Exploratory Data Analysis Using Python - Dipawali Sales Analysis Project

In [1]:
```
# import python libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

In [8]:
```
# import csv file
df = pd.read_csv(r'C:\Users\Denij\OneDrive\Desktop\Diwali Sales Data.csv', encoding = 'u
```

In [9]:
```
df.shape
```

Out[9]:
```
(11251, 15)
```

In [10]:
```
#checking top 5 rows of data
df.head()
```

Out[10]:

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

In [11]:
```
#checking bottom 5 rows of data
df.tail()
```

Out[11]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupatio
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemica
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcar
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textil
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agricultur
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcar

In [12]:
```
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
```

In [13]:
```
#drop unrelated/blank columns
df.drop(['Status','unnamed1'], axis = 1, inplace = True)
```

In [14]:
```
#check for null values
pd.isnull(df).sum()
```

Out[14]:
```
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
```

In [15]:
```
#drop null values
df.dropna(inplace = True)
```

In [16]:
```
# change data type
df['Amount'] = df['Amount'].astype('int')
```

In [17]:
```
#change data type
df['Amount'].dtypes
```

Out[17]:
```
dtype('int32')
```

In [18]:
```
df.columns
```

Out[18]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

In [19]:
```
#describe() method returns description of the data in the dataframe (i.e count, mean, st
df.describe()
```

Out[19]:

	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

Exploratory Data Analysis

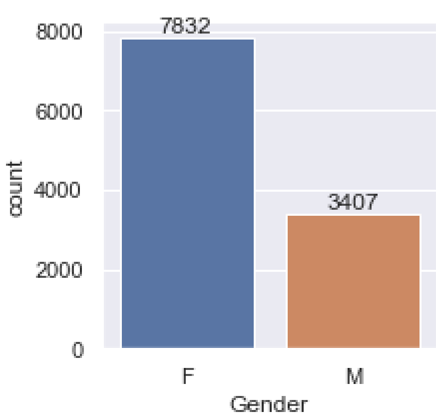
Gender

In [20]:
```
#use describe() for specific columns
df[['Age','Orders','Amount']].describe()
```

Out[20]:

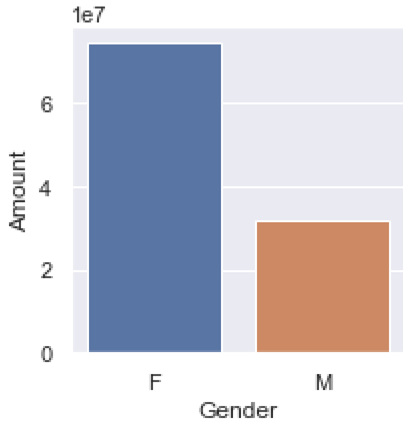
	Age	Orders	Amount
count	11239.000000	11239.000000	11239.000000
mean	35.410357	2.489634	9453.610553
std	12.753866	1.114967	5222.355168
min	12.000000	1.000000	188.000000
25%	27.000000	2.000000	5443.000000
50%	33.000000	2.000000	8109.000000
75%	43.000000	3.000000	12675.000000
max	92.000000	4.000000	23952.000000

In [22]:
```
#plotting a bar chart for Gender and it's count
sns.set(rc = {'figure.figsize':(3,3)})
ax = sns.countplot(x = 'Gender', data = df)

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



In [23]:
```
# plotting a bar chart for gender vs total amount
sales_gen = df.groupby(['Gender'],as_index = False)['Amount'].sum().sort_values(by = 'Am

sns.set(rc = {'figure.figsize':(3,3)})
sns.barplot(x = 'Gender', y = 'Amount', data = sales_gen)
```

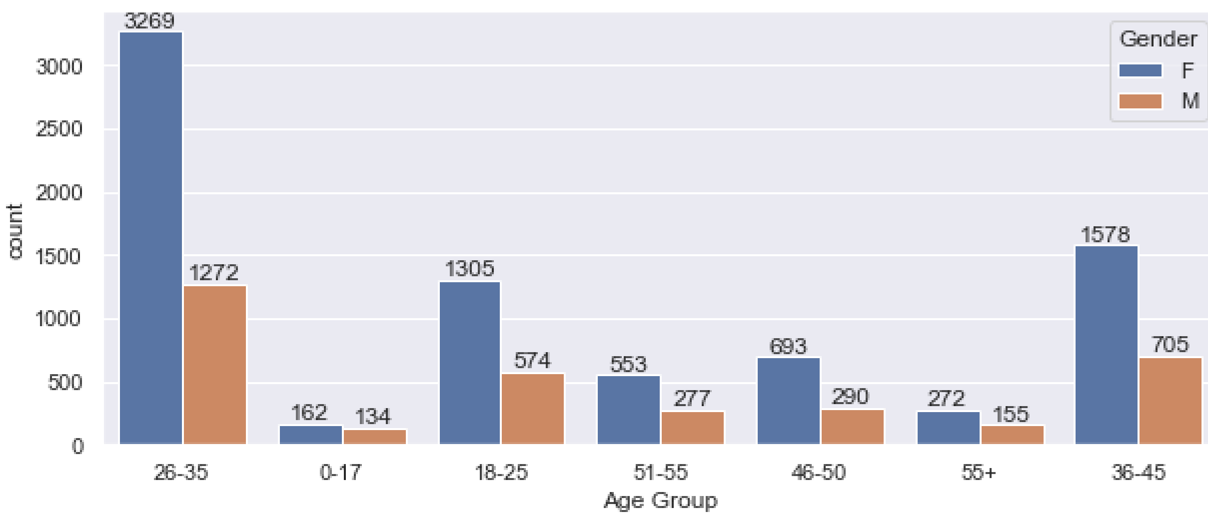
Out[23]: <AxesSubplot:xlabel='Gender', ylabel='Amount'>



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

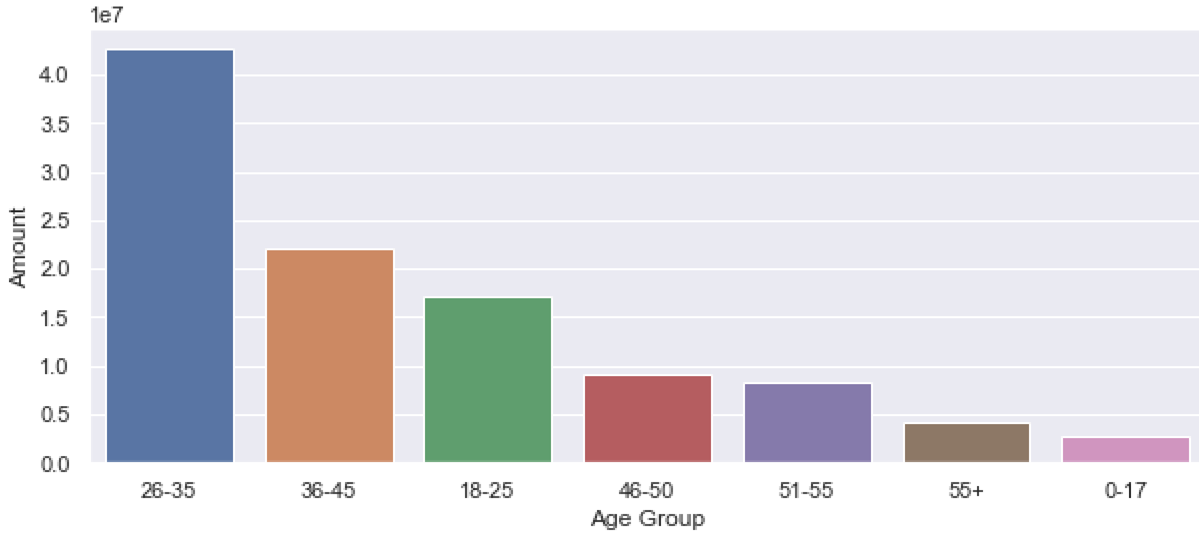
Age

In [24]:
```
sns.set(rc = {'figure.figsize':(10,4)})
ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

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



In [25]:
```
#Total Amount vs Age Group
sales_age = df.groupby(['Age Group'], as_index = False)['Amount'].sum().sort_values(by =

sns.set(rc={'figure.figsize':(10,4)})
sns.barplot(x = 'Age Group', y= 'Amount', data = sales_age)
```

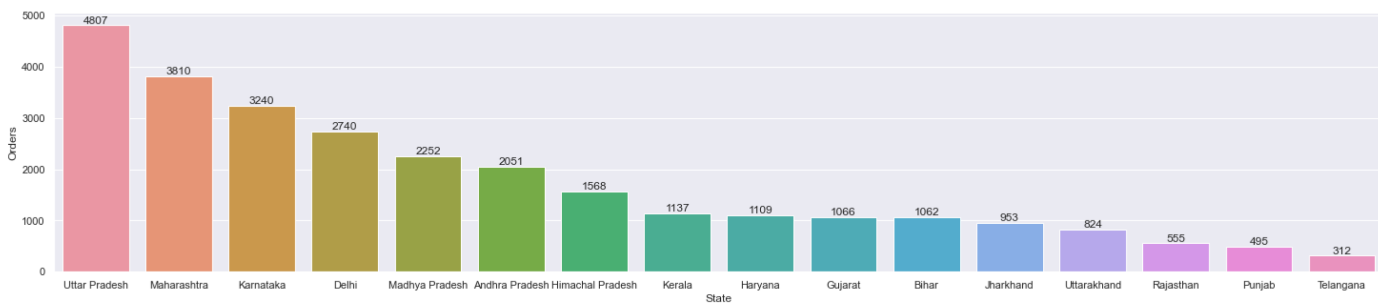
Out[25]:
```
<AxesSubplot:xlabel='Age Group', ylabel='Amount'>
```



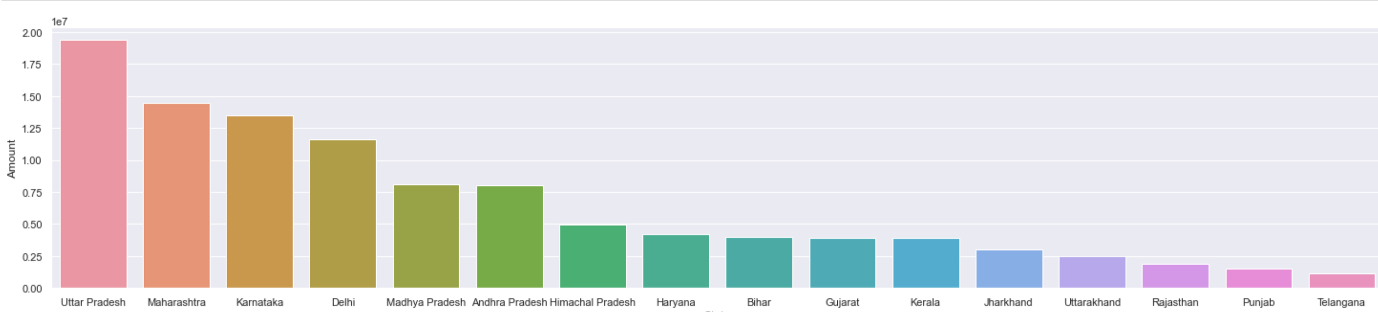
From above graphs we can see that most of the buyers are of age group between 26-35 years female.

State

In [30]:
```
#total number of orders from top 10 states
sales_state = df.groupby(['State'],as_index = False)['Orders'].sum().sort_values(by='Ord

sns.set(rc={'figure.figsize':(25,5)})
ax = sns.barplot(data = sales_state, x = 'State', y = 'Orders')

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



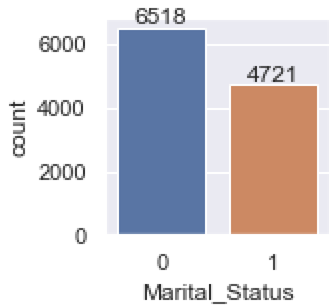
In [43]:
```
#total amount/sales from top 10 states
sales_state = df.groupby(['State'],as_index = False)['Amount'].sum().sort_values(by='Amo

sns.set(rc={'figure.figsize':(25,5)})
ax = sns.barplot(data = sales_state, x = 'State', y = 'Amount')
```



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

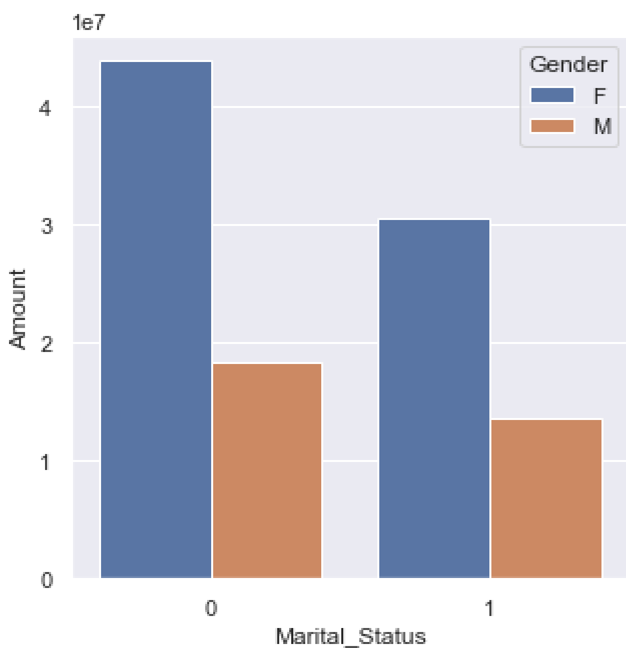
Marital Status

In [38]:
```
# Marital Status
ax = sns.countplot(data = df, x = 'Marital_Status')

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



In [44]:
```
sales_state = df.groupby(['Marital_Status','Gender'],as_index = False)['Amount'].sum().s

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

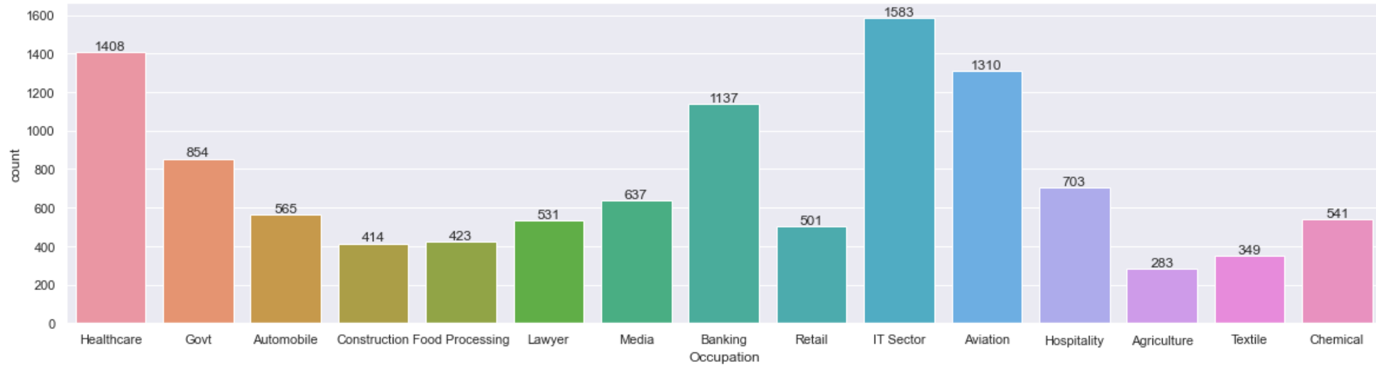
Out[44]:
```
<AxesSubplot:xlabel='Marital_Status', ylabel='Amount'>
```



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

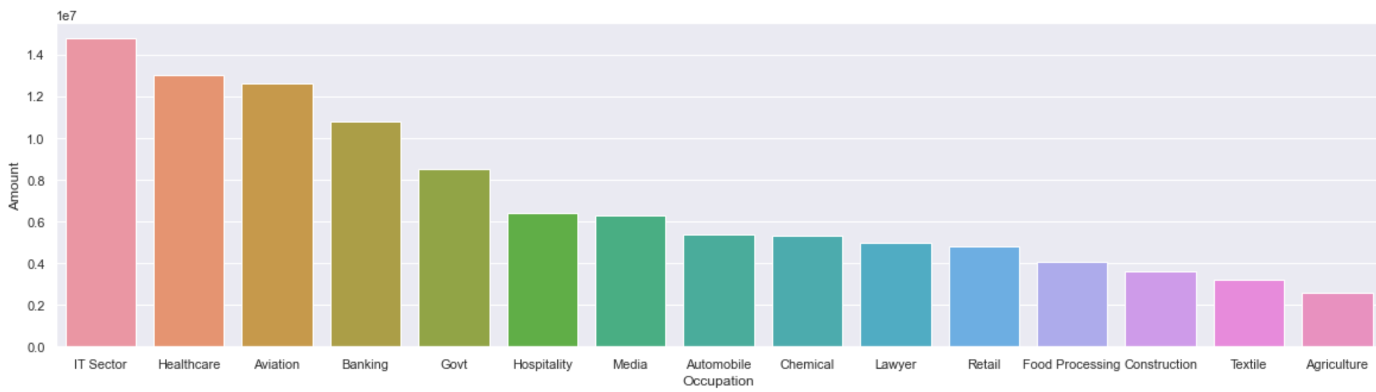
Occupation

In [48]:
```
#Occupation
ax = sns.countplot(data = df, x = 'Occupation')

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



In [53]:
```
sales_state = df.groupby(['Occupation'],as_index = False)['Amount'].sum().sort_values(by

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

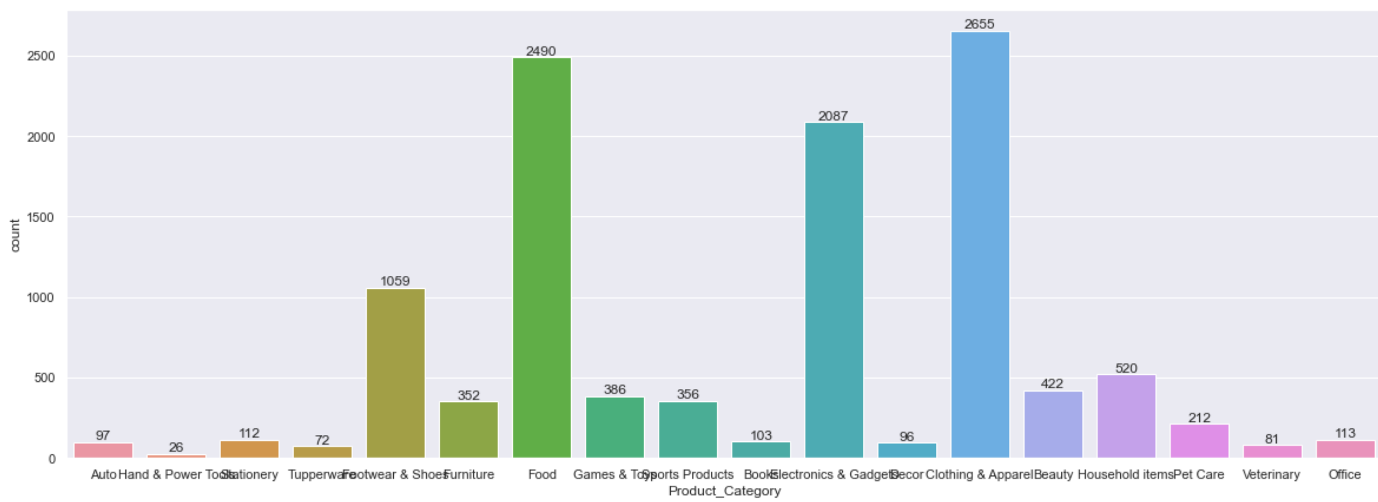
Out[53]:
```
<AxesSubplot:xlabel='Occupation', ylabel='Amount'>
```



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

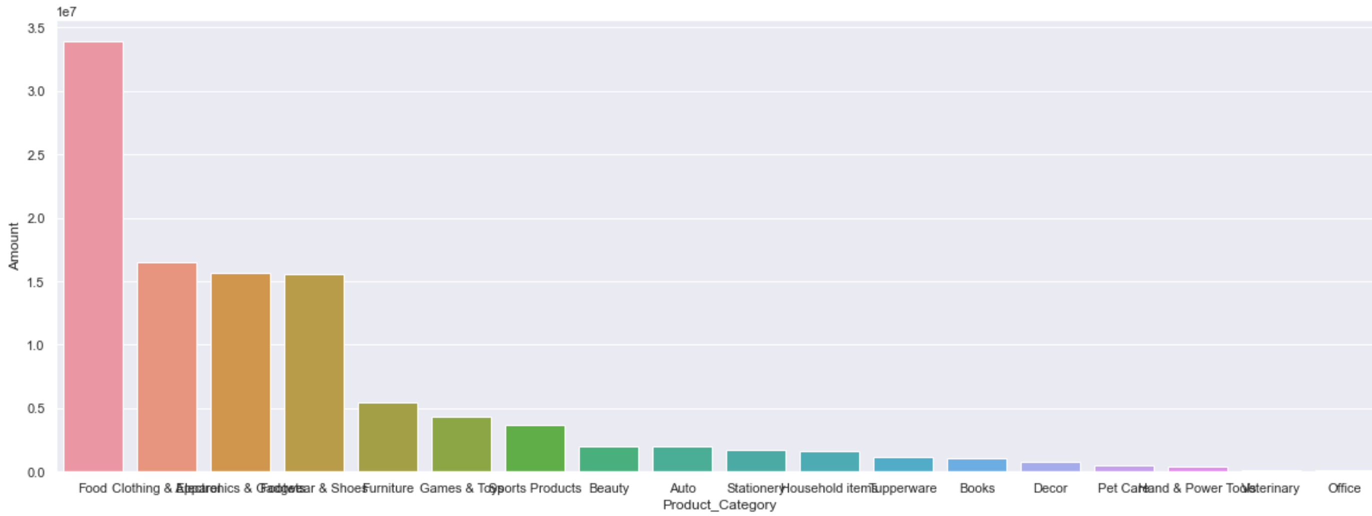
Product Category

In [52]:
```
sns.set(rc={'figure.figsize':(20,7)})
ax = sns.countplot(data = df, x = 'Product_Category')

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



In [54]:
```
sales_state = df.groupby(['Product_Category'],as_index = False)['Amount'].sum().sort_val

sns.set(rc={'figure.figsize':(20,7)})
sns.barplot(data = sales_state, x = 'Product_Category', y = 'Amount')
```

Out[54]: <AxesSubplot:xlabel='Product_Category', ylabel='Amount'>

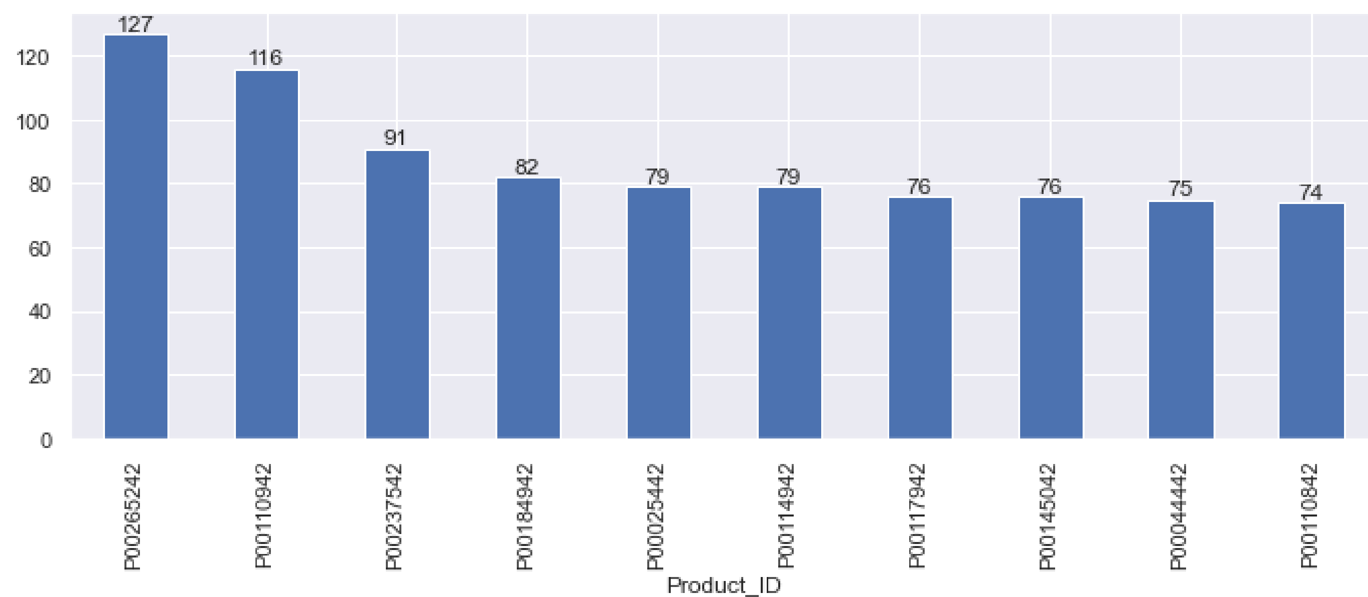


From above graphs we can see that most of the sold products are Food, Clothing and Electronics category.

Top Products

In [83]:

```
# Most sold products
fig1, ax1 = plt.subplots(figsize=(12,4))
ax = df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=False).

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



Conclusion: 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.