Diwali_Sales_Analysis

July 1, 2023

[1]:
```
# import python libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt # visualizing data
%matplotlib inline
import seaborn as sns
```

[18]:
```
# import csv file
df = pd.read_csv(r'C:\Users\Lenovo\Downloads\Diwali_Sales_Analysis.csv',␣
 ↪encoding='unicode_escape')
```

[19]:
```
df.shape
```

[19]:
```
(11251, 15)
```

[21]:
```
df.head(10)
```

[21]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2
5	1000588	Joni	P00057942	M	26-35	28	1	Himachal Pradesh	Northern	Food Processing	Auto	1
6	1001132	Balk	P00018042	F	18-25	25	1	Uttar Pradesh	Central	Lawyer	Auto	4
7	1002092	Shivangi	P00273442	F	55+	61	0					
8	1003224	Kushal	P00205642	M	26-35	35	0					
9	1003650	Ginny	P00031142	F	26-35	26	1					

```
7       Maharashtra   Western        IT Sector             Auto       1
8     Uttar Pradesh   Central             Govt             Auto       2
9    Andhra Pradesh   Southern           Media              Auto       4

     Amount  Status  unnamed1
0  23952.00     NaN       NaN
1  23934.00     NaN       NaN
2  23924.00     NaN       NaN
3  23912.00     NaN       NaN
4  23877.00     NaN       NaN
5  23877.00     NaN       NaN
6  23841.00     NaN       NaN
7       NaN     NaN       NaN
8  23809.00     NaN       NaN
9  23799.99     NaN       NaN
```

[22]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 15 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   User_ID           11251 non-null  int64
 1   Cust_name         11251 non-null  object
 2   Product_ID        11251 non-null  object
 3   Gender            11251 non-null  object
 4   Age Group         11251 non-null  object
 5   Age               11251 non-null  int64
 6   Marital_Status    11251 non-null  int64
 7   State             11251 non-null  object
 8   Zone              11251 non-null  object
 9   Occupation        11251 non-null  object
 10  Product_Category  11251 non-null  object
 11  Orders            11251 non-null  int64
 12  Amount            11239 non-null  float64
 13  Status            0 non-null      float64
 14  unnamed1          0 non-null      float64
dtypes: float64(3), int64(4), object(8)
memory usage: 1.3+ MB
```

[23]:
```
#drop unrelated/blank columns
df.drop(['Status', 'unnamed1'], axis=1, inplace=True)
```

[27]:
```
#check for null values
pd.isnull(df).sum()
```

[27]:
```
User_ID              0
Cust_name            0
Product_ID           0
Gender               0
Age Group            0
Age                  0
Marital_Status       0
State                0
Zone                 0
Occupation           0
Product_Category     0
Orders               0
Amount              12
dtype: int64
```

[34]:
```
# drop null values
df.dropna(inplace=True)
```

[35]:
```
# change data type
df['Amount'] = df['Amount'].astype('int')
```

[36]:
```
df['Amount'].dtypes
```

[36]:
```
dtype('int32')
```

[37]:
```
df.columns
```

[37]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

[38]:
```
#rename column
df.rename(columns= {'Marital_Status':'Shaadi'})
```

[38]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Shaadi
0	1002903	Sanskriti	P00125942	F	26-35	28	0
1	1000732	Kartik	P00110942	F	26-35	35	1
2	1001990	Bindu	P00118542	F	26-35	35	1
3	1001425	Sudevi	P00237842	M	0-17	16	0
4	1000588	Joni	P00057942	M	26-35	28	1
…	…	…	…	…	…	…	…
11246	1000695	Manning	P00296942	M	18-25	19	1
11247	1004089	Reichenbach	P00171342	M	26-35	33	0
11248	1001209	Oshin	P00201342	F	36-45	40	0
11249	1004023	Noonan	P00059442	M	36-45	37	0
11250	1002744	Brumley	P00281742	F	18-25	19	0

```
                State      Zone       Occupation Product_Category  Orders  \
0         Maharashtra   Western       Healthcare             Auto       1
1      Andhra Pradesh  Southern             Govt             Auto       3
2       Uttar Pradesh   Central       Automobile             Auto       3
3           Karnataka  Southern     Construction             Auto       2
4             Gujarat   Western  Food Processing             Auto       2
…                  …         …                …              …     …
11246     Maharashtra   Western         Chemical           Office       4
11247         Haryana  Northern       Healthcare       Veterinary       3
11248  Madhya Pradesh   Central          Textile           Office       4
11249       Karnataka  Southern      Agriculture           Office       3
11250     Maharashtra   Western       Healthcare           Office       3

       Amount
0       23952
1       23934
2       23924
3       23912
4       23877
…         …
11246     370
11247     367
11248     213
11249     206
11250     188

[11239 rows x 13 columns]
```

[39]:
```
# describe() method returns description of the data in the DataFrame (i.e.␣
↪count, mean, std, etc)
df.describe()
```

[39]:
```
             User_ID           Age  Marital_Status        Orders        Amount
count   1.123900e+04  11239.000000    11239.000000  11239.000000  11239.000000
mean    1.003004e+06     35.410357        0.420055      2.489634   9453.610553
std     1.716039e+03     12.753866        0.493589      1.114967   5222.355168
min     1.000001e+06     12.000000        0.000000      1.000000    188.000000
25%     1.001492e+06     27.000000        0.000000      2.000000   5443.000000
50%     1.003064e+06     33.000000        0.000000      2.000000   8109.000000
75%     1.004426e+06     43.000000        1.000000      3.000000  12675.000000
max     1.006040e+06     92.000000        1.000000      4.000000  23952.000000
```

[40]:
```
# use describe() for specific columns
df[['Age', 'Orders', 'Amount']].describe()
```

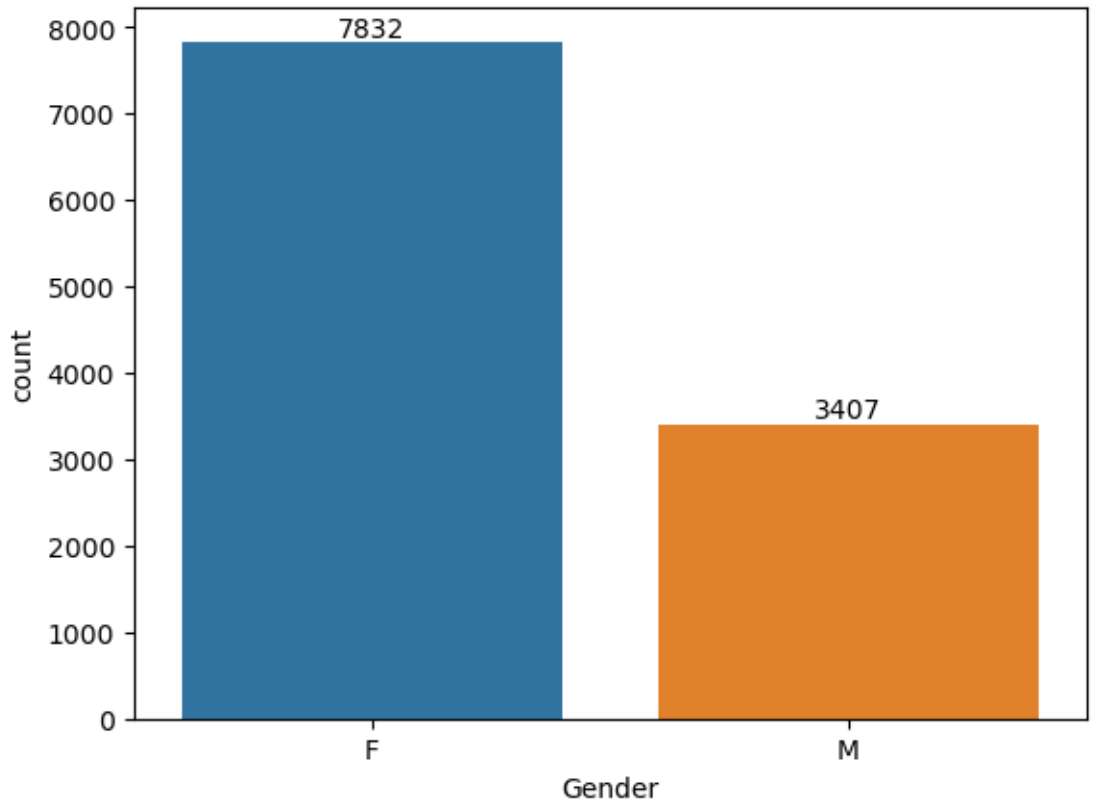
[40]:
```
                Age        Orders        Amount
count  11239.000000  11239.000000  11239.000000
```

```
mean      35.410357      2.489634   9453.610553
std       12.753866      1.114967   5222.355168
min       12.000000      1.000000    188.000000
25%       27.000000      2.000000   5443.000000
50%       33.000000      2.000000   8109.000000
75%       43.000000      3.000000  12675.000000
max       92.000000      4.000000  23952.000000
```

1 Exploratory Data Analysis

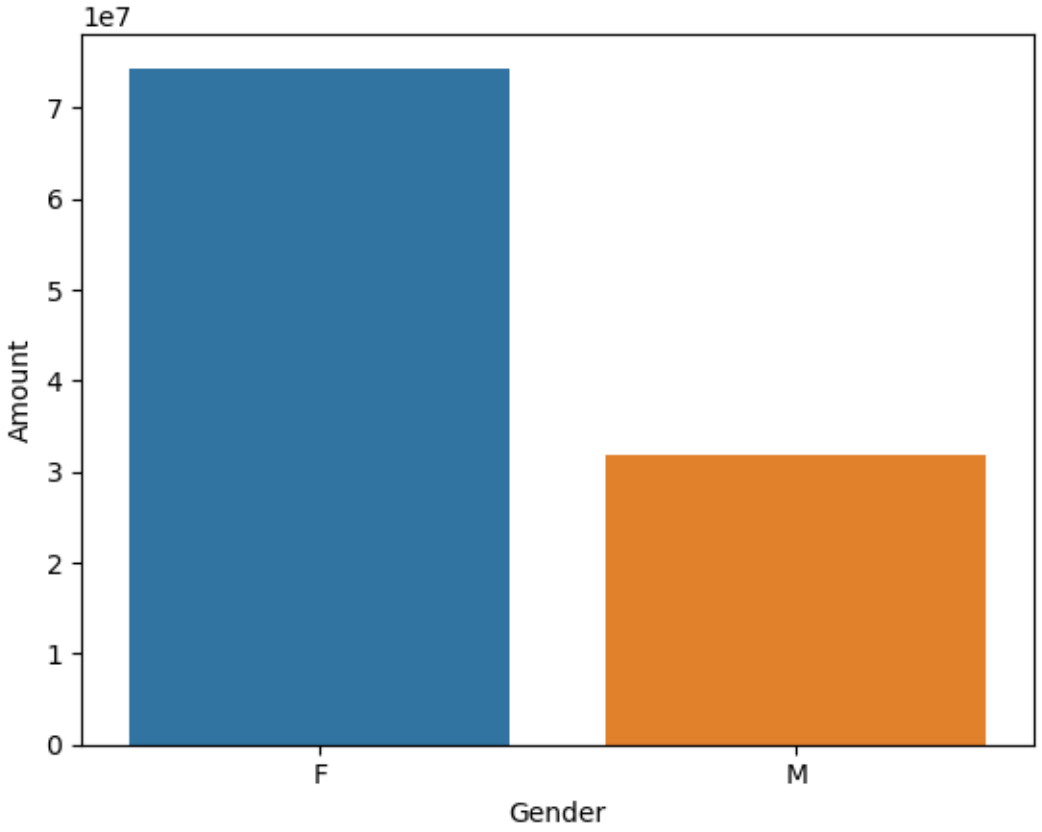
1.0.1 Gender

[41]:
```
# plotting a bar chart for Gender and it's count

ax = sns.countplot(x = 'Gender',data = df)

for bars in ax.containers:
    ax.bar_label(bars)
```



[42]:
```
# plotting a bar chart for gender vs total amount

sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().
 ↪sort_values(by='Amount', ascending=False)

sns.barplot(x = 'Gender',y= 'Amount' ,data = sales_gen)
```

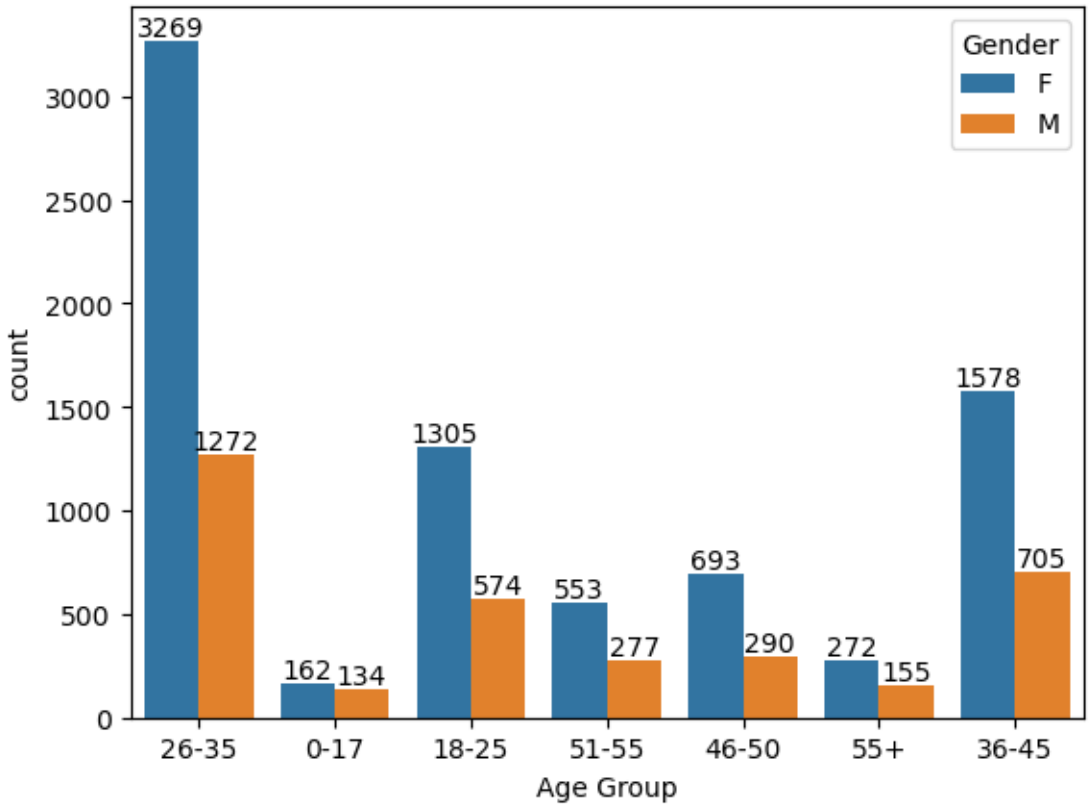
[42]:
```
<AxesSubplot:xlabel='Gender', ylabel='Amount'>
```



From above graphs we can see that most of the buyers are females and even the purchasing power of females are greater than men

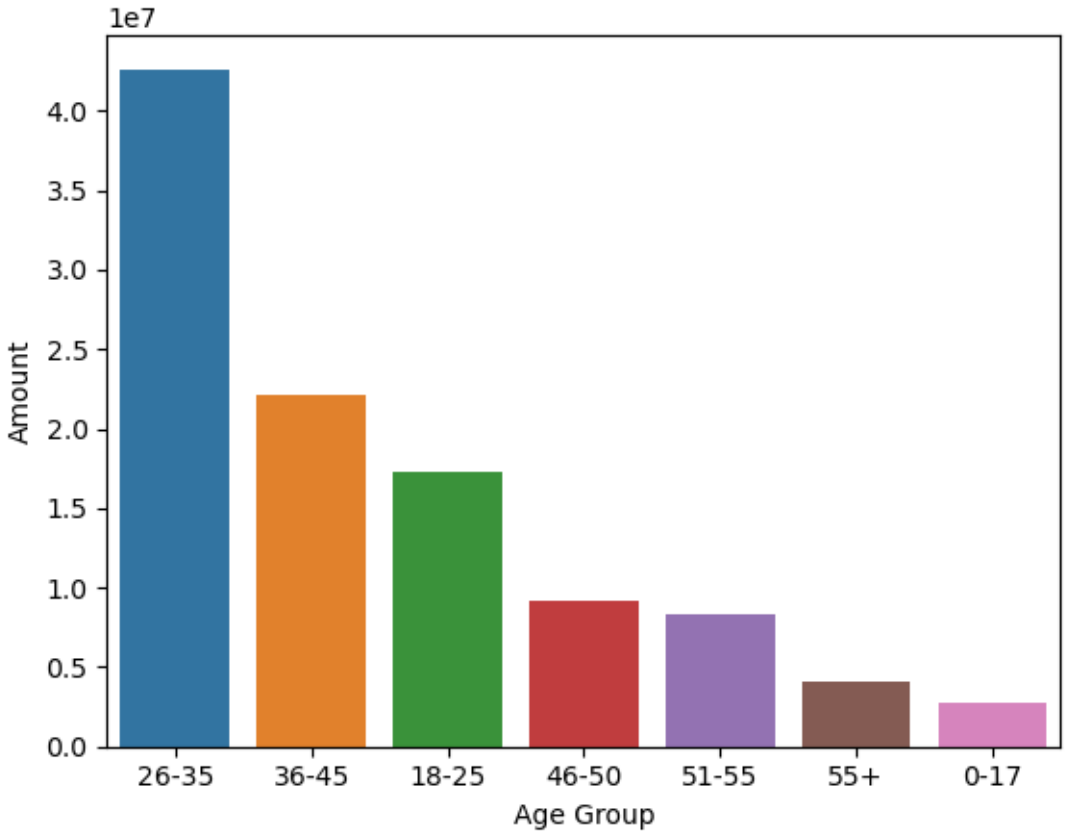
1.0.2 Age

[43]:
```
ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

for bars in ax.containers:
    ax.bar_label(bars)
```



[44]:
```
# Total Amount vs Age Group
sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().
 ↪sort_values(by='Amount', ascending=False)

sns.barplot(x = 'Age Group',y= 'Amount' ,data = sales_age)
```

[44]: <AxesSubplot:xlabel='Age Group', ylabel='Amount'>

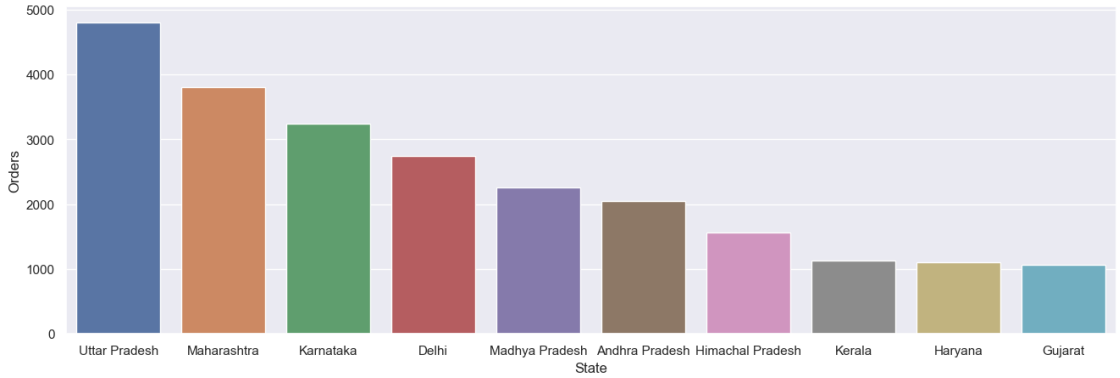


From above graphs we can see that most of the buyers are of age group between 26-35 yrs female

1.0.3 State

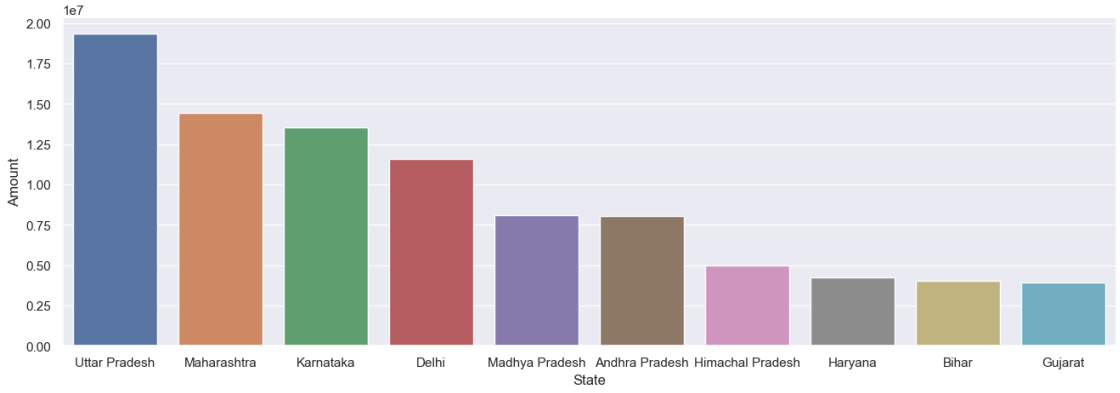
[47]:
```
# total number of orders from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Orders'].sum().
 ↪sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(16,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Orders')
```

[47]:
```
<AxesSubplot:xlabel='State', ylabel='Orders'>
```



[49]:
```
# total amount/sales from top 10 states

sales_state = df.groupby(['State'], as_index=False)['Amount'].sum().
 ↪sort_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(16,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Amount')
```

[49]:
```
<AxesSubplot:xlabel='State', ylabel='Amount'>
```

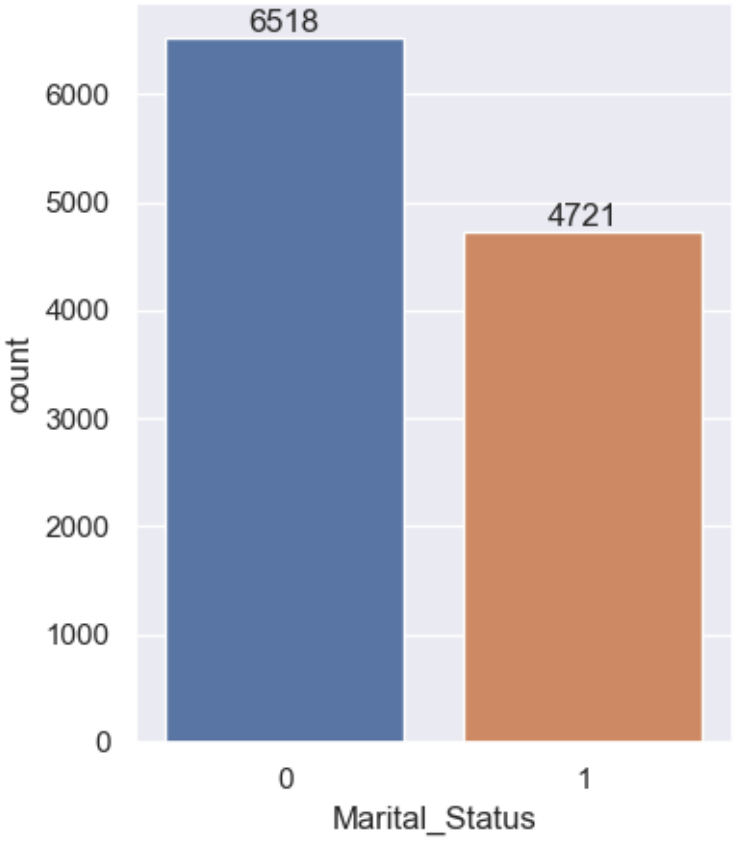


From above graphs we can see that most of the orders & total sales/amount are from Uttar Pradesh, Maharashtra and Karnataka respectively

1.0.4 Marital Status

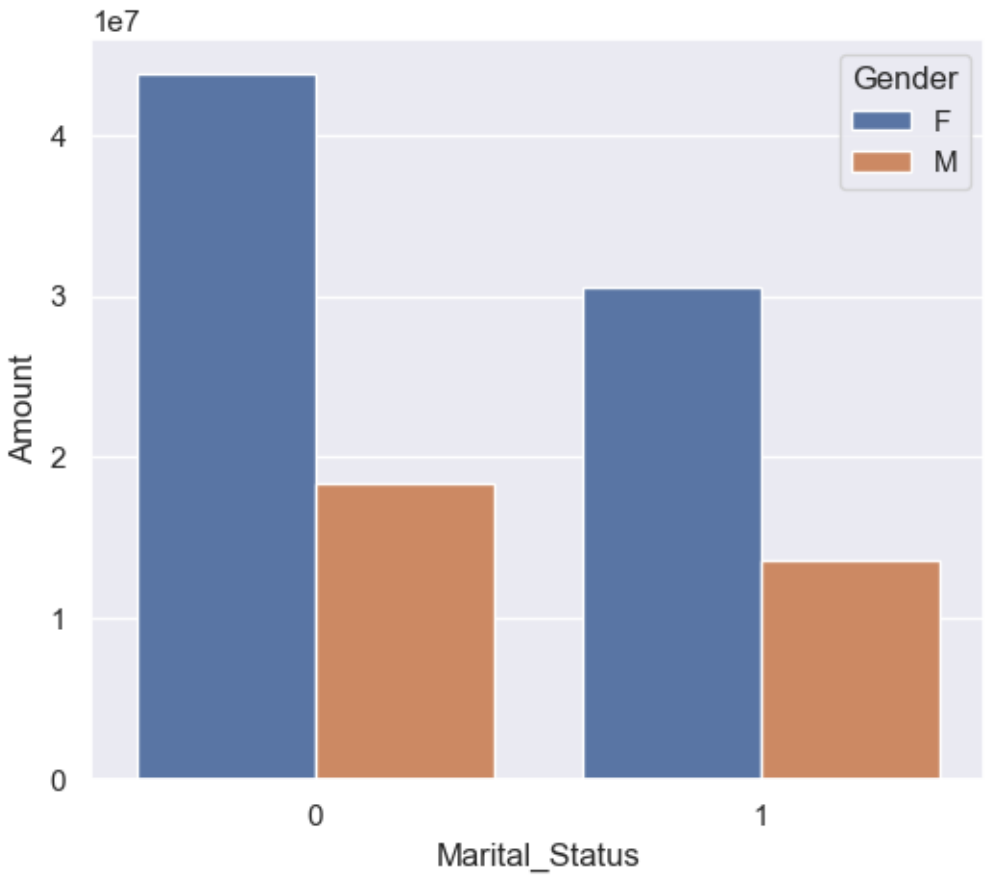
[53]:
```
ax = sns.countplot(data = df, x = 'Marital_Status')

sns.set(rc={'figure.figsize':(4,5)})
for bars in ax.containers:
```

```
    ax.bar_label(bars)
```



[54]:
```
sales_state = df.groupby(['Marital_Status', 'Gender'],␣
 ↪as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.set(rc={'figure.figsize':(6,5)})
sns.barplot(data = sales_state, x = 'Marital_Status',y= 'Amount', hue='Gender')
```

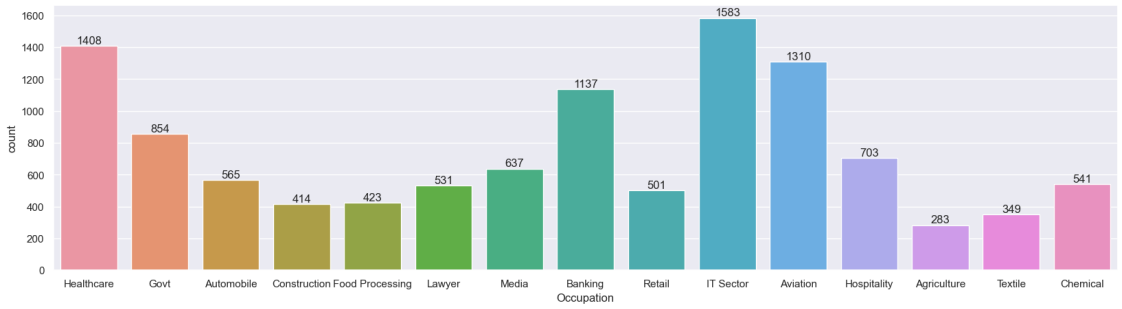
[54]:
```
<AxesSubplot:xlabel='Marital_Status', ylabel='Amount'>
```



From above graphs we can see that most of the buyers are married (women) and they have high purchasing power

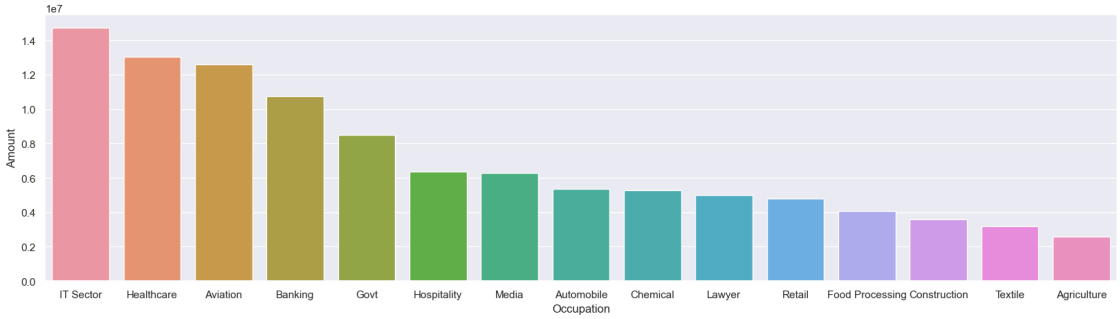
1.0.5 Occupation

[55]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')

for bars in ax.containers:
    ax.bar_label(bars)
```



[56]:
```
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().
↪sort_values(by='Amount', ascending=False)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Occupation',y= 'Amount')
```

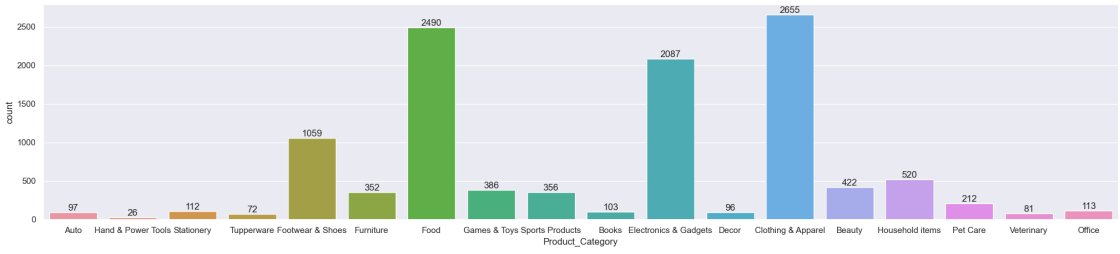
[56]:
```
<AxesSubplot:xlabel='Occupation', ylabel='Amount'>
```



From above graphs we can see that most of the buyers are working in IT, Healthcare and Aviation sector

1.0.6 Product Category

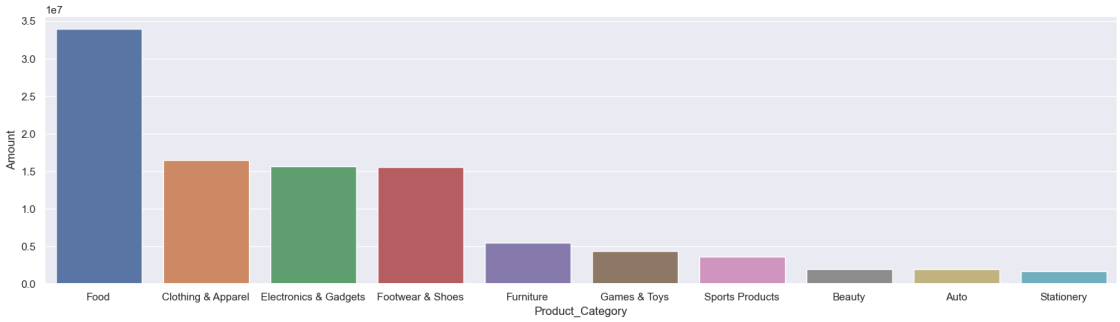
[58]:
```
sns.set(rc={'figure.figsize':(25,5)})
ax = sns.countplot(data = df, x = 'Product_Category')

for bars in ax.containers:
    ax.bar_label(bars)
```



[59]:
```
sales_state = df.groupby(['Product_Category'], as_index=False)['Amount'].sum().
↪sort_values(by='Amount', ascending=False).head(10)
```

```
sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Product_Category',y= 'Amount')
```

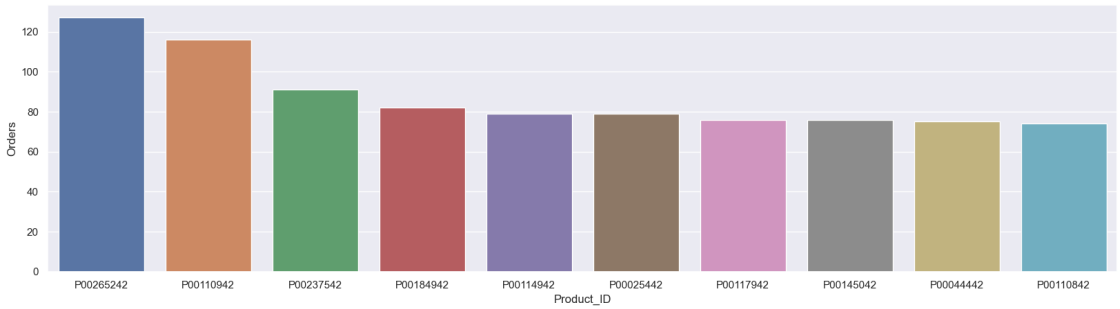
[59]: <AxesSubplot:xlabel='Product_Category', ylabel='Amount'>



From above graphs we can see that most of the sold products are from Food, Clothing and Electronics category

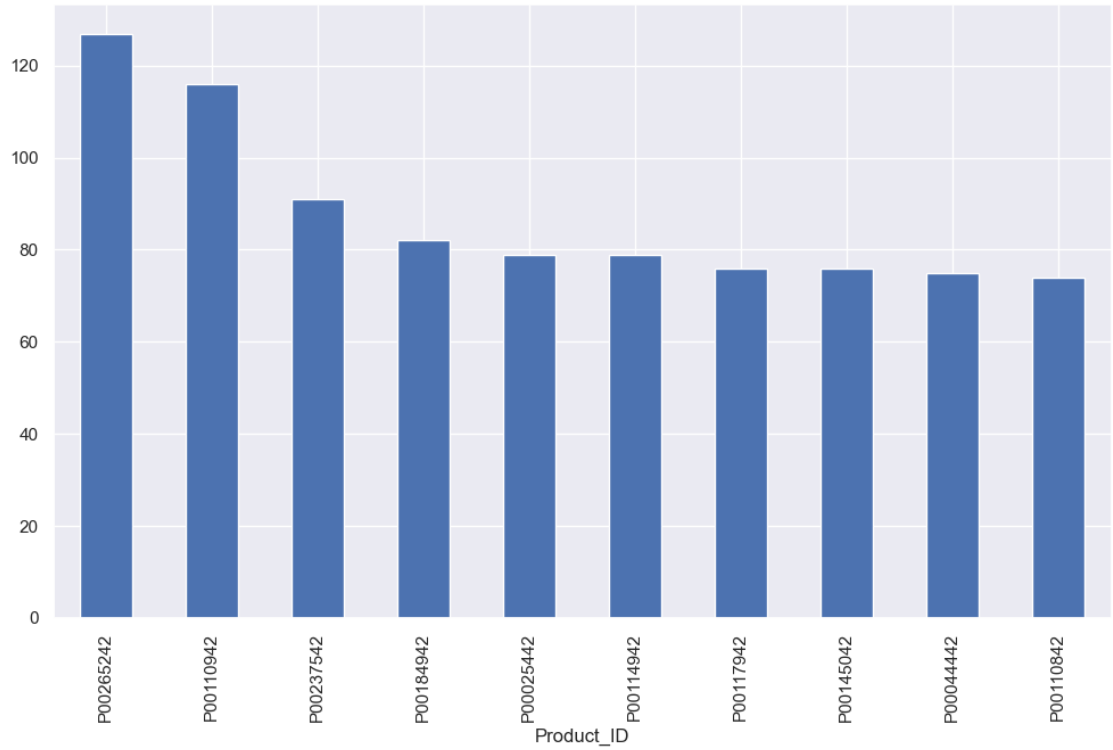
[60]:
```
sales_state = df.groupby(['Product_ID'], as_index=False)['Orders'].sum().
 ↪sort_values(by='Orders', ascending=False).head(10)

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Product_ID',y= 'Orders')
```

[60]: <AxesSubplot:xlabel='Product_ID', ylabel='Orders'>



[61]:
```
# top 10 most sold products (same thing as above)

fig1, ax1 = plt.subplots(figsize=(12,7))
df.groupby('Product_ID')['Orders'].sum().nlargest(10).
 ↪sort_values(ascending=False).plot(kind='bar')
```

[61]: <AxesSubplot:xlabel='Product_ID'>



1.1 Conclusion:

1.1.1

Married women age group 26-35 yrs from UP, Maharastra and Karnataka working in IT, Healthcare and Aviation are more likely to buy products from Food, Clothing and Electronics category