Diwali Sales Analysis

Dataset Overview

The **Diwali Sales Data** consists of **11,251** entries and **15 columns**. It contains information about customers, their demographics, and purchase behavior during the Diwali season.

Key Columns and Their Details

- **User_ID**: Unique identifier for each customer.
- **Cust_name**: Customer names.
- **Product_ID**: Unique identifier for each product.
- **Gender**: Gender of customers (Male/Female).
- **Age Group & Age**: Age category and exact age of customers.
- **Marital_Status**: Indicates if the customer is married (1) or not (0).
- **State & Zone**: Geographical information of customers.
- **Occupation**: Job or profession of customers.
- **Product_Category**: Category to which the product belongs.
- **Orders**: Number of orders placed.
- **Amount**: Purchase amount (contains some missing values).
- **Status & unnamed1**: Columns with **0 non-null values** — unrelated/blank.

1. Loading the Dataset and Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

2. Load the dataset

```
df = pd.read_csv('Diwali Sales Data.csv', encoding='latin1')
print('Shape of the data:\n',df.shape)
print('\nFirst 5 rows:\n',df.head())
print('\nDataset information:\n',df.info())
```

Shape of the data:
(11251, 15)

First 5 rows:

	User_ID	Cust_name	Product_ID	Gender	Age Group	...	Product_Category	Orders	Amount	Status	unnamed1
0	1002903	Sanskriti	P00125942	F	26-35	...	Auto	1	23952.0	NaN	NaN
1	1000732	Kartik	P00110942	F	26-35	...	Auto	3	23934.0	NaN	NaN
2	1001990	Bindu	P00118542	F	26-35	...	Auto	3	23924.0	NaN	NaN
3	1001425	Sudevi	P00237842	M	0-17	...	Auto	2	23912.0	NaN	NaN
4	1000588	Joni	P00057942	M	26-35	...	Auto	2	23877.0	NaN	NaN

[5 rows x 15 columns]
<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
Dataset information:
None

3. Data Cleaning

```
# Drop unrelated/blank columns
df.drop(['Status', 'unnamed1'], axis=1, inplace=True)
print('\nChecking null values:\n',df.isnull().sum())
```

Checking null values:

```
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
```

```
# Drop rows with null values
print('\nShape of the data before dropping null values:',df.shape)
df.dropna(inplace=True)
print('\nShape of the data after dropping null values:',df.shape)
```

```
Shape of the data before dropping null values: (11251, 13)

Shape of the data after dropping null values: (11239, 13)
```

```
# Change data type of columns
df['Amount'] = df['Amount'].astype('int')
print('\nData type of Amount column:',df['Amount'].dtype)
```

```
Data type of Amount column: int64
```

```
# Rename columns
print('\nColumns before renaming:\n',df.columns)
df.rename(columns={'Marital_Status' : 'Marrige_Status'})
```

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

```
# describe() method returns description of the data in the DataFrame (like cont,
mean, std, etc)
print('\nDesctiption of the data:\n', df.describe())
```

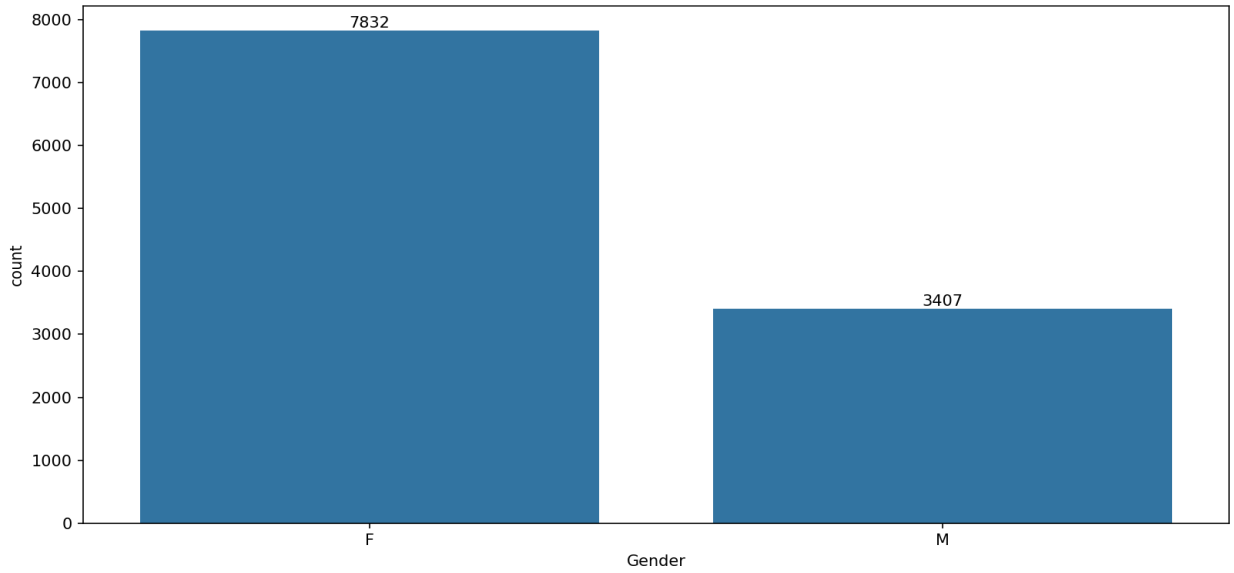
```
Desctiption of the data:
             User_ID          Age  Marital_Status        Orders        Amount
count  1.123900e+04  11239.000000    11239.000000  11239.000000  11239.000000
mean   1.003004e+06     35.410357        0.420055      2.489634   9453.610553
std    1.716039e+03     12.753866        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
```

```
# Use describe() for specific columns
print(df[['Age', 'Orders', 'Amount']].describe())
```

	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

4. Exploratory Data Analysis

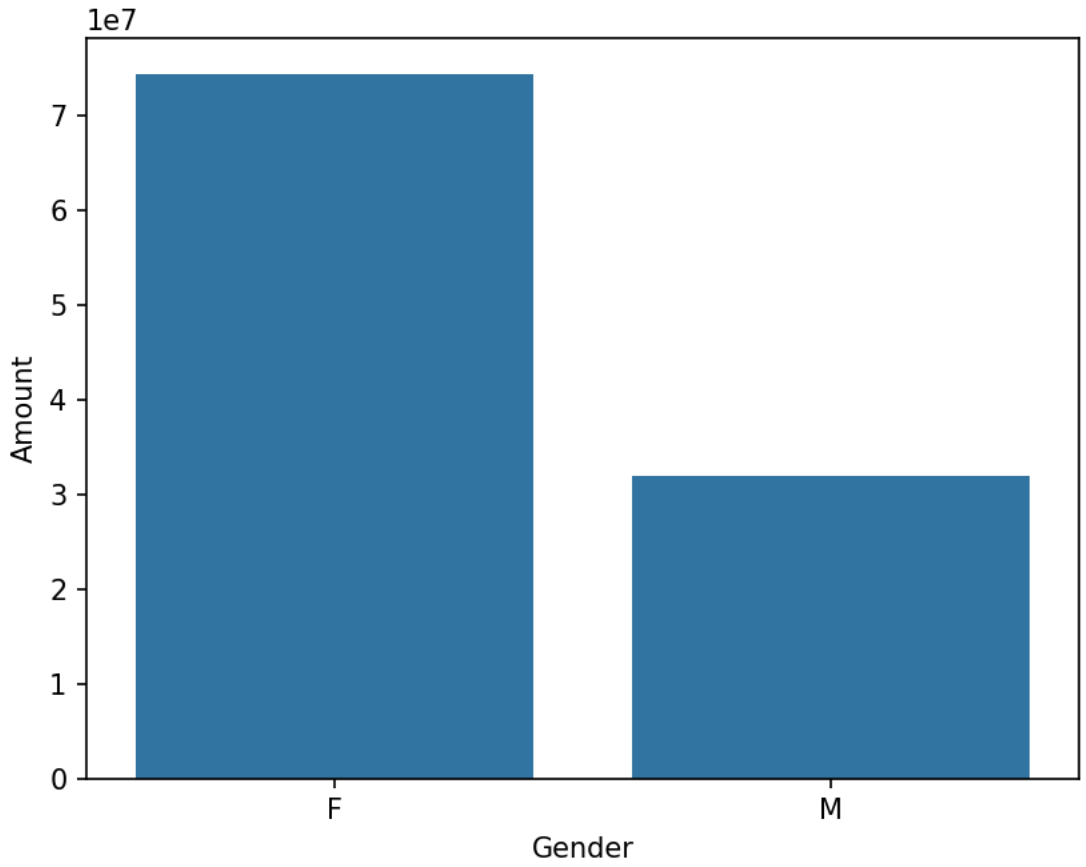
```
# Gender
ax = sns.countplot(x = 'Gender', data = df)
for bars in ax.containers:
    ax.bar_label(bars)
plt.show()
```




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

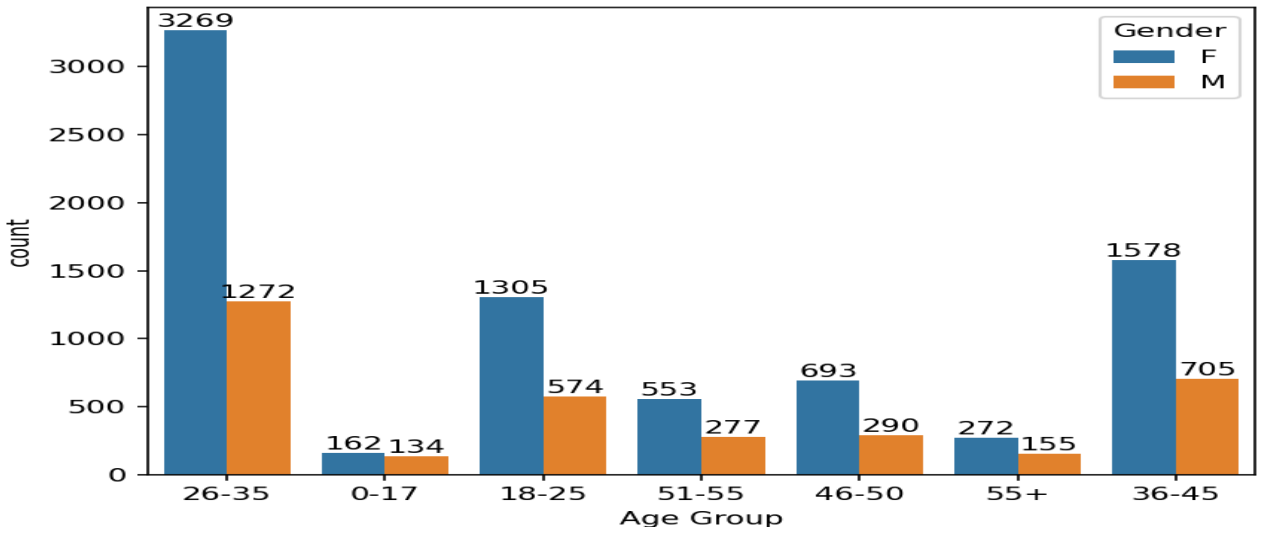
	Gender	Amount
0	F	74335853
1	M	31913276

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

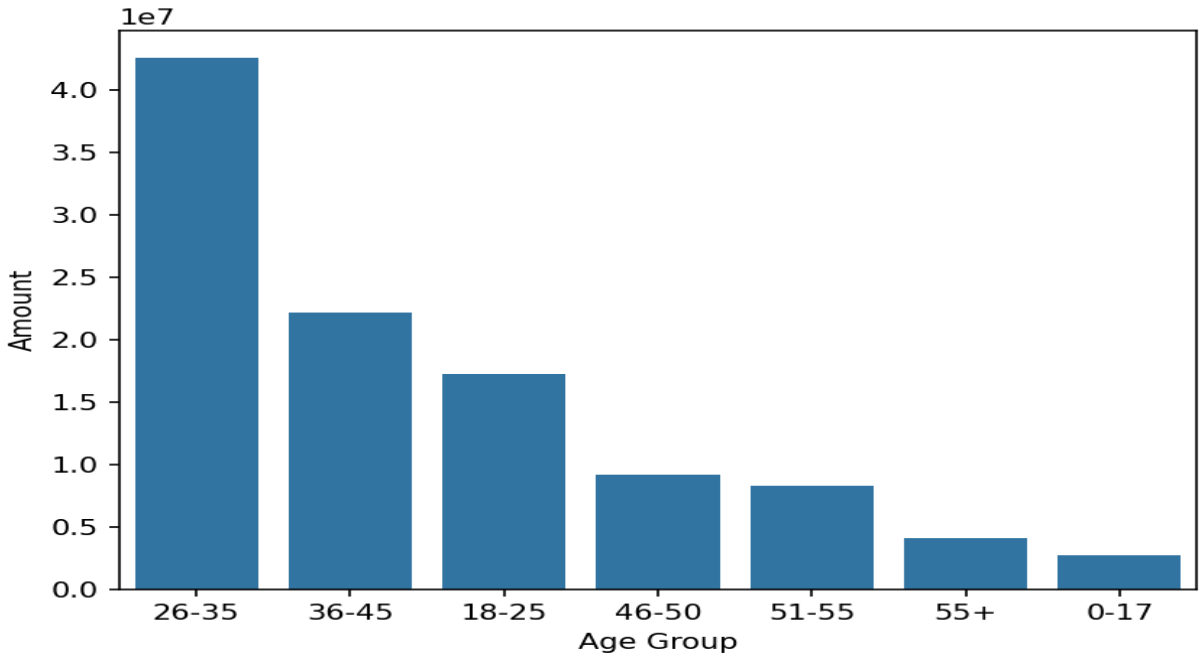


*From the above graphs we can see that most of the buyers are females and even the purchasing power of females are greater than men.

```
# Age
age = sns.countplot(data=df, x='Age Group', hue='Gender')
for bars in age.containers:
    age.bar_label(bars)
plt.show()
```

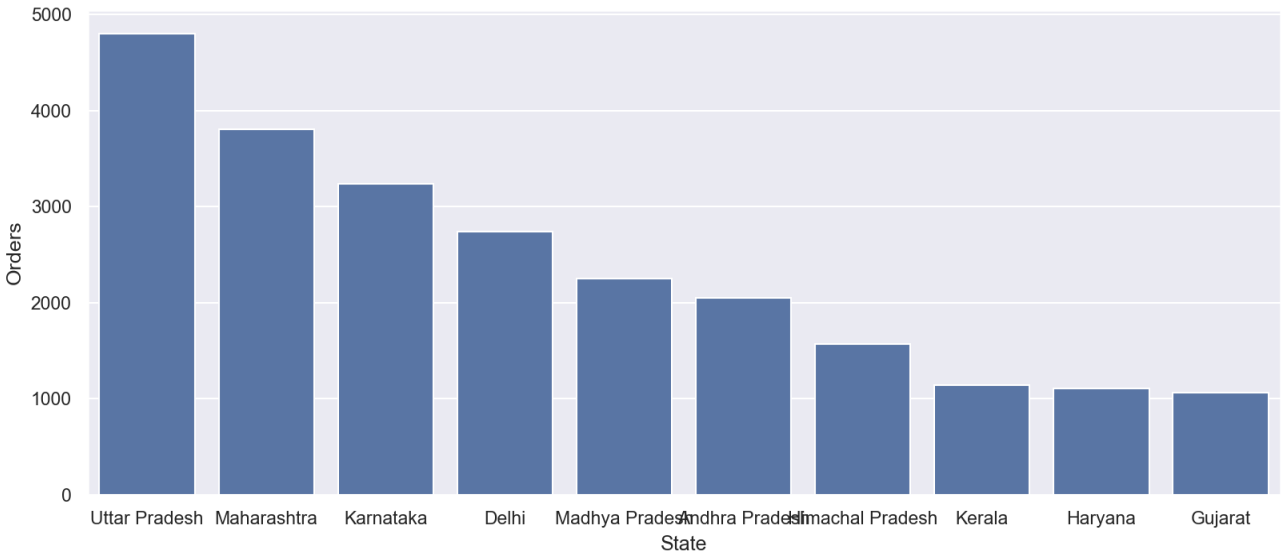


```
# Total amount vs Age Group
sales_age = df.groupby(['Age Group'],
as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

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

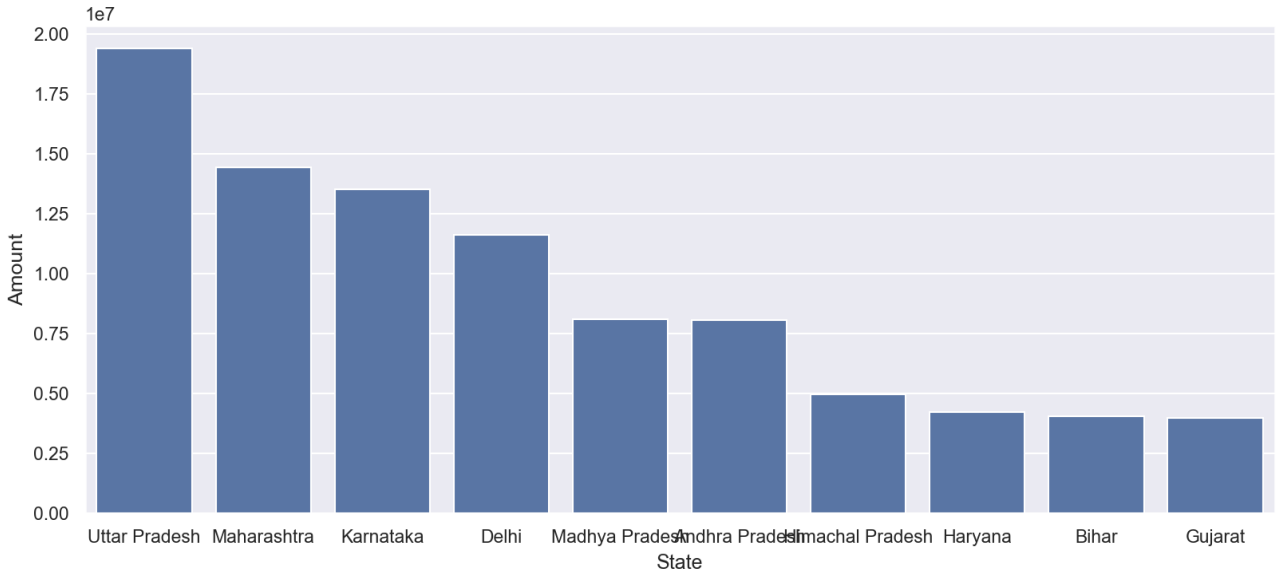


*From above graphs we can see most of the buyers are of age group between 26-35yrs female

```
# State
# 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':(15, 5)})
sns.barplot(data=sales_state, x='State', y='Orders')
plt.show()
```

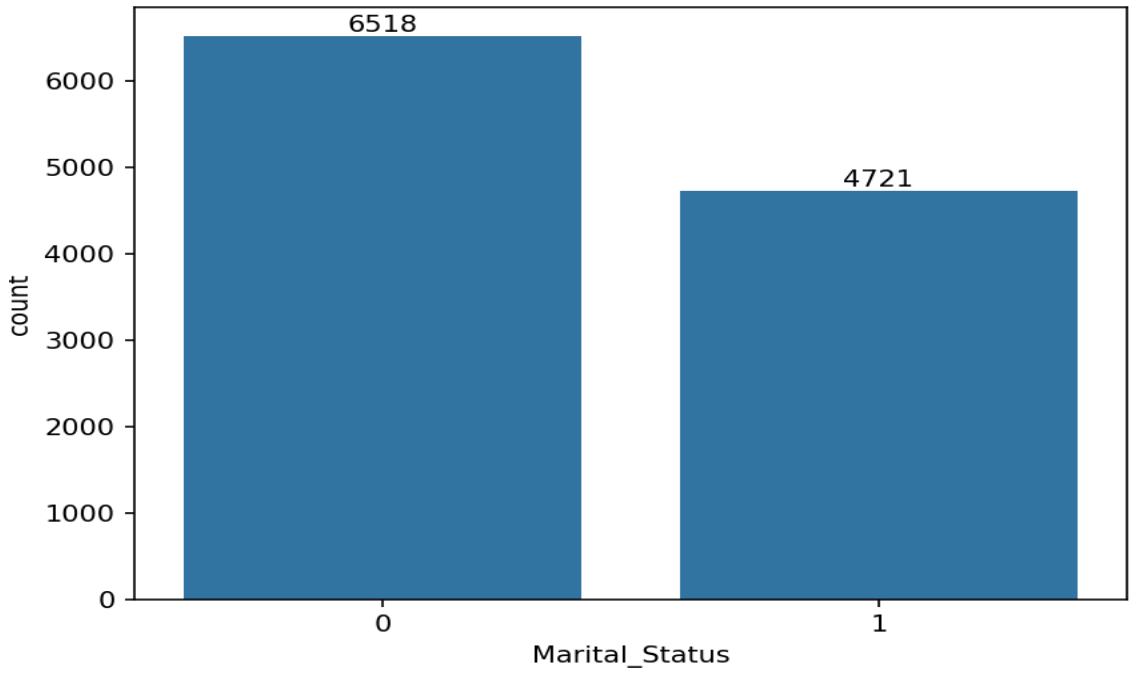


```
# Total amount/sales from top 10 states
sales_states = df.groupby(['State'],
as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(15, 5)})
sns.barplot(data=sales_states, x='State', y='Amount')
plt.show()
```

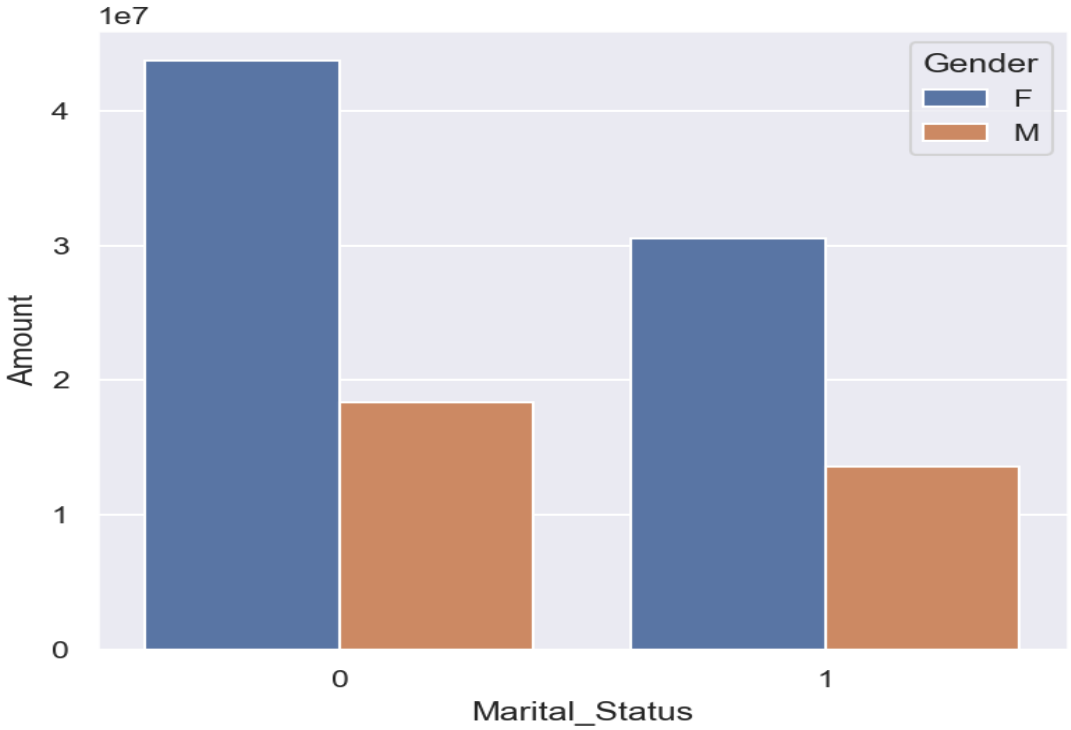


*From above graphs we can see that unexpectedly most of the orders are form Uttar Pradesh, Maharashtra and Karnataka Respectively but total sales/amount is from UP, Karnataka and then Maharashtra.

```
# Marital Status
ax = sns.countplot(data=df, x='Marital_Status')

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

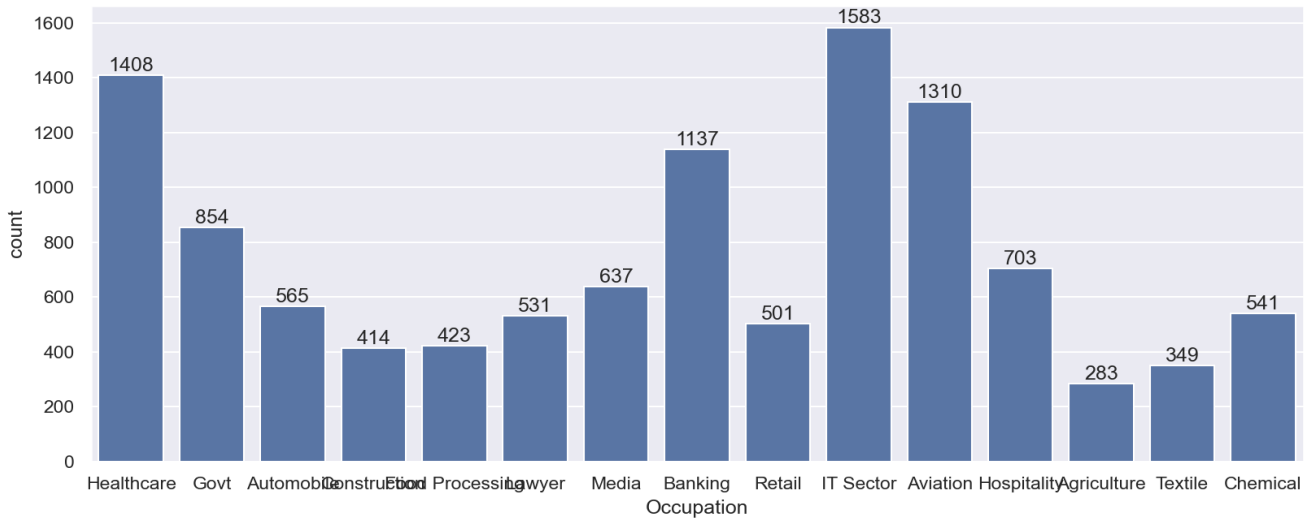


```
# Marital Status vs Amount
sales_marital = df.groupby(['Marital_Status', 'Gender'],
as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

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

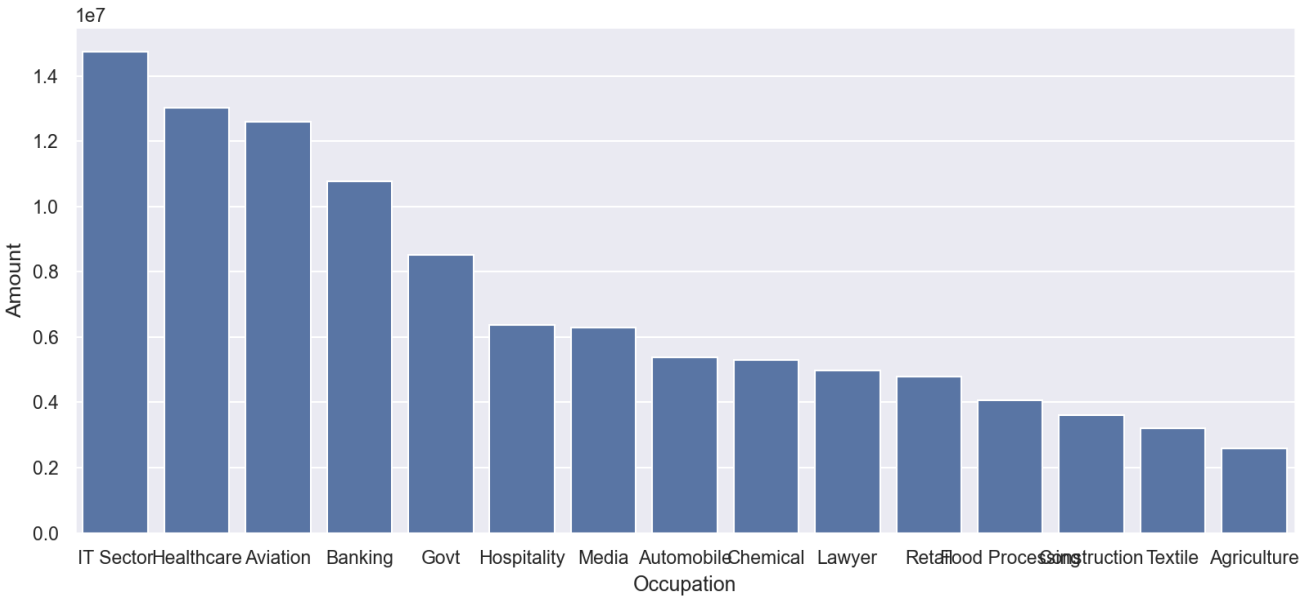


*From above graphs we can see that most of the buyers are married (women) and they have high purchasing power.

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

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

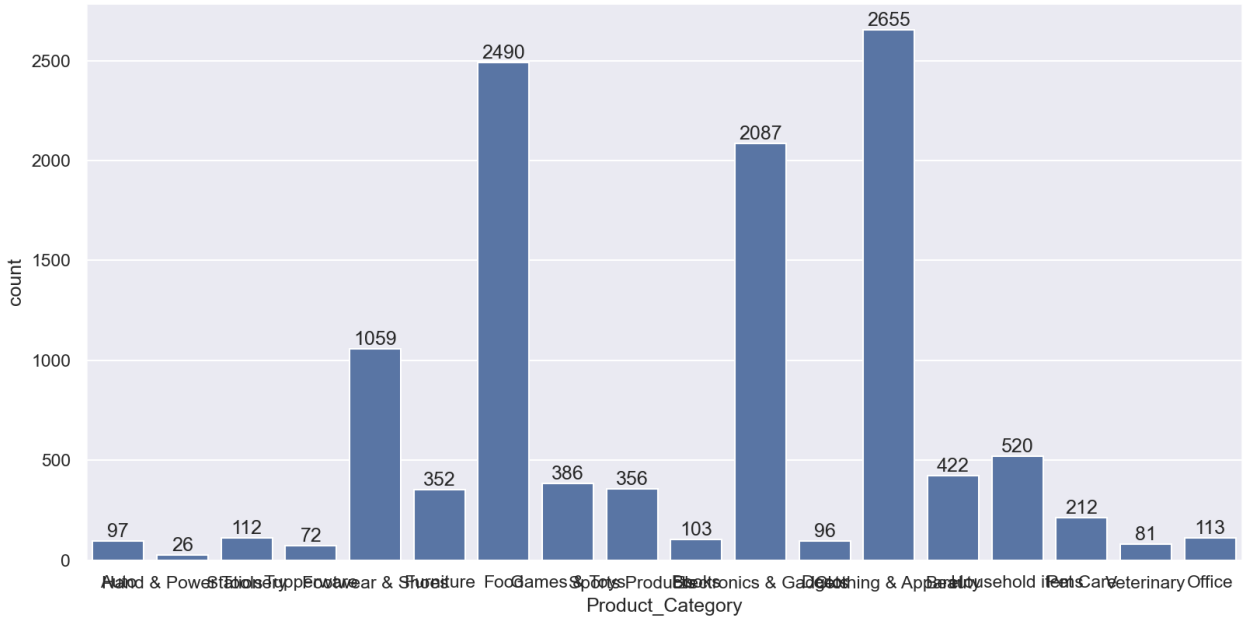


```
# Occupation vs Amount
sales_occ = df.groupby(['Occupation'],
as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

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

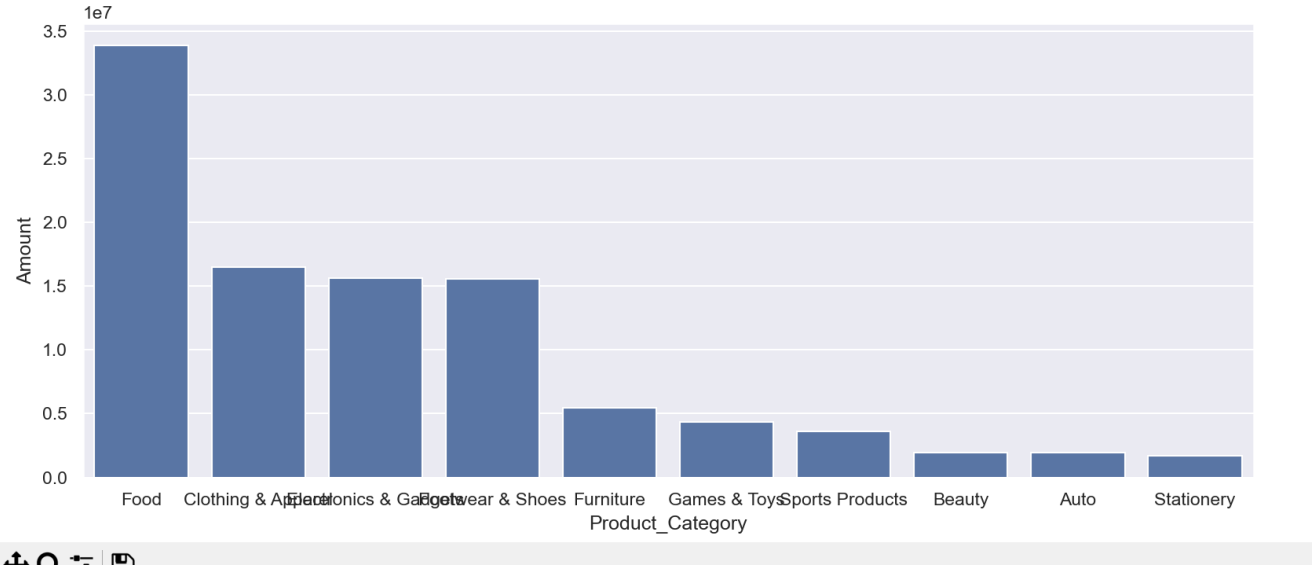


***From above graphs we can see that most of the buyers are working in IT, Aviation and Healthcare sector**

```
# Product Category
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data=df, x='Product_Category')

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

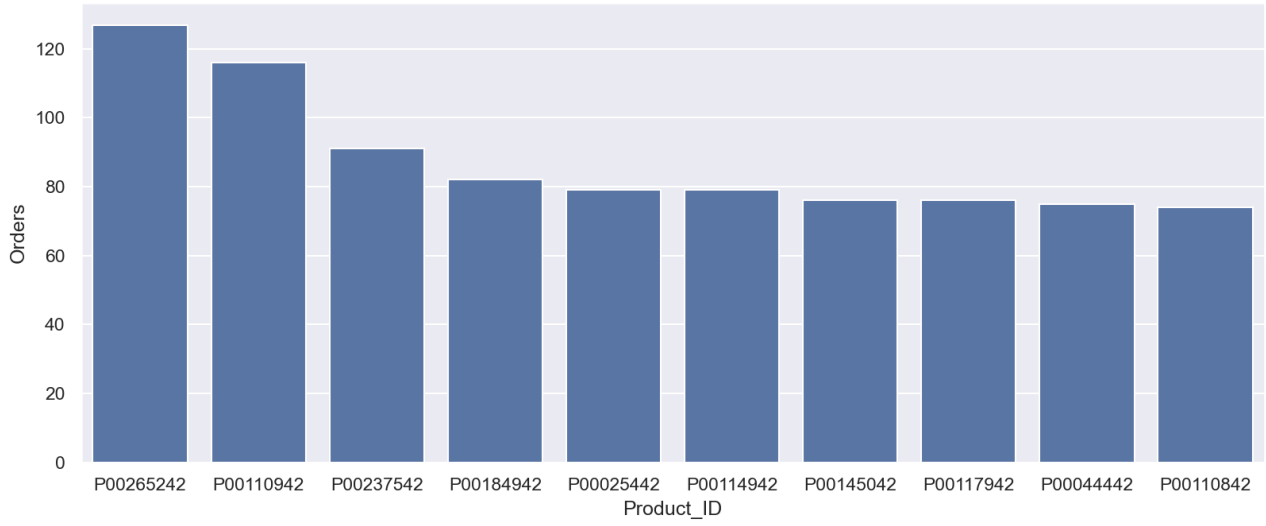


```
# Product Categeory vs Amount
sales_prod = df.groupby(['Product_Category'],
as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)
sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data=sales_prod, x='Product_Category', y='Amount')
plt.show()
```

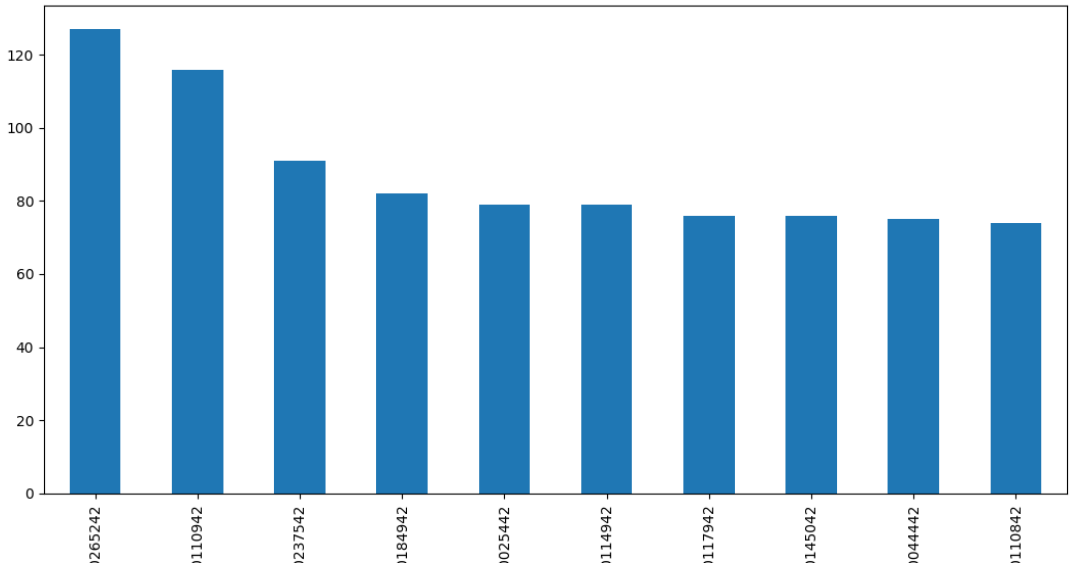


*From above graphs we can see the most of sold products are from Food, Footwear and Electronics category.

```
# Top Selling Products on Product_ID
sales_prod_id = 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_prod_id, x='Product_ID', y='Orders')
plt.show()
```



```
# Top 10 most sold products
fig1, ax1 = plt.subplots(figsize=(12,7))
df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=False).pl
ot(kind='bar')
plt.show()
```



Conclusion

Married women age group 25-35 yrs from UP, Maharashtra and Karnataka working in IT, Healthcare and Aviation are more likely to buy products from Food, Clothing and Electronics category