Amazon Diwali Sales - Exploratory Data Analytics Project

July 5, 2024

1 Data Cleaning

[1]:
```
# importing necessary packages

import pandas as pd # working with data sets
import numpy as np # perform mathematical operations on arrays
import matplotlib.pyplot as plt # visualizing data
%matplotlib inline
import seaborn as sns # visualizing data
```

[2]:
```
# import .CSV data file
df=pd.read_csv('Amazon Diwali Sales Data.csv', encoding = 'unicode_escape')
```

[3]:
```
# rows and columns count
df.shape
```

[3]: (11251, 15)

[4]:
```
# data view
df.head()
```

[4]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnamed1
0	1002903	Sanskriti	P00125942	F	26-35	28	Married	Maharashtra	Western	Healthcare	Auto	1	23952.0	NaN	NaN
1	1000732	Kartik	P00110942	F	26-35	35	Unmarried	Andhra Pradesh	Southern	Govt	Auto	3	23934.0	NaN	NaN
2	1001990	Bindu	P00118542	F	26-35	35	Unmarried	Uttar Pradesh	Central	Automobile	Auto	3			
3	1001425	Sudevi	P00237842	M	0-17	16	Married	Karnataka	Southern	Construction	Auto	2			
4	1000588	Joni	P00057942	M	26-35	28	Unmarried	Gujarat	Western	Food Processing	Auto	2			

```
 2  23924.0     NaN       NaN
 3  23912.0     NaN       NaN
 4  23877.0     NaN       NaN
```

[5]:
```
# data general info
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  object 
 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(3), object(9)
memory usage: 1.3+ MB
```

[6]:
```
# drop columns
df.drop(['Status', 'unnamed1'], axis=1, inplace=True)
```

[7]:
```
df.info()
```

```
<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  object
 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(3), object(9)
memory usage: 1.1+ MB
```

[8]:
```
# check null values
pd.isnull(df).sum()
```

[8]:
```
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
```

[9]:
```
# drop null values
df.dropna(inplace= True)
```

[10]:
```
pd.isnull(df).sum()
```

[10]:
```
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              0
dtype: int64
```

[11]:
```
# change data type
df['Amount']= df['Amount'].astype('int')
```

[12]:
```
df['Amount'].dtype
```

[12]:
```
dtype('int32')
```

[13]:
```
# retrieve columns
df.columns
```

[13]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

[14]:
```
# rename columns
df.rename(columns={'Cust_name': 'Customer_Name', 'Age Group': 'Age_Group'},␣
 ↪inplace= True)
```

[15]:
```
df.columns
```

[15]:
```
Index(['User_ID', 'Customer_Name', 'Product_ID', 'Gender', 'Age_Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

[16]:
```
# data description
df.describe()
```

[16]:

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

[17]:
```
df[['Age', 'Orders', 'Amount']].describe()
```

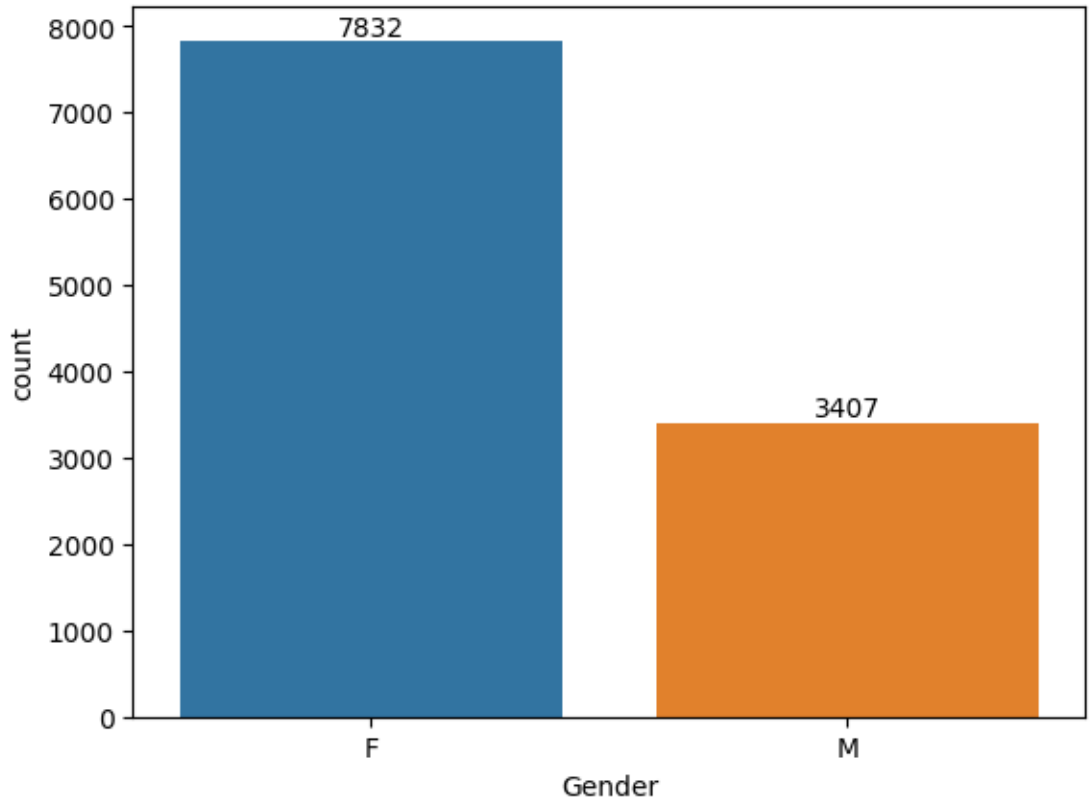
[17]:

	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

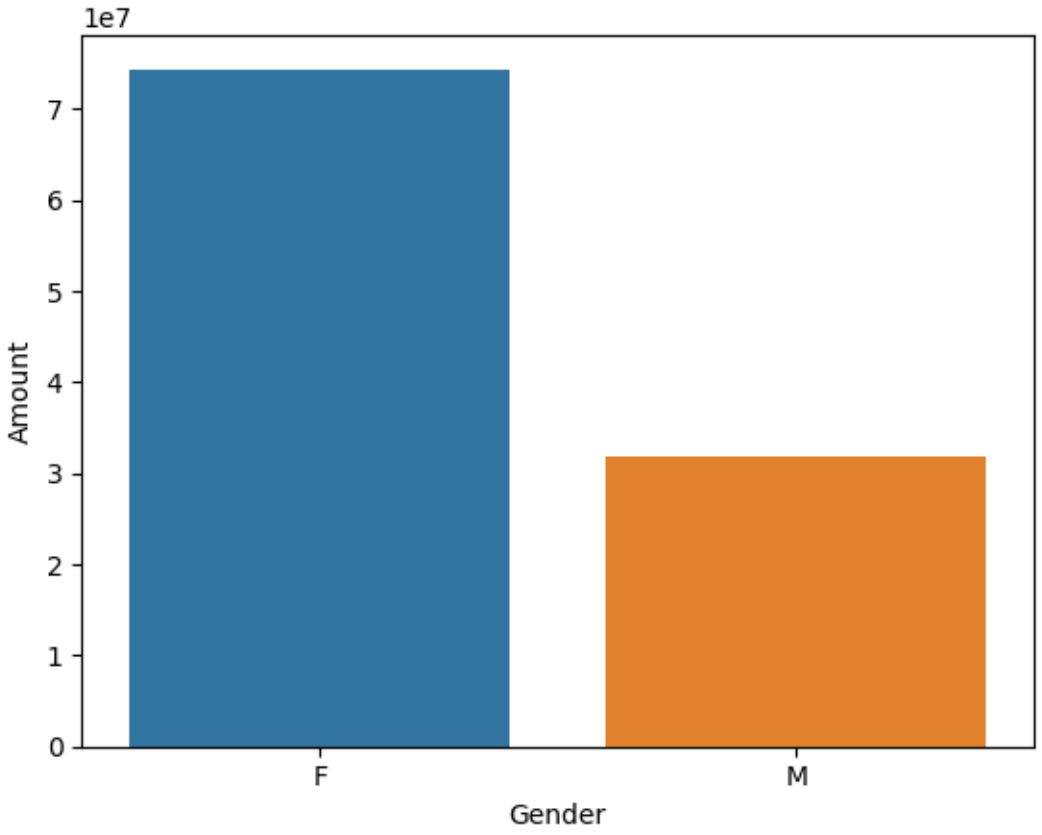
2.1 Gender

[18]:
```
# plotting a bar chart for Gender and it's count

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

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



[19]:
```
# 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, hue= 'Gender')
```

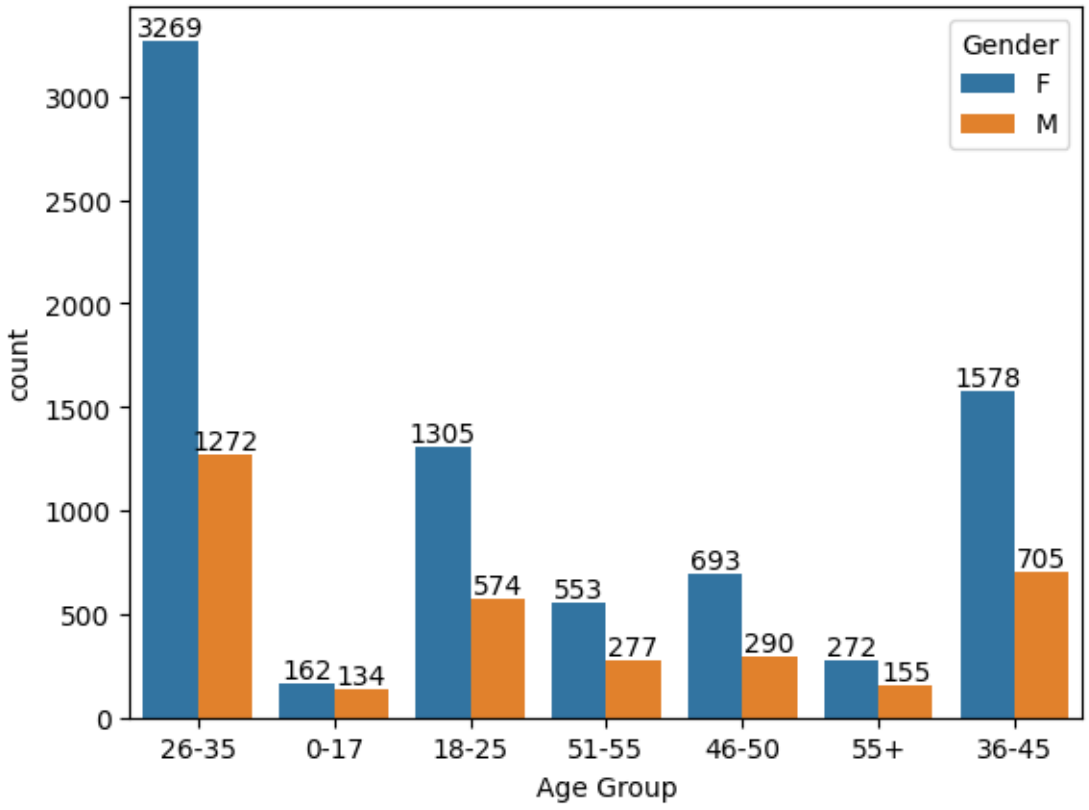
[19]:
```
<Axes: xlabel='Gender', ylabel='Amount'>
```



From the above graphs we can see that most of the buyers are Females and even the purchasing power of Females are greater than Men.

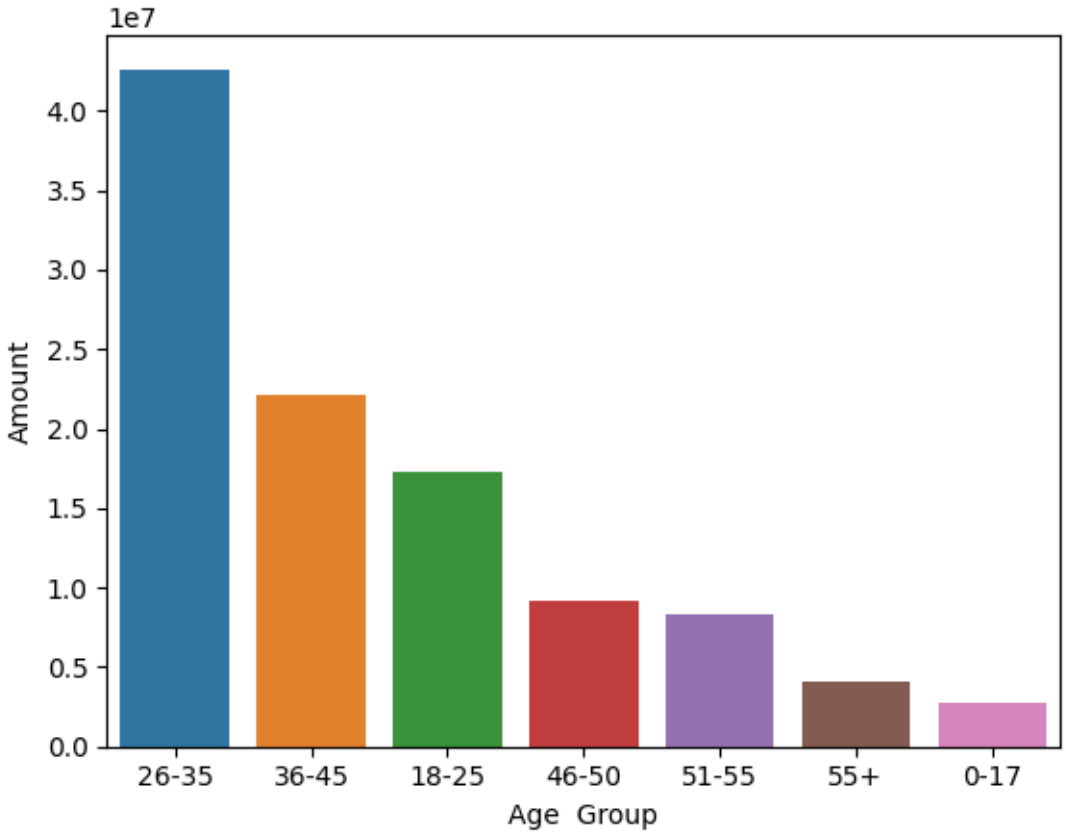
2.2 Age

[20]:
```
# plotting a bar chart for Age Group/Gender and it's count

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

for bars in ax.containers:
    ax.bar_label(bars)
    ax.set(xlabel='Age Group')
```



[21]:
```
# Total Amount vs Age Group

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

ax= sns.barplot(x= 'Age_Group', y= 'Amount', data = sales_age, hue= 'Age_Group')
ax.set(xlabel='Age  Group')
```

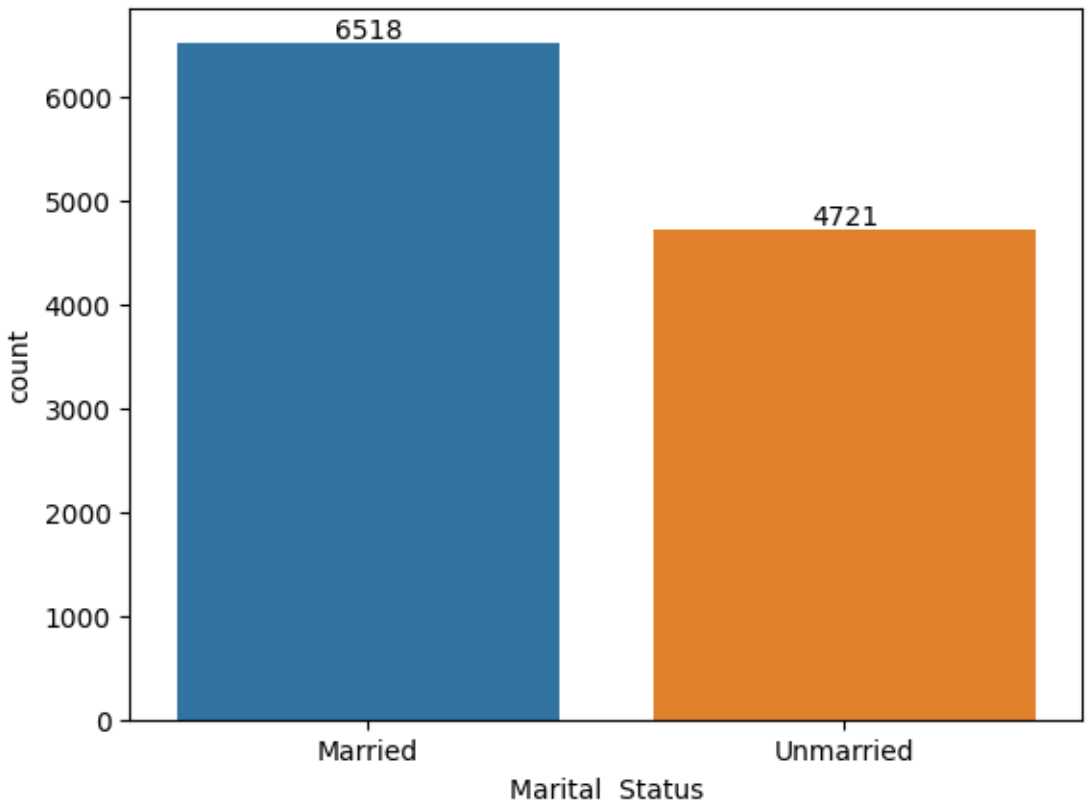
[21]:
```
[Text(0.5, 0, 'Age  Group')]
```



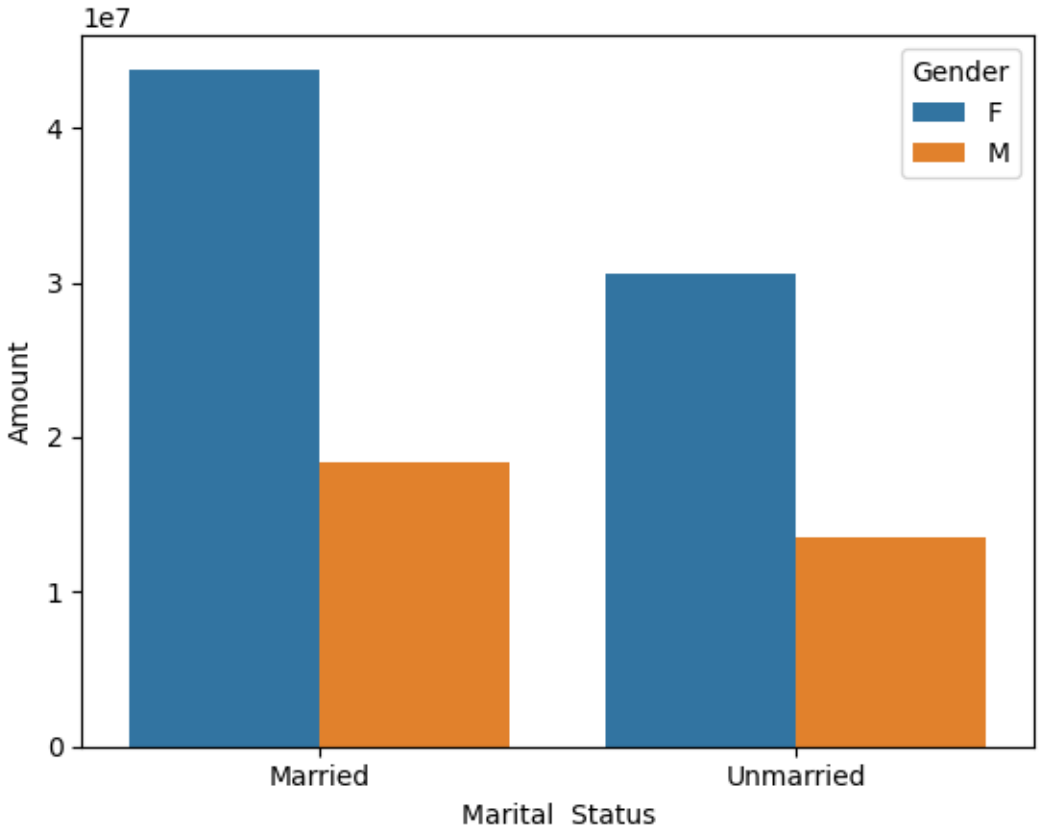
From the above graphs we can see that most of the buyers are of age group between 26-35 yrs Females.

2.3 Marital Status

[22]:
```
# plotting a bar chart for Marital Status and it's count

ax= sns.countplot(data= df, x= 'Marital_Status', hue= 'Marital_Status', legend=␣
 ↪False)

for bars in ax.containers:
    ax.bar_label(bars)
    ax.set_xlabel('Marital  Status')
```



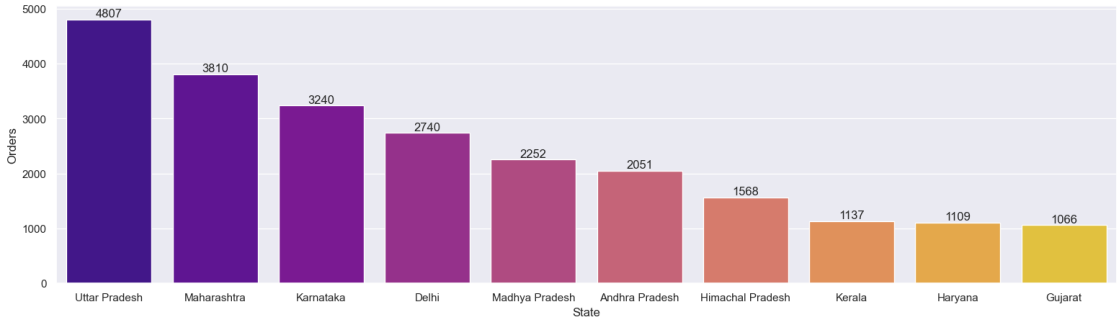
[23]:
```
# total amount vs marital status

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

ax= sns.barplot( data= sales_amount, x= 'Marital_Status', y= 'Amount', hue=␣
 ↪'Gender').set_xlabel('Marital  Status')
```



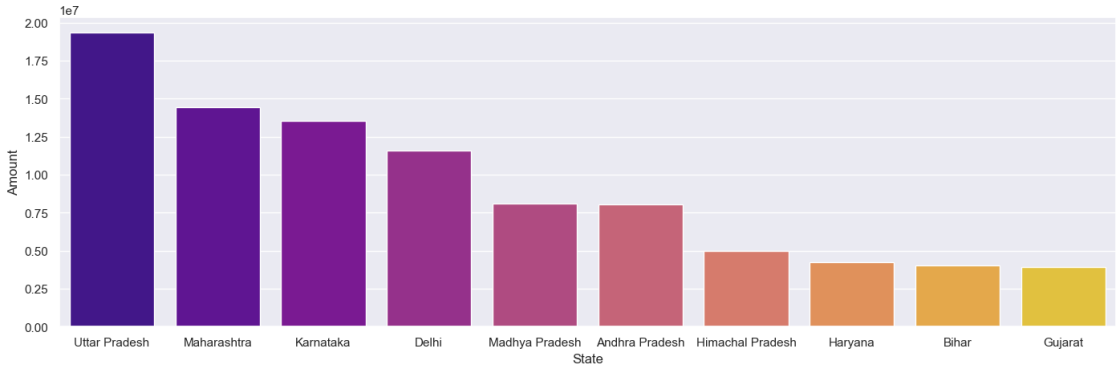
From the above graph we can say that most of the buyers are Married (Women) and they have high purchasing power.

2.4 State

[31]:
```
# 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)

ax= sns.barplot( data= sales_state, x= 'State', y= 'Orders', hue= 'State',␣
 ↪palette= 'plasma')
sns.set(rc={'figure.figsize': (17,5)})

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



[25]:
```
# 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.barplot( data= sales_state, x= 'State', y= 'Amount', hue= 'State',␣
 ↪palette='plasma')
sns.set(rc={'figure.figsize': (17,5)})
```



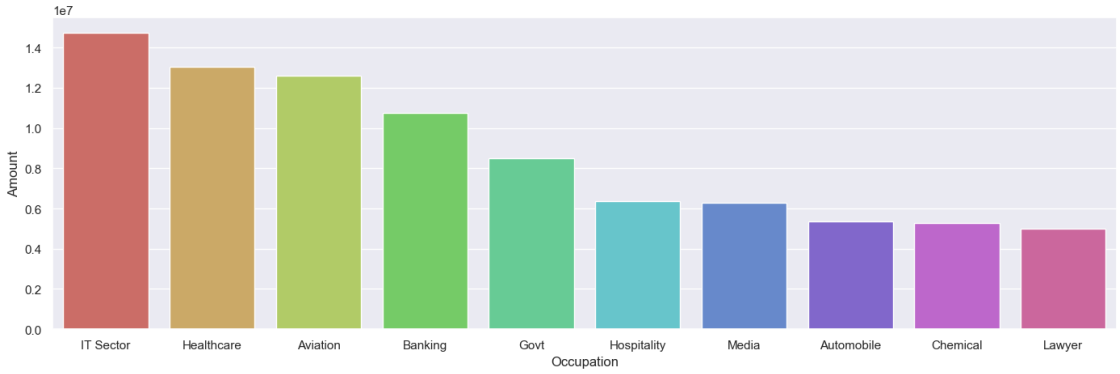
From the above graphs we can see that most of the Orders and Total Amonut/Sales are from Uttar Pradesh, Maharastra and Karnataka respectively.

2.5 Occupation

[34]:
```
# plotting a count chart for top 10 Occupation

sales_occ = df['Occupation'].value_counts().nlargest(10).index

ax= sns.countplot(data= df, x= 'Occupation', order= sales_occ, hue=␣
 ↪'Occupation', palette= 'hls')
sns.set(rc={'figure.figsize':(17,5)})
```

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

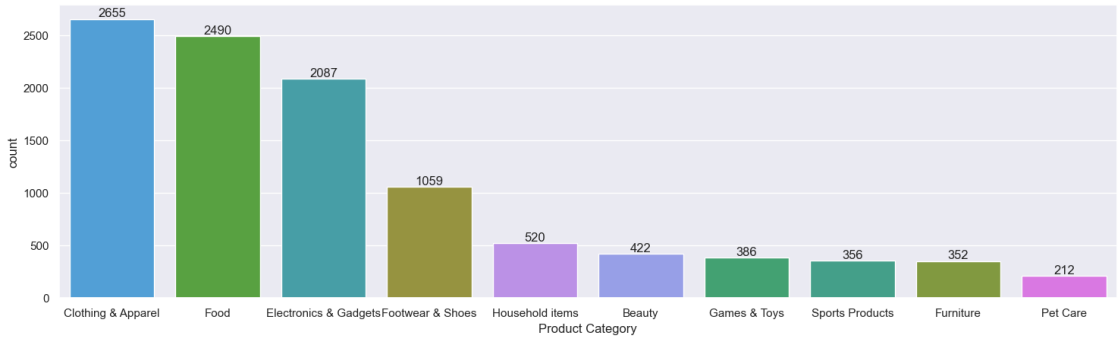
[35]:
```
# total amount vs Occupation

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

sns.barplot(data= sales_occ, x= 'Occupation', y= 'Amount', hue= 'Occupation',␣
 ↪palette= 'hls')
sns.set(rc= {'figure.figsize': (17,5)})
```



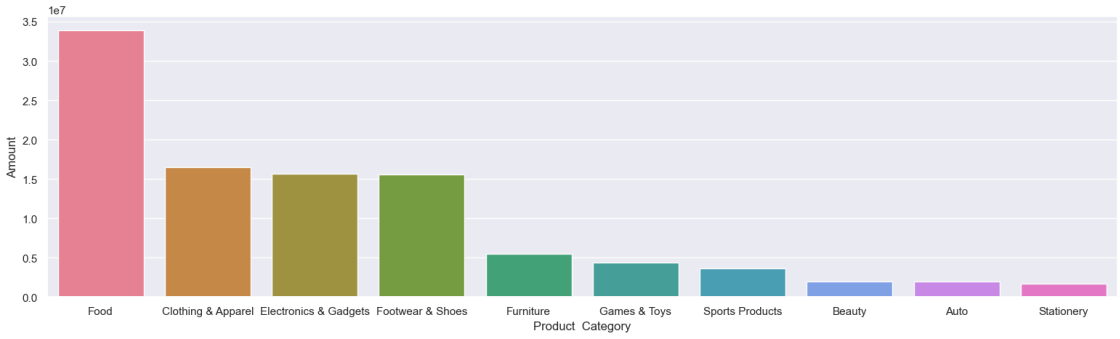
From the above graph we can see that most of the buyers are working in IT, Aviation and Healthcare sector.

2.6 Product Category

[37]:
```
# plotting a count chart for top 10 Product Category

sales_pro = df['Product_Category'].value_counts().nlargest(10).index

ax = sns.countplot(data=df, x='Product_Category', hue='Product_Category',␣
 ↪order= sales_pro, palette='husl')
sns.set(rc={'figure.figsize':(18,5)})

for container in ax.containers:
    ax.bar_label(container)
    ax.set_xlabel('Product Category')
```



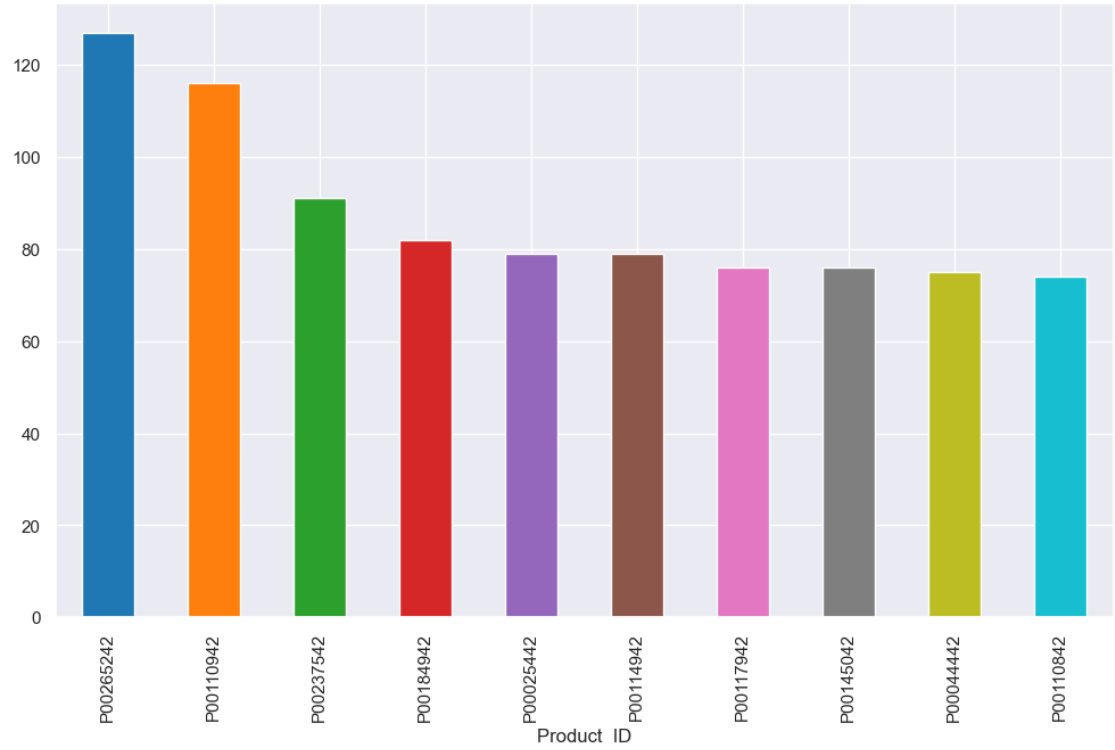
[39]:
```
# total amount vs product category

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

sns.barplot(data= sales_pro, x= 'Product_Category', y= 'Amount', hue=␣
 ↪'Product_Category', palette= 'husl').set_xlabel('Product  Category')
sns.set(rc= {'figure.figsize': (19,5)})
```



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

[30]:
```
# top 10 most sold products

plt.subplots(figsize= (12,7))
colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd',
          '#8c564b', '#e377c2', '#7f7f7f', '#bcbd22', '#17becf']
df.groupby('Product_ID')['Orders'].sum().nlargest(10).
 ↪sort_values(ascending=False).plot(kind='bar', color=colors).
 ↪set_xlabel('Product  ID')
```

[30]:
```
Text(0.5, 0, 'Product  ID')
```



3 Conclusion:

Married Women of age group 26-35 from Utter Pradesh, Maharastra and Karnataka working in IT, Aviation and Healthcare sectors are more likely to buy products from Food, Clothing and Electronics category.