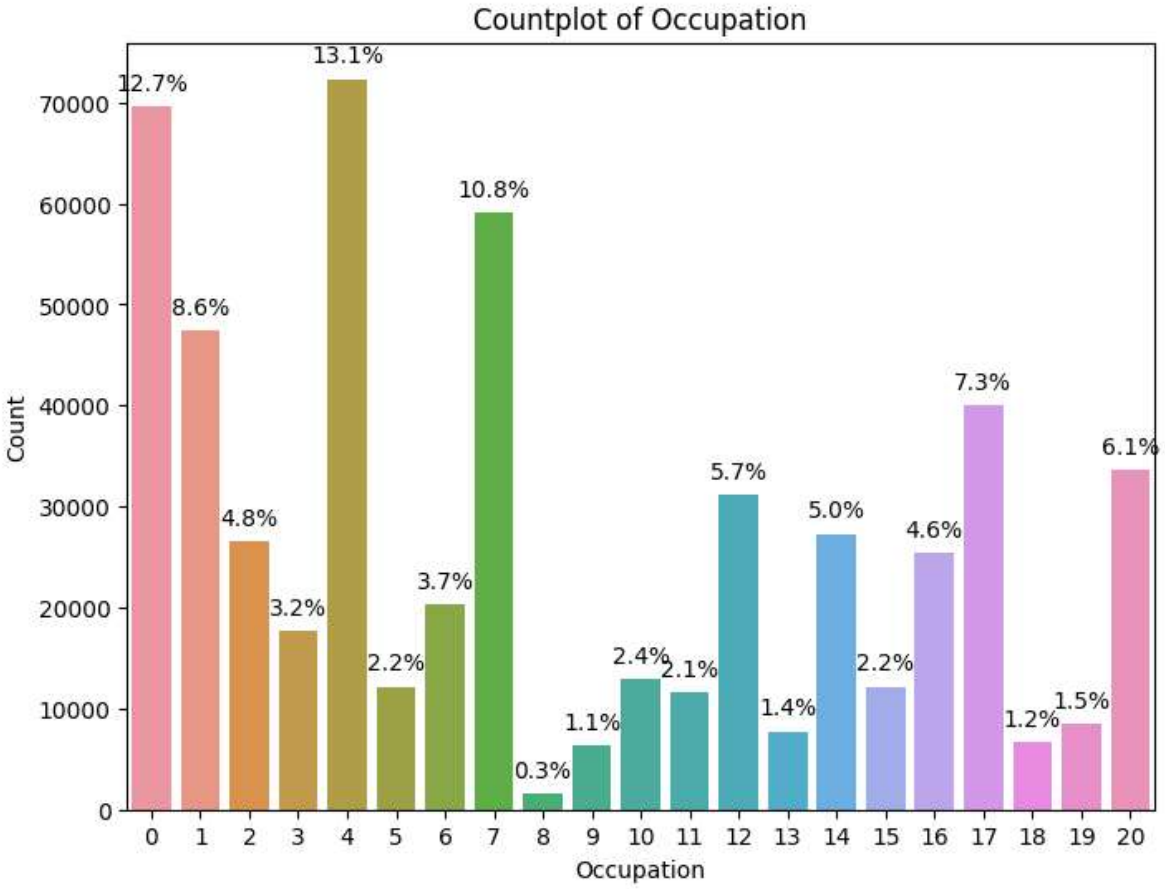
In [66]:
```
# Continuous Variables
continuous_vars = ["Purchase"]

# Categorical Variables
categorical_vars = ["Gender", "Age", "Occupation", "City_Category", "Stay_In_C
                    "Product_Category"]

# Plotting Distplot and Histogram for Continuous Variables
for var in continuous_vars:
    plt.figure(figsize=(8, 6))
    sns.distplot(df[var], kde=True)
    plt.title(f"Distribution of {var}")
    plt.xlabel(var)
    plt.ylabel("Density")
    plt.show()

# Plotting Countplot for Categorical Variables
for var in categorical_vars:
    plt.figure(figsize=(8, 6))
    ax = sns.countplot(x=var, data=df)
    plt.title(f"Countplot of {var}")
    plt.xlabel(var)
    plt.ylabel("Count")
    total = len(df[var])
    # Add percentage values above each bar
    for p in ax.patches:
        percentage = '{:.1f}%'.format(100 * p.get_height() / total)
        ax.annotate(percentage, (p.get_x() + p.get_width() / 2., p.get_height(
                    textcoords='offset points')
    plt.show()

# Plotting Boxplot for Categorical Variables
for var in categorical_vars:
    plt.figure(figsize=(8, 6))
    ax = sns.boxplot(x=var, y="Purchase", data=df)
    plt.title(f"Boxplot of Purchase by {var}")
    plt.xlabel(var)
    plt.ylabel("Purchase")
    # Add numerical values above each boxplot
    boxplot_stats = df.groupby(var)["Purchase"].describe()
    for i, box in enumerate(ax.artists):
        stats = boxplot_stats.loc[box.get_label()]
        ax.text(i, stats['75%'], f"Q3: {int(stats['75%'])}", ha='center', va='
        ax.text(i, stats['25%'], f"Q1: {int(stats['25%'])}", ha='center', va='
        ax.text(i, stats['max'], f"Max: {int(stats['max'])}", ha='center', va=
        ax.text(i, stats['min'], f"Min: {int(stats['min'])}", ha='center', va=
        ax.text(i, stats['median'], f"Median: {int(stats['50%'])}", ha='center
    plt.show()
```

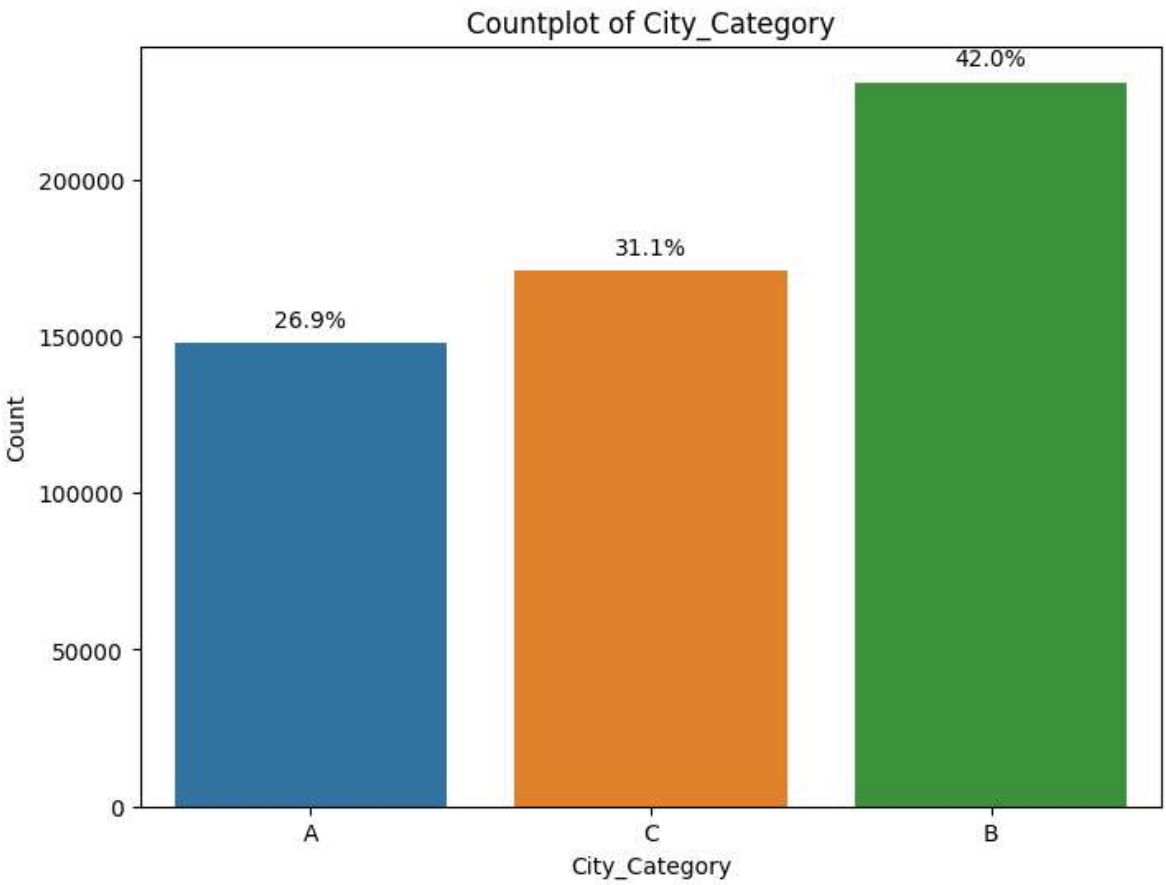
```
C:\Users\nidhi\AppData\Local\Temp\ipykernel_11668\409686796.py:11: UserWarnin
g:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751 (https://gis
t.github.com/mwaskom/de44147ed2974457ad6372750bbe5751)

  sns.distplot(df[var], kde=True)
```

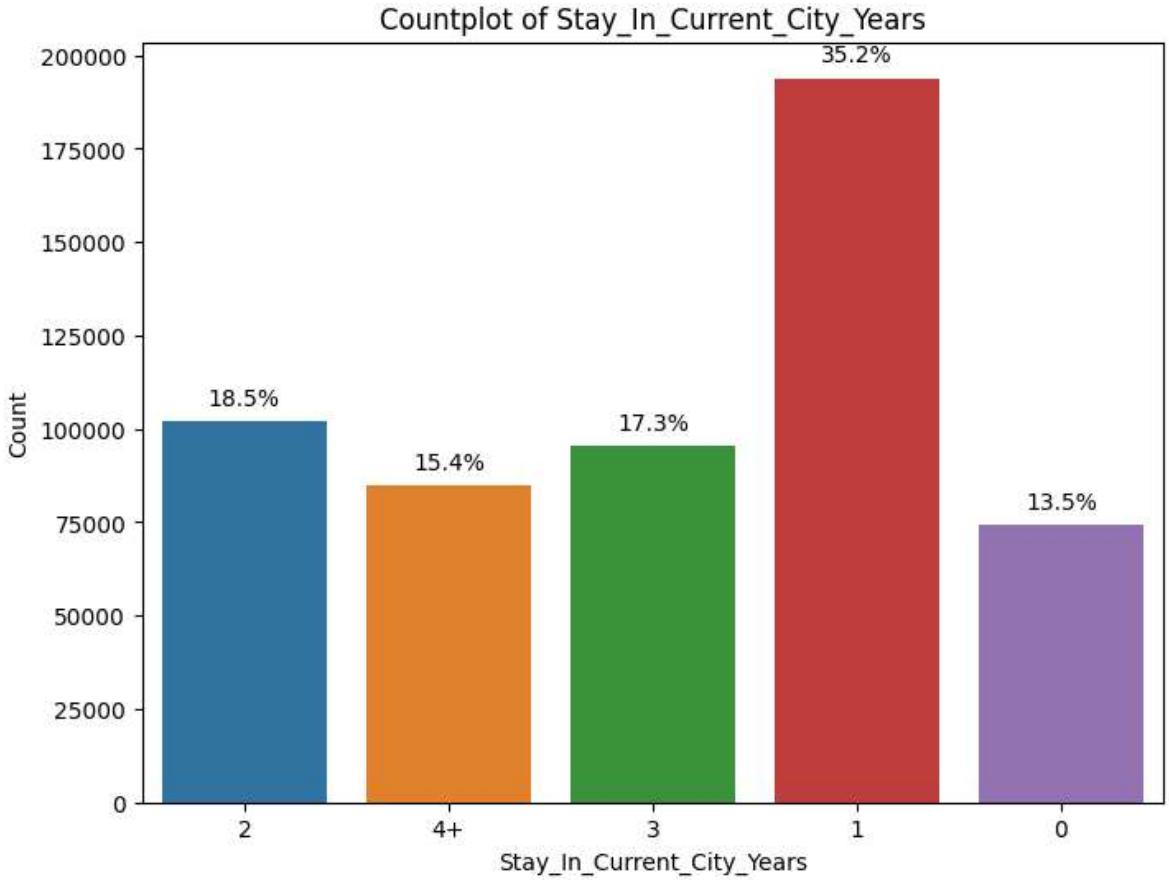


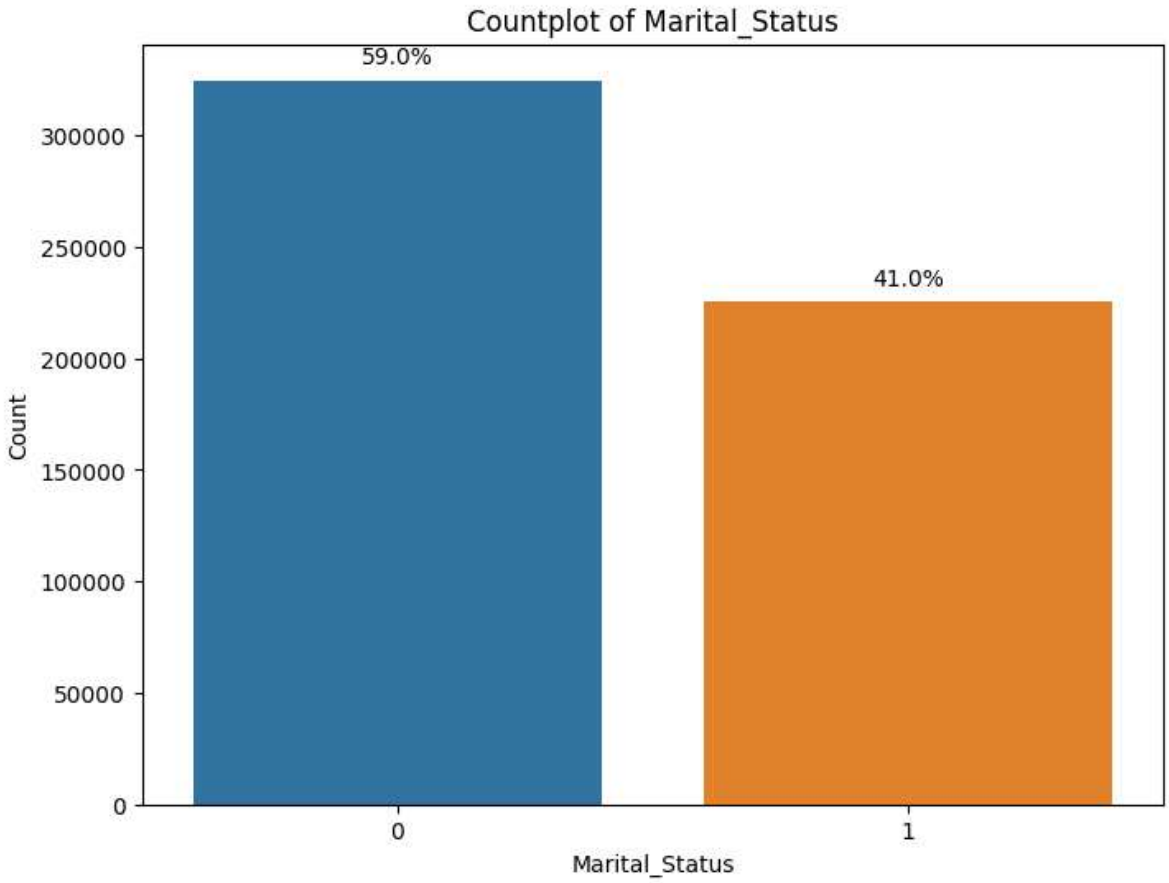

Countplot of Gender
75.3%
24.7%
400000
350000
300000
250000
200000
150000
100000
50000
0
Count
F
M
Gender

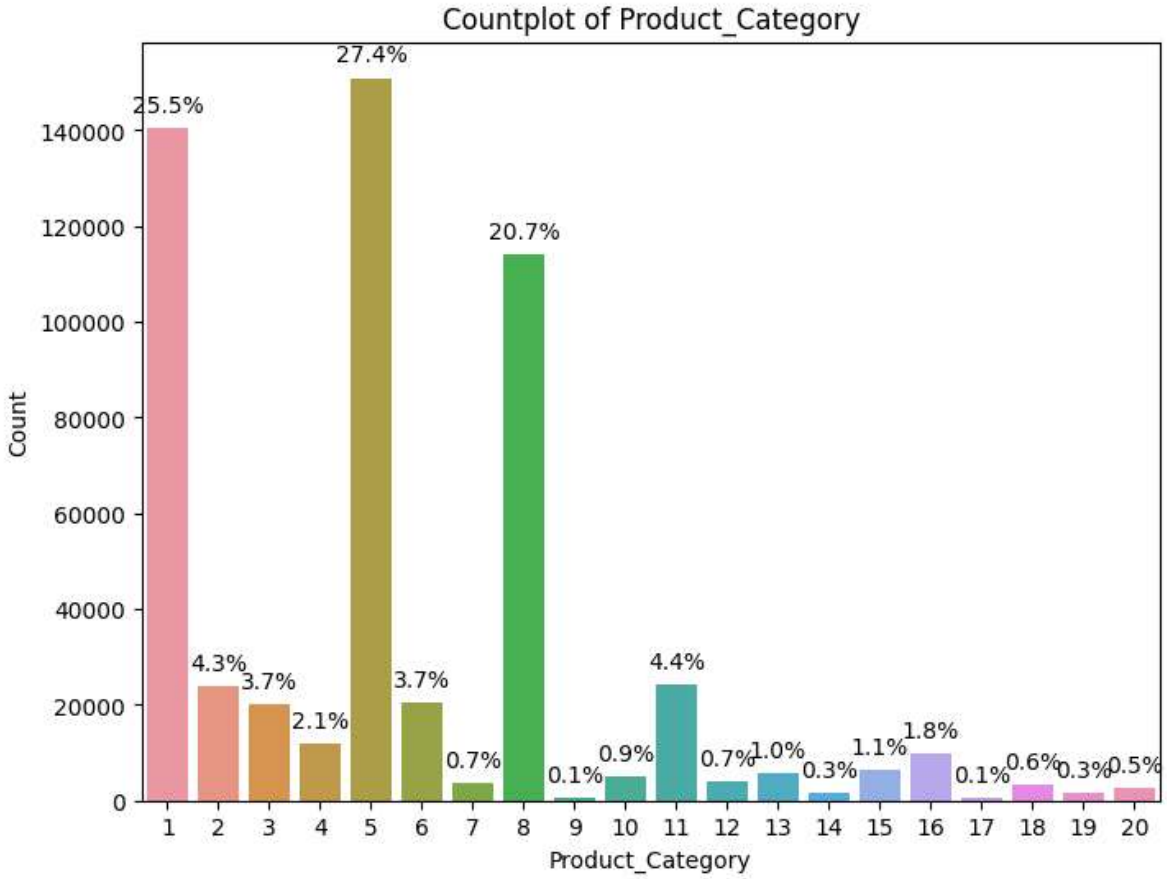

Countplot of Age
39.9%
20.0%
18.1%
8.3%
7.0%
3.9%
2.7%
200000
150000
100000
50000
0
Count
0-17
55+
26-35
46-50
51-55
36-45
18-25
Age

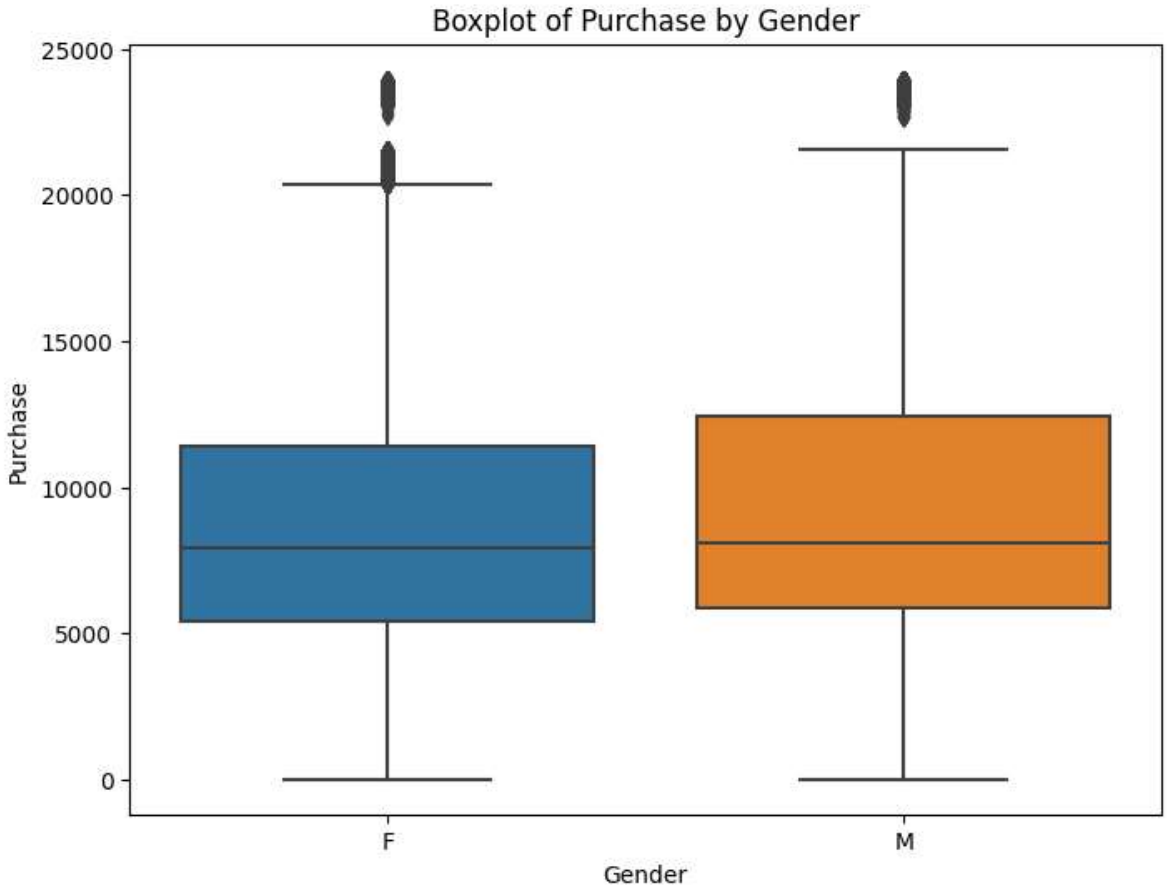

Countplot of Occupation
Count
Occupation
12.7%
8.6%
4.8%
3.2%
13.1%
2.2%
3.7%
10.8%
0.3%
1.1%
2.4%
2.1%
5.7%
1.4%
5.0%
2.2%
4.6%
7.3%
1.2%
1.5%
6.1%

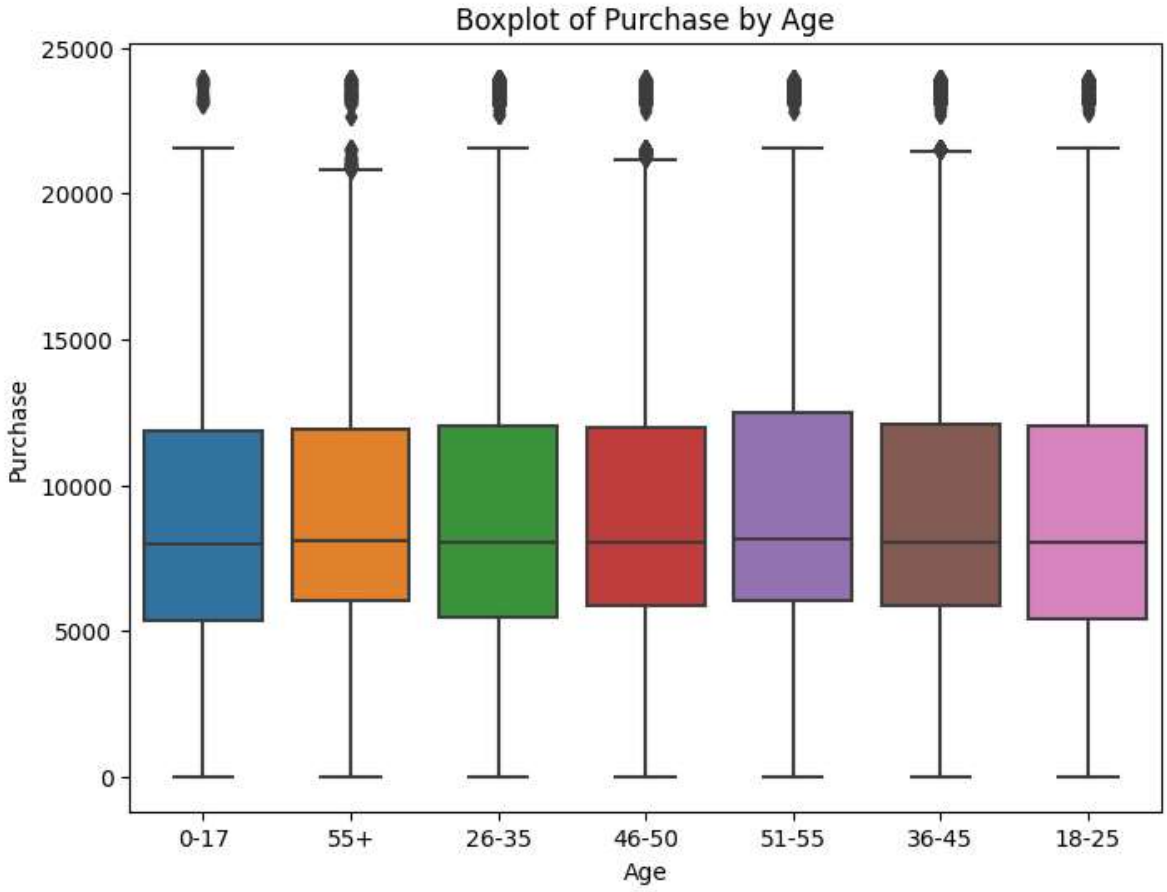

Countplot of City_Category
Count
City_Category
A
C
B
26.9%
31.1%
42.0%

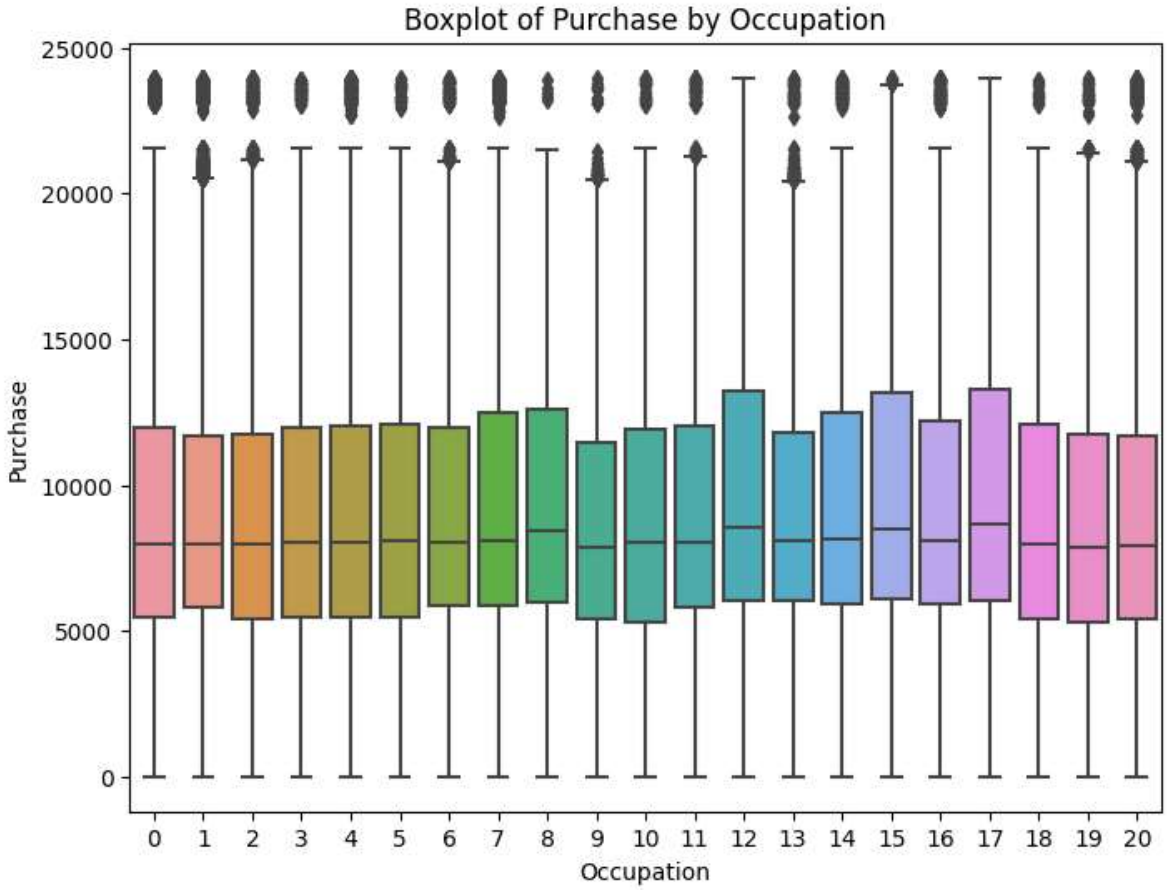

Countplot of Stay_In_Current_City_Years
Count
Stay_In_Current_City_Years
2
4+
3
1
0
18.5%
15.4%
17.3%
35.2%
13.5%
0
25000
50000
75000
100000
125000
150000
175000
200000

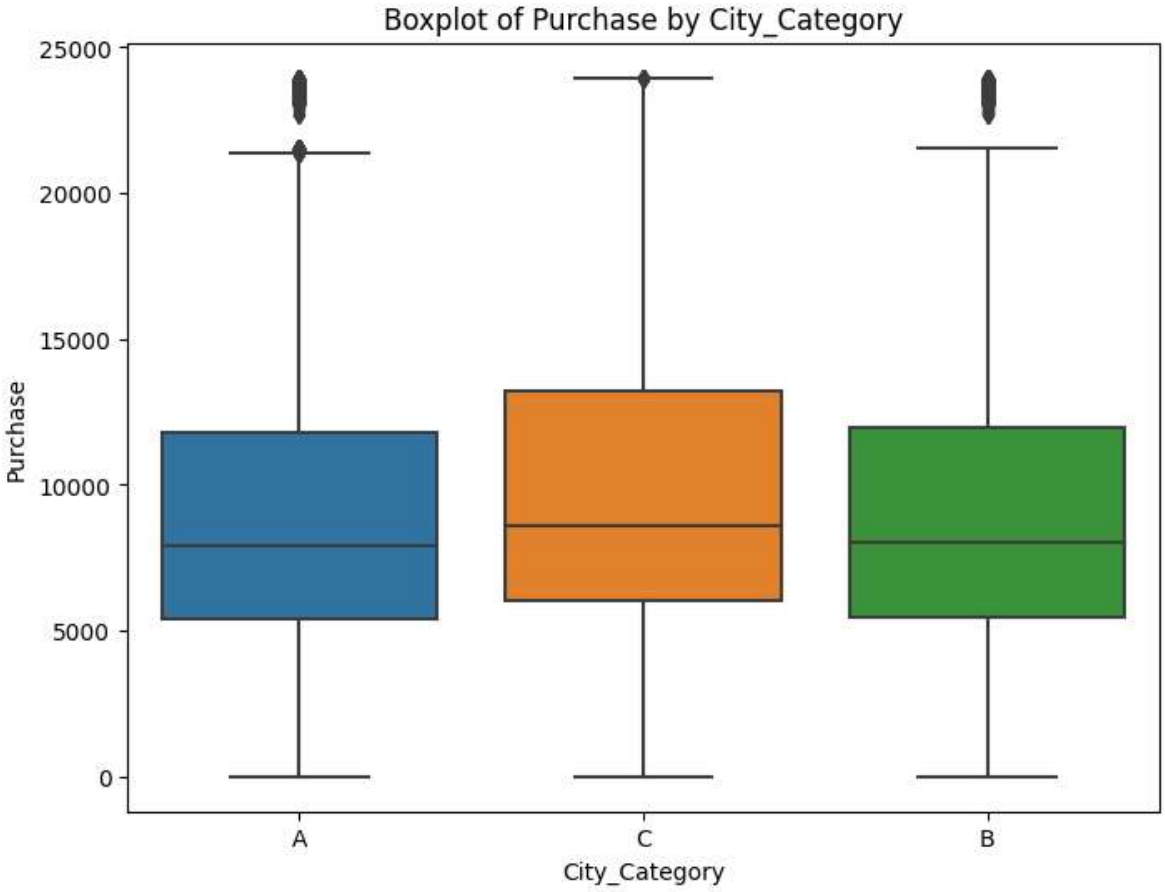

Countplot of Marital_Status
Count
Marital_Status
0
1
59.0%
41.0%
0
50000
100000
150000
200000
250000
300000

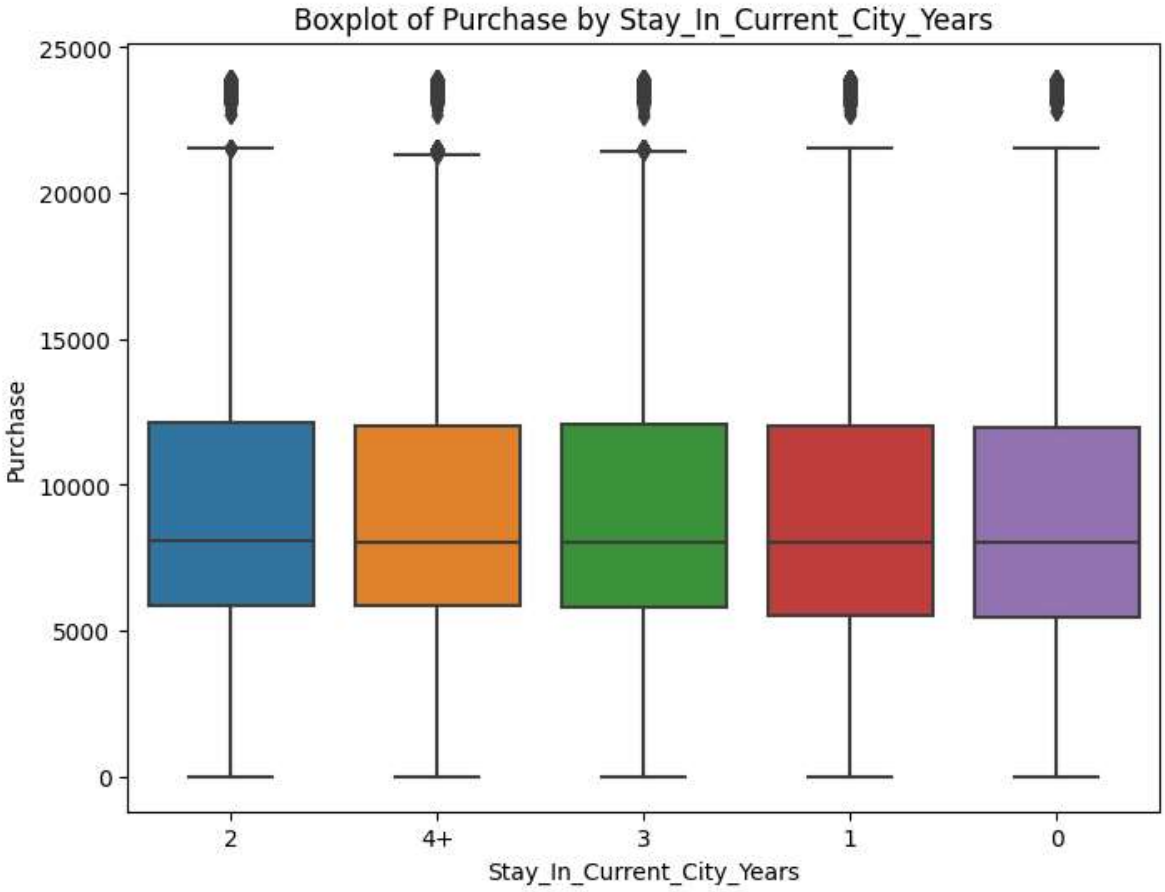

Countplot of Product_Category
Count
Product_Category
25.5%
4.3%
3.7%
2.1%
27.4%
3.7%
0.7%
20.7%
0.1%
0.9%
4.4%
0.7%
1.0%
0.3%
1.1%
1.8%
0.1%
0.6%
0.3%
0.5%

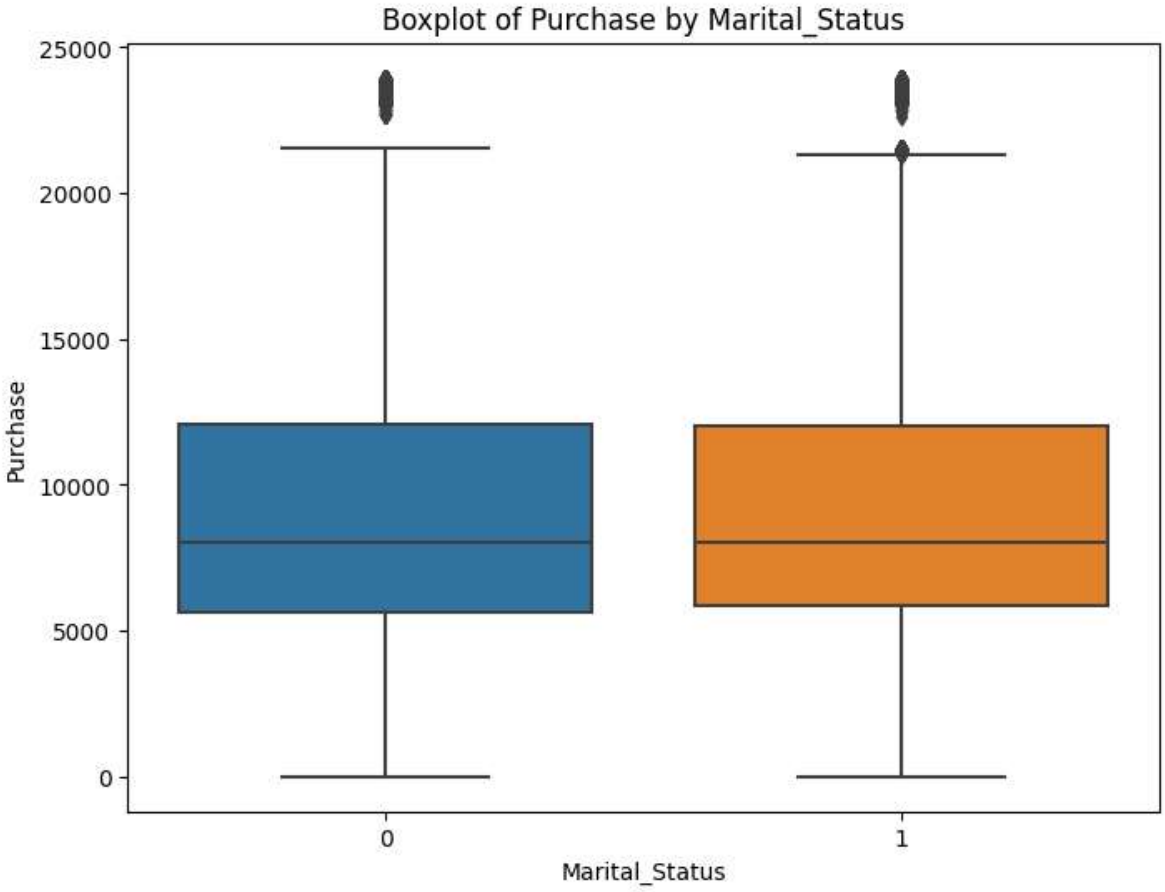

Boxplot of Purchase by Gender
Purchase
Gender
F
M

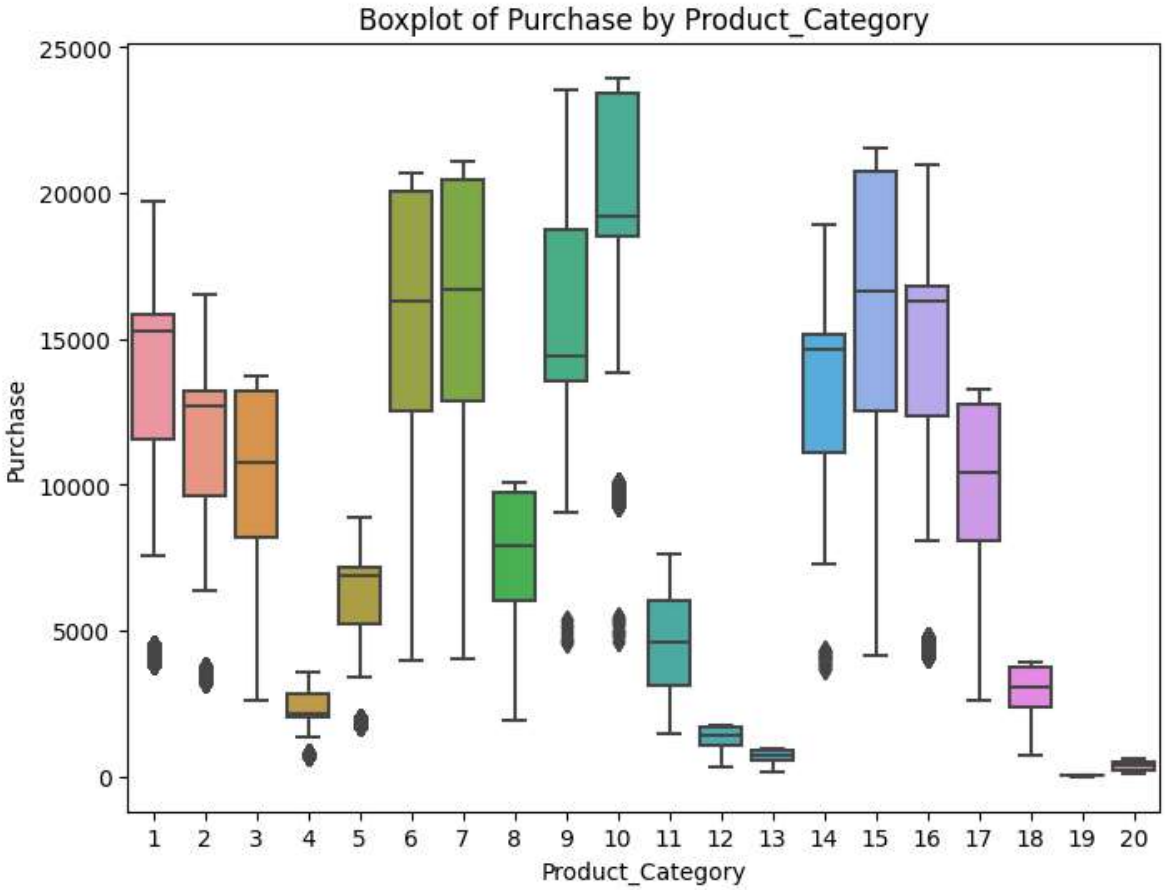

Boxplot of Purchase by Age
Purchase
25000
20000
15000
10000
5000
0
0-17
55+
26-35
46-50
51-55
36-45
18-25
Age


Boxplot of Purchase by Occupation
Purchase
25000
20000
15000
10000
5000
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
Occupation


Boxplot of Purchase by City_Category
Purchase
25000
20000
15000
10000
5000
0
A
C
B
City_Category


Boxplot of Purchase by Stay_In_Current_City_Years
Purchase
25000
20000
15000
10000
5000
0
2
4+
3
1
0
Stay_In_Current_City_Years


Boxplot of Purchase by Marital_Status
Purchase
Marital_Status
0
5000
10000
15000
20000
25000
0
1


Boxplot of Purchase by Product_Category
Purchase
Product_Category
0
5000
10000
15000
20000
25000
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20

In [69]:
```
# Categorical Variables
categorical_vars = ["Gender", "Age", "Occupation", "City_Category", "Stay_In_C
                    "Marital_Status", "Product_Category"]

# Groupby and Aggregation for Categorical Variables
for var in categorical_vars:
    var_stats = df.groupby(var)["Purchase"].describe()
    var_stats = var_stats[["count", "mean", "std", "min", "max"]]
    var_stats["percent_of_total"] = var_stats["count"] / df.shape[0] * 100
    print(f"Summary statistics of Purchase grouped by {var}:")
    print(var_stats)
    print()
```

```
Summary statistics of Purchase grouped by Gender:
           count         mean          std   min      max  percent_of_total
Gender
F       135809.0  8734.565765  4767.233289  12.0  23959.0         24.689493
M       414259.0  9437.526040  5092.186210  12.0  23961.0         75.310507

Summary statistics of Purchase grouped by Age:
          count         mean          std   min      max  percent_of_total
Age
0-17    15102.0  8933.464640  5111.114046  12.0  23955.0          2.745479
18-25   99660.0  9169.663606  5034.321997  12.0  23958.0         18.117760
26-35  219587.0  9252.690633  5010.527303  12.0  23961.0         39.919974
36-45  110013.0  9331.350695  5022.923879  12.0  23960.0         19.999891
46-50   45701.0  9208.625697  4967.216367  12.0  23960.0          8.308246
51-55   38501.0  9534.808031  5087.368080  12.0  23960.0          6.999316
55+     21504.0  9336.280459  5011.493996  12.0  23960.0          3.909335

Summary statistics of Purchase grouped by Occupation:
              count         mean          std   min      max  percent_of_tota
l
Occupation
0           69638.0  9124.428588  4971.757402  12.0  23961.0         12.65988
9
1           47426.0  8953.193270  4838.482159  12.0  23960.0          8.62184
3
2           26588.0  8952.481683  4939.418663  12.0  23955.0          4.83358
4
3           17650.0  9178.593088  5000.942719  12.0  23914.0          3.20869
4
4           72308.0  9213.980251  5043.674855  12.0  23961.0         13.14528
4
5           12177.0  9333.149298  5025.616603  12.0  23924.0          2.21372
6
6           20355.0  9256.535691  4989.216005  12.0  23951.0          3.70045
2
7           59133.0  9425.728223  5086.097089  12.0  23948.0         10.75012
5
8            1546.0  9532.592497  4916.641374  14.0  23869.0          0.28105
6
9            6291.0  8637.743761  4653.290986  13.0  23943.0          1.14367
7
10          12930.0  8959.355375  5124.339999  12.0  23955.0          2.35061
8
11          11586.0  9213.845848  5103.802992  12.0  23946.0          2.10628
5
12          31179.0  9796.640239  5140.437446  12.0  23960.0          5.66820
8
13           7728.0  9306.351061  4940.156591  12.0  23959.0          1.40491
7
14          27309.0  9500.702772  5069.600234  12.0  23941.0          4.96465
9
15          12165.0  9778.891163  5088.424301  12.0  23949.0          2.21154
5
16          25371.0  9394.464349  4995.918117  12.0  23947.0          4.61233
9
17          40043.0  9821.478236  5137.024383  12.0  23961.0          7.27964
5
```

```
18           6622.0  9169.655844  4987.697451  12.0  23894.0          1.20385
1
19           8461.0  8710.627231  5024.181000  12.0  23939.0          1.53817
3
20          33562.0  8836.494905  4919.662409  12.0  23960.0          6.10142
7

Summary statistics of Purchase grouped by City_Category:
                 count         mean          std   min      max  \
City_Category
A              147720.0  8911.939216  4892.115238  12.0  23961.0
B              231173.0  9151.300563  4955.496566  12.0  23960.0
C              171175.0  9719.920993  5189.465121  12.0  23961.0

               percent_of_total
City_Category
A                     26.854862
B                     42.026259
C                     31.118880

Summary statistics of Purchase grouped by Stay_In_Current_City_Years:
                              count         mean          std   min      max
\
Stay_In_Current_City_Years
0                             74398.0  9180.075123  4990.479940  12.0  23960.0
1                            193821.0  9250.145923  5027.476933  12.0  23961.0
2                            101838.0  9320.429810  5044.588224  12.0  23961.0
3                             95285.0  9286.904119  5020.343541  12.0  23961.0
4+                            84726.0  9275.598872  5017.627594  12.0  23958.0

                             percent_of_total
Stay_In_Current_City_Years
0                                  13.525237
1                                  35.235825
2                                  18.513711
3                                  17.322404
4+                                 15.402823

Summary statistics of Purchase grouped by Marital_Status:
                   count         mean          std   min      max  \
Marital_Status
0               324731.0  9265.907619  5027.347859  12.0  23961.0
1               225337.0  9261.174574  5016.897378  12.0  23961.0

                percent_of_total
Marital_Status
0                     59.034701
1                     40.965299

Summary statistics of Purchase grouped by Product_Category:
                    count          mean          std     min      max  \
Product_Category
1                  140378.0  13606.218596  4298.834894  3790.0  19708.0
2                   23864.0  11251.935384  3570.642713  3176.0  16504.0
3                   20213.0  10096.705734  2824.626957  2638.0  13717.0
4                   11753.0   2329.659491   812.540292   684.0   3556.0
5                  150933.0   6240.088178  1909.091687  1713.0   8907.0
```

6	20466.0	15838.478550	4011.233690	3981.0	20690.0
7	3721.0	16365.689600	4174.554105	4061.0	21080.0
8	113925.0	7498.958078	2013.015062	1939.0	10082.0
9	410.0	15537.375610	5330.847116	4528.0	23531.0
10	5125.0	19675.570927	4225.721898	4624.0	23961.0
11	24287.0	4685.268456	1834.901184	1472.0	7654.0
12	3947.0	1350.859894	362.510258	342.0	1778.0
13	5549.0	722.400613	183.493126	185.0	962.0
14	1523.0	13141.625739	4069.009293	3657.0	18931.0
15	6290.0	14780.451828	5175.465852	4148.0	21569.0
16	9828.0	14766.037037	4360.213198	4036.0	20971.0
17	578.0	10170.759516	2333.993073	2616.0	13264.0
18	3125.0	2972.864320	727.051652	754.0	3900.0
19	1603.0	37.041797	16.869148	12.0	62.0
20	2550.0	370.481176	167.116975	118.0	613.0

Product_Category	percent_of_total
1	25.520118
2	4.338373
3	3.674637
4	2.136645
5	27.438971
6	3.720631
7	0.676462
8	20.711076
9	0.074536
10	0.931703
11	4.415272
12	0.717548
13	1.008784
14	0.276875
15	1.143495
16	1.786688
17	0.105078
18	0.568112
19	0.291419
20	0.463579

Based on the analysis of the provided data, following insights can be drawn:

1. Range of Attributes:

   - Gender: The dataset includes both male and female customers, with approximately 75% being male and 25% being female.
   - Age: The age of customers ranges from 0 to 55+ years. The majority of customers fall within the age range of 26-35, followed by 36-45 and 18-25.
   - Occupation: The dataset covers various occupations, ranging from 0 to 20. Occupations 0, 4, and 7 have the highest representation in the dataset.
   - City_Category: Customers are distributed across three city categories: A, B, and C. Category B has the highest number of customers, followed by categories C and A.
   - Stay_In_Current_City_Years: Customers have varying durations of stay in their current cities. The majority of customers have been staying for 1 year, followed by 2, 3, 0 (recently moved), and 4+ years.

- Marital_Status: The dataset includes both married and unmarried customers, with approximately 59% being unmarried and 41% being married.
- Product_Category: The dataset covers a wide range of product categories, numbered from 1 to 20. Categories 1, 5, and 8 have the highest representation in the dataset.

2. Distribution of Variables and Relationships:

- Gender: The distribution of purchases between males and females shows that, on average, males have a slightly higher mean purchase value compared to females.
- Age: The distribution of purchases across different age groups shows that customers in the age range of 51-55 have the highest mean purchase value, followed by customers in the 55+ age group. Customers in the 0-17 age group have the lowest mean purchase value.
- Occupation: Different occupations show variations in their mean purchase values. Occupations 8 and 12 have the highest mean purchase values, while occupation 9 has the lowest mean purchase value.
- City_Category: Customers from city category C tend to have a higher mean purchase value compared to categories A and B.
- Stay_In_Current_City_Years: The duration of stay in the current city does not show a significant impact on the mean purchase value, as the values remain relatively consistent across different durations.
- Marital_Status: The distribution of purchases between married and unmarried customers does not show a significant difference in the mean purchase value.
- Product_Category: Different product categories exhibit varying mean purchase values. Categories 9, 10, 7, 6, and 15 have relatively higher mean purchase values, while categories 19, 13, 18, 12, and 17 have lower mean purchase values.

These insights provide an overview of the data and can help Walmart understand the customer segments that contribute to higher purchase values. However, further analysis and modeling may be required to uncover deeper insights and identify more specific patterns and relationships in the data.

In [70]:
```
# Check the difference between mean and median
purchase_mean = df["Purchase"].mean()
purchase_median = df["Purchase"].median()
difference = abs(purchase_mean - purchase_median)
print("The mean of overll purchases is: ",round(purchase_mean,2))
print("The median of overll purchases is: ",purchase_median)
print("Difference between mean and median of Purchase:", round(difference,2))
print("The mean is ",round((purchase_mean-purchase_median)*100/purchase_mean,2
# Here we observe that the mean is 13% bigger than the median,
# which indiactes that we must have some high spending outliers in the particu
# This could indicate that a portion of customers made large purchases,
# potentially taking advantage of Black Friday discounts or buying high-value
```

```
The mean of overll purchases is:  9263.97
The median of overll purchases is:  8047.0
Difference between mean and median of Purchase: 1216.97
The mean is  13.14 percent higher than the median
```

Here we observe that the mean is 13% bigger than the median, which indiactes that we must have some high spending outliers in this particular dataset.This could indicate that a portion of customers made large purchases, potentially taking advantage of Black Friday discounts or buying high-value items.

In [71]:
```
# Checking the difference between mean and median in case of male customers
purchase_mean_male = round(df[df["Gender"] == "M"]["Purchase"].mean(),2)
purchase_median_male = df[df["Gender"] == "M"]["Purchase"].median()
difference_male = round(abs(purchase_mean_male - purchase_median_male),2)
print("The mean of male purchases is: ",purchase_mean_male)
print("The median of male purchases is: ",purchase_median_male)
print("Difference between mean and median of male Purchase:", difference_male)
print("The mean is ",round((purchase_mean_male-purchase_median_male)*100/purch
```

```
The mean of male purchases is:  9437.53
The median of male purchases is:  8098.0
Difference between mean and median of male Purchase: 1339.53
The mean is  14.19 percent higher than the median
```

So, the male mean-median % difference is similar to that of the overall purchases. From a retail business perspective, this may suggest that certain male customers are more inclined to make larger purchases on Black Friday, possibly driven by specific product preferences or shopping behaviors.

In [72]:
```
# Checking the difference between mean and median in case of female customers
purchase_mean_female = round(df[df["Gender"] == "F"]["Purchase"].mean(),2)
purchase_median_female = df[df["Gender"] == "F"]["Purchase"].median()
difference_female = round(abs(purchase_mean_male - purchase_median_male),2)
print("The mean of female purchases is: ",purchase_mean_female)
print("The median of female purchases is: ",purchase_median_female)
print("Difference between mean and median of female Purchase:", difference_fem
print("The mean is ",round((purchase_mean_female-purchase_median_female)*100/p
```

```
The mean of female purchases is:  8734.57
The median of female purchases is:  7914.0
Difference between mean and median of female Purchase: 1339.53
The mean is  9.39 percent higher than the median
```

The mean of female purchases is 9.39% higher than the median. Interestingly, the mean-median difference for female purchases is lower compared to the overall and male purchases. This suggests that there are fewer high spending female outliers on Black Friday. Retailers could interpret this finding as an indication that female

customers, on average, exhibit more consistent spending patterns and may not engage in as many large-scale purchases on this particular shopping day. This insight could help Walmart tailor their offers and promotions to better suit the spending habits of female customers, focusing on value, convenience, and targeted product categories.

Overall, analyzing the mean-median differences provides valuable insights into the spending behavior of different customer segments. By understanding these differences, Walmart can design targeted marketing strategies, personalized offers, and product assortments to cater to the distinct preferences and needs of various customer segments. This approach can lead to enhanced customer satisfaction, increased sales, and improved customer retention.

In [6]:
```
df["Purchase"].describe().round(2)
# Here, we can observe a lot of things. One of them being the range of values
# Purchase --> (12,23961)
```

Out[6]:
```
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
```

In [19]:
```
df2 = (df.groupby("Gender")["User_ID"]
.count()
.reset_index()
.rename(columns = {"User_ID":"Count"})
)
df2
```

Out[19]:

	Gender	Count
0	F	135809
1	M	414259

In [20]:
```
df3 = (df.groupby("Gender")["Purchase"]
.sum()
.reset_index()
.rename(columns = {"User_ID":"Total_Purchase"})
)
df3
```

Out[20]:

	Gender	Purchase
0	F	1186232642
1	M	3909580100

As we can see here, the females spent around 1.18 bn dollars while, on the same black friday, the males spent 3.9 bn dollars.

Now, we know that the population mean for Male customers is 50mn while that of female customers is 50mn as well. So, treating this black friday as a sample of the overall spending behaviour of the customers, we can take 1.18 bn as the sample mean for female spenders and 3.9 bn as that for their male counterparts.

In [46]:
```
df6 = (df.groupby("Age")["User_ID"]
.count()
.reset_index()
.rename(columns = {"User_ID":"Count"})
)
df6
```

Out[46]:

	Age	Count
0	0-17	15102
1	18-25	99660
2	26-35	219587
3	36-45	110013
4	46-50	45701
5	51-55	38501
6	55+	21504

In [9]:
```
purchase_range = (np.min(df["Purchase"]), np.max(df["Purchase"]))
purchase_range
```

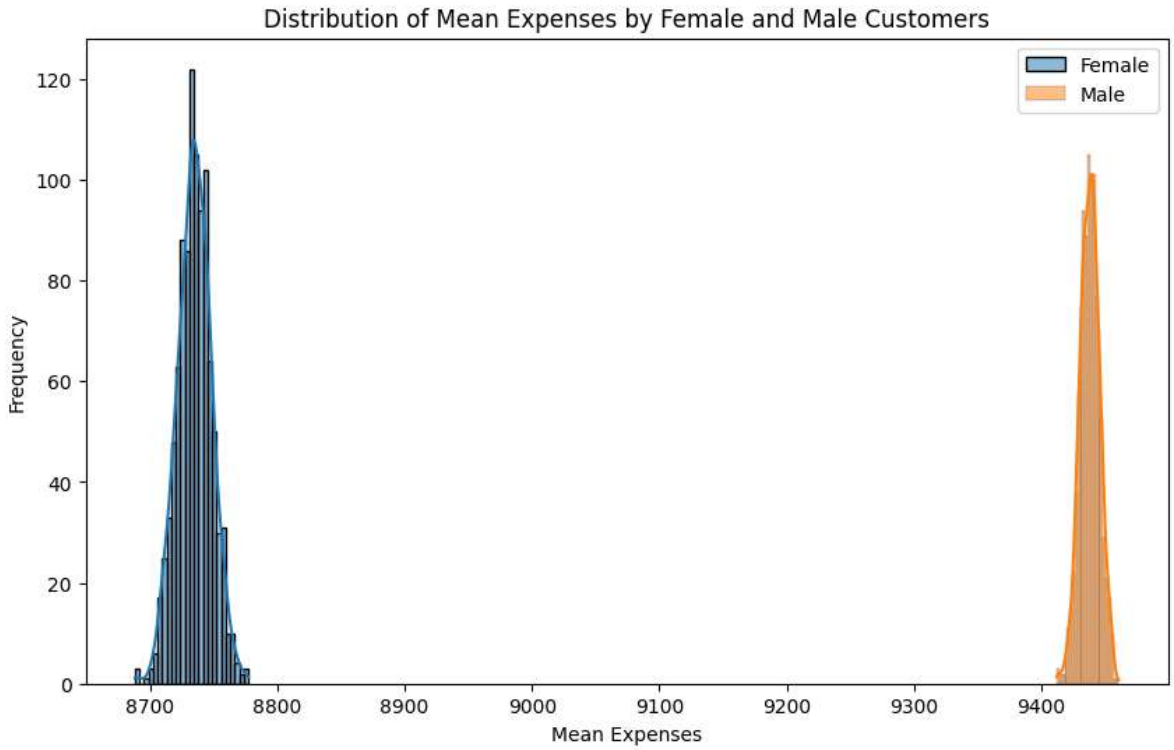
Out[9]:
```
(12, 23961)
```

In [9]:
```
# Set the random seed for reproducibility
np.random.seed(42)

# Set the number of bootstrap iterations
n_iterations = 1000

# Initialize empty arrays to store bootstrap statistics
bootstrap_female_mean = np.zeros(n_iterations)
bootstrap_male_mean = np.zeros(n_iterations)
bootstrap_married_mean = np.zeros(n_iterations)
bootstrap_unmarried_mean = np.zeros(n_iterations)
bootstrap_age1_mean = np.zeros(n_iterations)
bootstrap_age2_mean = np.zeros(n_iterations)
bootstrap_age3_mean = np.zeros(n_iterations)
bootstrap_age4_mean = np.zeros(n_iterations)
bootstrap_age5_mean = np.zeros(n_iterations)
bootstrap_age6_mean = np.zeros(n_iterations)
bootstrap_age7_mean = np.zeros(n_iterations)

# Perform bootstrap resampling
for i in range(n_iterations):
    # Resample female and male customers with replacement
    bootstrap_female_sample = df[df["Gender"] == "F"]["Purchase"].sample(n=135
    bootstrap_male_sample = df[df["Gender"] == "M"]["Purchase"].sample(n=41425

    # Resample married and unmarried customers with replacement
    bootstrap_married_sample = df[df["Marital_Status"] == 1]["Purchase"].sampl
    bootstrap_unmarried_sample = df[df["Marital_Status"] == 0]["Purchase"].sam

    # Resample customers by age group with replacement
    bootstrap_age_sample_1 = df[df["Age"] == "0-17"]["Purchase"].sample(n=1510
    bootstrap_age_sample_2 = df[df["Age"] == "18-25"]["Purchase"].sample(n=996
    bootstrap_age_sample_3 = df[df["Age"] == "26-35"]["Purchase"].sample(n=219
    bootstrap_age_sample_4 = df[df["Age"] == "36-45"]["Purchase"].sample(n=110
    bootstrap_age_sample_5 = df[df["Age"] == "46-50"]["Purchase"].sample(n=457
    bootstrap_age_sample_6 = df[df["Age"] == "51-55"]["Purchase"].sample(n=385
    bootstrap_age_sample_7 = df[df["Age"] == "55+"]["Purchase"].sample(n=21504

    # Calculate the mean for each bootstrap sample
    bootstrap_female_mean[i] = bootstrap_female_sample.mean()
    bootstrap_male_mean[i] = bootstrap_male_sample.mean()
    bootstrap_married_mean[i] = bootstrap_married_sample.mean()
    bootstrap_unmarried_mean[i] = bootstrap_unmarried_sample.mean()
    bootstrap_age1_mean[i] = bootstrap_age_sample_1.mean()
    bootstrap_age2_mean[i] = bootstrap_age_sample_2.mean()
    bootstrap_age3_mean[i] = bootstrap_age_sample_3.mean()
    bootstrap_age4_mean[i] = bootstrap_age_sample_4.mean()
    bootstrap_age5_mean[i] = bootstrap_age_sample_5.mean()
    bootstrap_age6_mean[i] = bootstrap_age_sample_6.mean()
    bootstrap_age7_mean[i] = bootstrap_age_sample_7.mean()

# Calculate the confidence intervals for average male and female spending
confidence_interval_female = np.percentile(bootstrap_female_mean, [2.5, 97.5])
confidence_interval_male = np.percentile(bootstrap_male_mean, [2.5, 97.5])

# Calculate the confidence intervals for average married and unmarried spendin
confidence_interval_married = np.percentile(bootstrap_married_mean, [2.5, 97.5
```

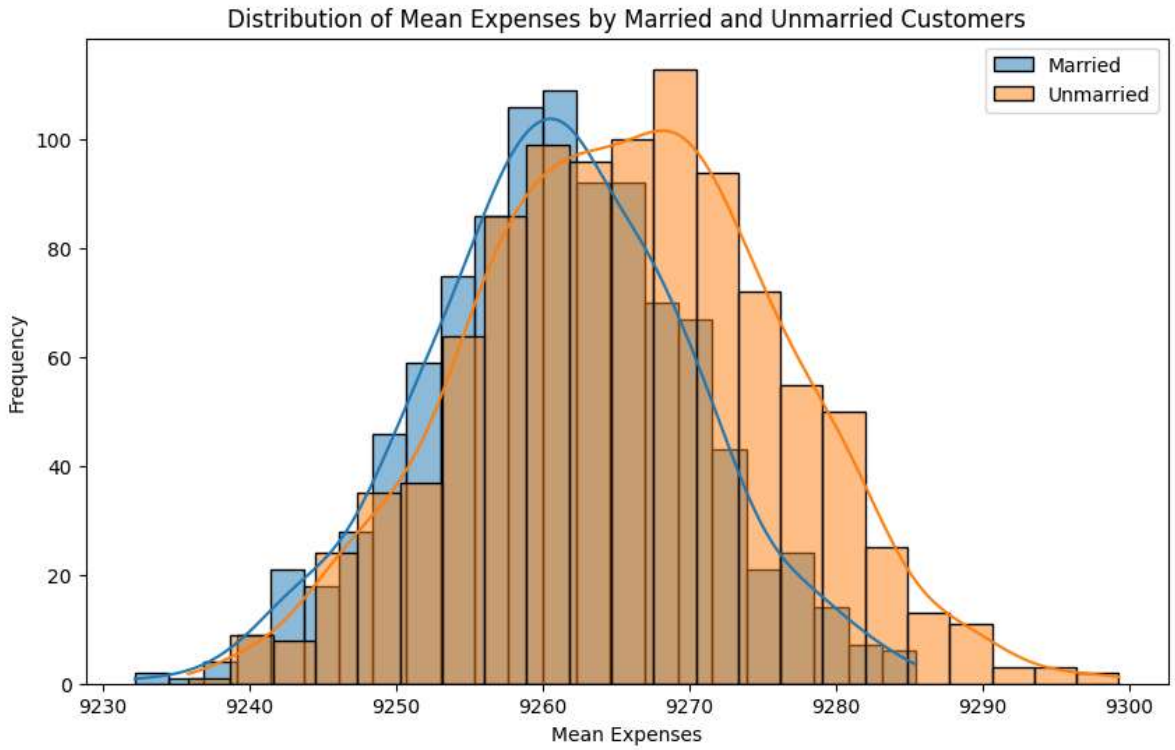
```
confidence_interval_unmarried = np.percentile(bootstrap_unmarried_mean, [2.5,

# Calculate the confidence intervals for average spending in different age gro
confidence_interval_age1 = np.percentile(bootstrap_age1_mean, [2.5, 97.5])
confidence_interval_age2 = np.percentile(bootstrap_age2_mean, [2.5, 97.5])
confidence_interval_age3 = np.percentile(bootstrap_age3_mean, [2.5, 97.5])
confidence_interval_age4 = np.percentile(bootstrap_age4_mean, [2.5, 97.5])
confidence_interval_age5 = np.percentile(bootstrap_age5_mean, [2.5, 97.5])
confidence_interval_age6 = np.percentile(bootstrap_age6_mean, [2.5, 97.5])
confidence_interval_age7 = np.percentile(bootstrap_age7_mean, [2.5, 97.5])

# Print the results
print("Results:")
print("1.(i) Are women spending more money per transaction than men?")
if confidence_interval_female[1] > confidence_interval_male[1]:
    print("   Yes, women are spending more money per transaction than men.")
else:
    print("   No, women are not spending more money per transaction than men."

# Plot the distribution of bootstrap means for female and male customers
plt.figure(figsize=(10, 6))
sns.histplot(bootstrap_female_mean, kde=True, label="Female")
sns.histplot(bootstrap_male_mean, kde=True, label="Male")
plt.title("Distribution of Mean Expenses by Female and Male Customers")
plt.xlabel("Mean Expenses")
plt.ylabel("Frequency")
plt.legend()
plt.show()

print("1.(ii) Confidence Intervals:")
print("   Female Customers: ", confidence_interval_female)
print("   Male Customers: ", confidence_interval_male)

print("1.(iii) Are confidence intervals of average male and female spending ov
if confidence_interval_female[1] >= confidence_interval_male[0] and confidence
    print("   Yes, the confidence intervals of average male and female spendin
else:
    print("   No, the confidence intervals of average male and female spending

print("2.(i) Are married customers spending more money per transaction than un
if confidence_interval_married[1] > confidence_interval_unmarried[1]:
    print("   Yes, married customers are spending more money per transaction t
else:
    print("   No, married customers are not spending more money per transactio

# Plot the distribution of bootstrap means for married and unmarried customers
plt.figure(figsize=(10, 6))
sns.histplot(bootstrap_married_mean, kde=True, label="Married")
sns.histplot(bootstrap_unmarried_mean, kde=True, label="Unmarried")
plt.title("Distribution of Mean Expenses by Married and Unmarried Customers")
plt.xlabel("Mean Expenses")
plt.ylabel("Frequency")
plt.legend()
plt.show()

print("2.(ii) Confidence Intervals:")
print("   Married Customers: ", confidence_interval_married)
```

```
print("   Unmarried Customers: ", confidence_interval_unmarried)

print("2.(iii) Are confidence intervals of average married and unmarried custo
if confidence_interval_married[1] >= confidence_interval_unmarried[0] and conf
    print("   Yes, the confidence intervals of average married and unmarried c
else:
    print("   No, the confidence intervals of average married and unmarried cu

print("3.(i) Results for Different Age Groups:")
print("   Confidence Interval: ")
print("   Age Group (0-17): ", confidence_interval_age1)
print("   Age Group (17-25): ", confidence_interval_age2)
print("   Age Group (26-35): ", confidence_interval_age3)
print("   Age Group (36-45): ", confidence_interval_age4)
print("   Age Group (46-50): ", confidence_interval_age5)
print("   Age Group (51-55): ", confidence_interval_age6)
print("   Age Group (55+): ", confidence_interval_age7)

# Plot the distribution of bootstrap means for all the different age groups of
plt.figure(figsize=(10, 6))
sns.histplot(bootstrap_age1_mean, kde=True, label="0-17")
sns.histplot(bootstrap_age2_mean, kde=True, label="17-25")
sns.histplot(bootstrap_age3_mean, kde=True, label="26-35")
sns.histplot(bootstrap_age4_mean, kde=True, label="36-45")
sns.histplot(bootstrap_age5_mean, kde=True, label="46-50")
sns.histplot(bootstrap_age6_mean, kde=True, label="51-55")
sns.histplot(bootstrap_age7_mean, kde=True, label="55+")
plt.title("Distribution of Mean Expenses by all the different age groups of cu
plt.xlabel("Mean Expenses")
plt.ylabel("Frequency")
plt.legend()
plt.show()
```

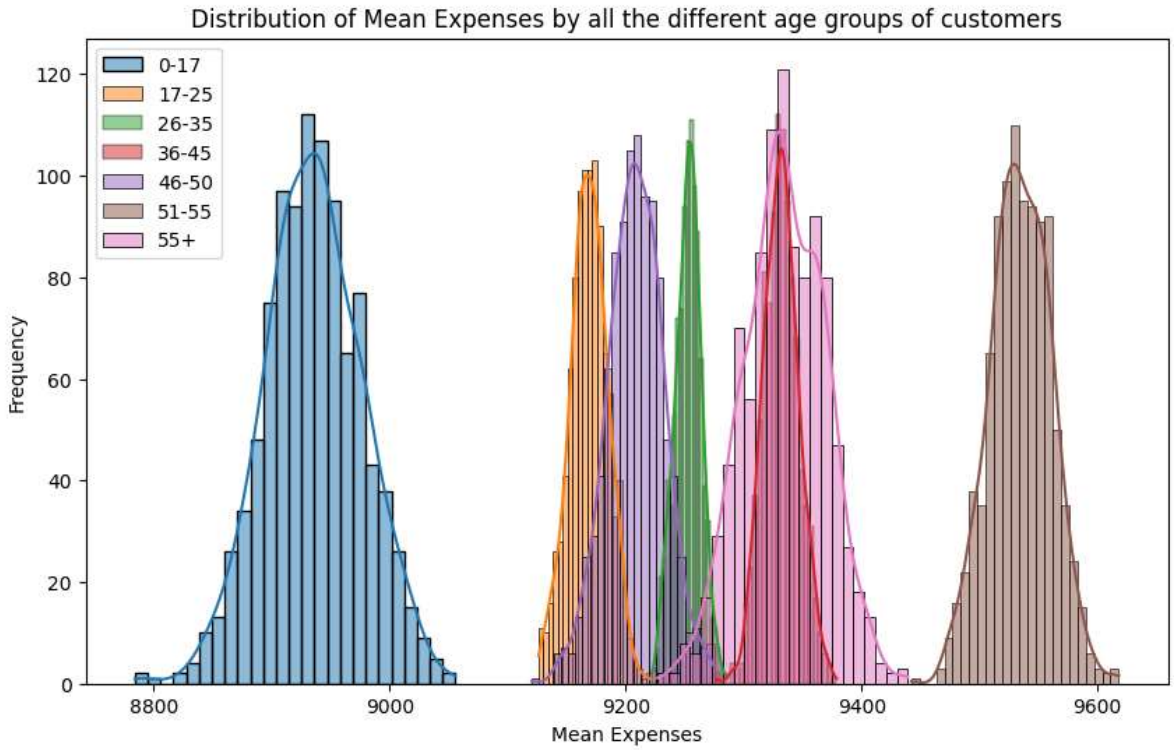
```
Results:
1.(i) Are women spending more money per transaction than men?
   No, women are not spending more money per transaction than men.
```



1.(ii) Confidence Intervals:
   Female Customers: [8708.95449031 8760.45516258]
   Male Customers: [9422.61592735 9453.06894624]
1.(iii) Are confidence intervals of average male and female spending overlapping?
   No, the confidence intervals of average male and female spending do not overlap.
2.(i) Are married customers spending more money per transaction than unmarried customers?
   No, married customers are not spending more money per transaction than unmarried customers.




```
2.(ii) Confidence Intervals:
   Married Customers:  [9242.86355907 9279.11584881]
   Unmarried Customers:  [9245.14125465 9286.76880838]
2.(iii) Are confidence intervals of average married and unmarried customers s
pending overlapping?
   Yes, the confidence intervals of average married and unmarried customers s
pending overlap.
3.(i) Results for Different Age Groups:
   Confidence Interval:
   Age Group (0-17):  [8854.69860449 9015.41389386]
   Age Group (17-25):  [9135.70551224 9199.23666993]
   Age Group (26-35):  [9230.53878952 9274.21350672]
   Age Group (36-45):  [9303.77401512 9361.11714207]
   Age Group (46-50):  [9161.14496127 9256.87812247]
   Age Group (51-55):  [9481.76767227 9584.36707423]
   Age Group (55+):  [9267.40035691 9401.18092448]
```



Based on the business problem faced by Walmart, here are modified recommendations tailored to their specific context:

Modified recommendations for Walmart, considering the analysis results:

1. **Promote Targeted Deals**: Design targeted deals and promotions that appeal to specific customer segments. While women may not spend more on average, create marketing campaigns highlighting products and offers that resonate with their preferences. At the same time, ensure that male customers are also provided with attractive deals to cater to their specific needs.
2. **Enhance Personalized Shopping Experience**: Leverage customer data to personalize the shopping experience for both married and unmarried customers. Use insights gained from the analysis to tailor product recommendations, offers, and discounts based on individual preferences. This approach can help increase customer satisfaction and encourage repeat purchases.
3. **Curate Product Assortment**: Optimize the product assortment by considering the preferences and spending patterns of different age groups. Analyze the confidence intervals of average spending within each age group and adjust the inventory accordingly. Offer a wide variety of products that align with the needs and preferences of customers across different age segments.
4. **Improve In-Store and Online Experience**: Focus on creating an exceptional in-store and online experience for all customers. Train store associates to provide personalized assistance to shoppers of all genders and age groups. Enhance the online shopping platform with intuitive navigation, customer reviews, and easy-to-use filters to facilitate product discovery for both married and unmarried customers.
5. **Customer Engagement and Feedback**: Actively engage with customers to gather feedback and understand their evolving preferences. Encourage customers to leave reviews, ratings, and suggestions to improve the shopping experience. This feedback can

help identify areas for improvement and drive customer loyalty among both male and female shoppers.

6. **Inclusive Marketing Campaigns**: Develop inclusive marketing campaigns that showcase diversity and cater to the interests of all customer segments. Highlight stories and experiences of diverse customers to create a sense of inclusivity. Promote equality and diversity within the organization and extend these values to external communications.
7. **Continuous Data Analysis**: Establish a dedicated analytics team to monitor and analyze customer spending behavior regularly. Continuously track the confidence intervals of average spending by gender, marital status, and age group. Leverage advanced analytics techniques to identify emerging trends and adapt strategies accordingly.

By incorporating these modified recommendations, Walmart can better address the needs and preferences of their diverse customer base. Understanding the spending behavior across genders, marital statuses, and age groups allows for more targeted marketing efforts, personalized experiences, and optimized product offerings. Ultimately, these actions can enhance customer satisfaction, drive sales, and strengthen Walmart's position in the retail

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: