```
!wget https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/ori
```

```
--2025-01-20 14:34:23--  https://d2beiqkhq929f0.cloudfront.net/public_asset
Resolving d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)...
Connecting to d2beiqkhq929f0.cloudfront.net (d2beiqkhq929f0.cloudfront.net)
HTTP request sent, awaiting response... 200 OK
Length: 23027994 (22M) [text/plain]
Saving to: ‘walmart_data.csv?1641285094.1’

walmart_data.csv?16 100%[===================>]  21.96M  42.5MB/s    in 0.5s

2025-01-20 14:34:24 (42.5 MB/s) - ‘walmart_data.csv?1641285094.1’ saved [23
```

```
import pandas as pd
df_orig=pd.read_csv('walmart_data.csv?1641285094')
df=df_orig
```

```
#Importing libraries
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import norm
```

```
df.head()
```

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Curre
0	1000001	P00069042	F	0-17	10	A	
1	1000001	P00248942	F	0-17	10	A	
2	1000001	P00087842	F	0-17	10	A	

```
#1. Import the dataset and do usual data analysis steps like checking the struc
#a The data type of all columns in the “customers” table.

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
```

```
#b find the number of rows and columns given in the dataset

df.shape
```

```
(550068, 10)
```

550068 Rows and 10 Columns

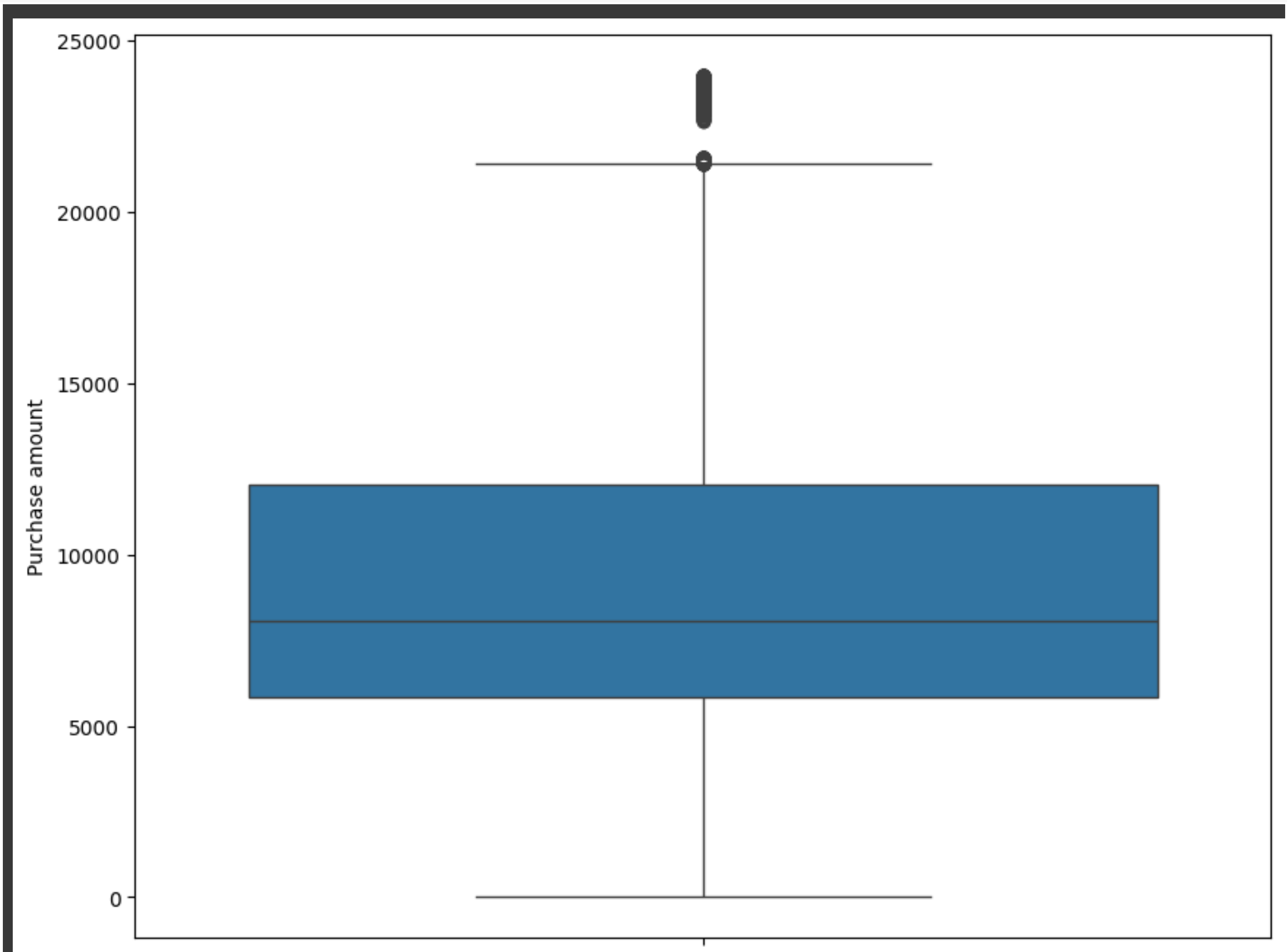
```
#c Check for the missing values and find the number of missing values in each c

df.isna().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

There are no missing values in any of the columns.

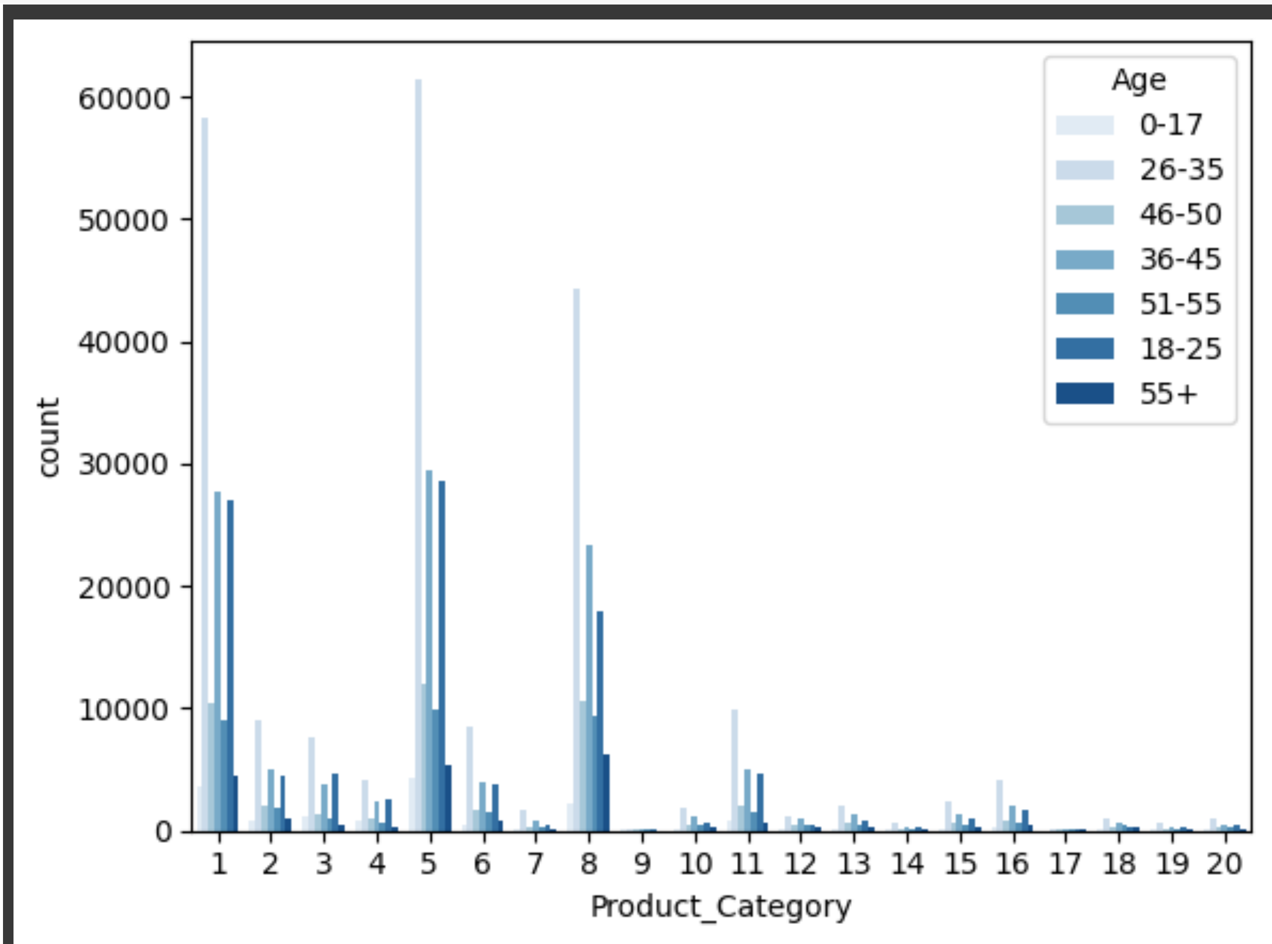
```
#2 Detect Null values and outliers

#a Find the outliers for every continuous variable in the dataset

plt.figure(figsize=[10,8])
plt.ylabel('Purchase amount')
sns.boxplot(y='Purchase',data=df)
plt.show()
```



As we can see , purchase amount above 22k or 23k are the outliers

```
#b. Remove/clip the data between the 5 percentile and 95 percentile

perc_5 = np.percentile(df['Purchase'],5)   #5th Percentile
perc_95 = np.percentile(df['Purchase'],95)  #95th Percentile

df['Purchase'] = np.clip(df['Purchase'],perc_5,perc_95)
```

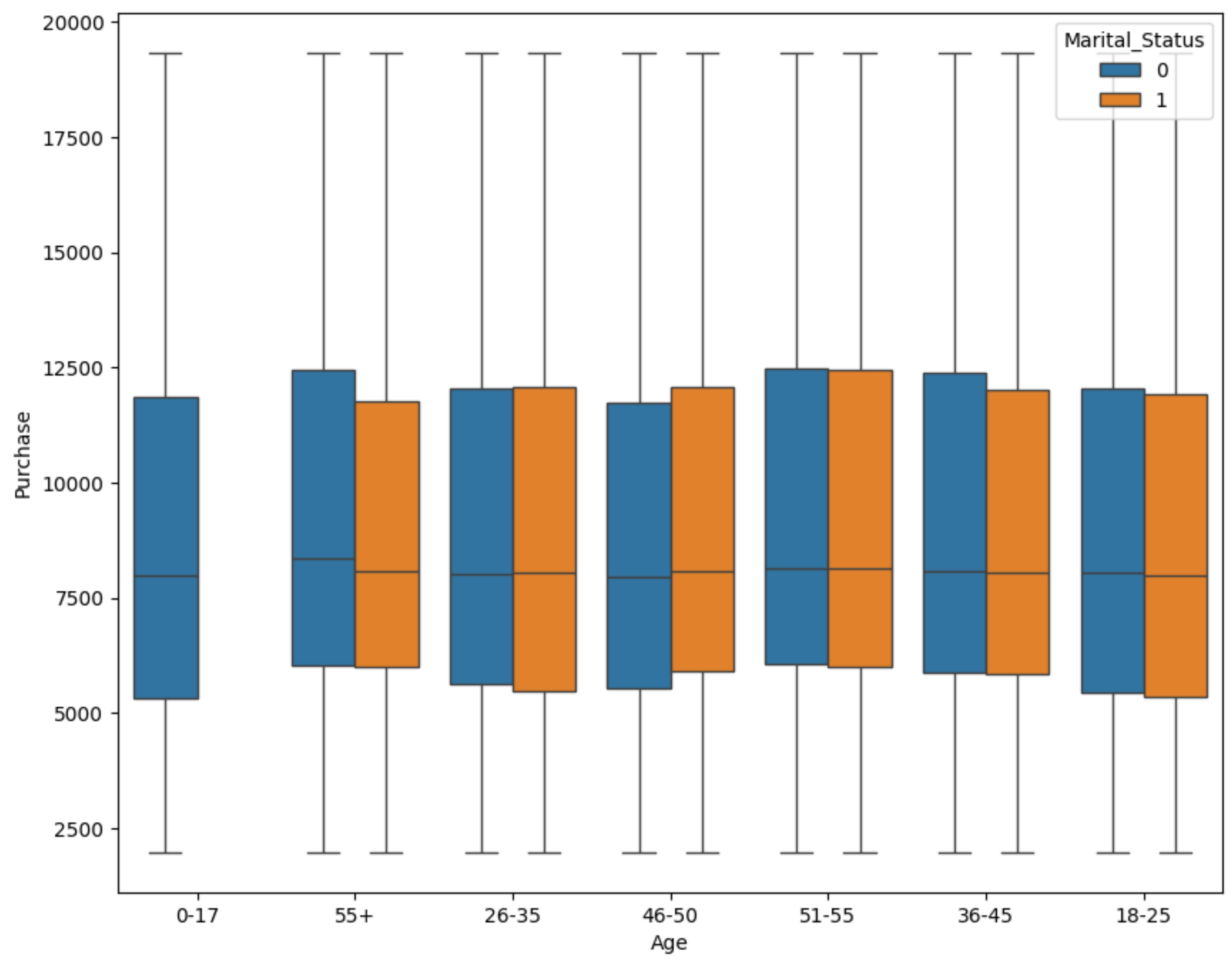
```
#3. Data Exploration

#a. What products are different age groups buying?
sns.countplot(x=df['Product_Category'],hue=df['Age'],palette='Blues')
plt.show()
```



Product 1,5 and 8 is being preferred most by age group 26-35

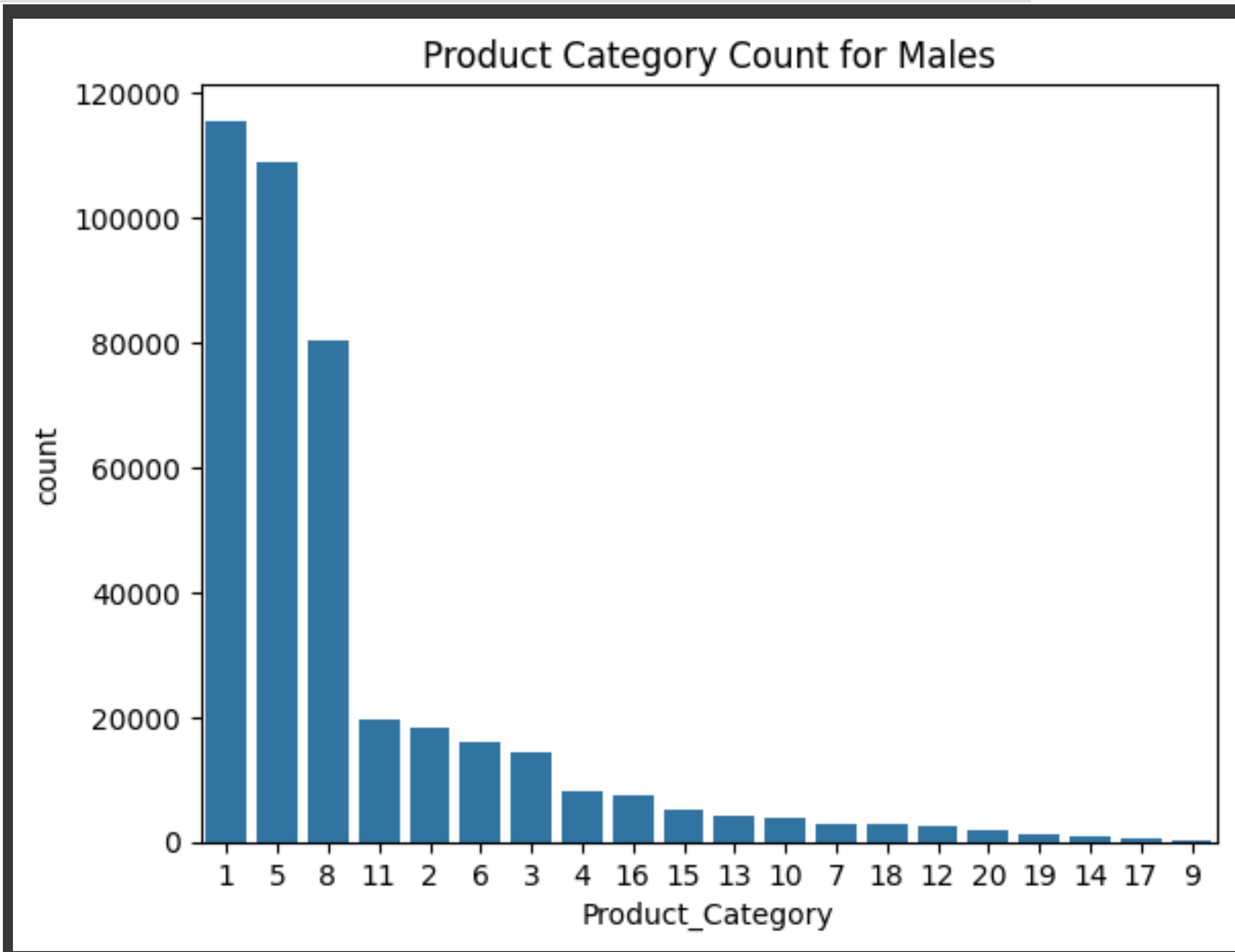
```
#b. Is there a relationship between age, marital status, and the amount spent?

plt.figure(figsize=[10,8])
sns.boxplot(x='Age',y='Purchase',hue='Marital_Status',data=df)
plt.legend(title='Marital_Status', loc='upper right')
```

```
<matplotlib.legend.Legend at 0x7f11e7195d10>
```



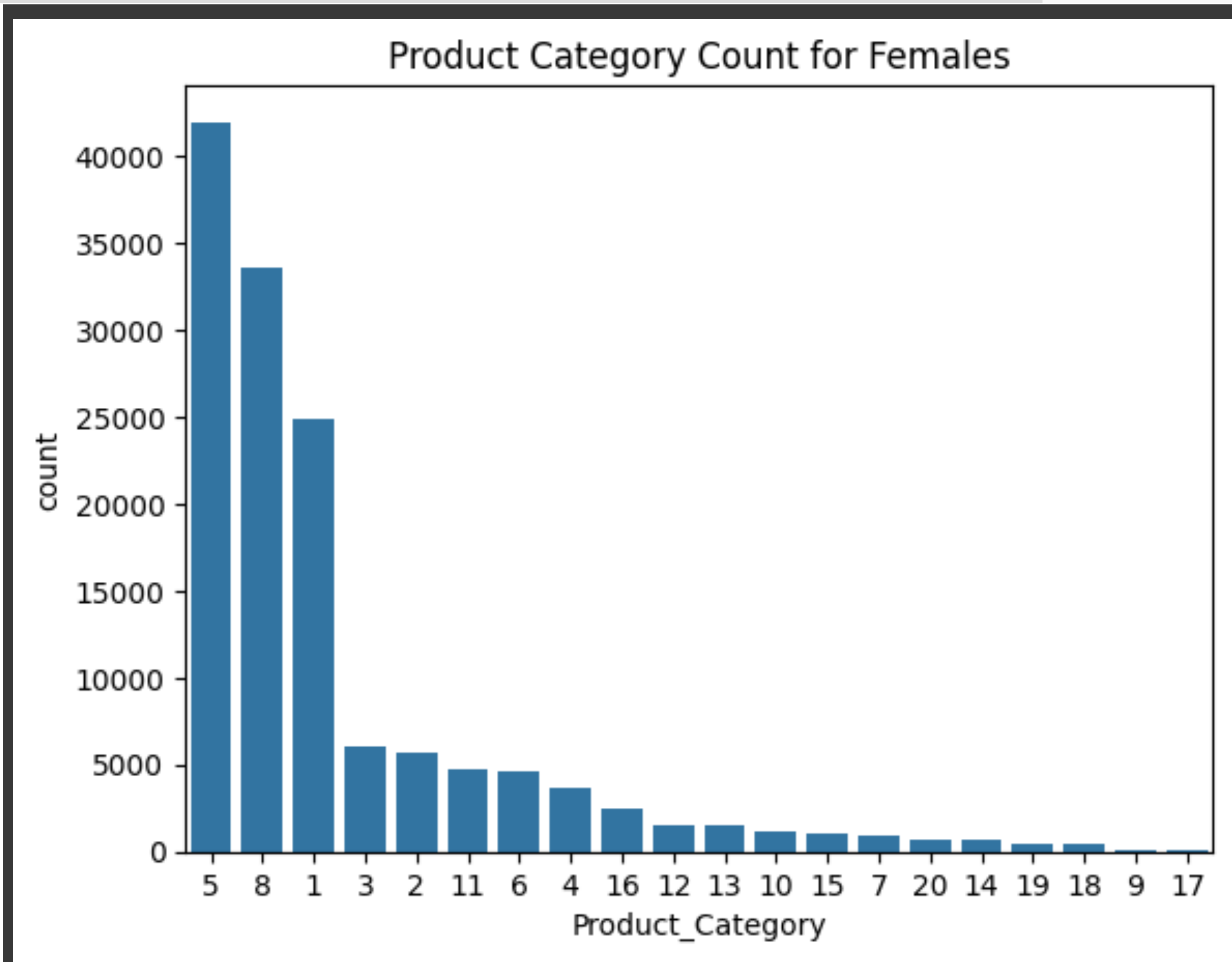
Median purchase and most of the purchase amount looks almost same for all the age groups and marital status.

```
#c Are there preferred product categories for different genders?
df_m = df[df['Gender']=='M']
plt.title('Product Category Count for Males')
sns.countplot(x='Product_Category',data=df_m,order=df_m['Product_Category'].val
plt.show()
```



Clearly , Males preferred the product category 1 most followed by 5 and 8

```
df_f = df[df['Gender']=='F']
plt.title('Product Category Count for Females')
sns.countplot(x='Product_Category',data=df_f,order=df_f['Product_Category'].val
plt.show()
```

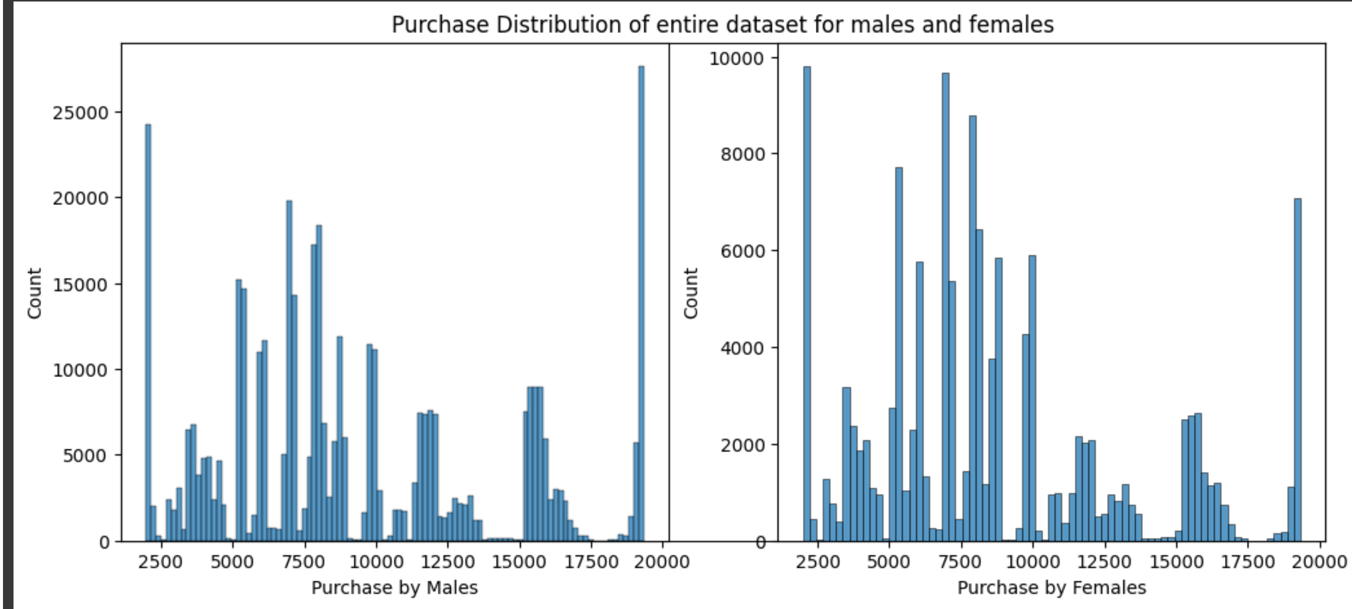


Females' most preferred product category is 5 followed by 8 and 1

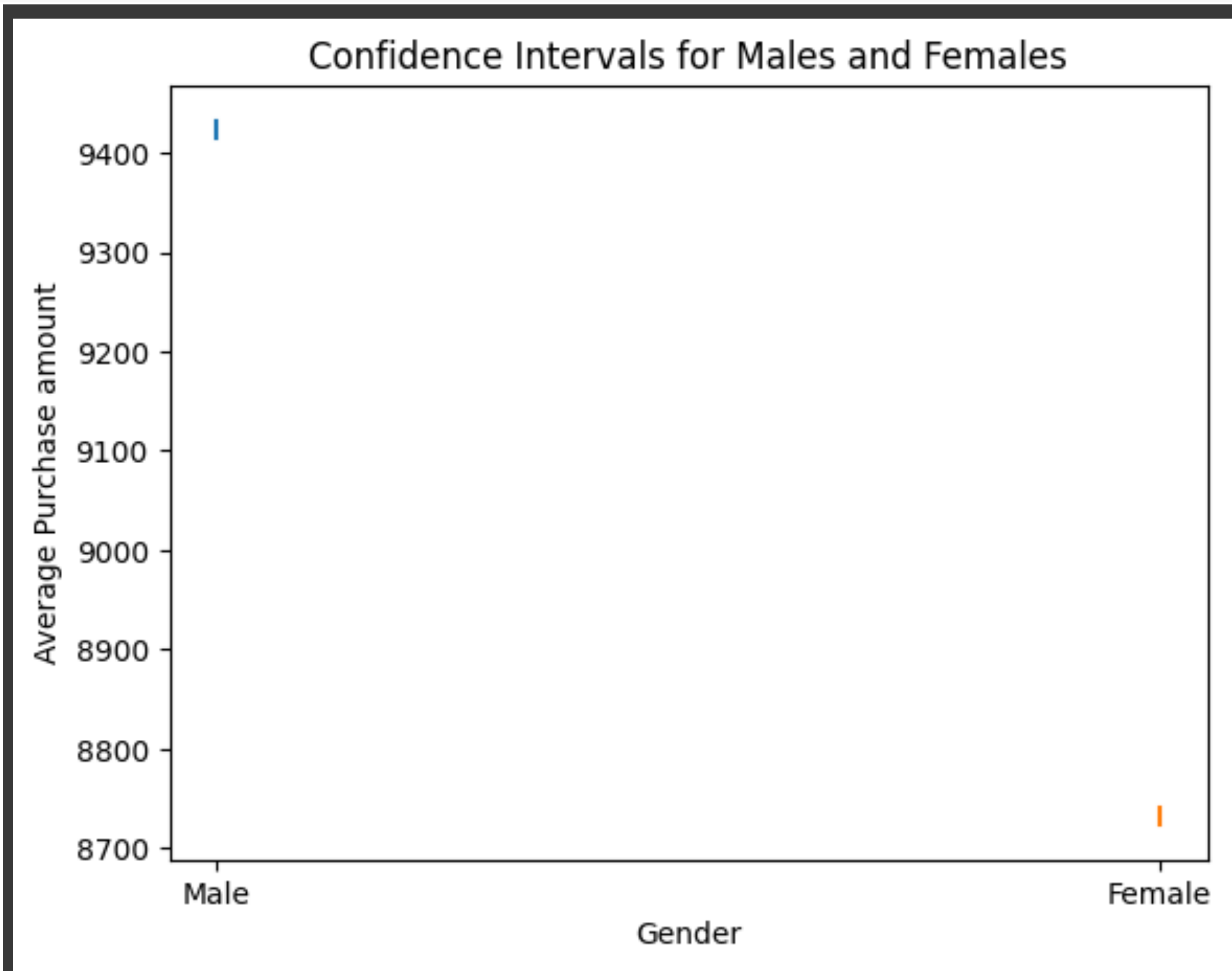
```
#4. How does gender affect the amount spent?

#CLT for entire dataset/population for Males

m_pop_mean = np.mean(df_m['Purchase'])  #population mean
m_pop_sd = np.std(df_m['Purchase'])     #Population standard deviation
m_std_error = m_pop_sd/np.sqrt(550068)    #Standard Error

z1=norm.ppf(0.025)  #Z score for 2.5 percentile
z2=norm.ppf(1-0.25) #Z score for remaining. To get 95 percent CI

m_x1 = m_pop_mean + z1*(m_std_error)
m_x2 = m_pop_mean + z2*(m_std_error)
m_x1=round(m_x1)
```

```
m_x2=round(m_x2)
print('95% confidence interval for the average amount spent for Males',m_x1,m_x
print('Width is ',m_x2-m_x1)


##CLT for entire dataset/population for Females

f_pop_mean = np.mean(df_f['Purchase'])  #population mean
f_pop_sd = np.std(df_f['Purchase'])     #Population standard deviation
f_std_error = f_pop_sd/np.sqrt(550068)    #Standard Error

f_x1=f_pop_mean + z1*(f_std_error)
f_x2=f_pop_mean + z2*(f_std_error)

f_x1=round(f_x1)
f_x2=round(f_x2)
print('95% confidence interval for the average amount spent for Females',f_x1,f
print('Width is ',f_x2-f_x1)


#Plotting both
#plt.title('Entire Dataset purchase distribution')
plt.figure(figsize=[12,5])
plt.title('Purchase Distribution of entire dataset for males and females')
plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Males')
sns.histplot(df_m['Purchase'])

plt.subplot(1,2,2)
plt.xlabel('Purchase by Females')
sns.histplot(df_f['Purchase'])

plt.show()
```

```
95% confidence interval for the average amount spent for Males 9414 9432
Width is  18
95% confidence interval for the average amount spent for Females 8724 8741
Width is  17
```



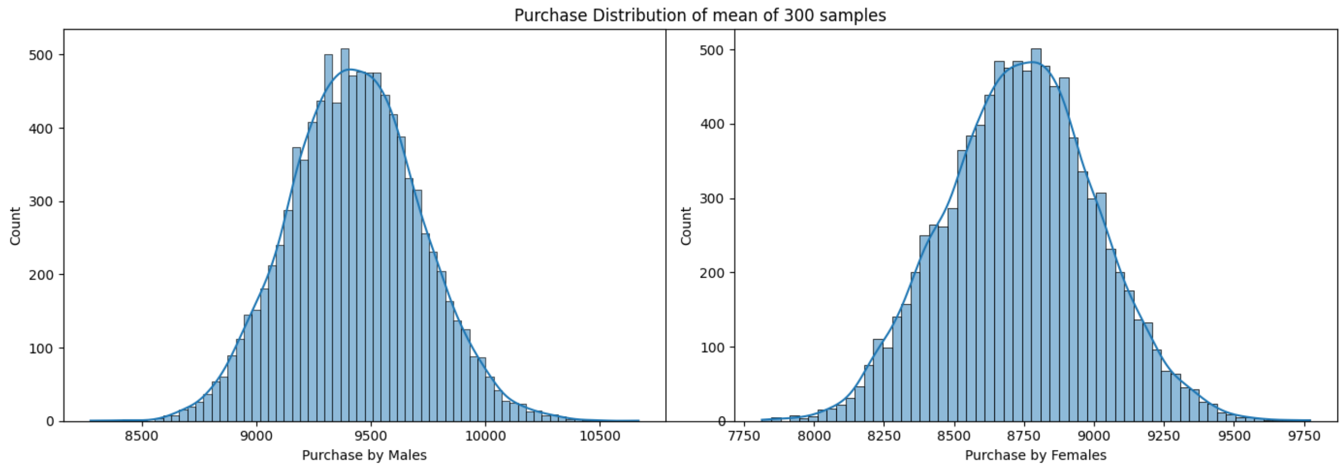
```
plt.title('Confidence Intervals for Males and Females')
plt.plot(['Male','Male'], [m_x1,m_x2])
plt.plot(['Female','Female'], [f_x1,f_x2])
plt.xlabel('Gender')
plt.ylabel('Average Purchase amount')
plt.show()
```




```
#Bootstrapping for sample of 300 Males

m_bootstrap_samples_300_means=[]

for i in range(10000):
  m_bootstrap_samples_300 = np.random.choice(df_m['Purchase'],size=300)
  m_bootstrap_samples_300_mean = np.mean(m_bootstrap_samples_300)
  m_bootstrap_samples_300_means.append(m_bootstrap_samples_300_mean)

a = np.percentile(m_bootstrap_samples_300_means,2.5)
b=np.percentile(m_bootstrap_samples_300_means,97.5)
a=round(a)
b=round(b)
print('300 sample confidence interval in Males',a,b)
print('Width is',b-a)
```

```
#Bootstrapping for sample of 300 Females
f_bootstrap_samples_300_means=[]

for i in range(10000):
  f_bootstrap_samples_300 = np.random.choice(df_f['Purchase'],size=300)
  f_bootstrap_samples_300_mean = np.mean(f_bootstrap_samples_300)
  f_bootstrap_samples_300_means.append(f_bootstrap_samples_300_mean)

a = np.percentile(f_bootstrap_samples_300_means,2.5)
b=np.percentile(f_bootstrap_samples_300_means,97.5)
a=round(a)
b=round(b)
print('300 sample confidence interval in Females',a,b)
print('Width is',b-a)



#Plotting
plt.figure(figsize=[14,5])
plt.title('Purchase Distribution of mean of 300 samples')
plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Males')
sns.histplot(m_bootstrap_samples_300_means,kde=True)


plt.subplot(1,2,2)
plt.xlabel('Purchase by Females')
sns.histplot(f_bootstrap_samples_300_means,kde=True)

plt.tight_layout()

plt.show()
```

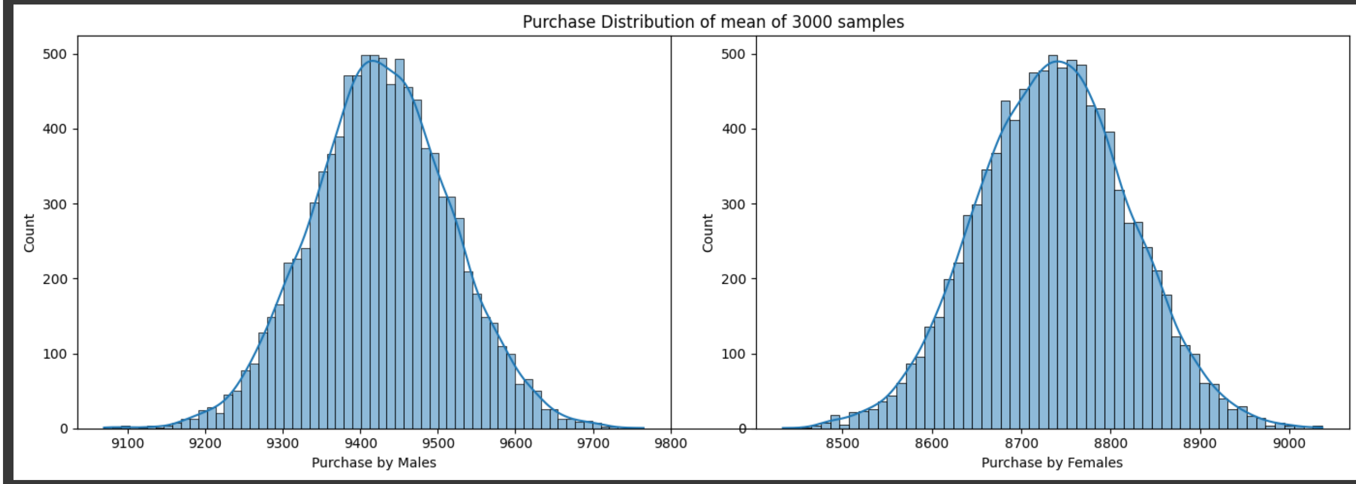
```
300 sample confidence interval in Males 8882 9994
Width is 1112
300 sample confidence interval in Females 8219 9267
Width is 1048
```




```
#Bootstrapping for sample of 3000 Males

m_bootstrap_samples_3000_means=[]

for i in range(10000):
  m_bootstrap_samples_3000 = np.random.choice(df_m['Purchase'],size=3000)
  m_bootstrap_samples_3000_mean = np.mean(m_bootstrap_samples_3000)
  m_bootstrap_samples_3000_means.append(m_bootstrap_samples_3000_mean)

a = np.percentile(m_bootstrap_samples_3000_means,2.5)
b=np.percentile(m_bootstrap_samples_3000_means,97.5)
a=round(a)
b=round(b)
print('3000 sample confidence interval in Males',a,b)
print('Width is',b-a)
```

```
#Bootstrapping for sample of 3000 Females
f_bootstrap_samples_3000_means=[]

for i in range(10000):
  f_bootstrap_samples_3000 = np.random.choice(df_f['Purchase'],size=3000)
  f_bootstrap_samples_3000_mean = np.mean(f_bootstrap_samples_3000)
  f_bootstrap_samples_3000_means.append(f_bootstrap_samples_3000_mean)

a = np.percentile(f_bootstrap_samples_3000_means,2.5)
b=np.percentile(f_bootstrap_samples_3000_means,97.5)
a=round(a)
b=round(b)
print('3000 sample confidence interval in Females',a,b)
print('Width is',b-a)



#Plotting
plt.figure(figsize=[14,5])
plt.title('Purchase Distribution of mean of 3000 samples')
plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Males')
sns.histplot(m_bootstrap_samples_3000_means,kde=True)


plt.subplot(1,2,2)
plt.xlabel('Purchase by Females')
sns.histplot(f_bootstrap_samples_3000_means,kde=True)

plt.tight_layout()

plt.show()
```

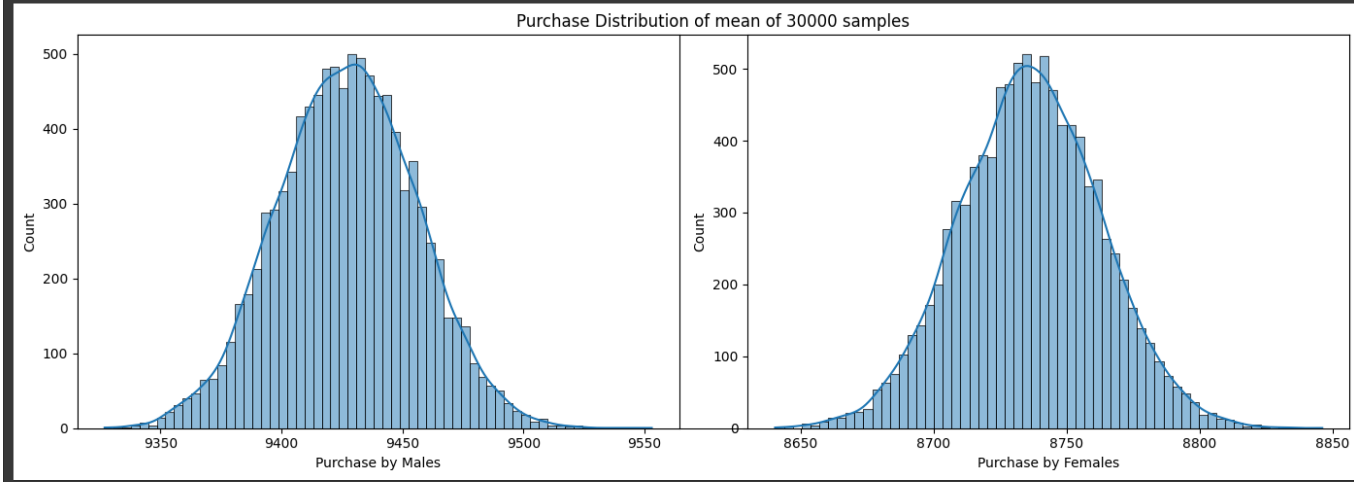
```
3000 sample confidence interval in Males 9252 9606
Width is 354
3000 sample confidence interval in Females 8571 8902
Width is 331
```




```
#Bootstrapping for sample of 30000 Males

m_bootstrap_samples_30000_means=[]

for i in range(10000):
  m_bootstrap_samples_30000 = np.random.choice(df_m['Purchase'],size=30000)
  m_bootstrap_samples_30000_mean = np.mean(m_bootstrap_samples_30000)
  m_bootstrap_samples_30000_means.append(m_bootstrap_samples_30000_mean)

a = np.percentile(m_bootstrap_samples_30000_means,2.5)
b=np.percentile(m_bootstrap_samples_30000_means,97.5)
a=round(a)
b=round(b)
print('30000 sample confidence interval in Males',a,b)
print('Width is',b-a)
```

```
#Bootstrapping for sample of 30000 Females
f_bootstrap_samples_30000_means=[]

for i in range(10000):
  f_bootstrap_samples_30000 = np.random.choice(df_f['Purchase'],size=30000)
  f_bootstrap_samples_30000_mean = np.mean(f_bootstrap_samples_30000)
  f_bootstrap_samples_30000_means.append(f_bootstrap_samples_30000_mean)

a = np.percentile(f_bootstrap_samples_30000_means,2.5)
b=np.percentile(f_bootstrap_samples_30000_means,97.5)
a=round(a)
b=round(b)
print('30000 sample confidence interval in Females',a,b)
print('Width is',b-a)



#Plotting
plt.figure(figsize=[14,5])
plt.title('Purchase Distribution of mean of 30000 samples')
plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Males')
sns.histplot(m_bootstrap_samples_30000_means,kde=True)


plt.subplot(1,2,2)
plt.xlabel('Purchase by Females')
sns.histplot(f_bootstrap_samples_30000_means,kde=True)

plt.tight_layout()

plt.show()
```

```
30000 sample confidence interval in Males 9371 9483
Width is 112
30000 sample confidence interval in Females 8684 8789
Width is 105
```

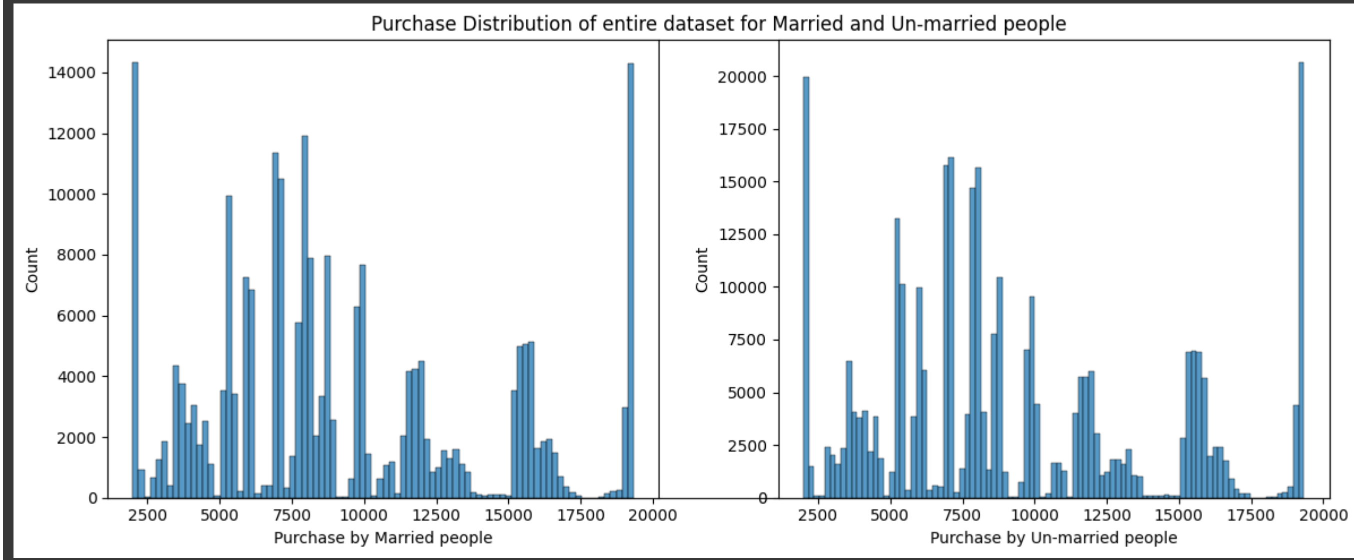


1. Confidence interval for entire dataset is slightly wider in males as compared to Females.
2. As the sample size increases , width of confidence interval decreases.
3. Confidence intervals do overlap for different sample sizes as the higher sample size's confidence interval are part of lower sample size's confidence interval
4. As the sample size increases distribution tends to become more towards normal.

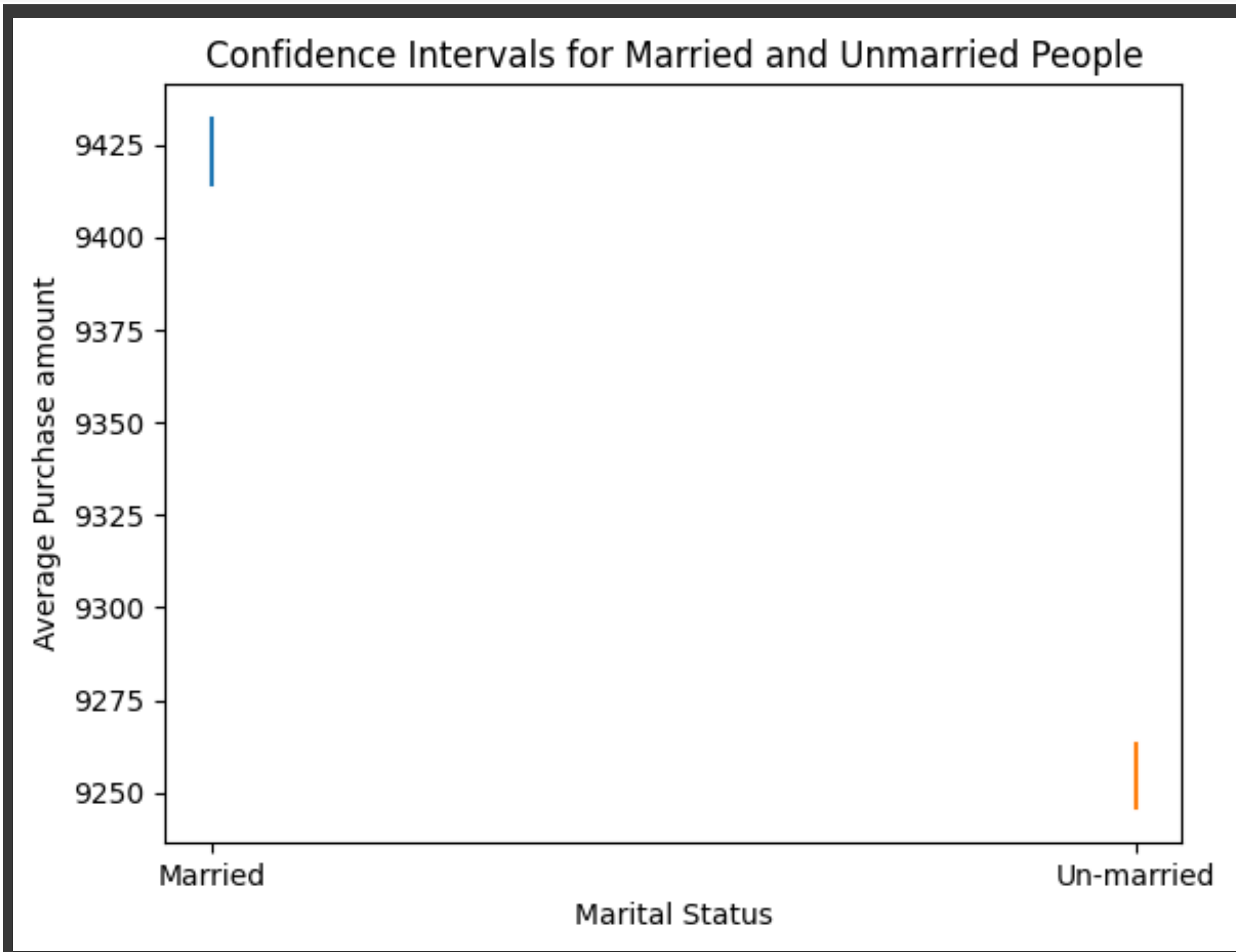
```
# 5. How does Marital_Status affect the amount spent?

df_ma=df[df['Marital_Status']==1]
df_um = df[df['Marital_Status']==0]

#CLT for entire dataset/population for married people : Marial_Status=1

ma_pop_mean = np.mean(df_ma['Purchase'])  #population mean
ma_pop_sd = np.std(df_ma['Purchase'])     #Population standard deviation
```

```
ma_std_error = m_pop_sd/np.sqrt(550068)    #Standard Error

ma_x1 = m_pop_mean + z1*(ma_std_error)
ma_x2 = m_pop_mean + z2*(ma_std_error)
ma_x1=round(m_x1)
ma_x2=round(m_x2)
print('95% confidence interval for the average amount spent for Married people'
print('Width is ',ma_x2-ma_x1)


##CLT for entire dataset/population for Females

um_pop_mean = np.mean(df_um['Purchase'])  #population mean
um_pop_sd = np.std(df_um['Purchase'])     #Population standard deviation
um_std_error = um_pop_sd/np.sqrt(550068)    #Standard Error

um_x1=um_pop_mean + z1*(um_std_error)
um_x2=um_pop_mean + z2*(um_std_error)

um_x1=round(um_x1)
um_x2=round(um_x2)
print('95% confidence interval for the average amount spent for Un-married peop
print('Width is ',um_x2-um_x1)


#Plotting both
#plt.title('Entire Dataset purchase distribution')
plt.figure(figsize=[12,5])
plt.title('Purchase Distribution of entire dataset for Married and Un-married p

plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Married people')
sns.histplot(df_ma['Purchase'])

plt.subplot(1,2,2)
plt.xlabel('Purchase by Un-married people')
sns.histplot(df_um['Purchase'])

plt.tight_layout()

plt.show()
```

```
95% confidence interval for the average amount spent for Married people 941
Width is  18
95% confidence interval for the average amount spent for Un-married people
Width is  17
```



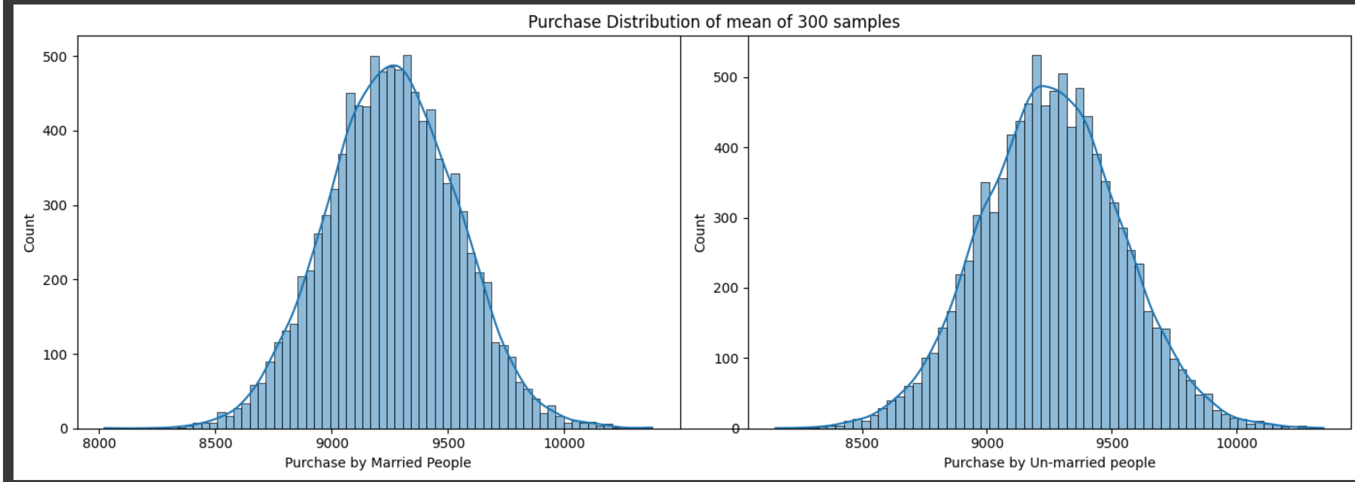
```
plt.title('Confidence Intervals for Married and Unmarried People')
plt.plot(['Married','Married'], [ma_x1,ma_x2])
plt.plot(['Un-married','Un-married'], [um_x1,um_x2])
plt.xlabel('Marital Status')
plt.ylabel('Average Purchase amount')
plt.show()
```




```
#Bootstrapping for sample of 300 Married people

m_bootstrap_samples_300_means=[]

for i in range(10000):
  m_bootstrap_samples_300 = np.random.choice(df_ma['Purchase'],size=300)
  m_bootstrap_samples_300_mean = np.mean(m_bootstrap_samples_300)
  m_bootstrap_samples_300_means.append(m_bootstrap_samples_300_mean)

a = np.percentile(m_bootstrap_samples_300_means,2.5)
b=np.percentile(m_bootstrap_samples_300_means,97.5)
a=round(a)
b=round(b)
print('300 sample confidence interval in Married People',a,b)
print('Width is',b-a)
```

```
#Bootstrapping for sample of 300 Un-married people
f_bootstrap_samples_300_means=[]

for i in range(10000):
  f_bootstrap_samples_300 = np.random.choice(df_um['Purchase'],size=300)
  f_bootstrap_samples_300_mean = np.mean(f_bootstrap_samples_300)
  f_bootstrap_samples_300_means.append(f_bootstrap_samples_300_mean)

a = np.percentile(f_bootstrap_samples_300_means,2.5)
b=np.percentile(f_bootstrap_samples_300_means,97.5)
a=round(a)
b=round(b)
print('300 sample confidence interval in Un-married people',a,b)
print('Width is',b-a)



#Plotting
plt.figure(figsize=[14,5])
plt.title('Purchase Distribution of mean of 300 samples')
plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Married People')
sns.histplot(m_bootstrap_samples_300_means,kde=True)


plt.subplot(1,2,2)
plt.xlabel('Purchase by Un-married people')
sns.histplot(f_bootstrap_samples_300_means,kde=True)

plt.tight_layout()

plt.show()
```

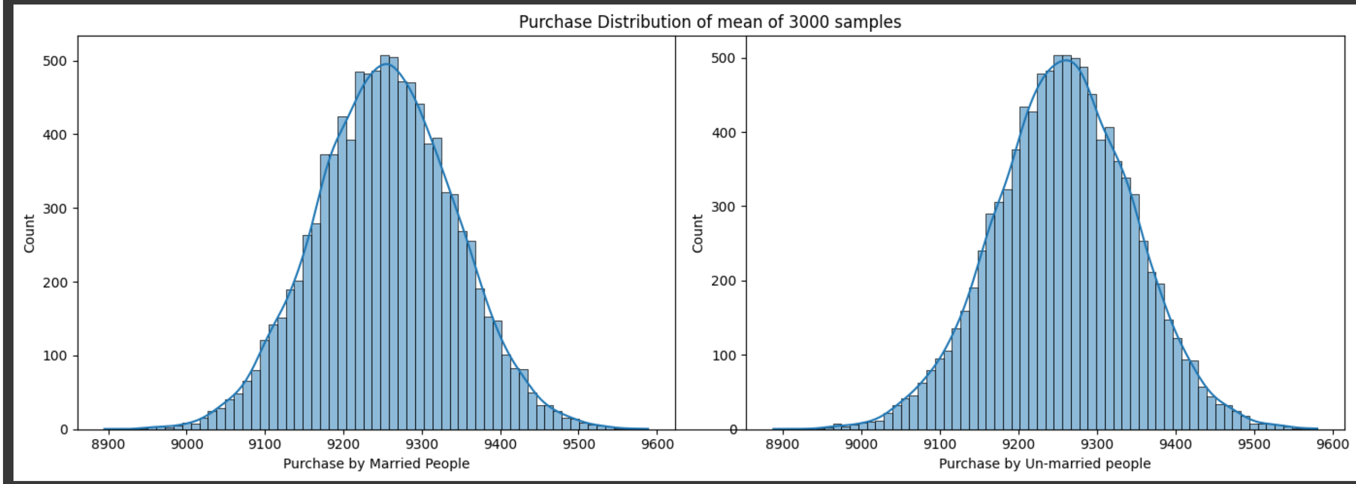
```
300 sample confidence interval in Married People 8718 9799
Width is 1081
300 sample confidence interval in Un-married people 8708 9809
Width is 1101
```




```
#Bootstrapping for sample of 3000 Married people

m_bootstrap_samples_3000_means=[]

for i in range(10000):
  m_bootstrap_samples_3000 = np.random.choice(df_ma['Purchase'],size=3000)
  m_bootstrap_samples_3000_mean = np.mean(m_bootstrap_samples_3000)
  m_bootstrap_samples_3000_means.append(m_bootstrap_samples_3000_mean)

a = np.percentile(m_bootstrap_samples_3000_means,2.5)
b=np.percentile(m_bootstrap_samples_3000_means,97.5)
a=round(a)
b=round(b)
print('3000 sample confidence interval in Married People',a,b)
print('Width is',b-a)
```

```
#Bootstrapping for sample of 3000 Un-married people
f_bootstrap_samples_3000_means=[]

for i in range(10000):
  f_bootstrap_samples_3000 = np.random.choice(df_um['Purchase'],size=3000)
  f_bootstrap_samples_3000_mean = np.mean(f_bootstrap_samples_3000)
  f_bootstrap_samples_3000_means.append(f_bootstrap_samples_3000_mean)

a = np.percentile(f_bootstrap_samples_3000_means,2.5)
b=np.percentile(f_bootstrap_samples_3000_means,97.5)
a=round(a)
b=round(b)
print('3000 sample confidence interval in Un-married people',a,b)
print('Width is',b-a)



#Plotting
plt.figure(figsize=[14,5])
plt.title('Purchase Distribution of mean of 3000 samples')
plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Married People')
sns.histplot(m_bootstrap_samples_3000_means,kde=True)


plt.subplot(1,2,2)
plt.xlabel('Purchase by Un-married people')
sns.histplot(f_bootstrap_samples_3000_means,kde=True)

plt.tight_layout()

plt.show()
```

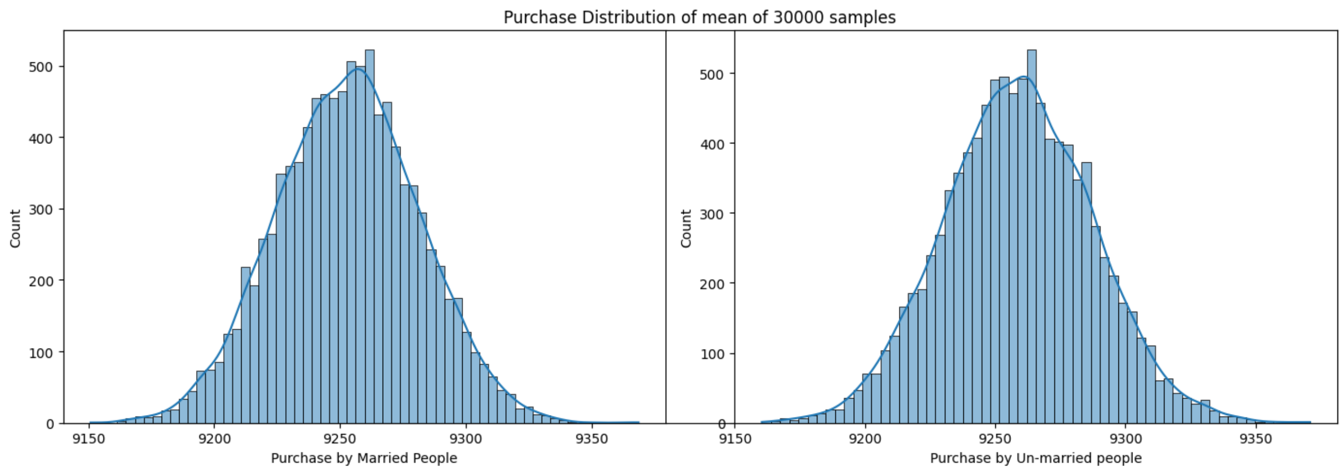
```
3000 sample confidence interval in Married People 9081 9428
Width is 347
3000 sample confidence interval in Un-married people 9082 9428
Width is 346
```




```
#Bootstrapping for sample of 30000 Married people

m_bootstrap_samples_30000_means=[]

for i in range(10000):
  m_bootstrap_samples_30000 = np.random.choice(df_ma['Purchase'],size=30000)
  m_bootstrap_samples_30000_mean = np.mean(m_bootstrap_samples_30000)
  m_bootstrap_samples_30000_means.append(m_bootstrap_samples_30000_mean)

a = np.percentile(m_bootstrap_samples_30000_means,2.5)
b=np.percentile(m_bootstrap_samples_30000_means,97.5)
a=round(a)
b=round(b)
print('30000 sample confidence interval in Married People',a,b)
print('Width is',b-a)
```

```
#Bootstrapping for sample of 30000 Un-married people
f_bootstrap_samples_30000_means=[]

for i in range(10000):
  f_bootstrap_samples_30000 = np.random.choice(df_um['Purchase'],size=30000)
  f_bootstrap_samples_30000_mean = np.mean(f_bootstrap_samples_30000)
  f_bootstrap_samples_30000_means.append(f_bootstrap_samples_30000_mean)

a = np.percentile(f_bootstrap_samples_30000_means,2.5)
b=np.percentile(f_bootstrap_samples_30000_means,97.5)
a=round(a)
b=round(b)
print('30000 sample confidence interval in Un-married people',a,b)
print('Width is',b-a)



#Plotting
plt.figure(figsize=[14,5])
plt.title('Purchase Distribution of mean of 30000 samples')
plt.xticks([])
plt.yticks([])

plt.subplot(1,2,1)
plt.xlabel('Purchase by Married People')
sns.histplot(m_bootstrap_samples_30000_means,kde=True)


plt.subplot(1,2,2)
plt.xlabel('Purchase by Un-married people')
sns.histplot(f_bootstrap_samples_30000_means,kde=True)

plt.tight_layout()

plt.show()
```

```
30000 sample confidence interval in Married People 9198 9308
Width is 110
30000 sample confidence interval in Un-married people 9203 9315
Width is 112
```



Double-click (or enter) to edit

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

```
array(['0-17', '55+', '26-35', '46-50', '51-55', '36-45', '18-25'],
      dtype=object)
```

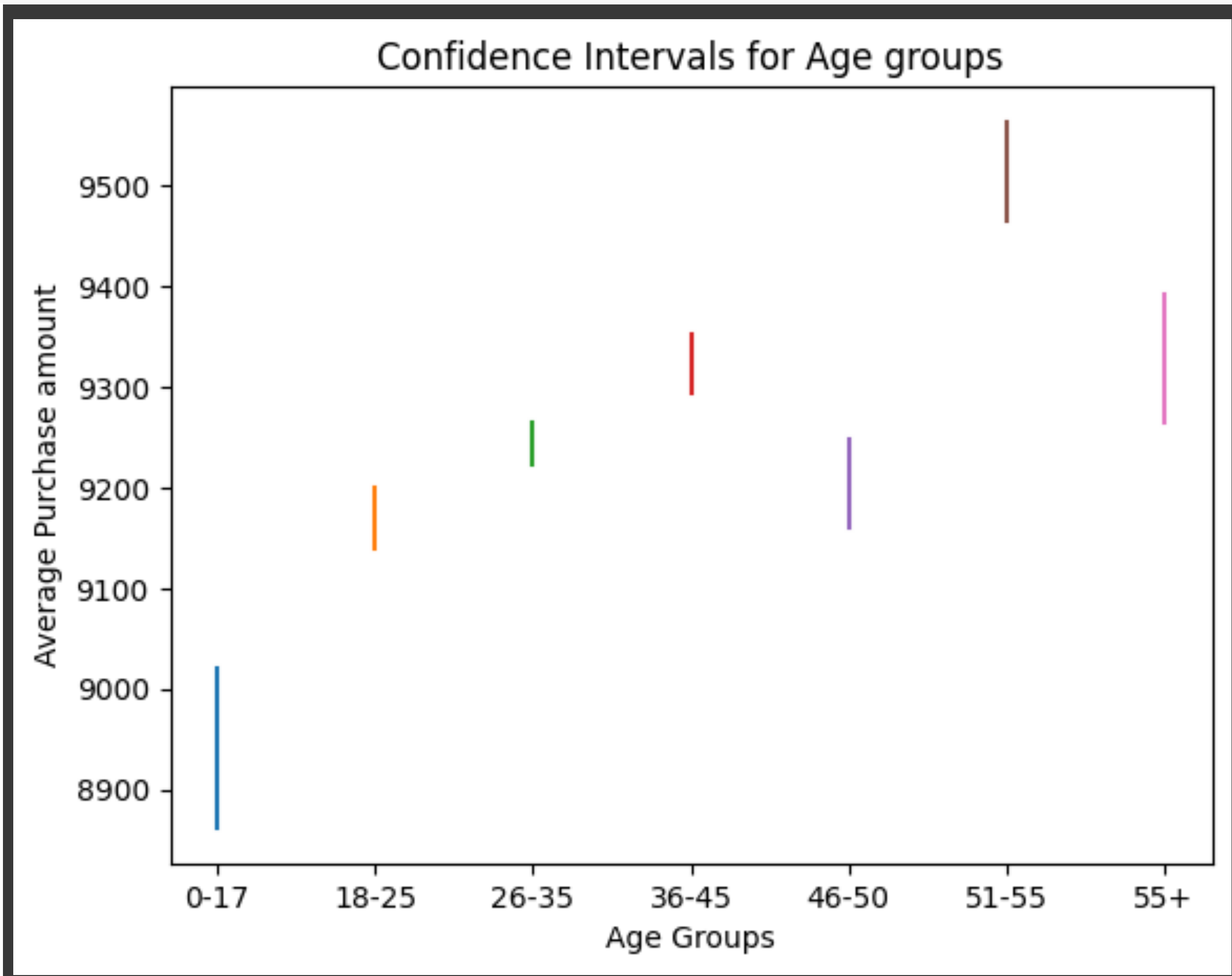
```
# 4. How does Age affect the amount spent?

#Entire Dataset
df_age1=df[df['Age']=='0-17']
df_age2=df[df['Age']=='55+']
df_age3=df[df['Age']=='26-35']
df_age4=df[df['Age']=='46-50']
df_age5=df[df['Age']=='51-55']
df_age6=df[df['Age']=='36-45']
```

```
df_age7=df[df['Age']=='18-25']

#Mean
m_1 = np.mean(df_age1['Purchase'])
m_2 =np.mean(df_age2['Purchase'])
m_3 =np.mean(df_age3['Purchase'])
m_4 =np.mean(df_age4['Purchase'])
m_5 =np.mean(df_age5['Purchase'])
m_6 =np.mean(df_age6['Purchase'])
m_7 =np.mean(df_age7['Purchase'])

#Standard Deviations
sd_1 = np.std(df_age1['Purchase'])
sd_2 = np.std(df_age2['Purchase'])
sd_3 = np.std(df_age3['Purchase'])
sd_4 = np.std(df_age4['Purchase'])
sd_5 = np.std(df_age5['Purchase'])
sd_6 = np.std(df_age6['Purchase'])
sd_7 = np.std(df_age7['Purchase'])



#Standard Error
se_1 = sd_1/np.sqrt(df_age1.shape[0])
se_2 = sd_2/np.sqrt(df_age2.shape[0])
se_3 = sd_3/np.sqrt(df_age3.shape[0])
se_4 = sd_4/np.sqrt(df_age4.shape[0])
se_5 = sd_5/np.sqrt(df_age5.shape[0])
se_6 = sd_6/np.sqrt(df_age6.shape[0])
se_7 = sd_7/np.sqrt(df_age7.shape[0])

#Ci for Age groups
x1,y1=norm.interval(0.95,loc=m_1,scale=se_1)
x2,y2=norm.interval(0.95,loc=m_2,scale=se_2)
x3,y3=norm.interval(0.95,loc=m_3,scale=se_3)
x4,y4=norm.interval(0.95,loc=m_4,scale=se_4)
x5,y5=norm.interval(0.95,loc=m_5,scale=se_5)
x6,y6=norm.interval(0.95,loc=m_6,scale=se_6)
x7,y7=norm.interval(0.95,loc=m_7,scale=se_7)



print('Confidence Intervals for Age group 0-17 is',round(x1,2),round(y1,2))
print('Width is',round(y1-x1,2))
print('\nConfidence Intervals for Age group 55+  is',round(x2,2),round(y2,2))
print('Width is',round(y2-x2,2))
print('\nConfidence Intervals for Age group 26-35 is',round(x3,2),round(y3,2))
print('Width is',round(y3-x3,2))
print('\nConfidence Intervals for Age group 46-50 is',round(x4,2),round(y4,2))
print('Width is',round(y4-x4,2))
```

```
print('\nConfidence Intervals for Age group 51-55 is',round(x5,2),round(y5,2))
print('Width is',round(y5-x5,2))
print('\nConfidence Intervals for Age group 36-45 is',round(x6,2),round(y6,2))
print('Width is',round(y6-x6,2))
print('\nConfidence Intervals for Age group 18-25 is',round(x7,2),round(y7,2))
print('Width is',round(y7-x7,2))
```

```
Confidence Intervals for Age group 0-17 is 8861.85 9019.44
Width is 157.59

Confidence Intervals for Age group 55+  is 9263.91 9391.68
Width is 127.77

Confidence Intervals for Age group 26-35 is 9223.47 9264.09
Width is 40.61

Confidence Intervals for Age group 46-50 is 9160.33 9248.09
Width is 87.76

Confidence Intervals for Age group 51-55 is 9466.18 9563.54
Width is 97.36

Confidence Intervals for Age group 36-45 is 9294.28 9351.57
Width is 57.29

Confidence Intervals for Age group 18-25 is 9138.66 9199.37
Width is 60.71
```

```
plt.title('Confidence Intervals for Age groups')
plt.plot(['0-17','0-17'], [x1,y1])
plt.plot(['18-25','18-25'], [x7,y7])
plt.plot(['26-35','26-35'], [x3,y3])
plt.plot(['36-45','36-45'], [x6,y6])
plt.plot(['46-50','46-50'], [x4,y4])
plt.plot(['51-55','51-55'], [x5,y5])
plt.plot(['55+','55+'], [x2,y2])
plt.xlabel('Age Groups')
plt.ylabel('Average Purchase amount')
plt.show()
```



Clearly , Overlap is for age groups of 18-25 and 26-35 , 36-45 and 55+

```
#Age group 0-17
print('95 percent Confidence Intervals for Age group 0-17 and their widths')
print('-'*50)
#sample size 300

df_age1_300 = np.random.choice(df_age1['Purchase'],size=300)
```

```
m_300 = np.mean(df_age1_300)
se_300 = sd_1 / np.sqrt(300)  #Standard error for 300 samples , sd1 is populati

x1,y1=norm.interval(0.95,loc=m_300,scale=se_300)
x1=round(x1,2)
y1=round(y1,2)
print('For 300 samples it would be',x1,y1)
print('Width would be',round(y1-x1,2))
print('\n')

#sample size 3000
df_age1_3000 = np.random.choice(df_age1['Purchase'],size=3000)

m_3000 = np.mean(df_age1_3000)
se_3000 = sd_1 / np.sqrt(3000)  #Standard error for 3000 samples , sd1 is popul

x1,y1=norm.interval(0.95,loc=m_3000,scale=se_3000)
x1=round(x1,2)
y1=round(y1,2)
print('For 3000 samples it would be',x1,y1)
print('Width would be',round(y1-x1,2))
print('\n')

#sample size 30000
df_age1_30000 = np.random.choice(df_age1['Purchase'],size=30000)

m_30000 = np.mean(df_age1_30000)
se_30000 = sd_1 / np.sqrt(30000)  #Standard error for 30000 samples , sd1 is pc

x1,y1=norm.interval(0.95,loc=m_30000,scale=se_30000)
x1=round(x1,2)
y1=round(y1,2)
print('For 30000 samples it would be',x1,y1)
print('Width would be',round(y1-x1,2))
print('\n')
```

```
95 percent Confidence Intervals for Age group 0-17 and their widths
--------------------------------------------------
For 300 samples it would be 8317.83 9435.93
Width would be 1118.1


For 3000 samples it would be 8731.7 9085.28
Width would be 353.58


For 30000 samples it would be 8908.23 9020.04
Width would be 111.81
```

```
#Age group 55+
print('95 percent Confidence Intervals for Age group 55+ and their widths')
print('-'*50)
#sample size 300

df_age2_300 = np.random.choice(df_age2['Purchase'],size=300)

m_300 = np.mean(df_age2_300)
se_300 = sd_2 / np.sqrt(300)  #Standard error for 300 samples , sd2 is populati

x2,y2=norm.interval(0.95,loc=m_300,scale=se_300)
x2=round(x2,2)
y2=round(y2,2)
print('For 300 samples it would be',x2,y2)
print('Width would be',round(y2-x2,2))
print('\n')

#sample size 3000
df_age2_3000 = np.random.choice(df_age2['Purchase'],size=3000)

m_3000 = np.mean(df_age2_3000)
se_3000 = sd_2 / np.sqrt(3000)  #Standard error for 3000 samples , sd2 is popul

x2,y2=norm.interval(0.95,loc=m_3000,scale=se_3000)
x2=round(x2,2)
y2=round(y2,2)
print('For 3000 samples it would be',x2,y2)
print('Width would be',round(y2-x2,2))
print('\n')

#sample size 30000
df_age2_30000 = np.random.choice(df_age2['Purchase'],size=30000)
```

```
m_30000 = np.mean(df_age2_30000)
se_30000 = sd_2 / np.sqrt(30000)  #Standard error for 30000 samples , sd2 is pc

x2,y2=norm.interval(0.95,loc=m_30000,scale=se_30000)
x2=round(x2,2)
y2=round(y2,2)
print('For 30000 samples it would be',x2,y2)
print('Width would be',round(y2-x2,2))
print('\n')
```

```
95 percent Confidence Intervals for Age group 55+ and their widths
--------------------------------------------------
For 300 samples it would be 8440.6 9522.35
Width would be 1081.75


For 3000 samples it would be 9099.45 9441.53
Width would be 342.08


For 30000 samples it would be 9275.18 9383.36
Width would be 108.18
```

```
#Age group 26-35
print('95 percent Confidence Intervals for Age group 26-35 and their widths')
print('-'*50)
#sample size 300

df_age3_300 = np.random.choice(df_age3['Purchase'],size=300)

m_300 = np.mean(df_age3_300)
se_300 = sd_3 / np.sqrt(300)  #Standard error for 300 samples , sd3 is populati

x3,y3=norm.interval(0.95,loc=m_300,scale=se_300)
x3=round(x3,2)
y3=round(y3,2)
print('For 300 samples it would be',x3,y3)
print('Width would be',round(y3-x3,2))
print('\n')

#sample size 3000
df_age3_3000 = np.random.choice(df_age3['Purchase'],size=3000)

m_3000 = np.mean(df_age3_3000)
se_3000 = sd_3 / np.sqrt(3000)  #Standard error for 3000 samples , sd3 is popul

x3,y3=norm.interval(0.95,loc=m_3000,scale=se_3000)
```

```
x3=round(x3,2)
y3=round(y3,2)
print('For 3000 samples it would be',x3,y3)
print('Width would be',round(y3-x3,2))
print('\n')

#sample size 30000
df_age3_30000 = np.random.choice(df_age3['Purchase'],size=30000)

m_30000 = np.mean(df_age3_30000)
se_30000 = sd_3 / np.sqrt(30000)  #Standard error for 30000 samples , sd3 is pc

x3,y3=norm.interval(0.95,loc=m_30000,scale=se_30000)
x3=round(x3,2)
y3=round(y3,2)
print('For 30000 samples it would be',x3,y3)
print('Width would be',round(y3-x3,2))
print('\n')
```

```
95 percent Confidence Intervals for Age group 26-35 and their widths
--------------------------------------------------
For 300 samples it would be 8732.39 9831.21
Width would be 1098.82


For 3000 samples it would be 8977.77 9325.24
Width would be 347.47


For 30000 samples it would be 9245.38 9355.26
Width would be 109.88
```

```
#Age group 46-50
print('95 percent Confidence Intervals for Age group 46-30 and their widths')
print('-'*50)
#sample size 300

df_age4_300 = np.random.choice(df_age4['Purchase'],size=300)

m_300 = np.mean(df_age4_300)
se_300 = sd_4 / np.sqrt(300)  #Standard error for 300 samples , sd4 is populati

x4,y4=norm.interval(0.95,loc=m_300,scale=se_300)
x4=round(x4,2)
y4=round(y4,2)
print('For 300 samples it would be',x4,y4)
print('Width would be',round(y4-x4,2))
```

```
print('\n')

#sample size 3000
df_age4_3000 = np.random.choice(df_age4['Purchase'],size=3000)

m_3000 = np.mean(df_age4_3000)
se_3000 = sd_4 / np.sqrt(3000)  #Standard error for 3000 samples , sd4 is popul

x4,y4=norm.interval(0.95,loc=m_3000,scale=se_3000)
x4=round(x4,2)
y4=round(y4,2)
print('For 3000 samples it would be',x4,y4)
print('Width would be',round(y4-x4,2))
print('\n')

#sample size 30000
df_age4_30000 = np.random.choice(df_age4['Purchase'],size=30000)

m_30000 = np.mean(df_age4_30000)
se_30000 = sd_4 / np.sqrt(30000)  #Standard error for 30000 samples , sd4 is pc

x4,y4=norm.interval(0.95,loc=m_30000,scale=se_30000)
x4=round(x4,2)
y4=round(y4,2)
print('For 30000 samples it would be',x4,y4)
print('Width would be',round(y4-x4,2))
print('\n')
```

```
95 percent Confidence Intervals for Age group 46-30 and their widths
--------------------------------------------------
For 300 samples it would be 8621.69 9704.81
Width would be 1083.12


For 3000 samples it would be 8986.82 9329.34
Width would be 342.52


For 30000 samples it would be 9144.59 9252.9
Width would be 108.31
```

```
#Age group 51-55
print('95 percent Confidence Intervals for Age group 51-55 and their widths')
print('-'*50)
#sample size 300

df_age5_300 = np.random.choice(df_age5['Purchase'],size=300)
```

```
m_300 = np.mean(df_age5_300)
se_300 = sd_5 / np.sqrt(300)  #Standard error for 300 samples , sd5 is populati

x5,y5=norm.interval(0.95,loc=m_300,scale=se_300)
x5=round(x5,2)
y5=round(y5,2)
print('For 300 samples it would be',x5,y5)
print('Width would be',round(y5-x5,2))
print('\n')

#sample size 3000
df_age5_3000 = np.random.choice(df_age5['Purchase'],size=3000)

m_3000 = np.mean(df_age5_3000)
se_3000 = sd_5 / np.sqrt(3000)  #Standard error for 3000 samples , sd5 is popul

x5,y5=norm.interval(0.95,loc=m_3000,scale=se_3000)
x5=round(x5,2)
y5=round(y5,2)
print('For 3000 samples it would be',x5,y5)
print('Width would be',round(y5-x5,2))
print('\n')

#sample size 30000
df_age5_30000 = np.random.choice(df_age5['Purchase'],size=30000)

m_30000 = np.mean(df_age5_30000)
se_30000 = sd_5 / np.sqrt(30000)  #Standard error for 30000 samples , sd5 is pc

x5,y5=norm.interval(0.95,loc=m_30000,scale=se_30000)
x5=round(x5,2)
y5=round(y5,2)
print('For 30000 samples it would be',x5,y5)
print('Width would be',round(y5-x5,2))
print('\n')
```

```
95 percent Confidence Intervals for Age group 51-55 and their widths
--------------------------------------------------
For 300 samples it would be 9431.42 10534.4
Width would be 1102.98


For 3000 samples it would be 9338.86 9687.65
Width would be 348.79


For 30000 samples it would be 9453.4 9563.69
Width would be 110.29
```

```
#Age group 36-45
print('95 percent Confidence Intervals for Age group 36-45 and their widths')
print('-'*50)
#sample size 300

df_age6_300 = np.random.choice(df_age6['Purchase'],size=300)

m_300 = np.mean(df_age6_300)
se_300 = sd_6 / np.sqrt(300)  #Standard error for 300 samples , sd6 is populati

x6,y6=norm.interval(0.95,loc=m_300,scale=se_300)
x6=round(x6,2)
y6=round(y6,2)
print('For 300 samples it would be',x6,y6)
print('Width would be',round(y6-x6,2))
print('\n')

#sample size 3000
df_age6_3000 = np.random.choice(df_age6['Purchase'],size=3000)

m_3000 = np.mean(df_age6_3000)
se_3000 = sd_6 / np.sqrt(3000)  #Standard error for 3000 samples , sd6 is popul

x6,y6=norm.interval(0.95,loc=m_3000,scale=se_3000)
x6=round(x6,2)
y6=round(y6,2)
print('For 3000 samples it would be',x6,y6)
print('Width would be',round(y6-x6,2))
print('\n')

#sample size 30000
df_age6_30000 = np.random.choice(df_age6['Purchase'],size=30000)
```

```
m_30000 = np.mean(df_age6_30000)
se_30000 = sd_6 / np.sqrt(30000)  #Standard error for 30000 samples , sd6 is pc

x6,y6=norm.interval(0.95,loc=m_30000,scale=se_30000)
x6=round(x6,2)
y6=round(y6,2)
print('For 30000 samples it would be',x6,y6)
print('Width would be',round(y6-x6,2))
print('\n')
```

```
95 percent Confidence Intervals for Age group 36-45 and their widths
--------------------------------------------------
For 300 samples it would be 8624.9 9721.99
Width would be 1097.09


For 3000 samples it would be 9298.34 9645.27
Width would be 346.93


For 30000 samples it would be 9261.13 9370.84
Width would be 109.71
```

```
#Age group 18-25
print('95 percent Confidence Intervals for Age group 18-25 and their widths')
print('-'*50)
#sample size 300

df_age7_300 = np.random.choice(df_age7['Purchase'],size=300)

m_300 = np.mean(df_age7_300)
se_300 = sd_7 / np.sqrt(300)  #Standard error for 300 samples , sd7 is populati

x7,y7=norm.interval(0.95,loc=m_300,scale=se_300)
x7=round(x7,2)
y7=round(y7,2)
print('For 300 samples it would be',x7,y7)
print('Width would be',round(y7-x7,2))
print('\n')

#sample size 3000
df_age7_3000 = np.random.choice(df_age7['Purchase'],size=3000)

m_3000 = np.mean(df_age7_3000)
se_3000 = sd_7 / np.sqrt(3000)  #Standard error for 3000 samples , sd7 is popul

x7,y7=norm.interval(0.95,loc=m_3000,scale=se_3000)
```

```
x7=round(x7,2)
y7=round(y7,2)
print('For 3000 samples it would be',x7,y7)
print('Width would be',round(y7-x7,2))
print('\n')

#sample size 30000
df_age7_30000 = np.random.choice(df_age7['Purchase'],size=30000)

m_30000 = np.mean(df_age7_30000)
se_30000 = sd_7 / np.sqrt(30000)  #Standard error for 30000 samples , sd7 is pc

x7,y7=norm.interval(0.95,loc=m_30000,scale=se_30000)
x7=round(x7,2)
y7=round(y7,2)
print('For 30000 samples it would be',x7,y7)
print('Width would be',round(y7-x7,2))
print('\n')
```

```
95 percent Confidence Intervals for Age group 18-25 and their widths
--------------------------------------------------
For 300 samples it would be 8464.06 9570.61
Width would be 1106.55


For 3000 samples it would be 8984.66 9334.59
Width would be 349.93


For 30000 samples it would be 9080.71 9191.36
Width would be 110.65
```

1. Confidence interval is widest for age group 0-17.
2. As the sample size increases,confidence interval decreases making it more certain about the sample mean being closer to population mean.
3. Yes , confidence interval overlaps for different sample sizes

```
#7. Create a report

# Report whether the confidence intervals for the average amount spent by males
#this conclusion to make changes or improvements?
```

Clearly , no overlap and hence average amount spent is high for Males than females.

```
#Report whether the confidence intervals for the average amount spent by marrie
#leverage this conclusion to make changes or improvements?
```

Clearly , no overlap and hence average amount spent is high for married people than un-married ones.

```
#Report whether the confidence intervals for the average amount spent by differ
#leverage this conclusion to make changes or improvements?
```

Clearly , Overlap is for age groups of 18-25 and 26-35 , 36-45 and 55+

```
# 8. Recommendations
```

1. Average purchase is higher for Males in Black Friday. More offers can be introduced for female customers to increase their purchase.
2. Walmart can introduce some discounts for products category like 1,5,8 to increase the revenue.
3. Walmart can introduce discounts for married couple buying products together to increase the sales further.