Diwali_Sales_Analysis

Objective:

Improve customer experience by analyzing Sales data and to Increase revenue

```
import numpy as np # for working with arrays
import pandas as pd # data structure manupulation
import matplotlib.pyplot as plt # visualize using chart
import seaborn as sns #advanced version of matplotlib
```

Import and data uderstandsing

```
df1 = pd.read_csv("Diwali Sales Data.csv", encoding =
'unicode_escape')
# to prevent the unicode error we use the encoding = unicode_escape

df1.shape
```

```
(11251, 15)
```

```
df1.head(10)
```

```
   User_ID  Cust_name Product_ID Gender Age Group  Age  Marital_Status
\
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

5  1000588       Joni  P00057942      M     26-35   28               1

6  1001132       Balk  P00018042      F     18-25   25               1

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


               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
5  Himachal Pradesh  Northern  Food Processing             Auto
1
6     Uttar Pradesh   Central           Lawyer             Auto
4
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

```
df1.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
```

Data Cleaning

```
#drop the unrelated columns we can use drop
df1.drop(['Status','unnamed1'], axis=1, inplace=True)
#here the axis = 1 represent that apply the code to total row and
inplace = true means whatever
#done in the line of code save the execution.

df1.isnull()
```

```
       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]
```

```
df1.isnull().sum()
```

```
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
```

```
df1.shape
```

```
(11251, 13)
```

```
df1.dropna(inplace=True)
```

```
df1.shape
```

```
(11239, 13)
```

```
df1.isnull().sum()
```

```
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
```

```
df1['Amount'] = df1['Amount'].astype('int64')
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11239 entries, 0 to 11250
Data columns (total 13 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   User_ID           11239 non-null  int64
 1   Cust_name         11239 non-null  object
 2   Product_ID        11239 non-null  object
 3   Gender            11239 non-null  object
 4   Age Group         11239 non-null  object
 5   Age               11239 non-null  int64
 6   Marital_Status    11239 non-null  int64
 7   State             11239 non-null  object
 8   Zone              11239 non-null  object
 9   Occupation        11239 non-null  object
 10  Product_Category  11239 non-null  object
 11  Orders            11239 non-null  int64
 12  Amount            11239 non-null  int64
dtypes: int64(5), object(8)
memory usage: 1.2+ MB
```

```
df1.columns
```

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

```
df1.describe()
```

```
            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
```

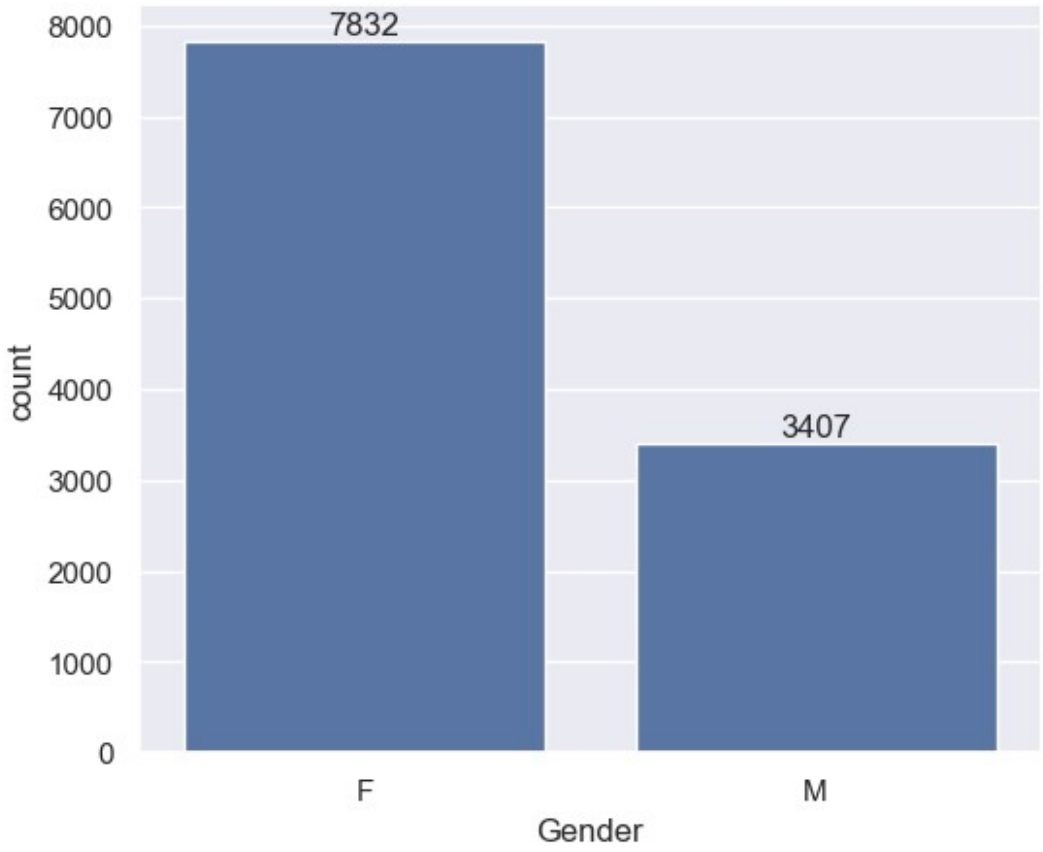
Exploratory Data analysis

On basis of Gender

```
df1.columns
```

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

```
Dx = sns.countplot(x = 'Gender', data = df1)

sns.set(rc={"figure.figsize":(10,5)})# 10 = width, 5 = height
for bars in Dx.containers:
    Dx.bar_label(bars)
#used to count the male to female ratio in sales
```

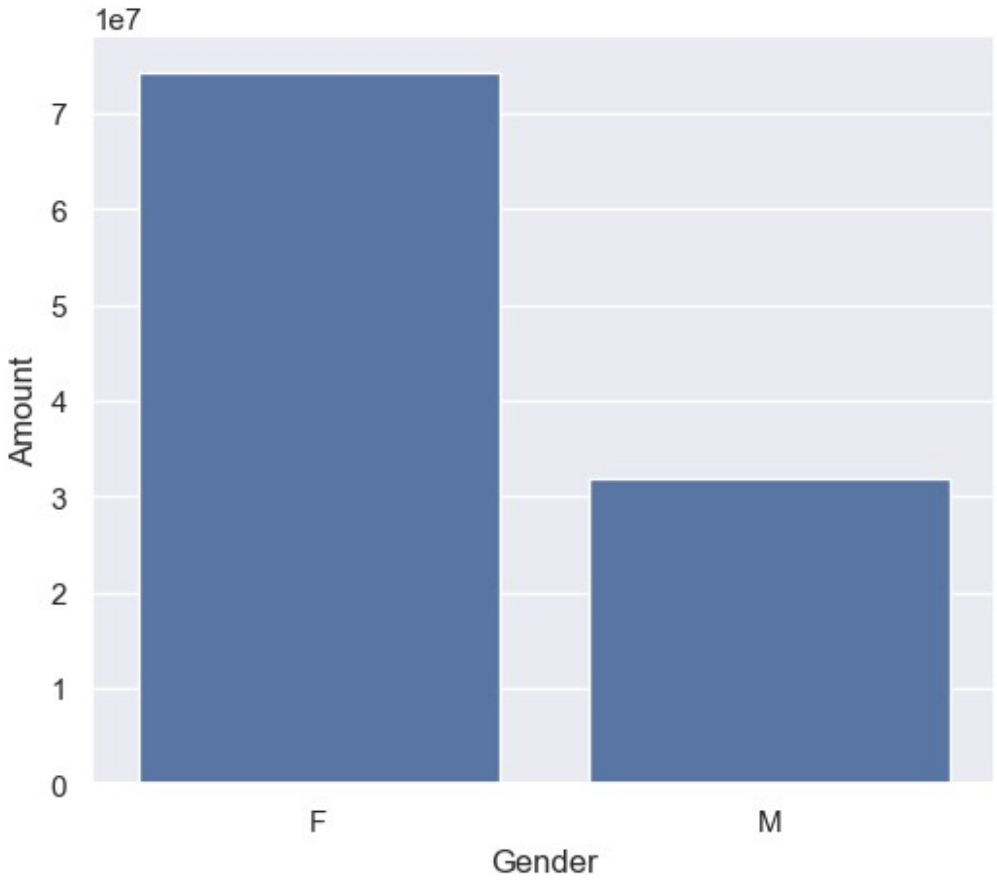



```
df1.groupby(['Gender'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)
```

	Gender	Amount
0	F	74335853
1	M	31913276

```
sales_gender = df1.groupby(['Gender'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)
sns.set(rc={"figure.figsize":(6,5)})# 6 = width, 5 = height
sns.barplot(x = 'Gender', y = 'Amount', data = sales_gender)
```

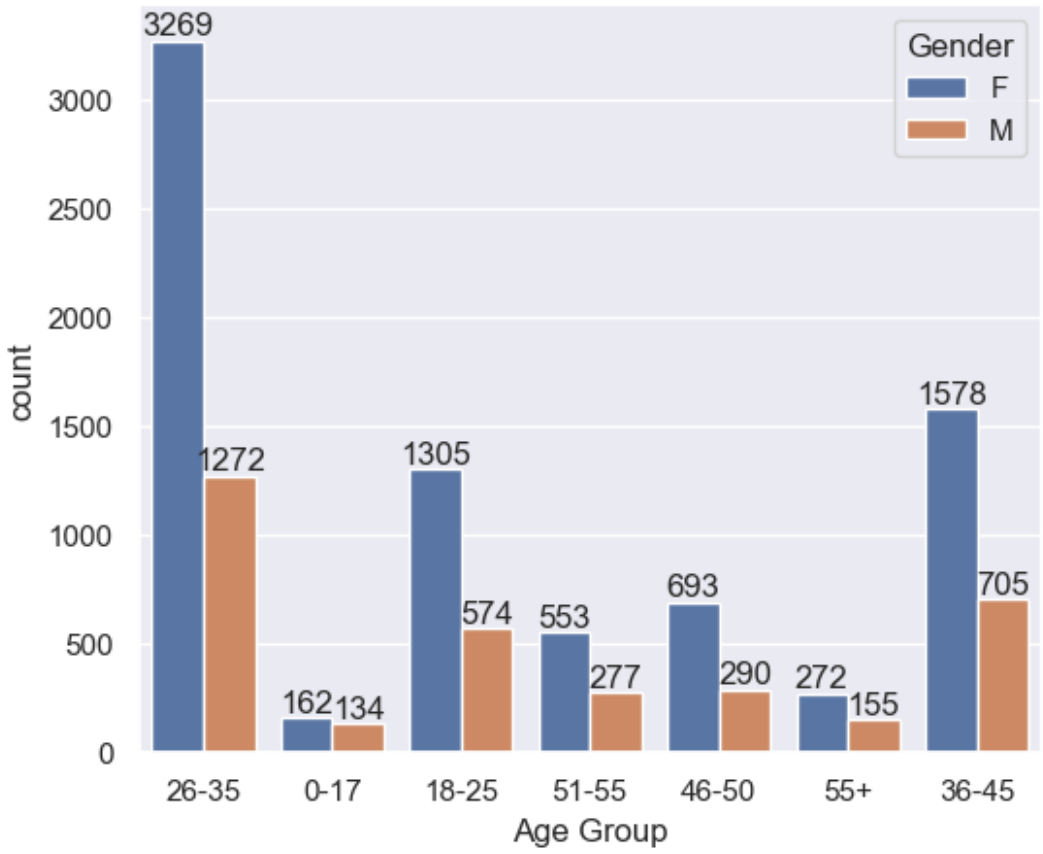
```
<Axes: xlabel='Gender', ylabel='Amount'>
```



From the above graphs we can say more buyers are from females and the female purchasing power is more than the males

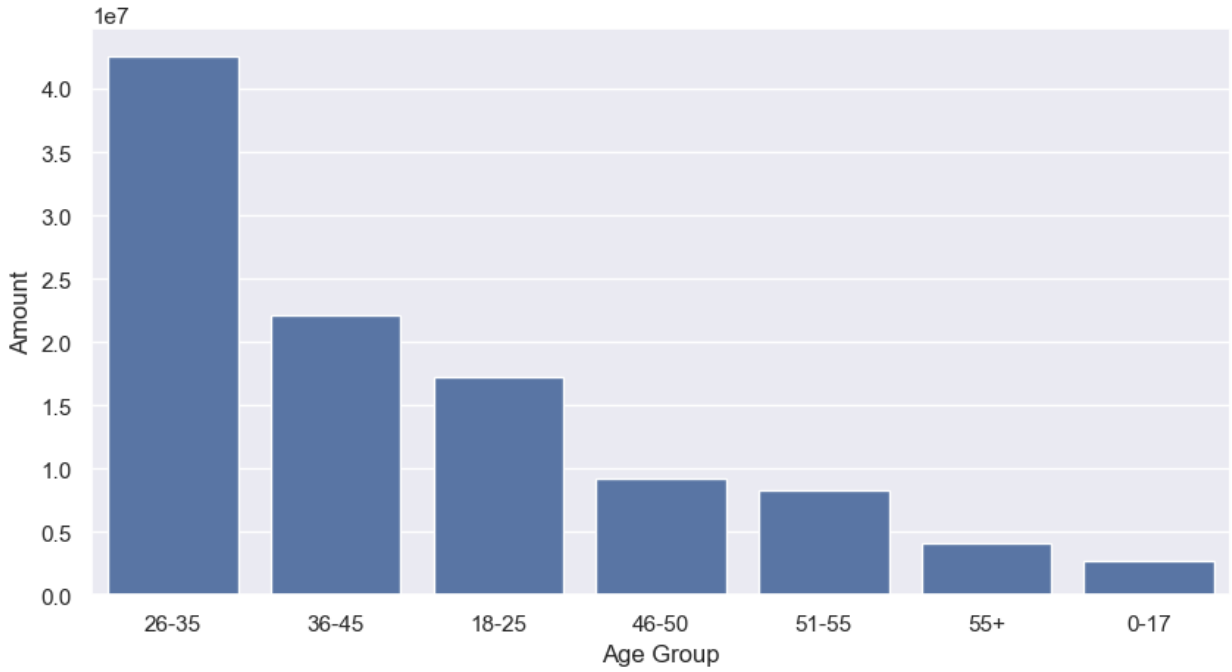
Age

```
Gx = sns.countplot(x = 'Age Group', hue = 'Gender', data = df1)
sns.set(rc={"figure.figsize":(10,5)})# 10 = width, 5 = height
for bars in Gx.containers:
    Gx.bar_label(bars)
#here hue is used to divide the result by the column provided
```




```
sales_age = df1.groupby(['Age Group'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)
sns.set(rc={"figure.figsize":(10,5)})# 10 = width, 5 = height
sns.barplot(x = 'Age Group', y = 'Amount', data = sales_age)
```

```
<Axes: xlabel='Age Group', ylabel='Amount'>
```



From the above graphs we can say that the most purchases are from the age group 26-35 and in the especially from the females

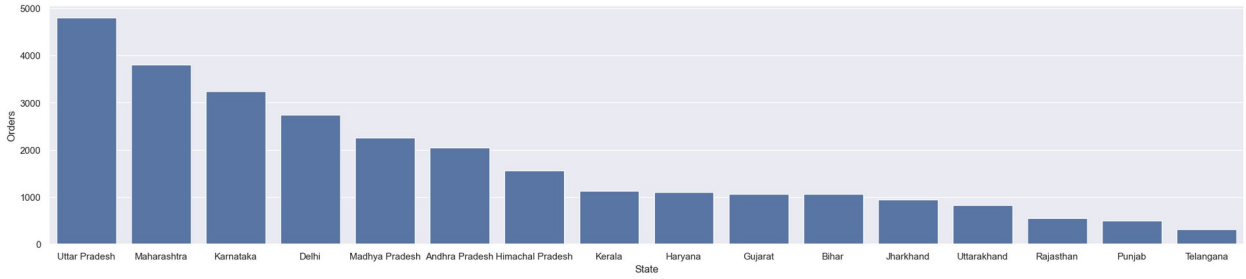
State

```
df1.columns
```

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

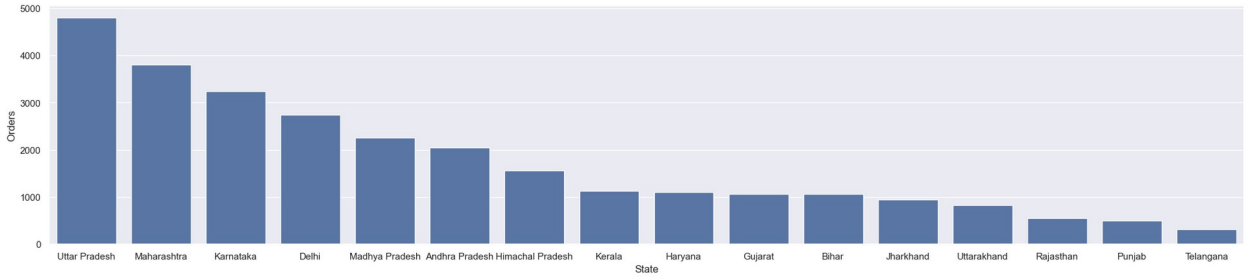
```
# total number of orders from different states
sales_stat = df1.groupby(['State'], as_index=False)
['Orders'].sum().sort_values(by='Orders', ascending=False)
sns.set(rc={"figure.figsize":(25,5)})# 25 = width, 5 = height
sns.barplot(x = 'State', y = 'Orders', data = sales_stat)

#as the plot is not clear we can adjust the size by using the below
code
```

```
<Axes: xlabel='State', ylabel='Orders'>
```



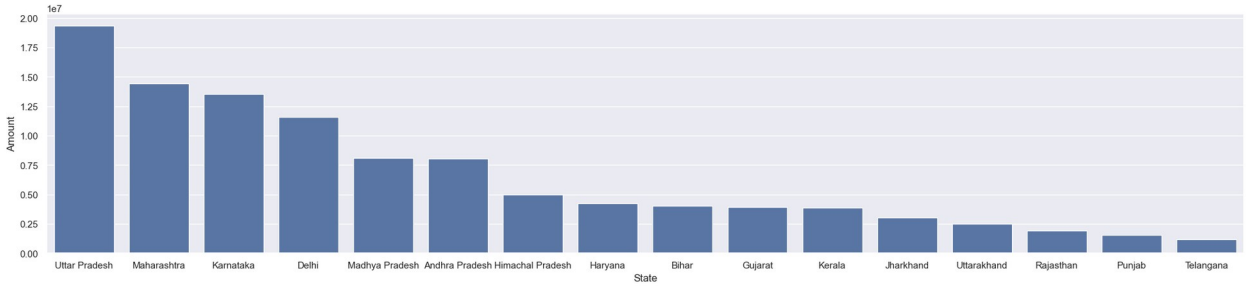

```
sales_stat = df1.groupby(['State'], as_index=False)
['Orders'].sum().sort_values(by='Orders', ascending=False)

sns.set(rc={"figure.figsize":(25,5)})# 25 = width, 5 = height
sns.barplot(x = 'State', y = 'Orders', data = sales_stat)
```

```
<Axes: xlabel='State', ylabel='Orders'>
```




```
sales_stat_amount = df1.groupby(['State'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)

sns.set(rc={"figure.figsize":(25,5)})# 25 = width, 5 = height
sns.barplot(x = 'State', y = 'Amount', data = sales_stat_amount)
```

```
<Axes: xlabel='State', ylabel='Amount'>
```

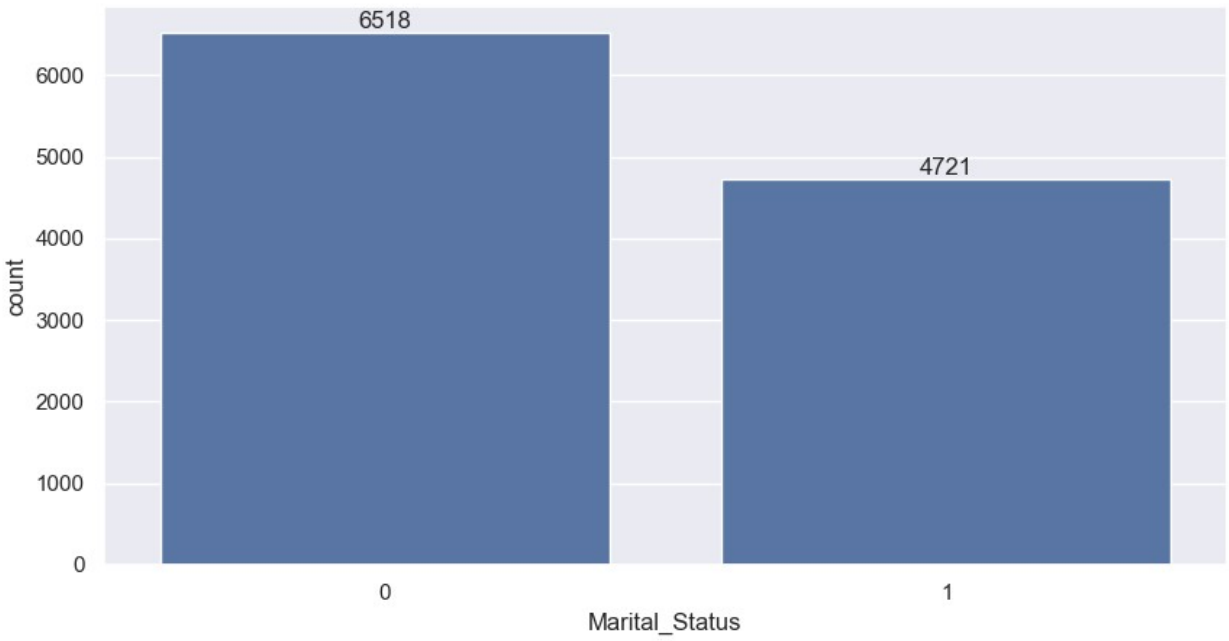


From the above graphs we can say that the most of the orders and the sales are from the states uttar pradesh, maharastra, karnataka.

Marital status

```
Mx = sns.countplot(x = 'Marital_Status', data = df1)
```

```
sns.set(rc={"figure.figsize":(7,5)})# 7 = width, 5 = height
for bars in Mx.containers:
    Mx.bar_label(bars)
```

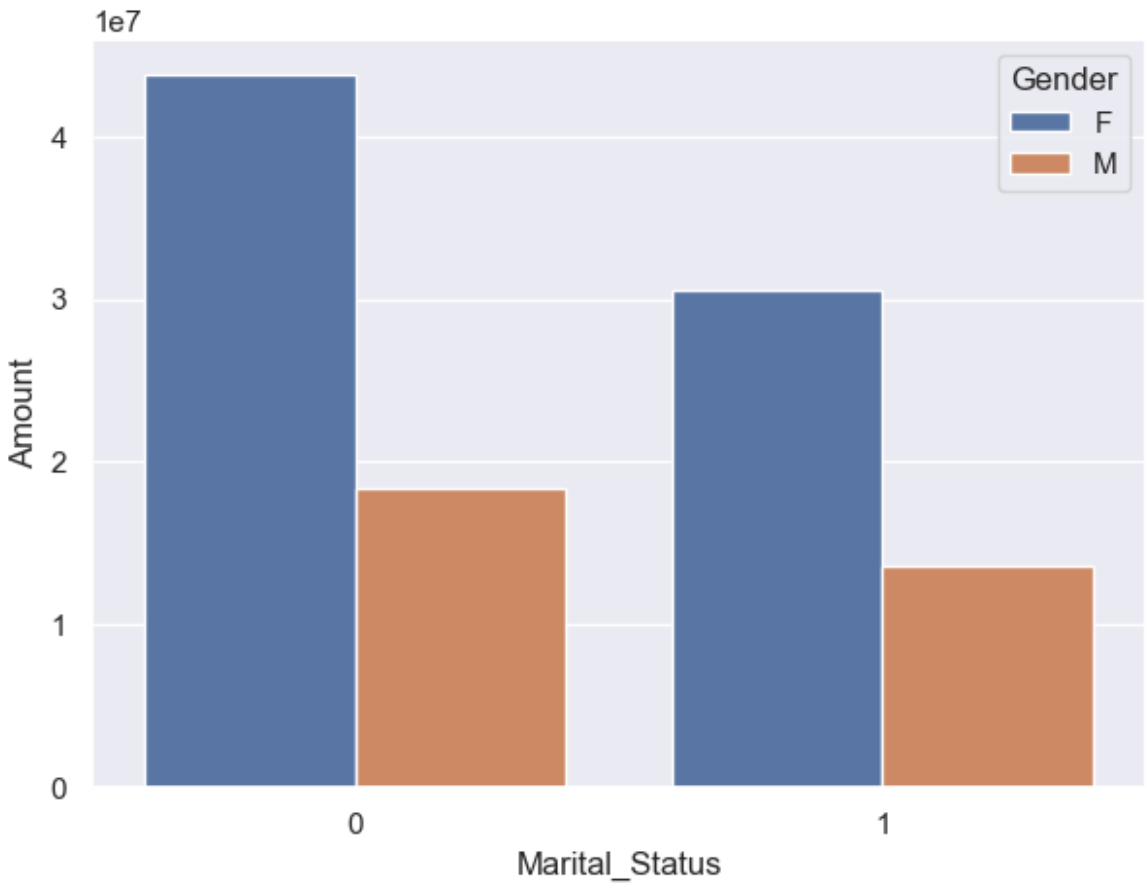



```
df1.columns
```

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

```
sales_ar = df1.groupby(['Marital_Status', 'Gender'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)
sns.set(rc={"figure.figsize":(7,5)})# 7 = width, 5 = height
sns.barplot(x ='Marital_Status', y ='Amount', hue ='Gender', data =
sales_ar)
```

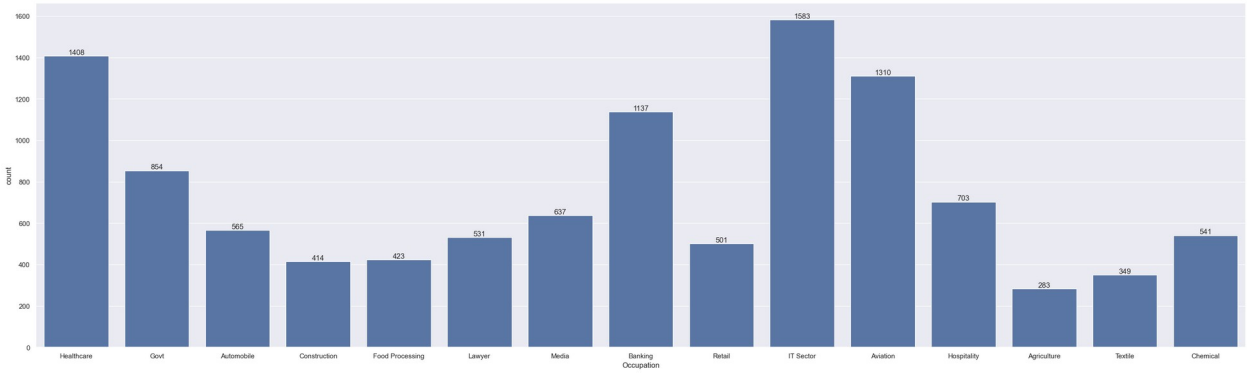
```
<Axes: xlabel='Marital_Status', ylabel='Amount'>
```



By the above graphs we can say that the married women are the most in the buyers and are with highest purchaing power

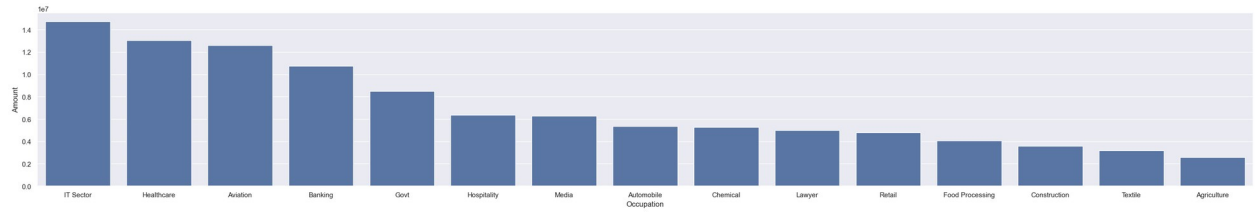
Occupation

```
Ox = sns.countplot(x = 'Occupation', data = df1)

sns.set(rc={"figure.figsize":(35,200)})# 35 = width, 200 = height
for bars in Ox.containers:
    Ox.bar_label(bars)
```



```
sales_Oc = df1.groupby(['Occupation'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)
sns.set(rc={"figure.figsize":(35,5)})# 35 = width, 5 = height
sns.barplot(x ='Occupation', y ='Amount',  data = sales_Oc)
```

```
<Axes: xlabel='Occupation', ylabel='Amount'>
```



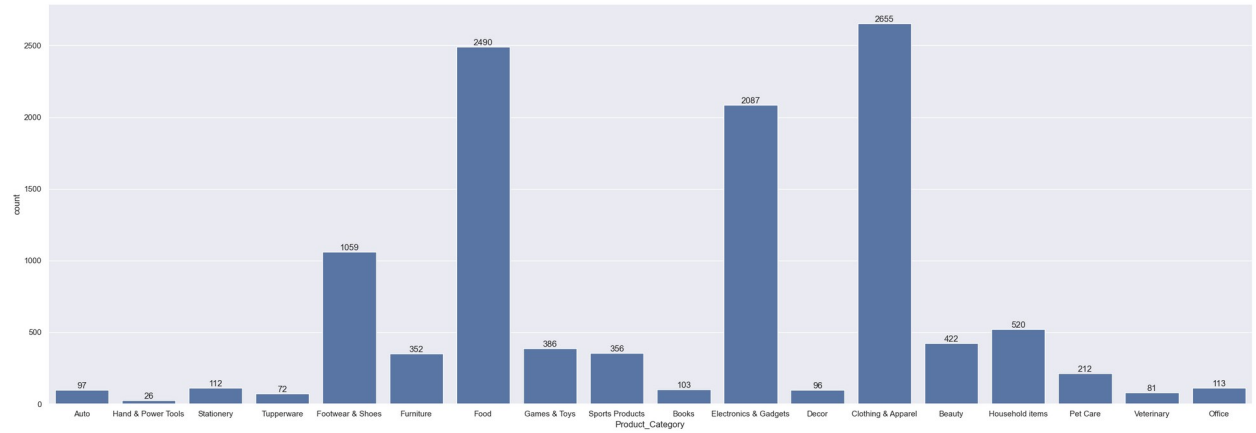
From the above analysis the most of the buyers are from the IT sector followed by Health care then comes the Aviation.

```
df1.columns
```

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

Product category

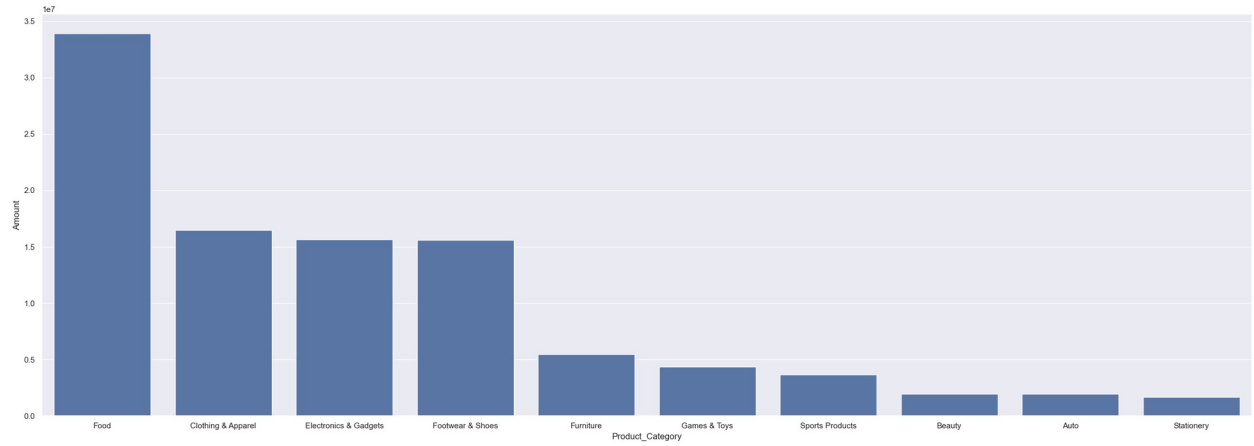
```
sns.set(rc={'figure.figsize':(30,10)})
Px = sns.countplot(data = df1, x = 'Product_Category')

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




```
sales_Pro = df1.groupby(['Product_Category'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)
```

```
sns.set(rc={'figure.figsize':(30,10)})
sns.barplot(data = sales_Pro, x = 'Product_Category',y= 'Amount')
```

```
<Axes: xlabel='Product_Category', ylabel='Amount'>
```



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

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