```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
import gdown
from scipy.stats import norm
!gdown 1lcjtmvtSjco6cnWaJvgJNTGjGJoKieh1
```

```
Downloading...
From: https://drive.google.com/uc?id=1lcjtmvtSjco6cnWaJvgJNTGjGJoKieh1
To: /content/walmart-data.csv
100% 23.0M/23.0M [00:00<00:00, 180MB/s]
```

```
# Loading the dataset -
data = pd.read_csv('https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/walmart_data.csv?1641285094')
data.sample(10)
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	P
97760	1003169	P00274042	M	51-55	0	C	2	0	1	
378227	1004183	P00171742	M	26-35	1	B	4+	0	5	
244745	1001694	P00028842	M	26-35	12	B	3	1	6	
87270	1001451	P00109242	M	36-45	20	B	0	0	6	
320160	1001320	P00289242	M	26-35	4	A	2	0	1	
357959	1001150	P00051542	F	26-35	20	A	1	0	8	
421183	1004796	P00311542	M	36-45	7	A	4+	1	5	
214199	1003087	P00136442	F	0-17	1	B	0	0	14	
130764	1002073	P00186942	F	18-25	4	B	4+	0	5	
66520	1004179	P00086742	F	36-45	20	C	3	1	5	

```
data.shape
```

```
(550068, 10)
```

```
data.info()
```

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

```
data.isnull().sum()
```

	0
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

```
data.isnull().sum().sum()
```

```
0
```

Unique Attributes and Value Counts

```
data.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purcha
0	1000001	P00069042	F	0-17	10	A	2	0	3	83
1	1000001	P00248942	F	0-17	10	A	2	0	1	152
2	1000001	P00087842	F	0-17	10	A	2	0	12	14

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

User_ID	count
1001680	1026
1004277	979
1001941	898
1001181	862
1000889	823
...	...
1002690	7
1002111	7
1005810	7
1004991	7
1000708	6

5891 rows × 1 columns

dtype: int64

```
data['User_ID'].nunique()
```

```
5891
```

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

Product_ID	count
P00265242	1880
P00025442	1615
P00110742	1612
P00112142	1562
P00057642	1470
...	...
P00314842	1
P00298842	1
P00231642	1
P00204442	1
P00066342	1

3631 rows × 1 columns

dtype: int64

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

```
3631
```

Visualizing the data

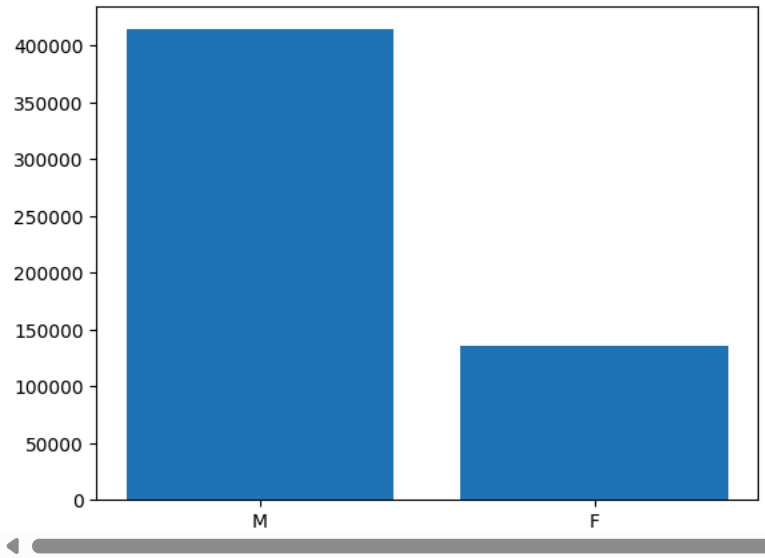
```
gender=data['Gender'].value_counts()
gender
```

Gender	count
M	414259
F	135809

dtype: int64

```
x_bar=gender.index
y_bar=gender
plt.bar(x_bar,y_bar)
```

```
<BarContainer object of 2 artists>
```

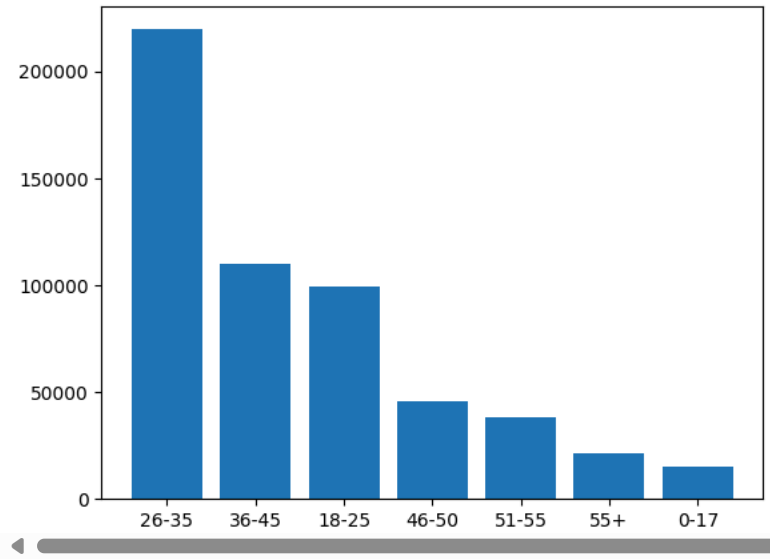


On the whole from the given data, 75% are males and rest are females.

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

```
x_bar=age.index
y_bar=age
plt.bar(x_bar,y_bar)
```

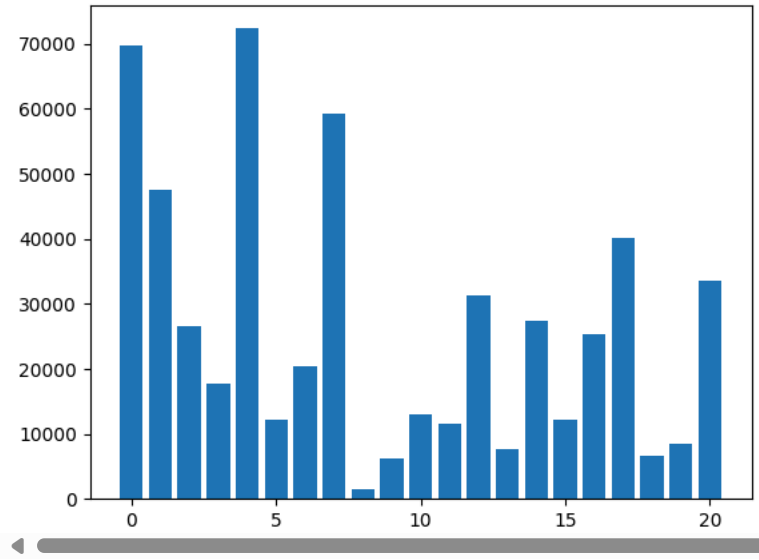
```
<BarContainer object of 7 artists>
```



Most of the Walmart customers are from the age range between 26 and 35.

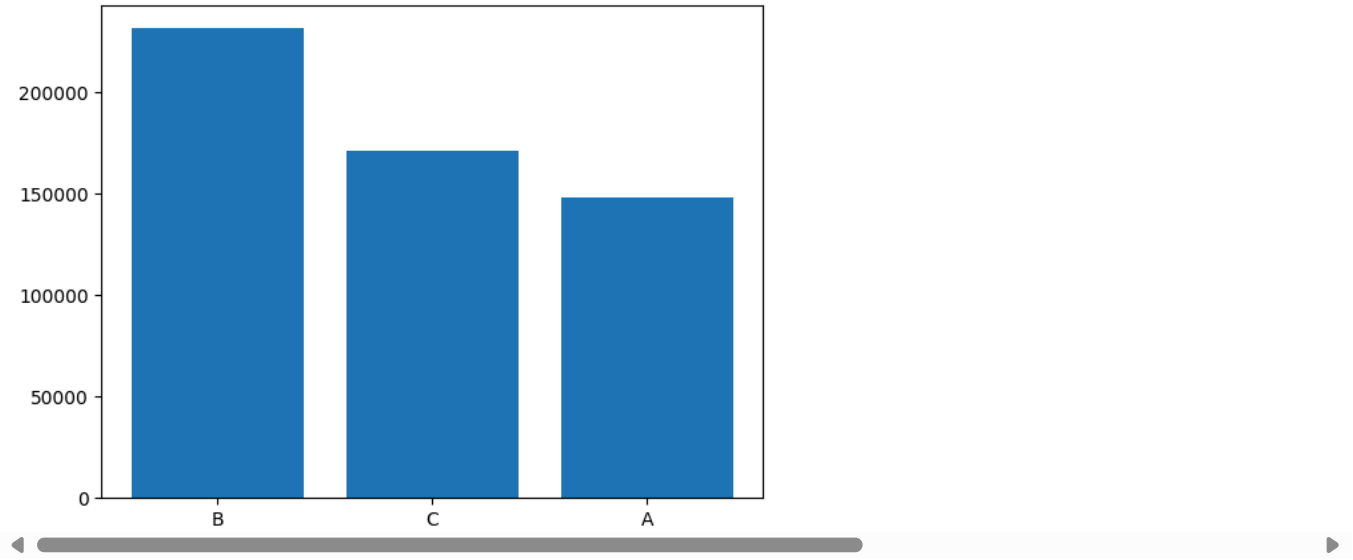
```
occ=data['Occupation'].value_counts()
x_bar=occ.index
y_bar=occ
plt.bar(x_bar,y_bar)
```

```
<BarContainer object of 21 artists>
```




```
city_cat=data['City_Category'].value_counts()
x_bar=city_cat.index
y_bar=city_cat
plt.bar(x_bar,y_bar)
```

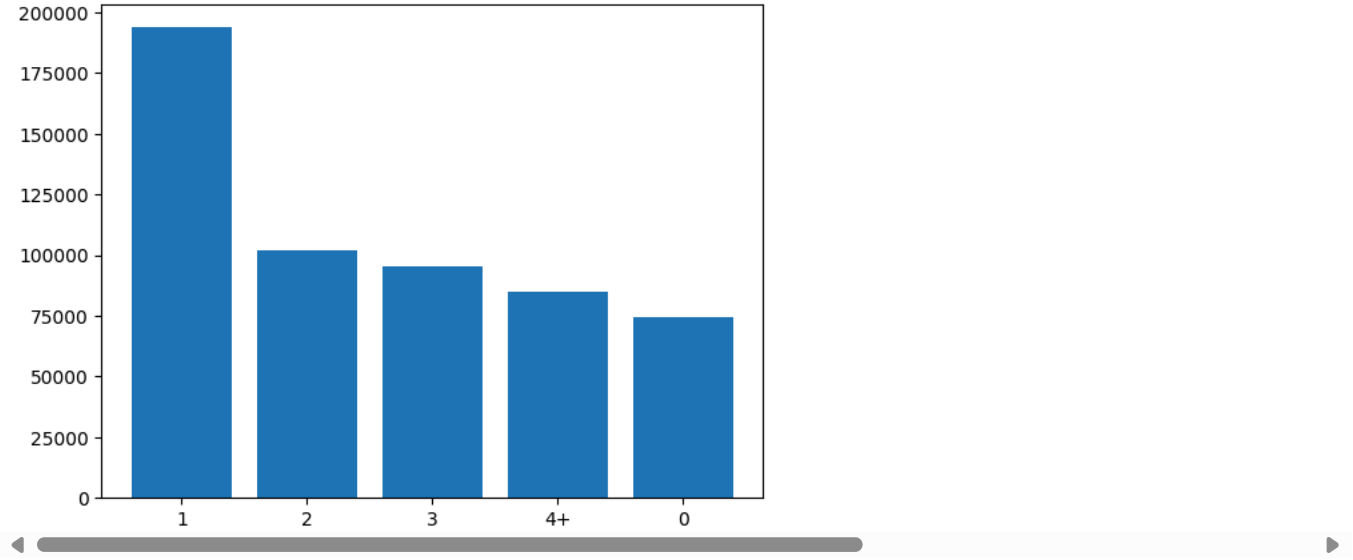
```
<BarContainer object of 3 artists>
```



The Customers from B category cities are more followed C and A.

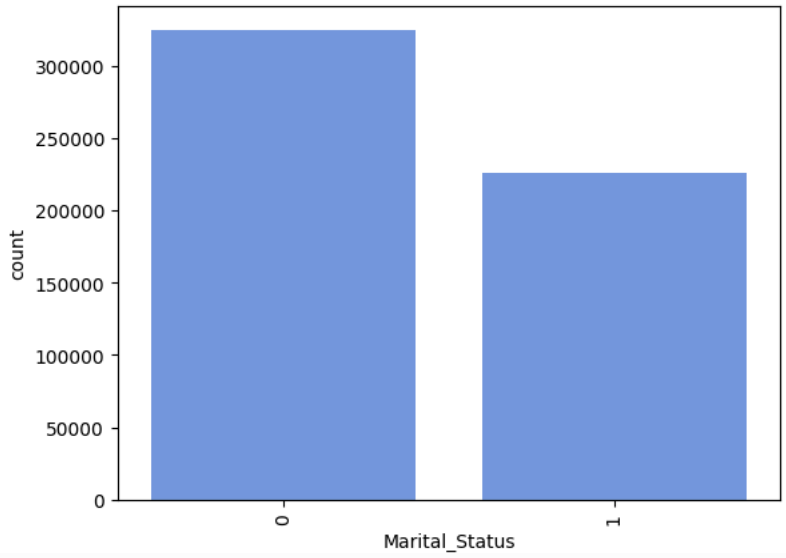
```
city_stay=data['Stay_In_Current_City_Years'].value_counts()
x_bar=city_stay.index
y_bar=city_stay
plt.bar(x_bar,y_bar)
```

```
<BarContainer object of 5 artists>
```



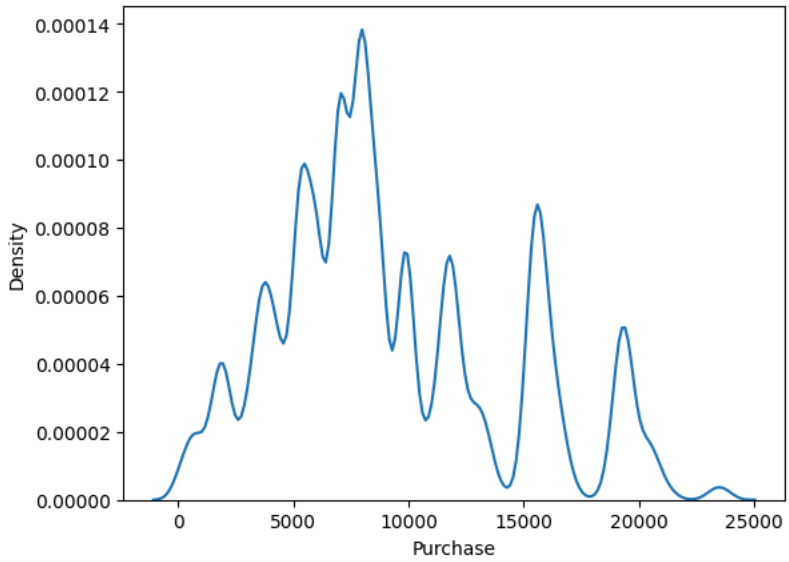

```
sns.countplot(x='Marital_Status', data=data, color='cornflowerblue')
plt.xticks(rotation=90)
```

```
([0, 1], [Text(0, 0, '0'), Text(1, 0, '1')])
```




```
sns.kdeplot(data['Purchase'])
```

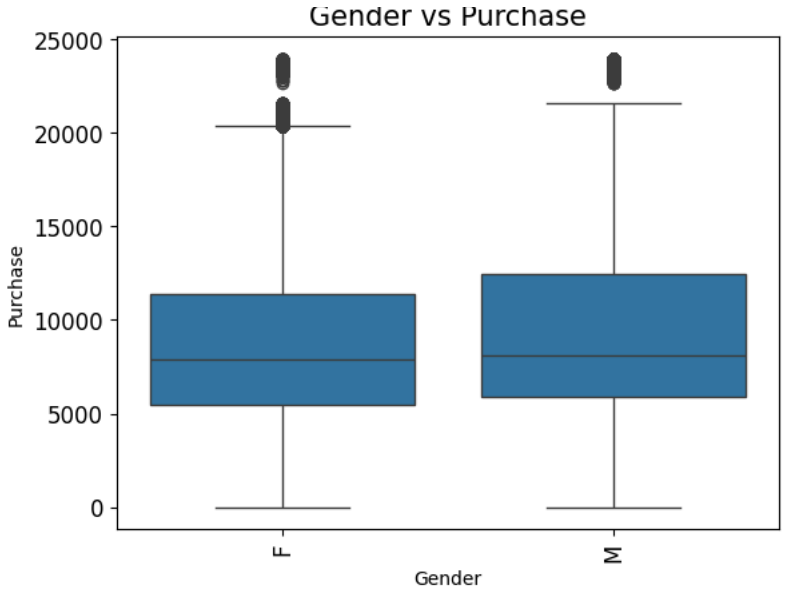
```
<Axes: xlabel='Purchase', ylabel='Density'>
```

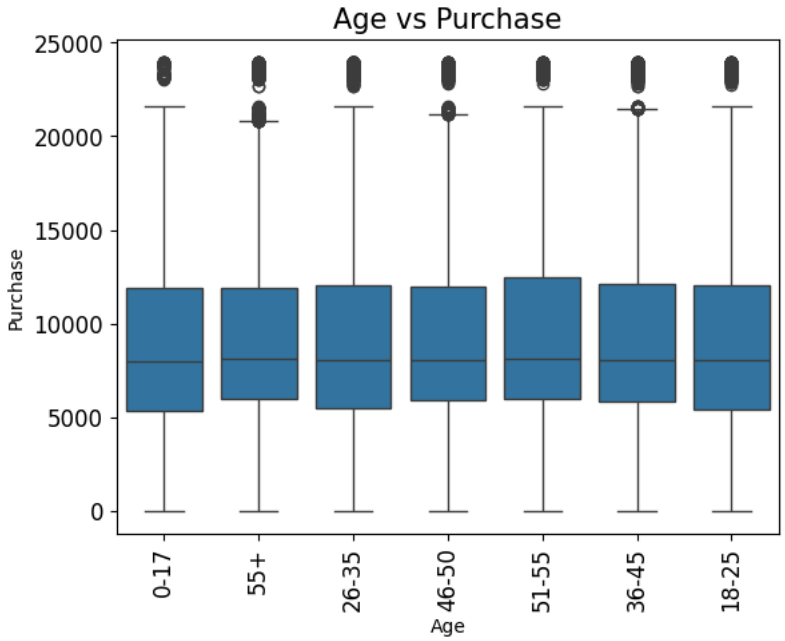


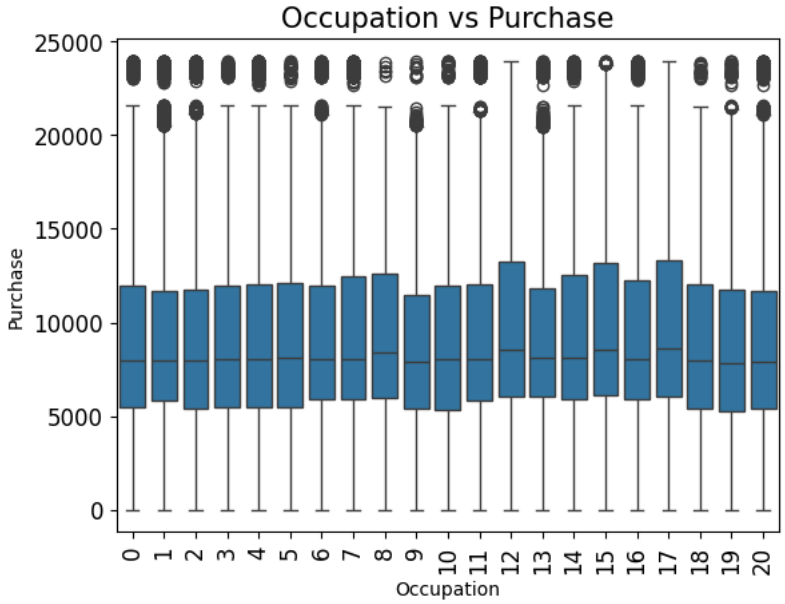
Bivariate Analysis

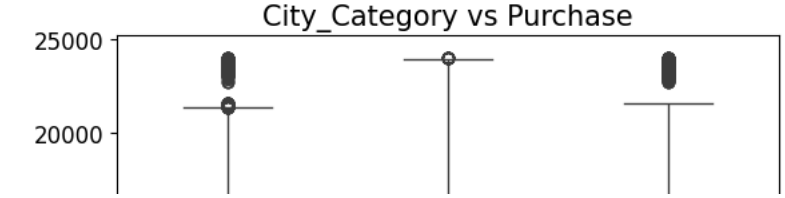
```
col = ['Gender', 'Age', 'Occupation', 'City_Category', 'Stay_In_Current_City_Years', 'Marital_Status']

for i in col:
  sns.boxplot(x=i, y='Purchase', data=data)
  plt.xticks(rotation=90,fontsize=12)
  plt.yticks(fontsize=12)
  plt.title(f'{i} vs Purchase', fontsize=15)
  plt.show()
```

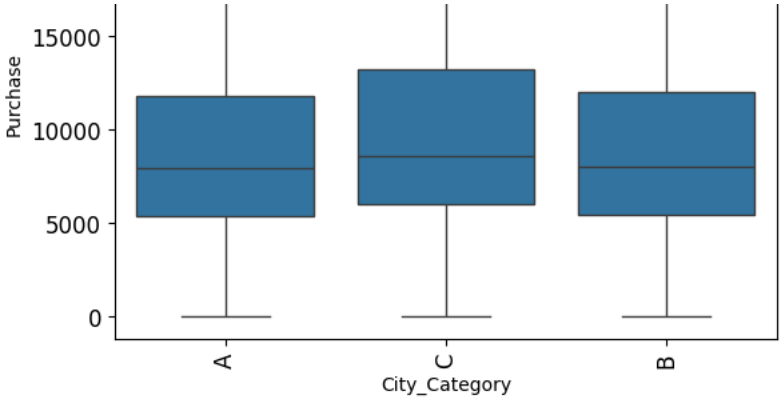
[4]

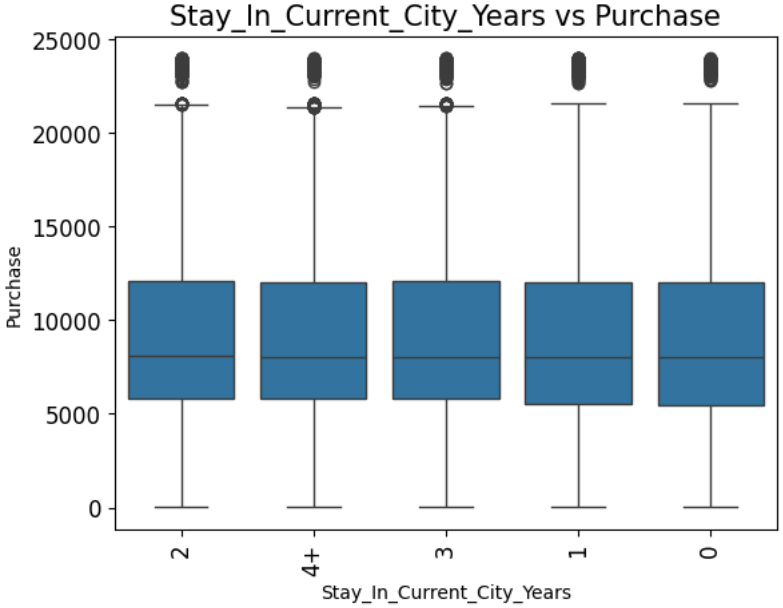

Gender vs Purchase
Purchase
25000
20000
15000
10000
5000
0
F
M
Gender

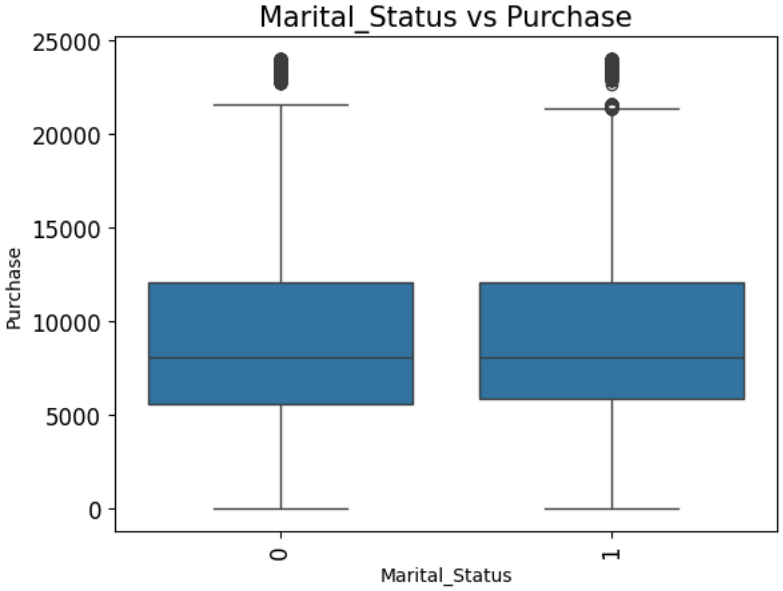

Age vs Purchase
Purchase
25000
20000
15000
10000
5000
0
0-17
55+
26-35
46-50
51-55
36-45
18-25
Age


Occupation vs Purchase
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

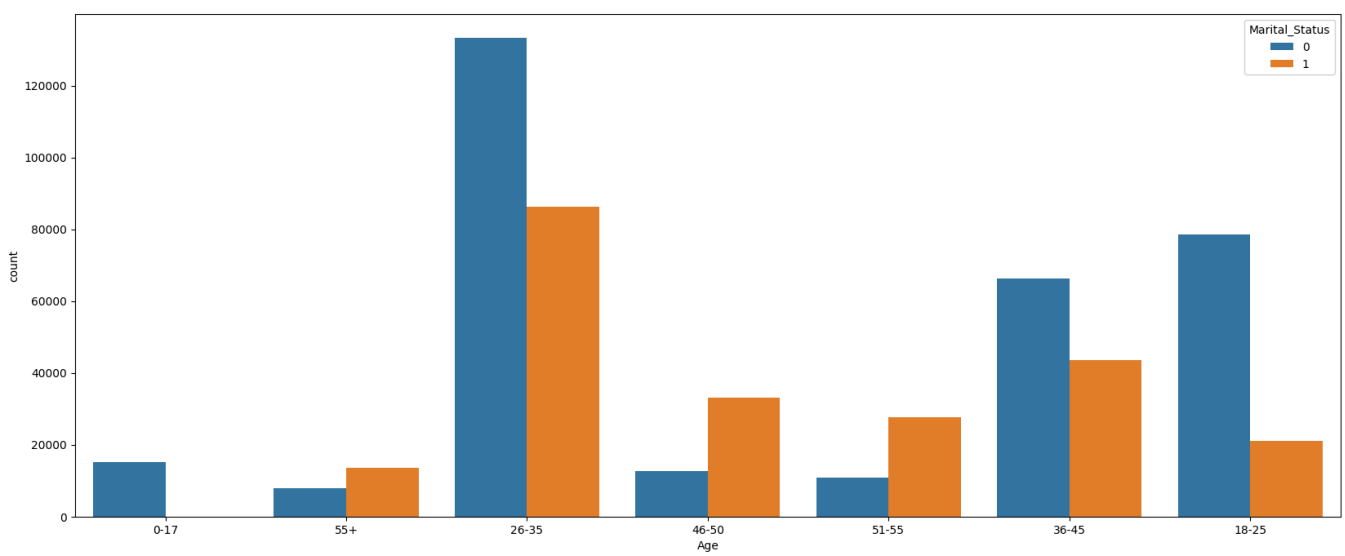

City_Category vs Purchase
25000
20000


15000
10000
5000
0
Purchase
A
C
B
City_Category

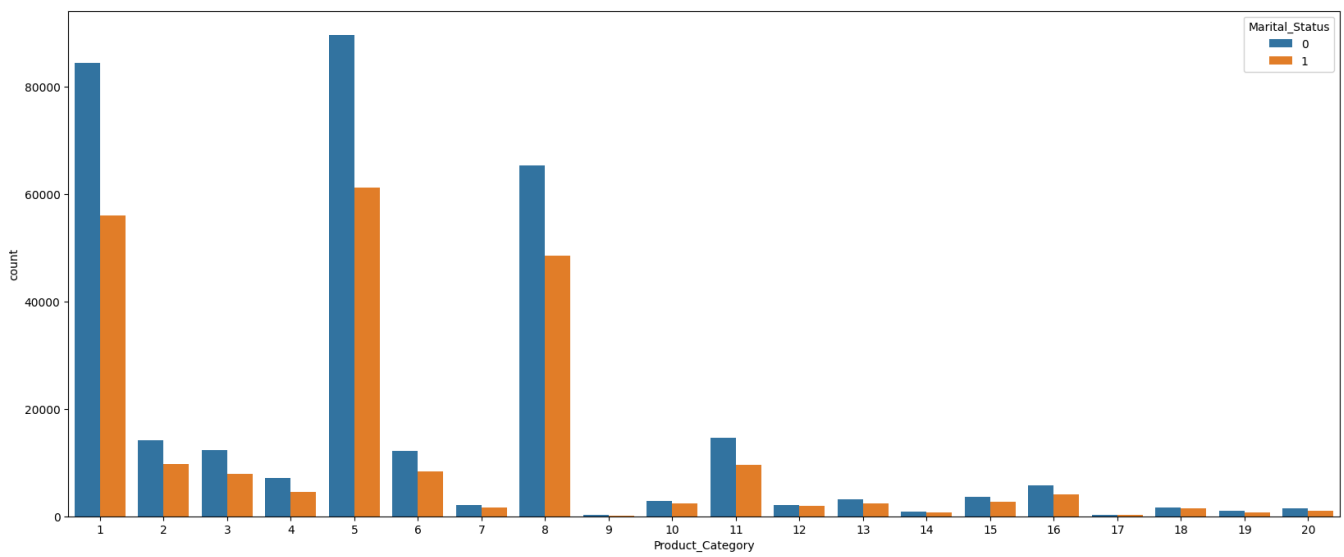

Stay_In_Current_City_Years vs Purchase
25000
20000
15000
10000
5000
0
Purchase
2
4+
3
1
0
Stay_In_Current_City_Years


Marital_Status vs Purchase
25000
20000
15000
10000
5000
0
Purchase
0
1
Marital_Status

```
plt.figure(figsize=(20, 8))
sns.countplot(data=data, x='Age',hue='Marital_Status')
plt.show()
```

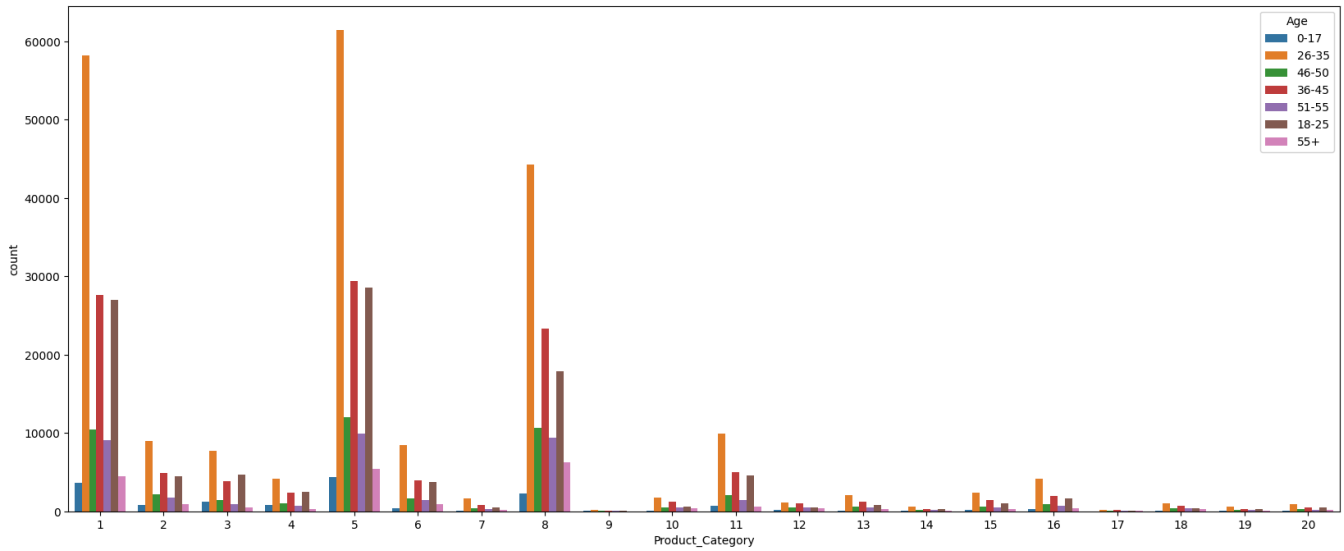



```
plt.figure(figsize=(20, 8))
sns.countplot(data=data, x='Product_Category',hue='Marital_Status')
plt.show()
```



Most of the purchases are done by males

```
plt.figure(figsize=(20, 8))
sns.countplot(data=data, x='Product_Category',hue='Age')
plt.show()
```



1, 5 and 8 were the most purchased product_category sold in Walmart.

```
data[col].melt().groupby(['variable', 'value'])['value'].count()*100/len(data)
```

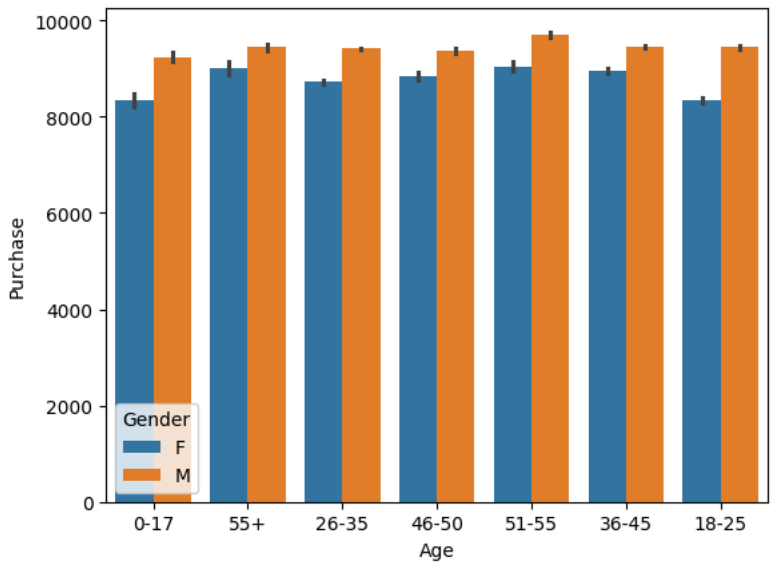
		value
variable	value	
Age	**0-17**	2.745479
	18-25	18.117760
	26-35	39.919974
	36-45	19.999891
	46-50	8.308246
	51-55	6.999316
	55+	3.909335
City_Category	**A**	26.854862
	B	42.026259
	C	31.118880
Gender	**F**	24.689493
	M	75.310507
Marital_Status	**0**	59.034701
	1	40.965299
Occupation	**0**	12.659889
	1	8.621843
	2	4.833584
	3	3.208694
	4	13.145284
	5	2.213726
	6	3.700452
	7	10.750125
	8	0.281056
	9	1.143677
	10	2.350618
	11	2.106285
	12	5.668208
	13	1.404917
	14	4.964659
	15	2.211545
	16	4.612339
	17	7.279645
	18	1.203851
	19	1.538173
	20	6.101427
Stay_In_Current_City_Years	**0**	13.525237
	1	35.235825
	2	18.513711
	3	17.322404
	4+	15.402823

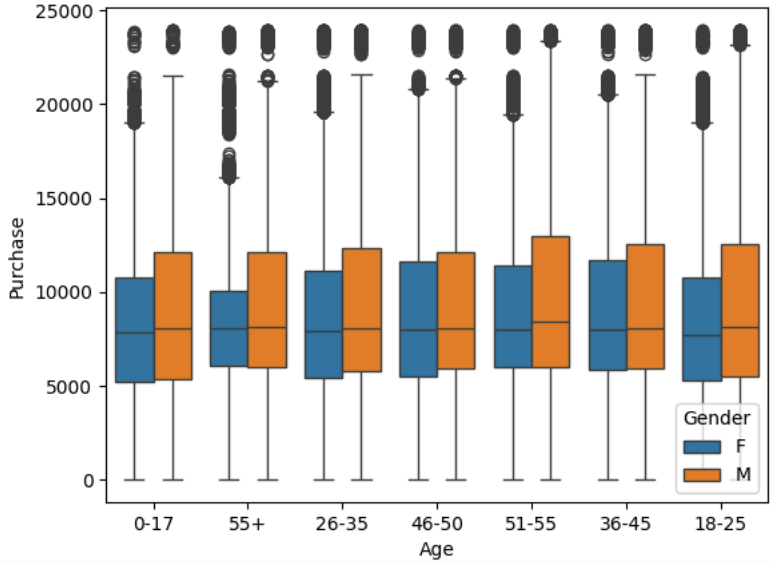
dtype: float64

Multi-variate Analysis

```
sns.barplot(data=data, x='Age', y='Purchase',hue="Gender")
plt.show()
```

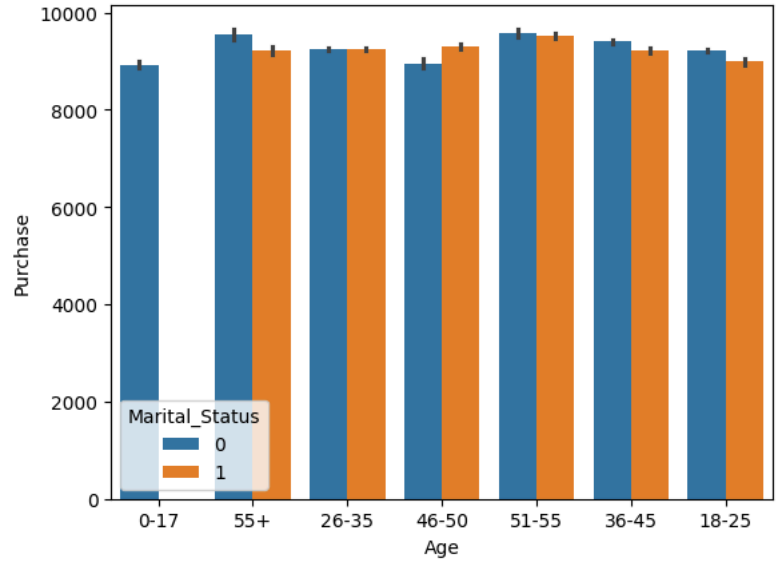
```
sns.boxplot(x='Age',y='Purchase',hue='Gender',data=data)
plt.show()
```

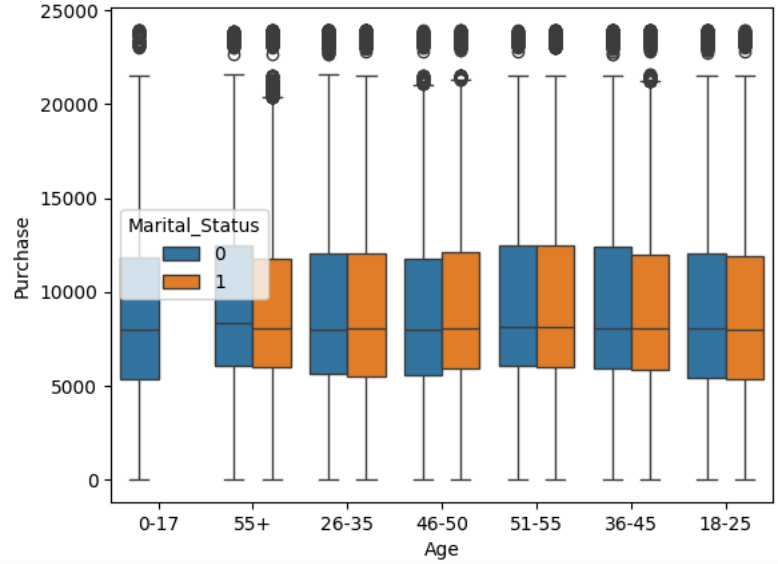





```
sns.barplot(data=data, x='Age', y='Purchase',hue="Marital_Status")
plt.show()
```

```
sns.boxplot(x='Age',y='Purchase',hue='Marital_Status',data=data)
plt.show()
```






```
data.describe(include='all').T
```

	count	unique	top	freq	mean	std	min	25%	50%	75%
User_ID	550068.0	NaN	NaN	NaN	1003028.842401	1727.591586	1000001.0	1001516.0	1003077.0	1004478.0
Product_ID	550068	3631	P00265242	1880	NaN	NaN	NaN	NaN	NaN	NaN
Gender	550068	2	M	414259	NaN	NaN	NaN	NaN	NaN	NaN
Age	550068	7	26-35	219587	NaN	NaN	NaN	NaN	NaN	NaN
Occupation	550068.0	NaN	NaN	NaN	8.076707	6.52266	0.0	2.0	7.0	14.0
City_Category	550068	3	B	231173	NaN	NaN	NaN	NaN	NaN	NaN
Stay_In_Current_City_Years	550068	5	1	193821	NaN	NaN	NaN	NaN	NaN	NaN
Marital_Status	550068.0	NaN	NaN	NaN	0.409653	0.49177	0.0	0.0	0.0	1.0
Product_Category	550068.0	NaN	NaN	NaN	5.40427	3.936211	1.0	1.0	5.0	8.0
Purchase	550068.0	NaN	NaN	NaN	9263.968713	5023.065394	12.0	5823.0	8047.0	12054.0

Distribution of Expenses based on Gender

```
data.groupby('Gender').describe()
```

	User_ID								Occupation		...	Product_Cate	
	count	mean	std	min	25%	50%	75%	max	count	mean	...	75%	max
Gender													
F	135809.0	1.003130e+06	1786.630589	1000001.0	1001569.0	1003159.0	1004765.0	1006039.0	135809.0	6.74054	...	8.0	
M	414259.0	1.002996e+06	1706.493873	1000002.0	1001505.0	1003041.0	1004411.0	1006040.0	414259.0	8.51475	...	8.0	

2 rows × 40 columns

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

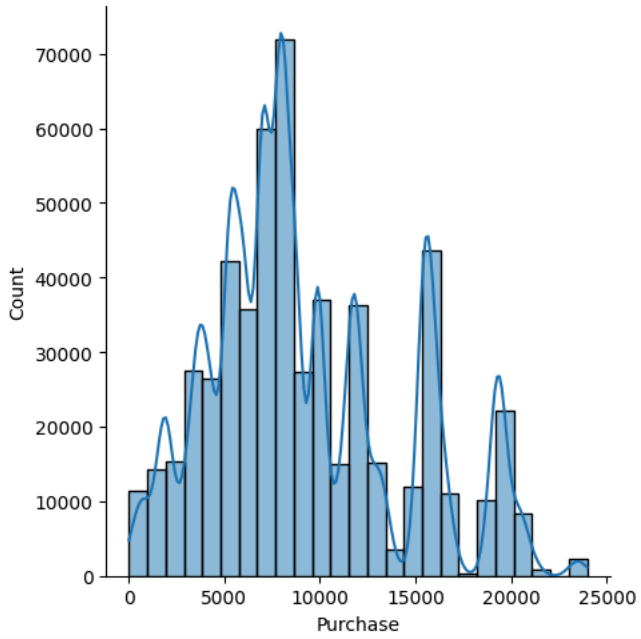
	count	mean	std	min	25%	50%	75%	max
Gender								
F	135809.0	8734.565765	4767.233289	12.0	5433.0	7914.0	11400.0	23959.0
M	414259.0	9437.526040	5092.186210	12.0	5863.0	8098.0	12454.0	23961.0

```
# Female Purchases
Female_data = data[data["Gender"]=="F"]
print("Female Purchase amount Mean:"+ str(Female_data["Purchase"].mean()))
print("Female Purchase amount SD:"+ str(Female_data["Purchase"].std()))

# Male Purchases
Male_data = data[data["Gender"]=="M"]
print("Male Purchase amount Mean:"+ str(Male_data["Purchase"].mean()))
print("Male Purchase amount SD:"+ str(Male_data["Purchase"].std()))
```

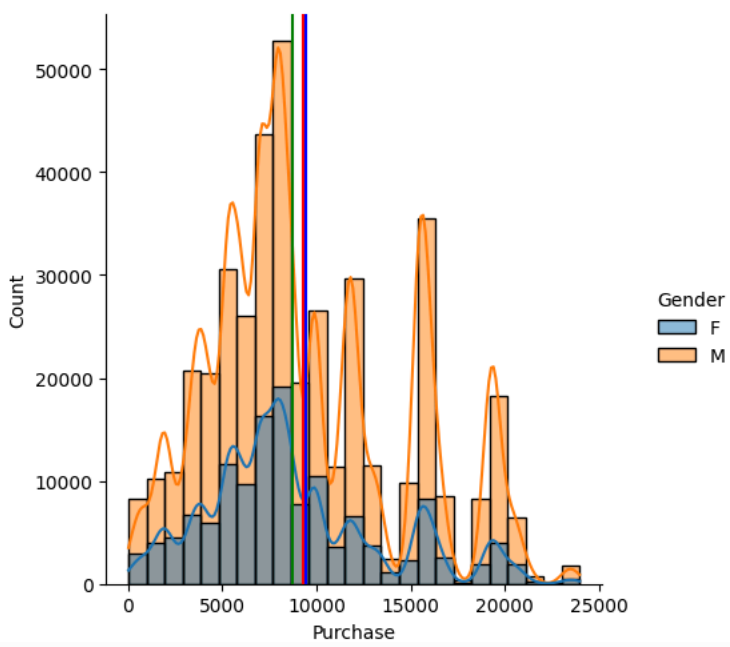
```
Female Purchase amount Mean:8734.565765155476
Female Purchase amount SD:4767.233289291444
Male Purchase amount Mean:9437.526040472265
Male Purchase amount SD:5092.186209777949
```

```
sns.displot(x='Purchase', bins=25, kde=True, data=data )
```

```
<seaborn.axisgrid.FacetGrid at 0x7e0734de3880>
```




```
sns.displot(x='Purchase', bins=25, kde=True, hue='Gender', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Gender']=='M']['Purchase'].mean(), color='b')
plt.axvline(x=data[data['Gender']=='F']['Purchase'].mean(), color='g')
plt.show()
```



Almost male and female means coincide with overall means.

Exploring data using CLT.

```
data.sample(300).groupby('Gender')['Purchase'].describe()
```

Gender	count	mean	std	min	25%	50%	75%	max
F	69.0	8855.246377	5568.838694	480.0	5438.0	7985.0	10699.0	23365.0
M	231.0	9372.939394	5257.901485	12.0	5652.5	8008.0	12303.0	23775.0

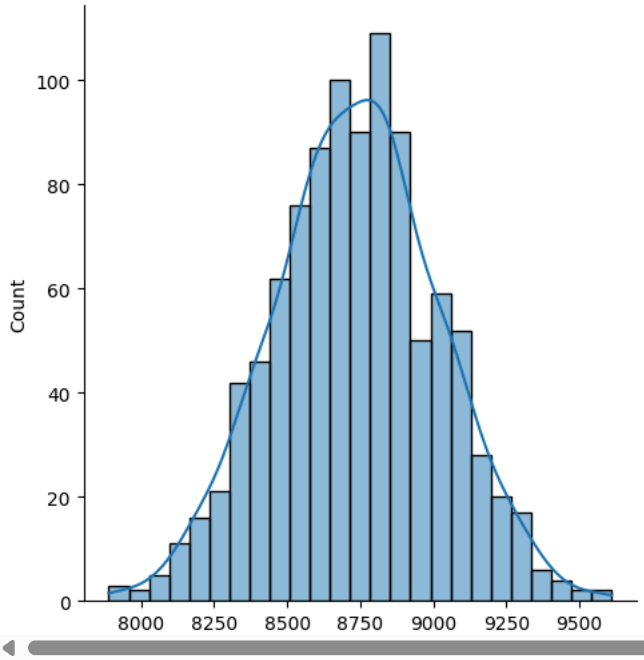
```
data.sample(00).groupby('Gender')['Purchase'].describe()
```

count	mean	std	min	25%	50%	75%	max

```
female_sample_means_for_300=[Female_data.sample(300, replace=True)['Purchase'].mean() for i in range(1000)]
```

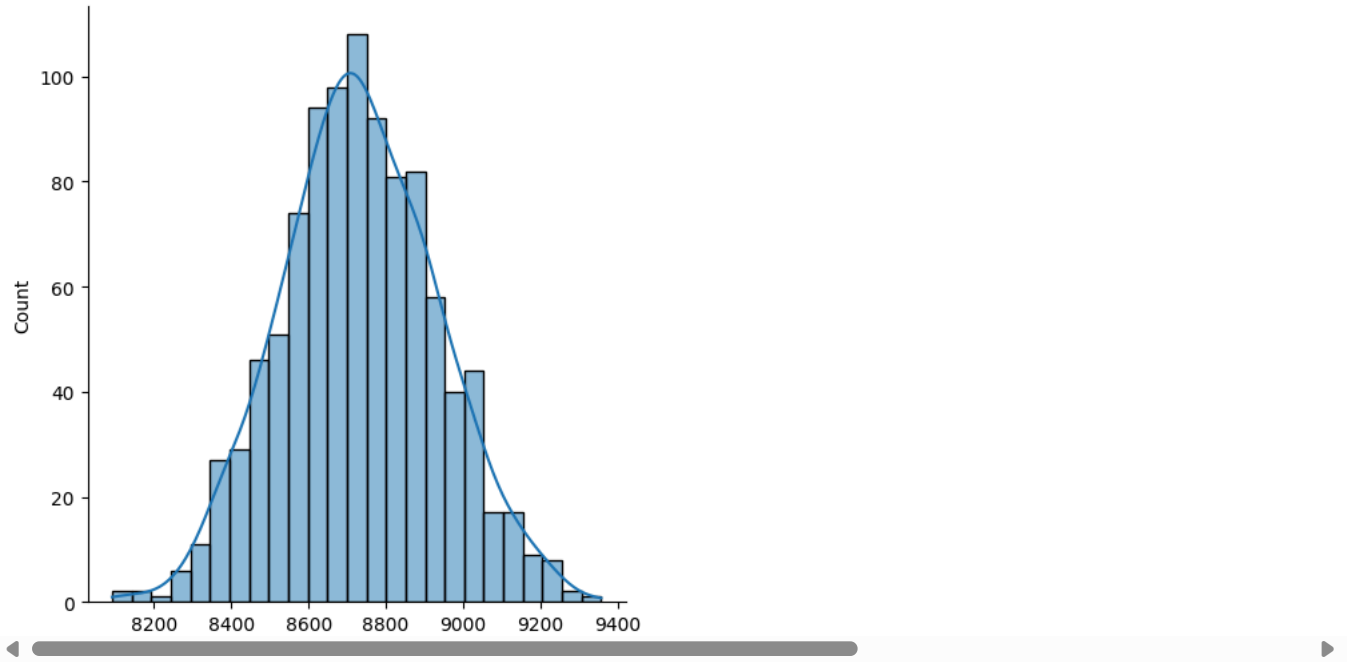
```
sns.displot( bins=25, kde=True,  data=female_sample_means_for_300 )
```

```
<seaborn.axisgrid.FacetGrid at 0x7e07321e9840>
```



```
female_sample_means_for_600=[Female_data.sample(600, replace=True)['Purchase'].mean() for i in range(1000)]
sns.displot( bins=25, kde=True,  data=female_sample_means_for_600 )
```

```
<seaborn.axisgrid.FacetGrid at 0x7e073618c6a0>
```

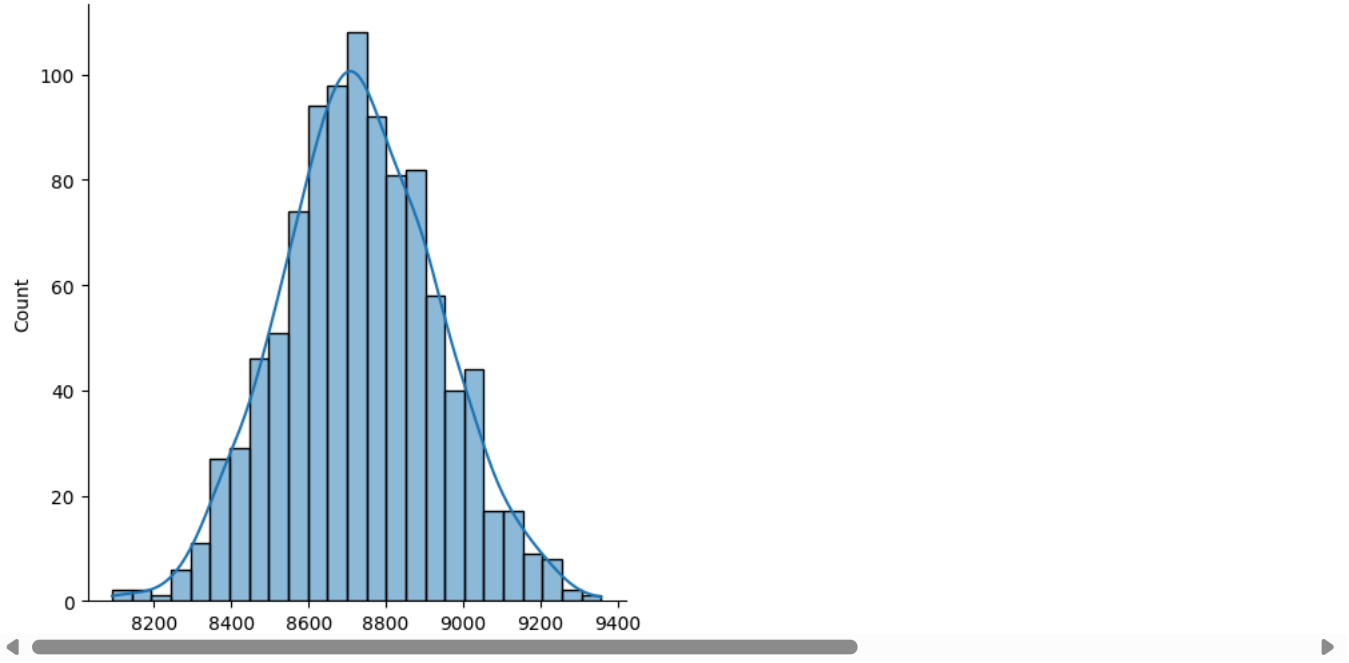



```
np.mean(female_sample_means_for_600)
```

```
8732.294033333334
```

```
male_sample_means_for_600=[Male_data.sample(600, replace=True)['Purchase'].mean() for i in range(1000)]
sns.displot( bins=25, kde=True,  data=female_sample_means_for_600 )
```

```
<seaborn.axisgrid.FacetGrid at 0x7e07361a0850>
```




```
np.mean(male_sample_means_for_600)
```

```
9443.043593333334
```

```
f_stderror=np.std(female_sample_means_for_600)/np.sqrt(1000)
```

```
m_stderror=np.std(male_sample_means_for_600)/np.sqrt(1000)
```

```
norm.interval(confidence=.95,loc=np.mean(female_sample_means_for_600),scale=f_stderror)
```

```
(8719.866794165076, 8744.72127250159)
```

```
norm.interval(confidence=.95,loc=np.mean(male_sample_means_for_600),scale=m_stderror)
```

```
(9430.085888350462, 9456.001298316205)
```

The confidence interval of the sample average of male and female does not coincide.

Distribution of Expenses based on Maritial Status

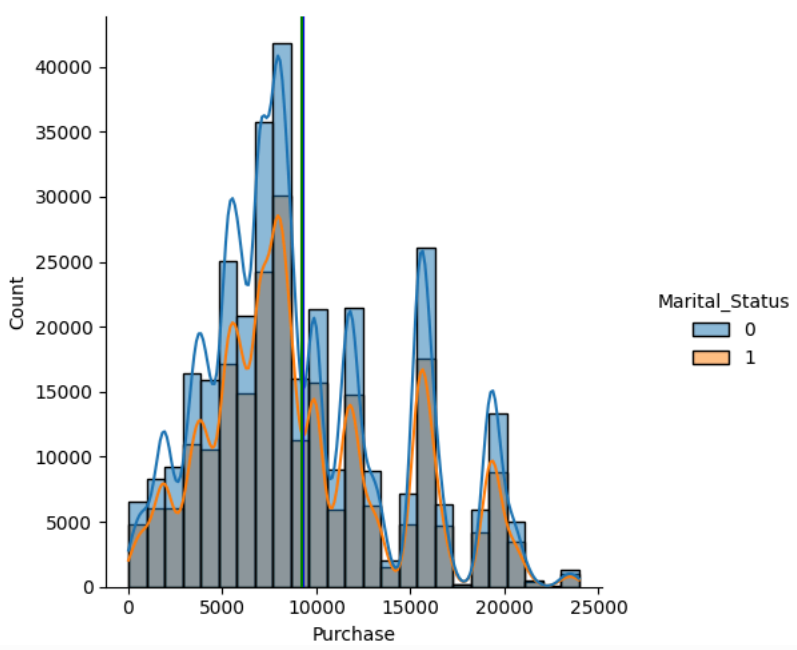
```
data.groupby('Marital_Status')['Purchase'].describe()
```

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

```
# Single Purchases
Single_data = data[data["Marital_Status"]==0]
print("Single Purchase amount Mean:"+ str(Single_data["Purchase"].mean()))
print("Single Purchase amount SD:"+ str(Single_data["Purchase"].std()))

# Married Purchases
Married_data = data[data["Marital_Status"]==1]
print("Married Purchase amount Mean:"+ str(Married_data["Purchase"].mean()))
print("Married Purchase amount SD:"+ str(Married_data["Purchase"].std()))
```

```
Single Purchase amount Mean:9265.907618921507
Single Purchase amount SD:5027.347858674457
Married Purchase amount Mean:9261.174574082374
Married Purchase amount SD:5016.89737779313
```

```
sns.displot(x='Purchase', bins=25, kde=True, hue='Marital_Status', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Marital_Status']==0]['Purchase'].mean(), color='b')
plt.axvline(x=data[data['Marital_Status']==1]['Purchase'].mean(), color='g')
plt.show()
```

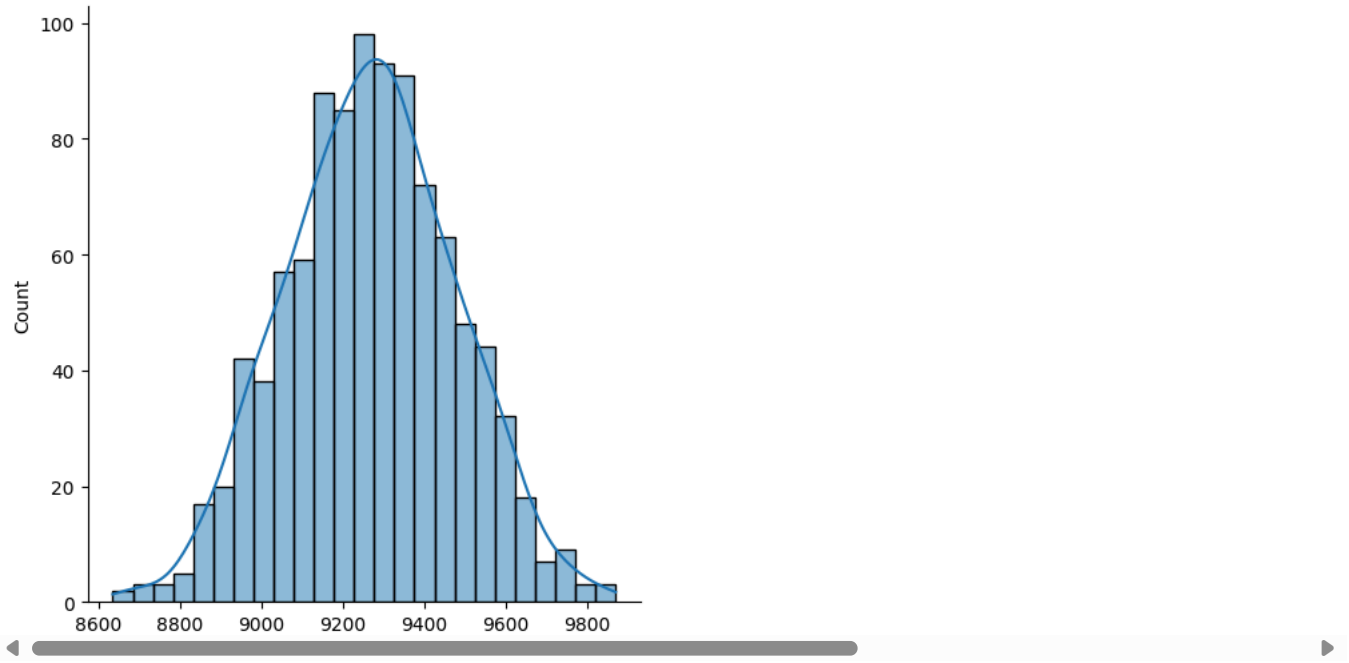



```
data.sample(600).groupby('Marital_Status')['Purchase'].describe()
```

Marital_Status	count	mean	std	min	25%	50%	75%	max
0	346.0	9375.910405	4965.922135	191.0	5942.25	8615.5	11903.75	23809.0
1	254.0	9423.377953	4878.291733	566.0	5977.75	8138.0	12064.25	23262.0

```
single_sample_means_for_600=[Single_data.sample(600, replace=True)['Purchase'].mean() for i in range(1000)]
sns.displot( bins=25, kde=True,  data=single_sample_means_for_600 )
```

```
<seaborn.axisgrid.FacetGrid at 0x7e07338154e0>
```

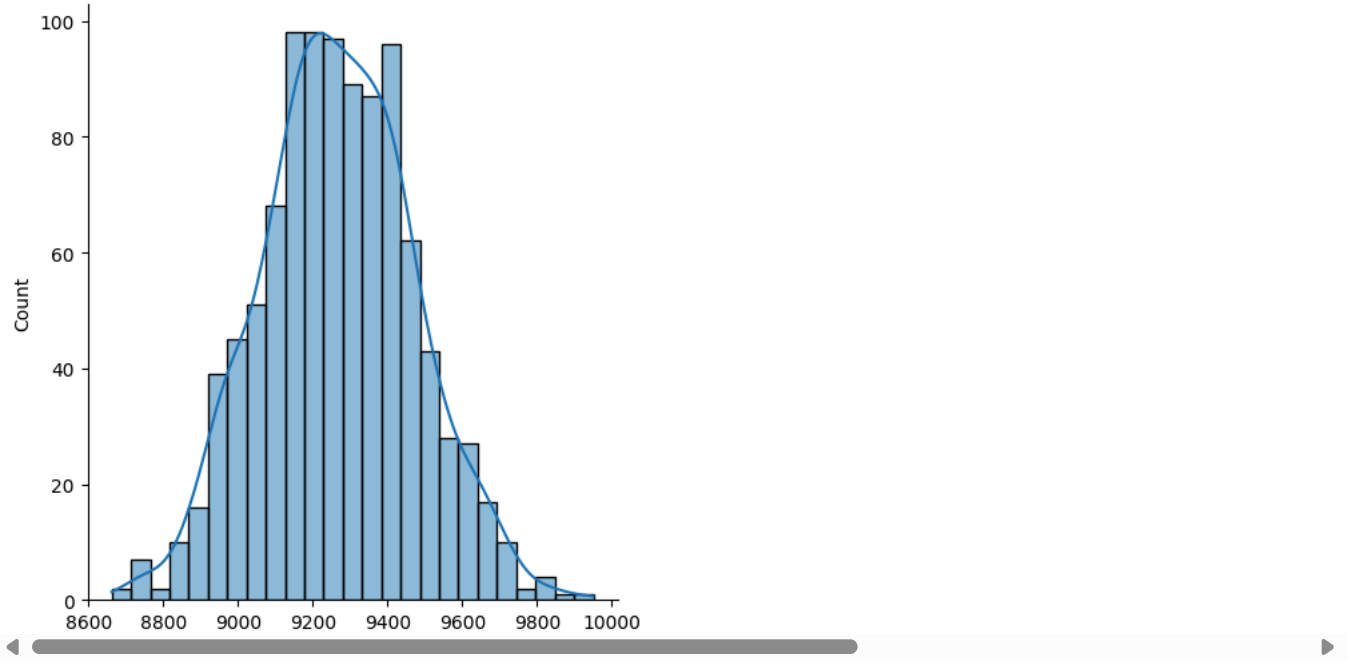



```
np.mean(single_sample_means_for_600)
```

```
9268.45226
```

```
married_sample_means_for_600=[Married_data.sample(600, replace=True)['Purchase'].mean() for i in range(1000)]
sns.displot( bins=25, kde=True,  data=married_sample_means_for_600 )
```

```
<seaborn.axisgrid.FacetGrid at 0x7e072f6001c0>
```




```
np.mean(married_sample_means_for_600)
```

```
9266.183008333335
```

```
s_stderror=np.std(single_sample_means_for_600)/np.sqrt(1000)
```

```
m_stderror=np.std(married_sample_means_for_600)/np.sqrt(1000)
```

```
norm.interval(confidence=.95,loc=np.mean(single_sample_means_for_600),scale=s_stderror)
```

```
(9255.62389375113, 9281.28062624887)
```

```
norm.interval(confidence=.95,loc=np.mean(married_sample_means_for_600),scale=m_stderror)
```

```
(9253.660797812818, 9278.705218853853)
```

Average of both sample means taking Maritial Status as a account coincides each other.

Distribution of Expenses based on Age

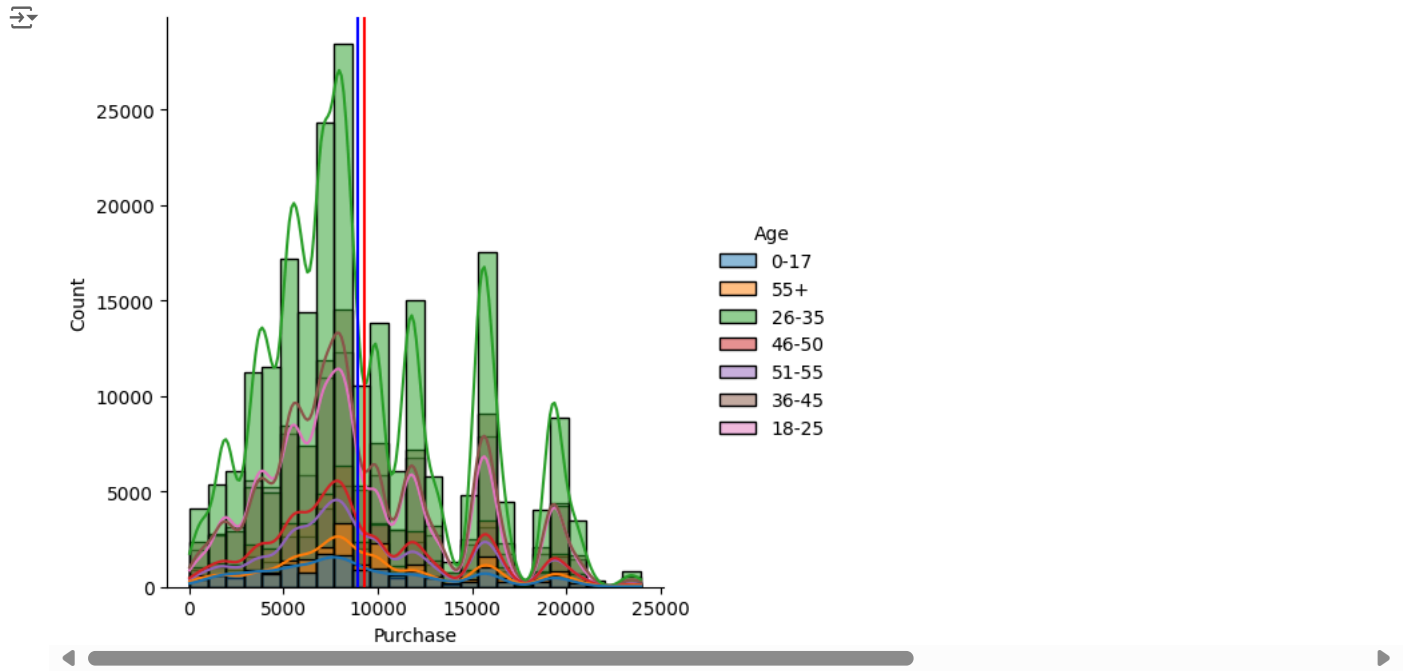
```
data.groupby('Age')['Purchase'].describe()
```

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

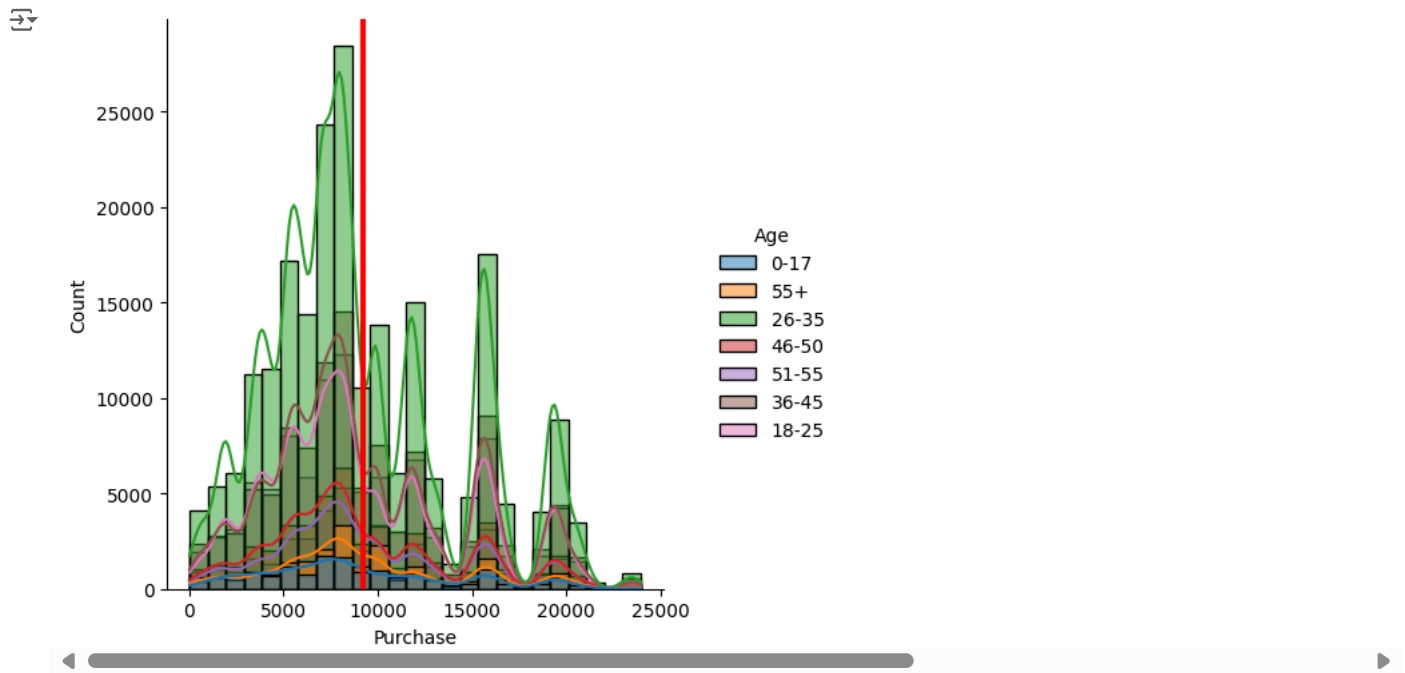
```
# Single Purchases
Age_grp1 = data[data["Age"]=='0-17']
print("Age_grp1 Purchase amount Mean:"+ str(Age_grp1["Purchase"].mean()))
print("Age_grp1 Purchase amount SD:"+ str(Age_grp1["Purchase"].std()))

# Married Purchases
Age_grp2 = data[data["Age"]=='18-25']
print("Age_grp2 Purchase amount Mean:"+ str(Age_grp2["Purchase"].mean()))
print("Age_grp2 Purchase amount SD:"+ str(Age_grp2["Purchase"].std()))

# Married Purchases
Age_grp3 = data[data["Age"]=='26-35']
print("Age_grp3 Purchase amount Mean:"+ str(Age_grp3["Purchase"].mean()))
print("Age_grp3 Purchase amount SD:"+ str(Age_grp3["Purchase"].std()))

# Married Purchases
Age_grp4 = data[data["Age"]=='36-45']
print("Age_grp4 Purchase amount Mean:"+ str(Age_grp4["Purchase"].mean()))
print("Age_grp4 Purchase amount SD:"+ str(Age_grp4["Purchase"].std()))

# Married Purchases
Age_grp5 = data[data["Age"]=='46-50']
print("Age_grp5 Purchase amount Mean:"+ str(Age_grp5["Purchase"].mean()))
print("Age_grp5 Purchase amount SD:"+ str(Age_grp5["Purchase"].std()))

# Married Purchases
Age_grp6 = data[data["Age"]=='51-55']
print("Age_grp6 Purchase amount Mean:"+ str(Age_grp6["Purchase"].mean()))
print("Age_grp6 Purchase amount SD:"+ str(Age_grp6["Purchase"].std()))

# Married Purchases
Age_grp7 = data[data["Age"]=='55+']
print("Age_grp7 Purchase amount Mean:"+ str(Age_grp7["Purchase"].mean()))
print("Age_grp7 Purchase amount SD:"+ str(Age_grp7["Purchase"].std()))
```

```
Age_grp1 Purchase amount Mean:8933.464640444974
Age_grp1 Purchase amount SD:5111.11404600277
Age_grp2 Purchase amount Mean:9169.663606261289
Age_grp2 Purchase amount SD:5034.321997176577
Age_grp3 Purchase amount Mean:9252.690632869888
Age_grp3 Purchase amount SD:5010.527303002927
Age_grp4 Purchase amount Mean:9331.350694917874
Age_grp4 Purchase amount SD:5022.923879204652
Age_grp5 Purchase amount Mean:9208.625697468327
Age_grp5 Purchase amount SD:4967.216367142921
Age_grp6 Purchase amount Mean:9534.808030960236
Age_grp6 Purchase amount SD:5087.368079602116
Age_grp7 Purchase amount Mean:9336.280459449405
Age_grp7 Purchase amount SD:5011.493995603418
```

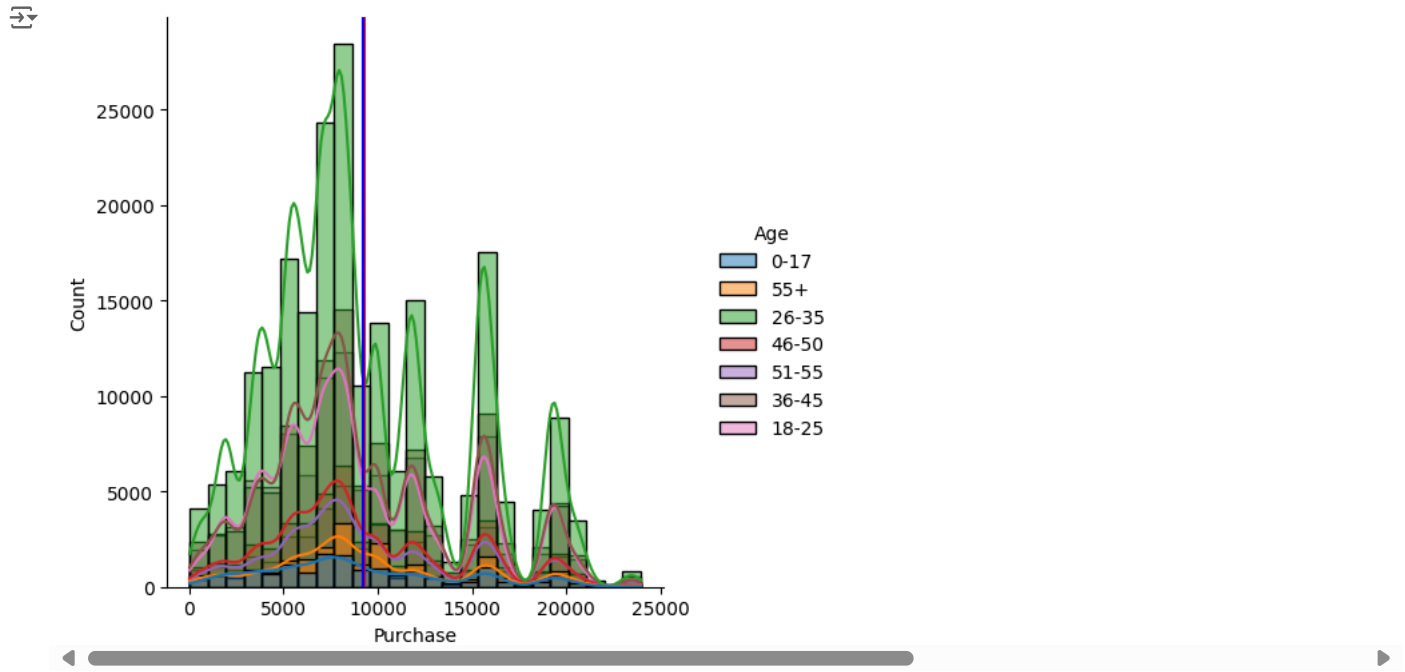
```
sns.displot(x='Purchase', bins=25, kde=True, hue='Age', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Age']=='0-17']['Purchase'].mean(), color='b')
plt.show()
```



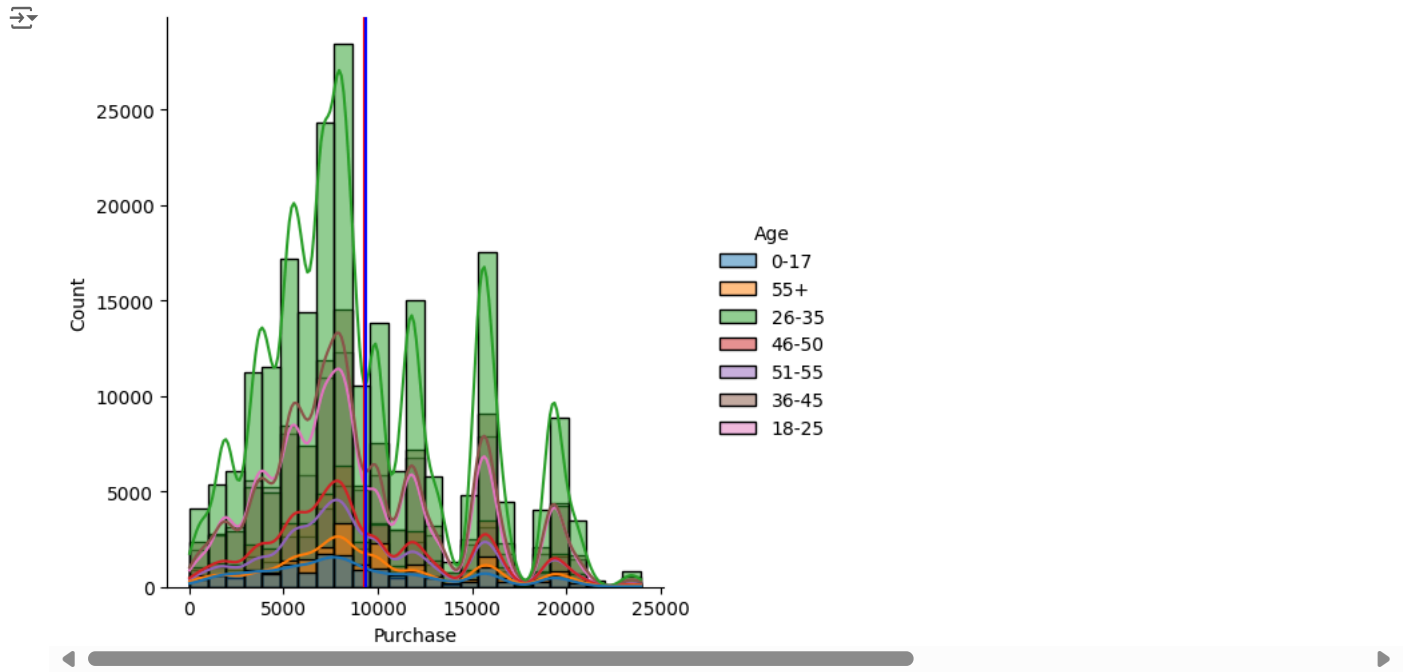

```
sns.displot(x='Purchase', bins=25, kde=True, hue='Age', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Age']=='18-25']['Purchase'].mean(), color='r')
plt.show()
```



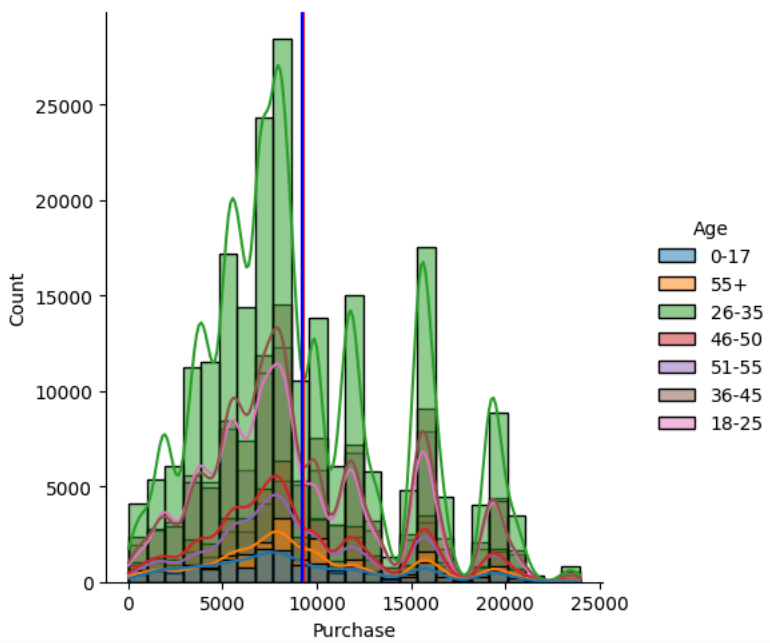

```
sns.displot(x='Purchase', bins=25, kde=True, hue='Age', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Age']=='26-35']['Purchase'].mean(), color='b')
plt.show()
```



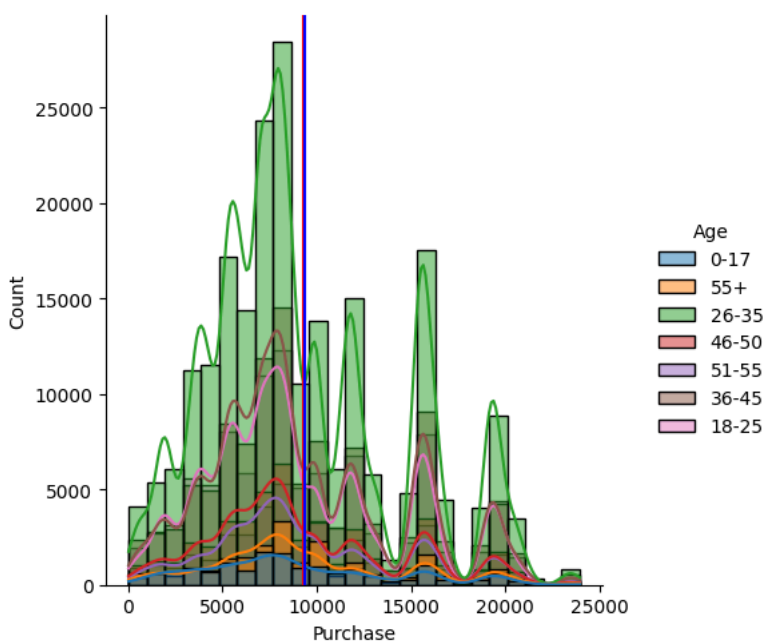

```
sns.displot(x='Purchase', bins=25, kde=True, hue='Age', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Age']=='36-45']['Purchase'].mean(), color='b')
plt.show()
```




```
sns.displot(x='Purchase', bins=25, kde=True, hue='Age', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Age']=='46-50']['Purchase'].mean(), color='b')
plt.show()
```




```
sns.displot(x='Purchase', bins=25, kde=True, hue='Age', data=data )
plt.axvline(x=data['Purchase'].mean(), color='r')
plt.axvline(x=data[data['Age']=='55+']['Purchase'].mean(), color='b')
plt.show()
```




```
Age_grp1_means_for_600=[Age_grp1.sample(600, replace=True)['Purchase'].mean() for i in range(1000)]
sns.displot( bins=25, kde=True,  data=married_sample_means_for_600 )
```