Data Analysis (Exploratory Data Analysis)

February 6, 2024

[9]:
```
 import numpy as np
import pandas as pd
import matplotlib.pyplot as plt #for
visualizing data %matplotlib inline
import seaborn as sns
```

1 Data Cleaning

[14]:
```
df=pd.read_csv("Downloads/Diwali_Sales_Data.csv",encoding="unicode_escape")
#to avoid encoding error, use "unicode_escape"
df.shape
```

[14]: (11251, 15)

[17]:
```
 df.head(10)
```

[17]:

	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
```

[16]:
```
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
```

[18]:
```
#drop unrelated / blank columns
df.drop(['Status','unnamed1'], axis=1 , inplace=True)
```

[19]:
```
df.info() #when we again do df.info() it shows that the two rows
 i.e Status␣ and unnamed1 has been dropped
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
Data columns (total 13 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
dtypes: float64(1), int64(4), object(8)
memory usage: 1.1+ MB
```

[20]:
```
pd.isnull(df)
```

[20]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	\
0	False	False	False	False	False	False	
1	False	False	False	False	False	False	
2	False	False	False	False	False	False	
3	False	False	False	False	False	False	
4	False	False	False	False	False	False	
…	…	…	…	…	…	…	
11246	False	False	False	False	False	False	
11247	False	False	False	False	False	False	
11248	False	False	False	False	False	False	
11249	False	False	False	False	False	False	
11250	False	False	False	False	False	False	

	Marital_Status	State	Zone	Occupation	Product_Category	Orders	\
0	False	False	False	False	False	False	
1	False	False	False	False	False	False	
2	False	False	False	False	False	False	
3	False	False	False	False	False	False	
4	False	False	False	False	False	False	
…	…	…	…	…	…	…	
11246	False	False	False	False	False	False	
11247	False	False	False	False	False	False	
11248	False	False	False	False	False	False	
11249	False	False	False	False	False	False	

```
11250         False False False      False           False  False

       Amount
0       False
1       False
2       False
3       False
4       False
……
11246   False
11247   False
11248   False
11249   False
11250   False

[11251 rows x 13 columns]
```

[21]:
```
#checking for null values
pd.isnull(df).sum()
```

[21]:
```
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
```

[22]:
```
df.shape
```

[22]:
```
(11251, 13)
```

[24]:
```
#drop null values
df.dropna(inplace=True)
```

[25]:
```
df.shape #here we can see that 12 rows which are null are deducted
 which is 11251 initially
```

[25]:
```
(11239, 13)
```

[26]:
```
#change data type of 'Amount' as it is in float
df["Amount"]=df["Amount"].astype('int')
```

[27]:
```
df["Amount"].dtype #after run this we can see that data type of
 'Amount' changes from float to int
```

[27]:
```
dtype('int32')
```

[28]:
```
 df.columns
```

[28]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age
      Group', 'Age', 'Marital_Status', 'State', 'Zone',
      'Occupation', 'Product_Category', 'Orders', 'Amount'],
     dtype='object')
```

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

[29]:
```
 #describe() method return description of the data in the
  database (i.e count,mean,std etc.)
df[["Age","Orders","Amount"]].describe()
```

[29]:

	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

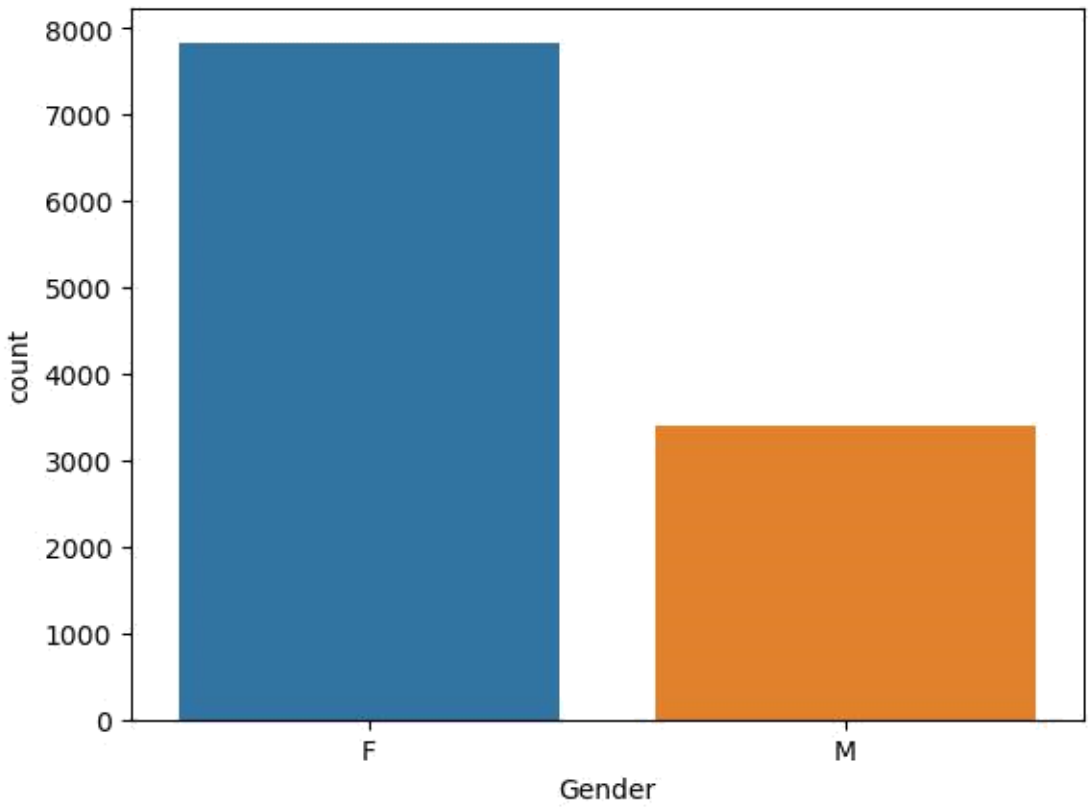
2 Exploratory Data Analysis

3 1. Gender

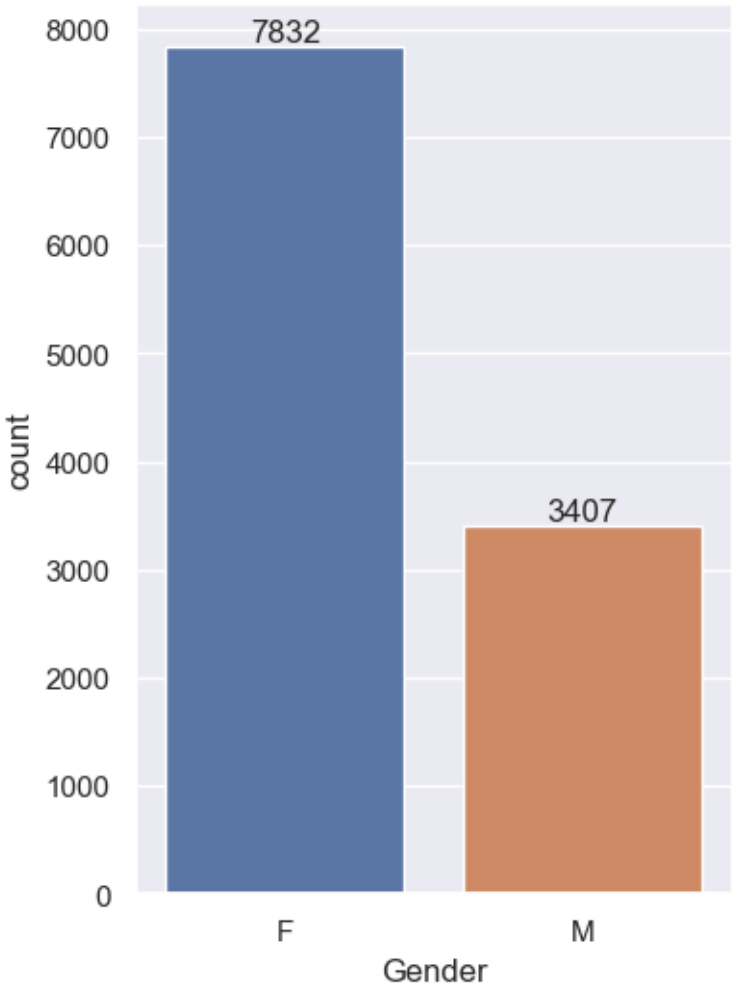
[32]:
```
 df.columns
```

[32]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age
      Group', 'Age', 'Marital_Status', 'State', 'Zone',
      'Occupation', 'Product_Category', 'Orders', 'Amount'],
     dtype='object')
```

[33]:
```
 ax = sns.countplot(x ="Gender",data=df)
```



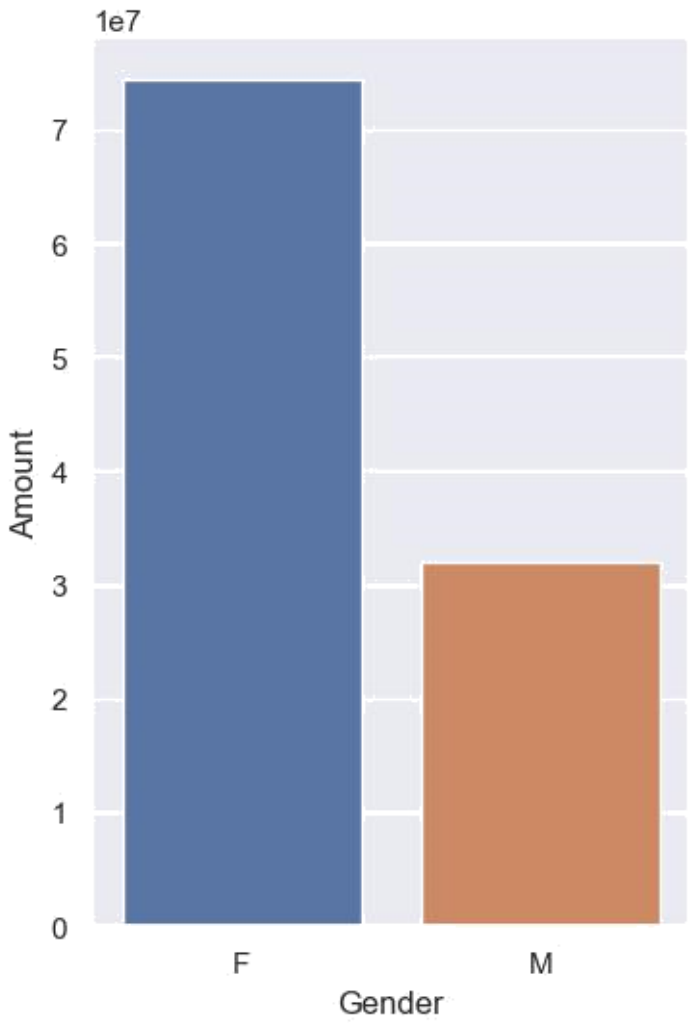
[88]:
```
ax = sns.countplot(x ="Gender",data=df)

sns.set(rc={"figure.figsize":(4,6)})
for bars in ax.containers: #for values of no. of females and males i.e we can␣
 ↪see there are 7832 females and 3407 males...
    ax.bar_label(bars)
```



[37]:
```
sales_gen = df.groupby(['Gender'],as_index=False)['Amount'].sum().
  sort_values(by="Amount",ascending=False)
```

[87]:
```
sales_gen = df.groupby(['Gender'],as_index=False)['Amount'].sum().
  sort_values(by="Amount",ascending=False)
sns.set(rc={"figure.figsize":(4,6)})
sns.barplot(x="Gender",y="Amount", data=sales_gen)
```

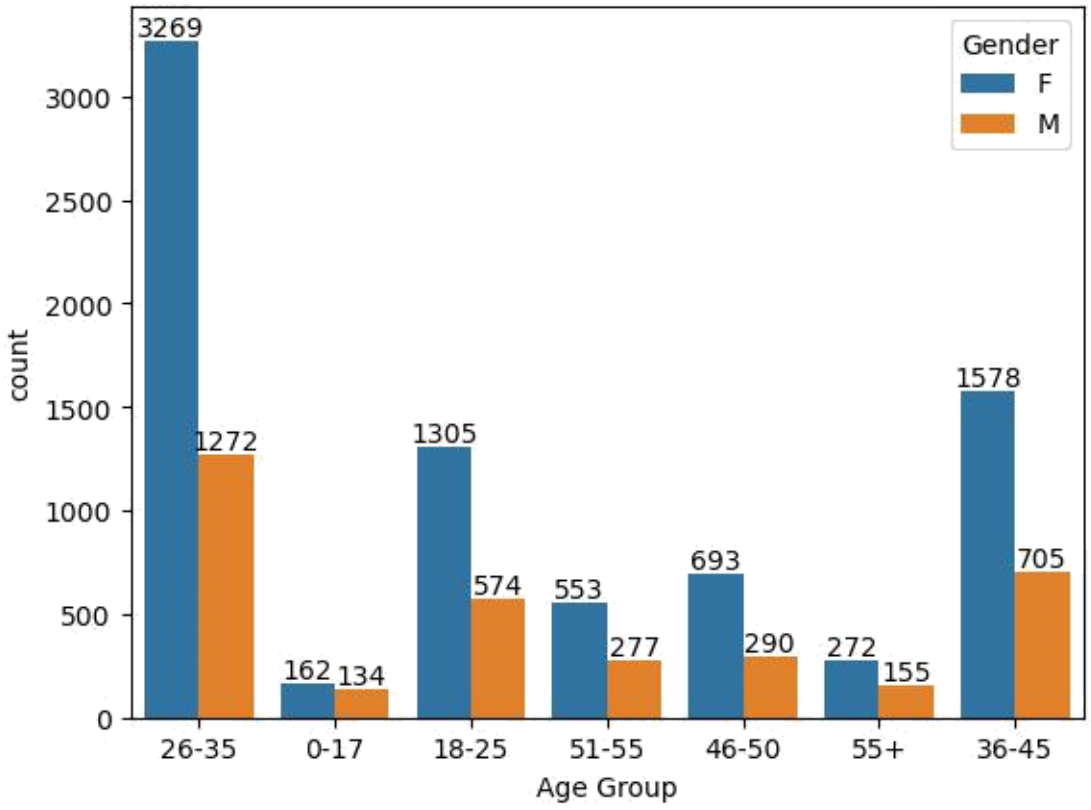
[87]: <Axes: xlabel='Gender', ylabel='Amount'>



From above graph we can see that most of the buyers are females and even the purchasing power of females are greater than males.

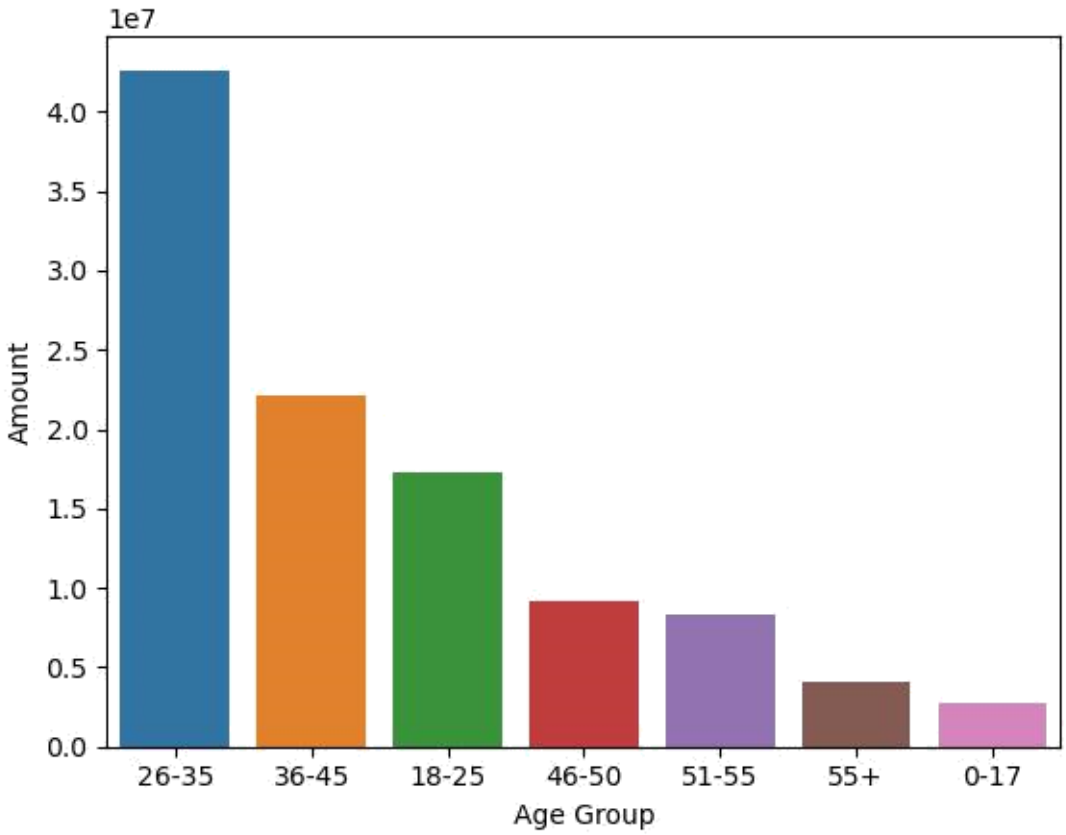
4 2. Age

[38]:
```
ax = sns.countplot(data=df, x= "Age Group",hue="Gender") #hue
 will divide␣ ↪gender bars in male and female


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



[40]:
```
#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)
```

[40]: <Axes: xlabel='Age Group', ylabel='Amount'>

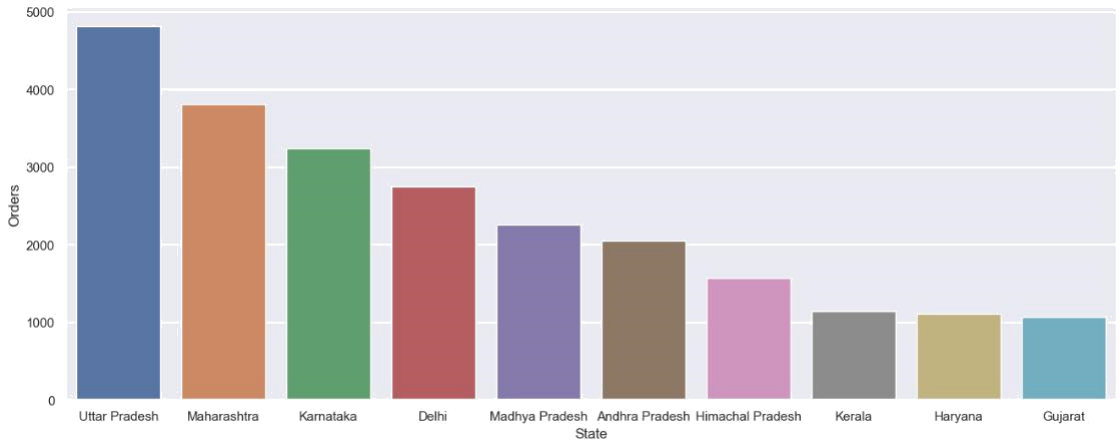


From above Graphs we can see that most of the buyers are of age group between 26-35 years (female)

5 3. State

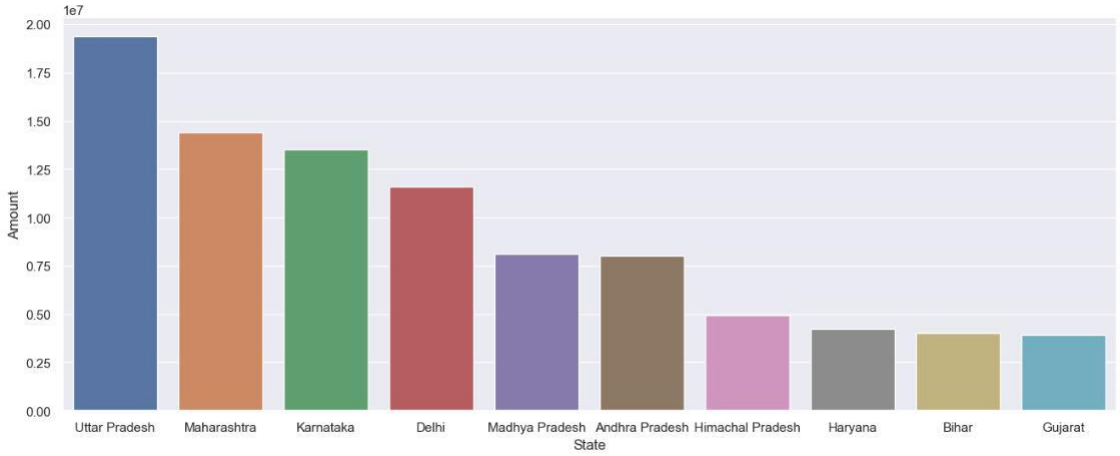
[49]:
```
#Total no. of orders from top 10 States

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

sns.set(rc={"figure.figsize":(16,6)}) #to give size of the
chart/graph sns.barplot(data = sales_state,x="State", y="Orders")
```

[49]: <Axes: xlabel='State', ylabel='Orders'>



[51]:
```
#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":(16,6)}) #to give size of the
chart/graph sns.barplot(data = sales_state,x="State", y="Amount")
```

[51]: <Axes: xlabel='State', ylabel='Amount'>



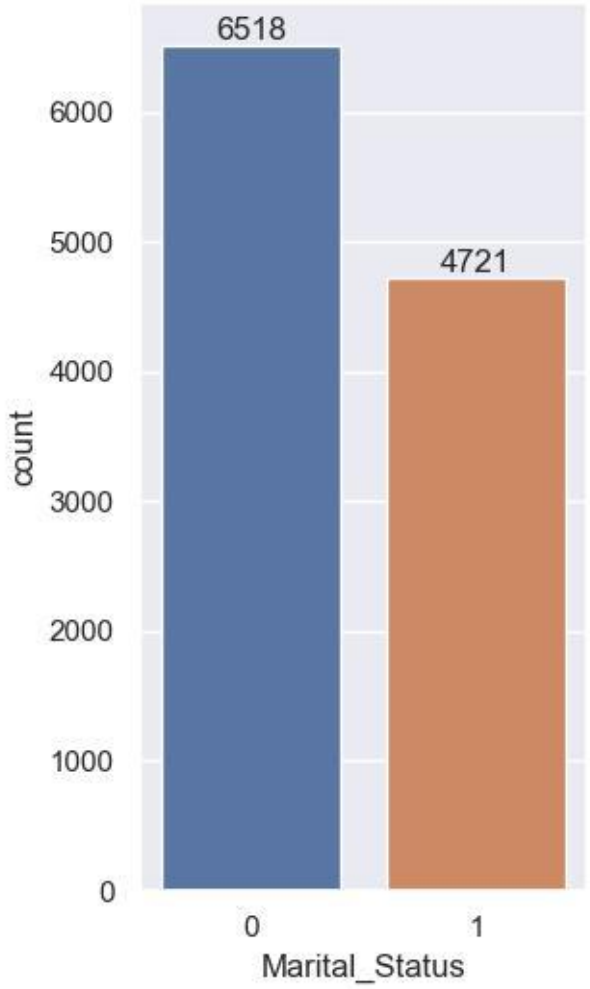
From above Graphs we can see that most of the orders and total sales/amount are from Uttar Pradesh, Maharashtra, Karnataka respectively.

6 4.Martial Status

[55]:
```
df.columns
```

[55]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age
       Group', 'Age', 'Marital_Status', 'State', 'Zone',
       'Occupation', 'Product_Category', 'Orders', 'Amount'],
      dtype='object')
```

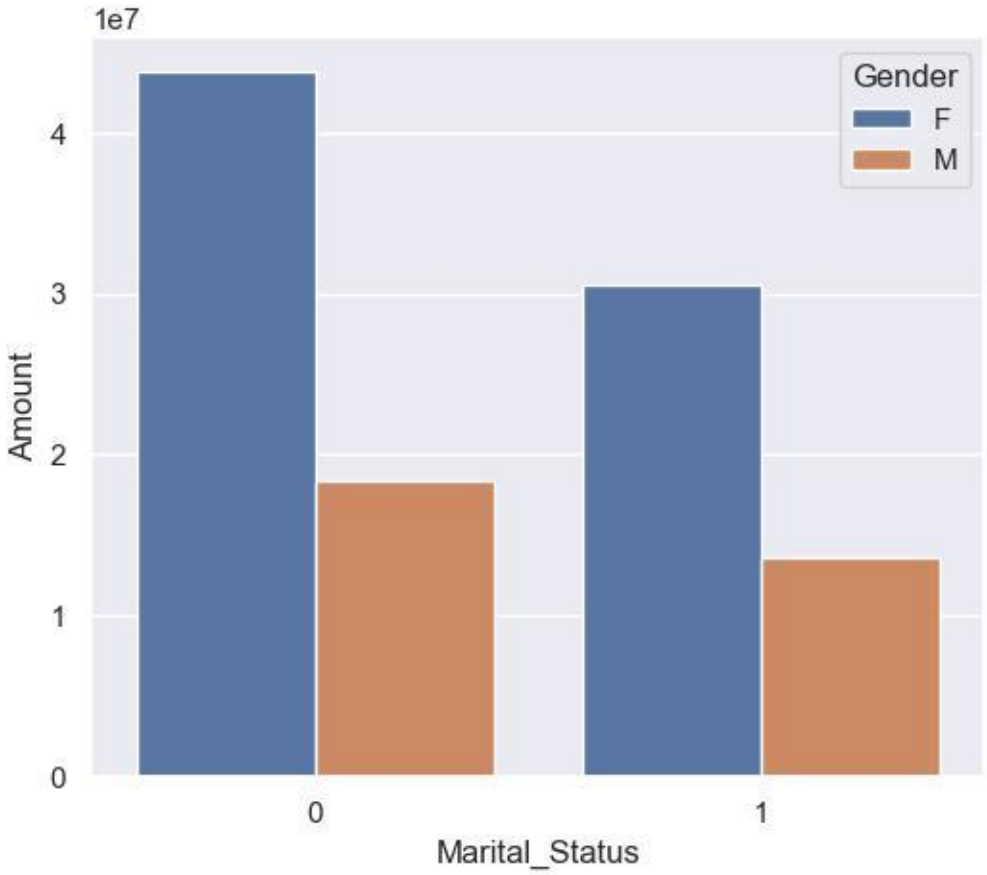
[62]:
```
ax = sns.countplot(data=df , x ="Marital_Status")

sns.set(rc={"figure.figsize":(3,7)})
for bars in ax.containers:
    ax.bar_label(bars)
```



[64]:
```
sales_state = df.groupby(["Marital_Status","Gender"],as_index=False).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")
```

```
C:\Users\Dell\AppData\Local\Temp\ipykernel_18684\2957534978.py:1: FutureWarning:
The default value of numeric_only in DataFrameGroupBy.sum is deprecated. In a
future version, numeric_only will default to False. Either specify numeric_only
or select only columns which should be valid for the function.
  sales_state = df.groupby(["Marital_Status","Gender"],as_index=False).sum().sor
t_values(by="Amount",ascending = False)
```

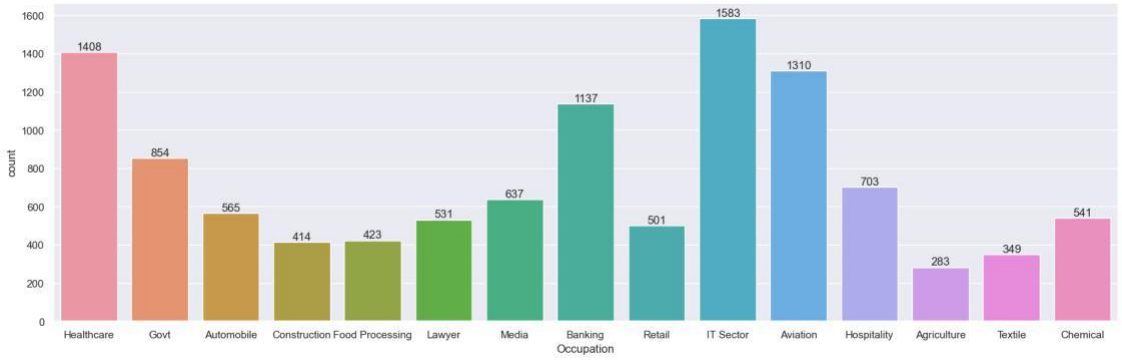
[64]: <Axes: xlabel='Marital_Status', ylabel='Amount'>



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

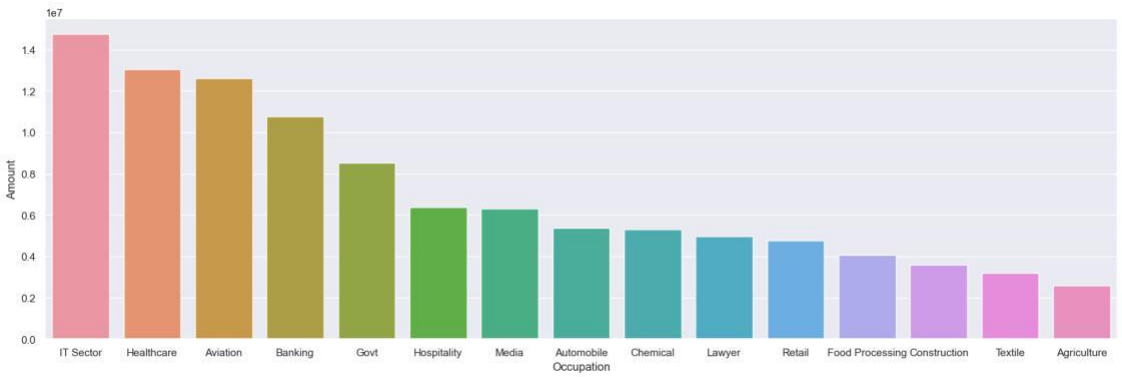
7 5. Occupation

[66]:
```
  sns.set(rc={'figure.figsize':(20,6)})
ax= sns.countplot(data=df, x="Occupation")

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



[68]:
```
sales_state = df.groupby(["Occupation"], as_index=False)["Amount"].sum().
 sort_values(by="Amount",ascending=False)
sns.set(rc={"figure.figsize":(20,6)})
sns.barplot(data = sales_state , x="Occupation",y ="Amount")
```

[68]: <Axes: xlabel='Occupation', ylabel='Amount'>



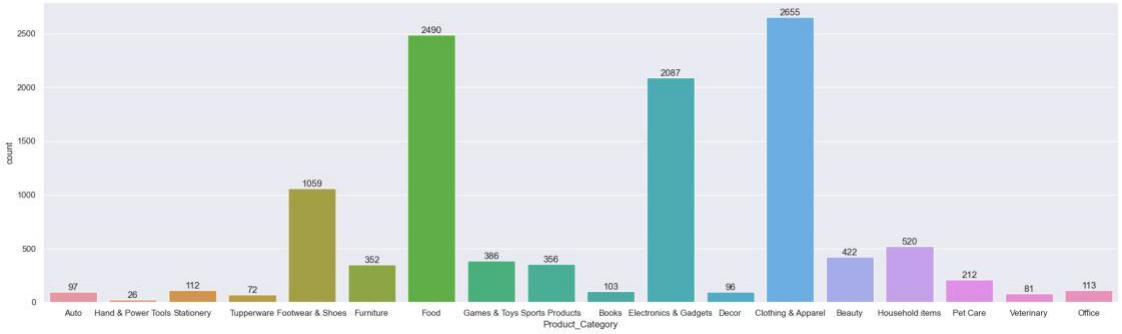
From above graphs we can see that most of the buyers are working in IT, Aviation and Healthcare sector.

[69]:
```
  df.columns
```

[69]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age
       Group', 'Age', 'Marital_Status', 'State', 'Zone',
       'Occupation', 'Product_Category', 'Orders', 'Amount'],
      dtype='object')
```

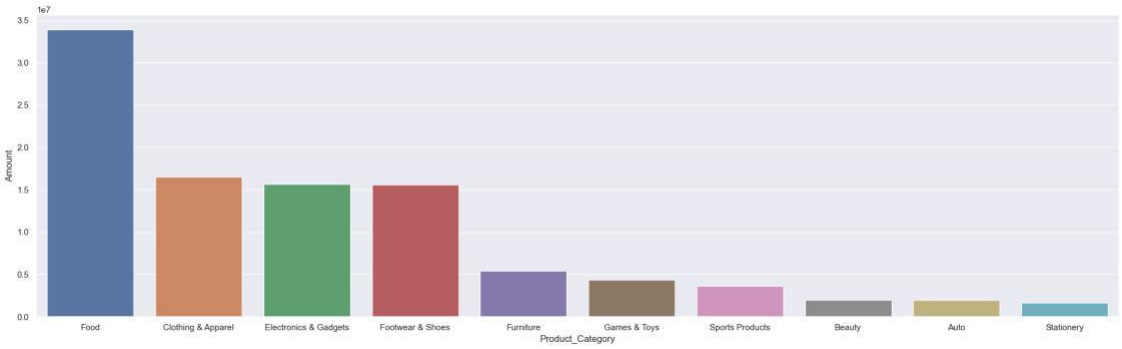
8 *6. Occupation*

[72]:
```
  sns.set(rc={'figure.figsize':(25,7)})
ax= sns.countplot(data=df, x="Product_Category")

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



[79]:
```
sales_state = df.groupby(["Product_Category"], as_index=False)["Amount"].sum().
 sort_values(by="Amount",ascending=False).head(10)
sns.set(rc={"figure.figsize":(25,7)})
sns.barplot(data = sales_state , x="Product_Category",y ="Amount")
```

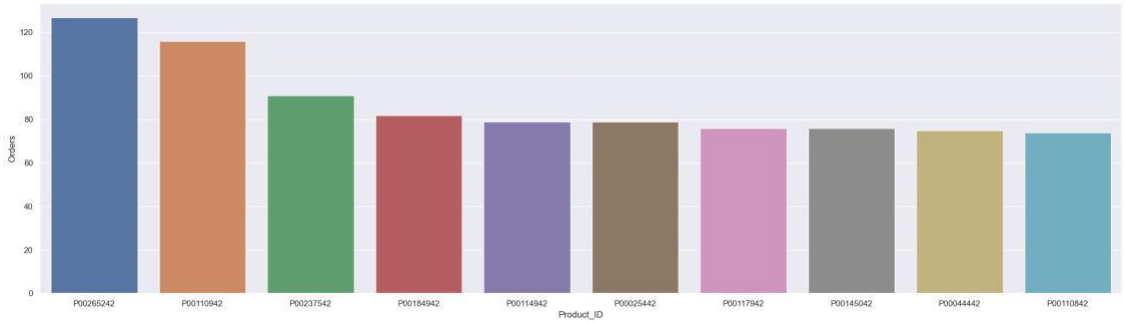
[79]: <Axes: xlabel='Product_Category', ylabel='Amount'>



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

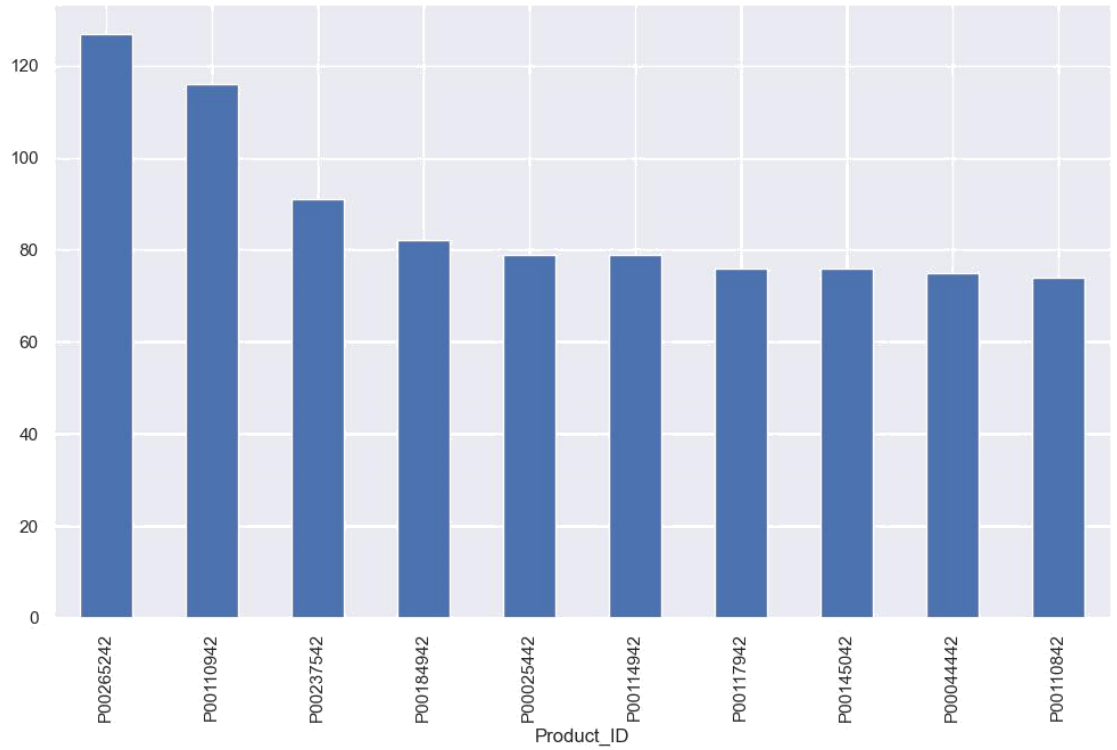
[82]:
```
sales_state = df.groupby(["Product_ID"], as_index=False)["Orders"].sum().
 sort_values(by="Orders",ascending=False).head(10)
sns.set(rc={"figure.figsize":(26,7)})
sns.barplot(data = sales_state , x="Product_ID",y ="Orders")
```

[82]: <Axes: xlabel='Product_ID', ylabel='Orders'>



[83]:
```
 #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")
```

[83]: <Axes: xlabel='Product_ID'>



9 Conclusion

Married women age group 26-35 years from UP, Maharashtra and Karnataka working in IT, Health-care and Aviation are more likely to buy products from Food, Clothing and Electronics category.