Data analysis of Diwali Sales

Data cleaning to exploratory analysis

Sachin Borse

https://github.com/SachinBorse009

```
import numpy as np # for arrays(mathamatical use)
import pandas as pd # for dataframe
import matplotlib.pyplot as plt # visualing data
%matplotlib inline
import seaborn as sns

#import csv file
df=pd.read_csv('Diwali Sales Data.csv', encoding= 'unicode_escape') #
to avoid ecoding error, use 'unicode_escape'

#find count of rows and columns
df.shape
```

```
(11251, 15)
```

```
# to view top 5 rows
df.head()
```

```
   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


            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
```

```
    Amount  Status  unnamed1
0  23952.0     NaN       NaN
1  23934.0     NaN       NaN
2  23924.0     NaN       NaN
3  23912.0     NaN       NaN
4  23877.0     NaN       NaN
```

```
#to view more rows enter rows number into bracket
df.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
```

start data cleaning

```
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  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
```

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

#cheack null values
pd.isnull(df)  # if you find true value that means that value is null
```

```
       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]
```

```
#find the sum of null values
pd.isnull(df).sum()
#as we can see all the columns has 0 null value except Amount col
```

```
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
```

```
df.shape
```

```
(11251, 13)
```

```
#so,we have to drop null values from Amount col
df.dropna(inplace=True)
```

```
df.shape
# as we see null vlaues are deleted
```

```
(11239, 13)
```

```
pd.isnull(df).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
```

```
#find data types of schema
df.dtypes
```

```
User_ID               int64
Cust_name            object
Product_ID           object
Gender               object
Age Group            object
Age                   int64
Marital_Status        int64
State                object
Zone                 object
Occupation           object
Product_Category     object
Orders                int64
Amount              float64
dtype: object
```

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

df['Amount'].dtypes
```

```
dtype('int32')
```

```
# find coloums
df.columns
```

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

```
#rename col name
df.rename(columns = {'Marital_Status' : 'Shaadi'})
```

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

...        ...          ...        ...    ...       ...  ...     ...
```

```
11246  1000695      Manning  P00296942      M     18-25   19       1

11247  1004089  Reichenbach  P00171342      M     26-35   33       0

11248  1001209        Oshin  P00201342      F     36-45   40       0

11249  1004023       Noonan  P00059442      M     36-45   37       0

11250  1002744      Brumley  P00281742      F     18-25   19       0


                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
...               ...       ...              ...              ...
...
11246     Maharashtra   Western         Chemical           Office
4
11247         Haryana  Northern       Healthcare       Veterinary
3
11248  Madhya Pradesh   Central          Textile           Office
4
11249       Karnataka  Southern      Agriculture           Office
3
11250     Maharashtra   Western       Healthcare           Office
3

       Amount
0       23952
1       23934
2       23924
3       23912
4       23877
...       ...
11246     370
11247     367
11248     213
11249     206
11250     188

[11239 rows x 13 columns]
```

```
#decription() method return description of the data in the
DataFrame(i.e. count,mean,std,etc)
df.describe()
```

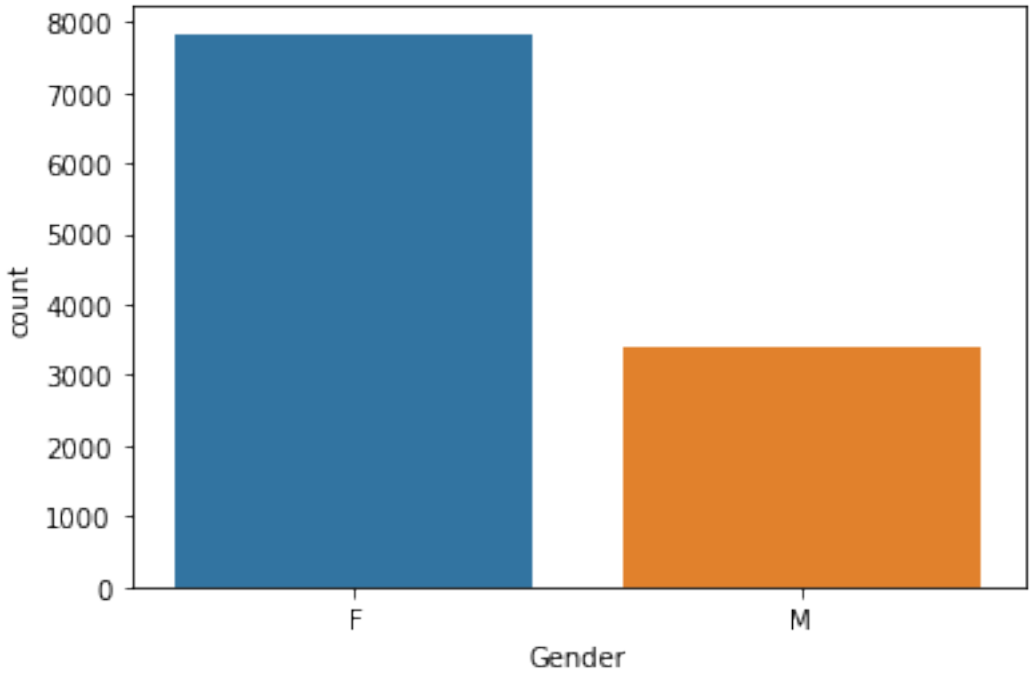
```
             User_ID           Age  Marital_Status        Orders
Amount
count  1.123900e+04  11239.000000    11239.000000  11239.000000
11239.000000
mean   1.003004e+06     35.410357        0.420055      2.489634
9453.610553
std    1.716039e+03     12.753866        0.493589      1.114967
5222.355168
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
```

```
#use describe for specific col
df[['Age', 'Orders']].describe()
```

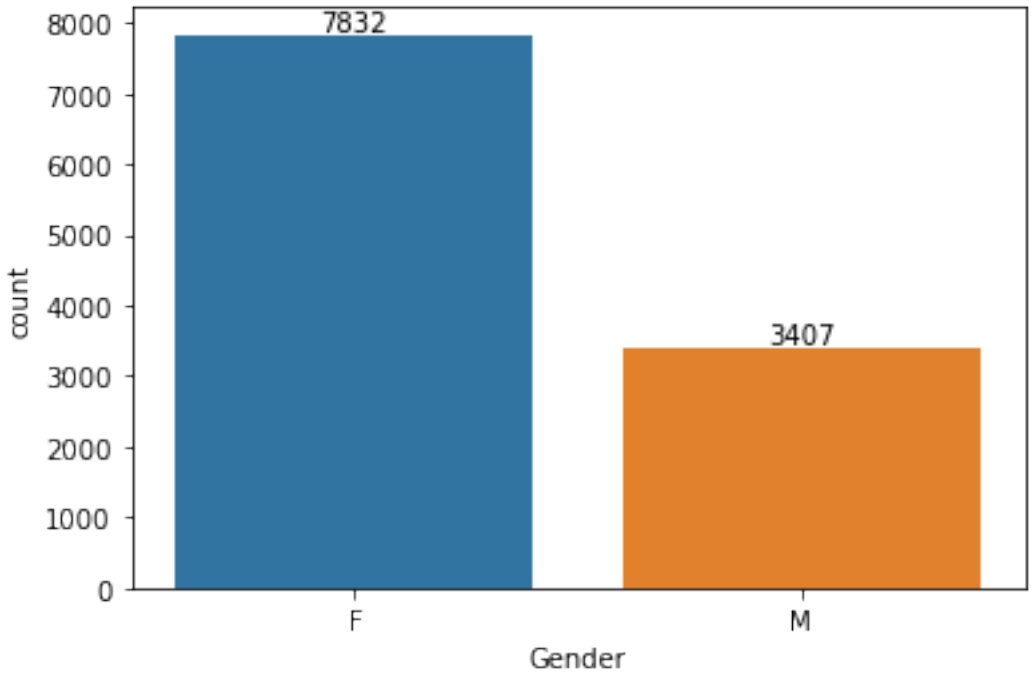
```
                Age        Orders
count  11239.000000  11239.000000
mean      35.410357      2.489634
std       12.753866      1.114967
min       12.000000      1.000000
25%       27.000000      2.000000
50%       33.000000      2.000000
75%       43.000000      3.000000
max       92.000000      4.000000
```

Exploratory Data Analysis

```
#gender count visually by seaborn
ax = sns.countplot(x = 'Gender', data=df)
```




```
# there is no value in the bar so we can write following code
ax = sns.countplot(x='Gender',data=df)

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

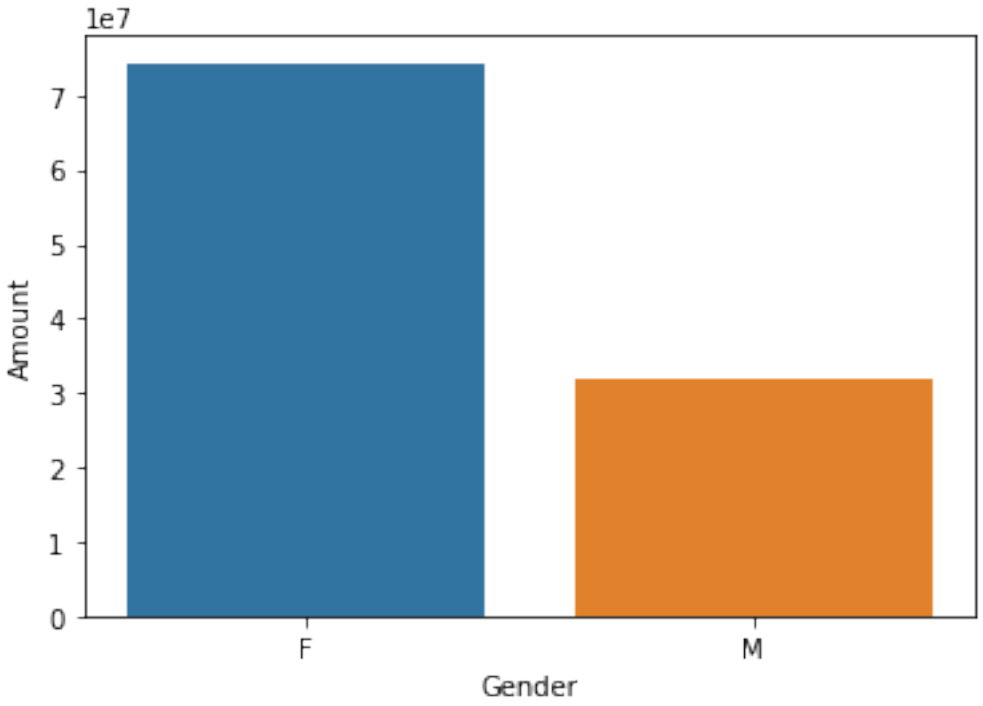


```
#total sum of genderwise sales
sales_gen = df.groupby(['Gender'],as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)

sales_gen
```

	Gender	Amount
0	F	74335853
1	M	31913276

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

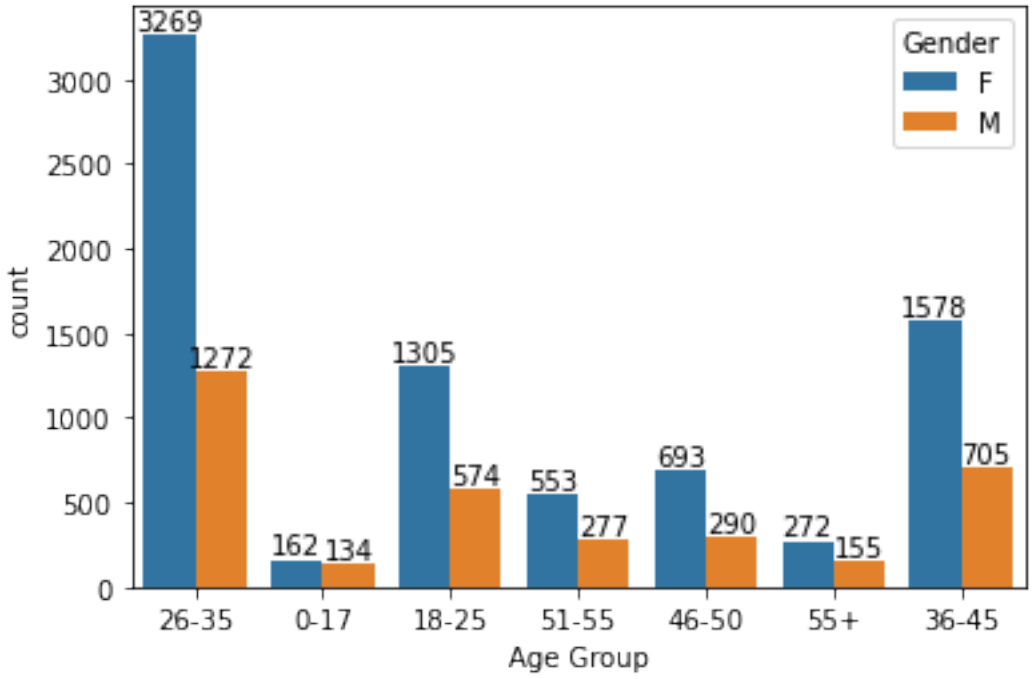
```
<AxesSubplot:xlabel='Gender', ylabel='Amount'>
```



from the above graph we can see that most of the buyes are females and even the purchasing power of female are greater than men

Age

```
#find buyers age-group
ax = sns.countplot(x = 'Age Group', hue = 'Gender', data=df)

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

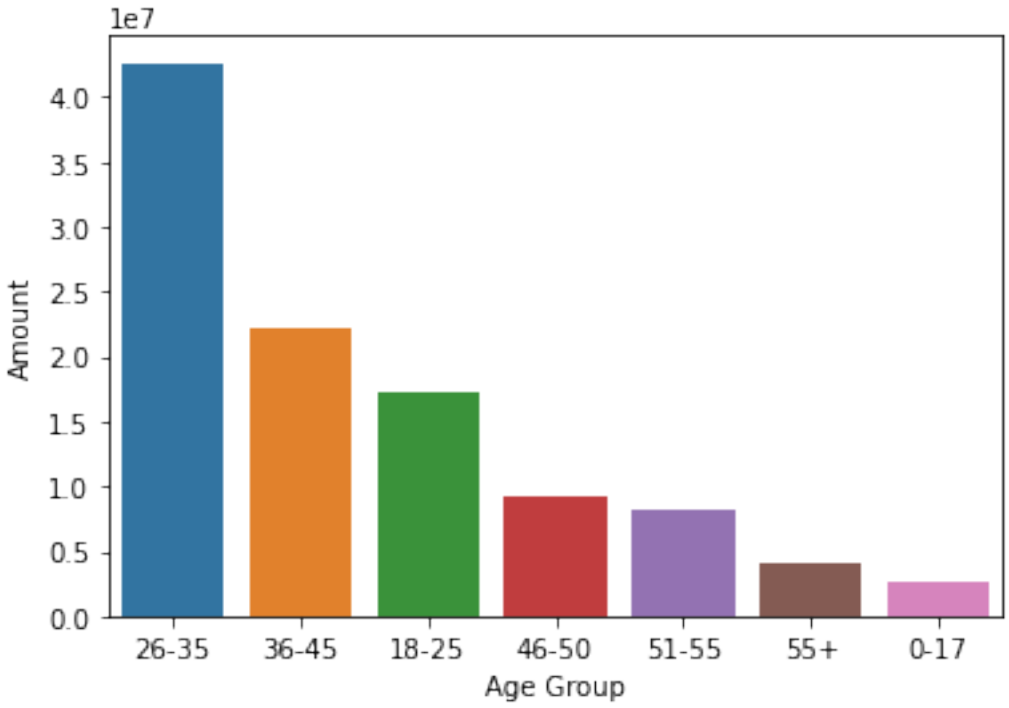



```
#Total sales vs Age Group
sales_age = df.groupby(['Age Group'],as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)

sales_age
```

	Age Group	Amount
2	26-35	42613442
3	36-45	22144994
1	18-25	17240732
4	46-50	9207844
5	51-55	8261477
6	55+	4080987
0	0-17	2699653

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

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



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

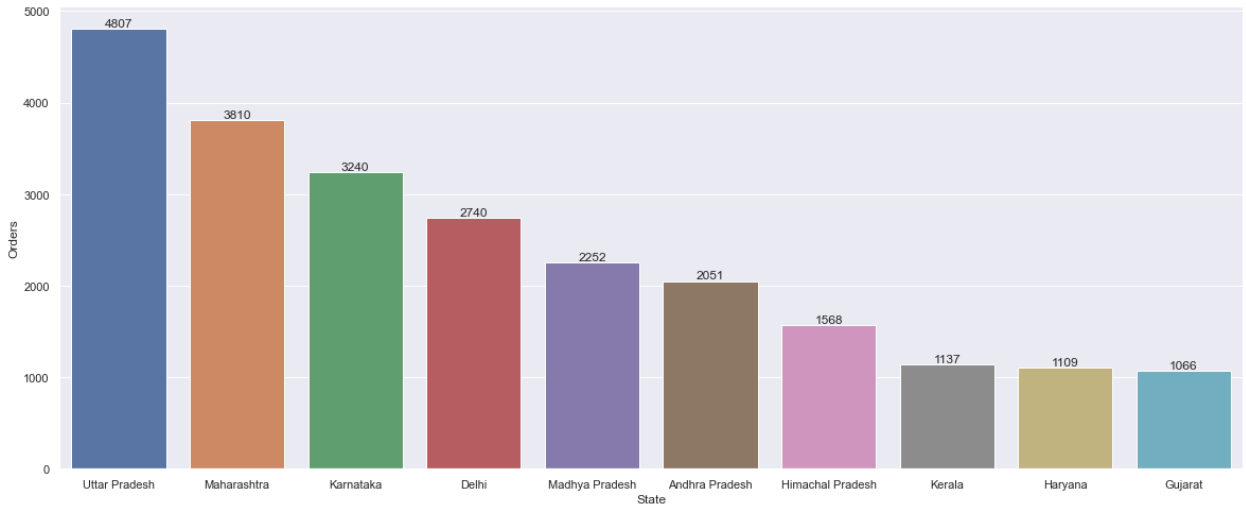
State

```
# Total number of order from top 10 states
sales_state = df.groupby(['State'], as_index=False)
['Orders'].sum().sort_values(by=['Orders'], ascending=False).head(10)

sales_state
```

	State	Orders
14	Uttar Pradesh	4807
10	Maharashtra	3810
7	Karnataka	3240
2	Delhi	2740
9	Madhya Pradesh	2252
0	Andhra Pradesh	2051
5	Himachal Pradesh	1568
8	Kerala	1137
4	Haryana	1109
3	Gujarat	1066

```
sns.set(rc={'figure.figsize':(20,8)})
ax = sns.barplot(data=sales_state, x= 'State' , y = 'Orders')
```

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

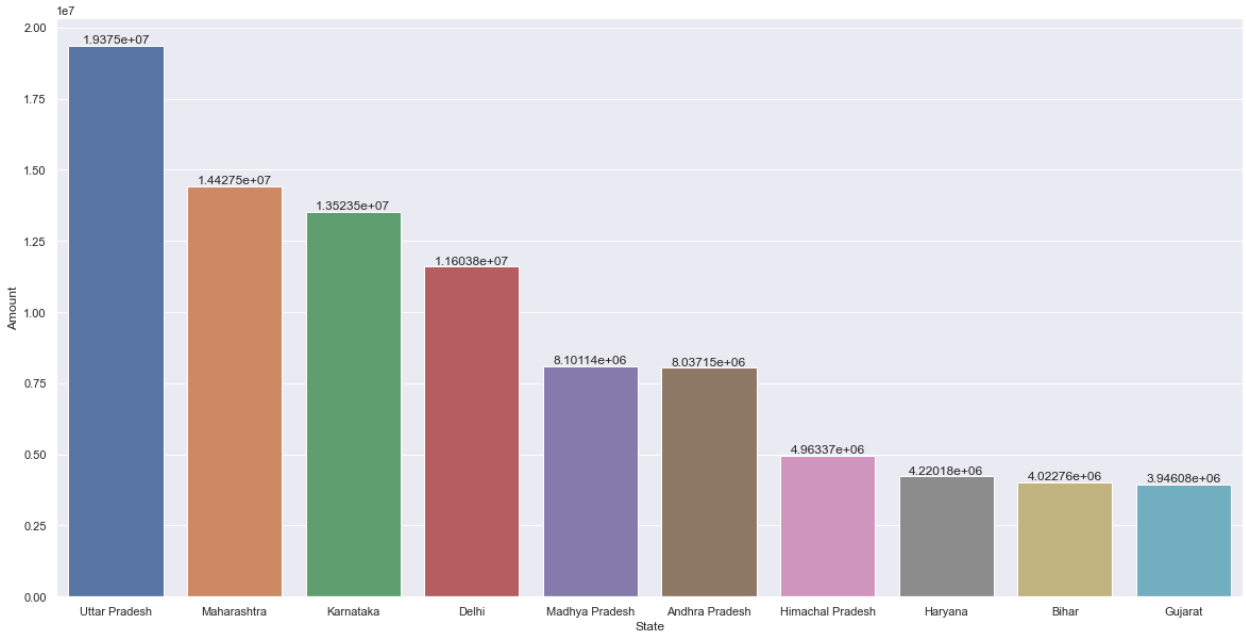


From above grapth we can see top 10 states which has total number of orders

```
## Total amount/sales from top 10 states

sales_amount = df.groupby(['State'] , as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending = False).head(10)
sales_amount
```

	State	Amount
14	Uttar Pradesh	19374968
10	Maharashtra	14427543
7	Karnataka	13523540
2	Delhi	11603818
9	Madhya Pradesh	8101142
0	Andhra Pradesh	8037146
5	Himachal Pradesh	4963368
4	Haryana	4220175
1	Bihar	4022757
3	Gujarat	3946082

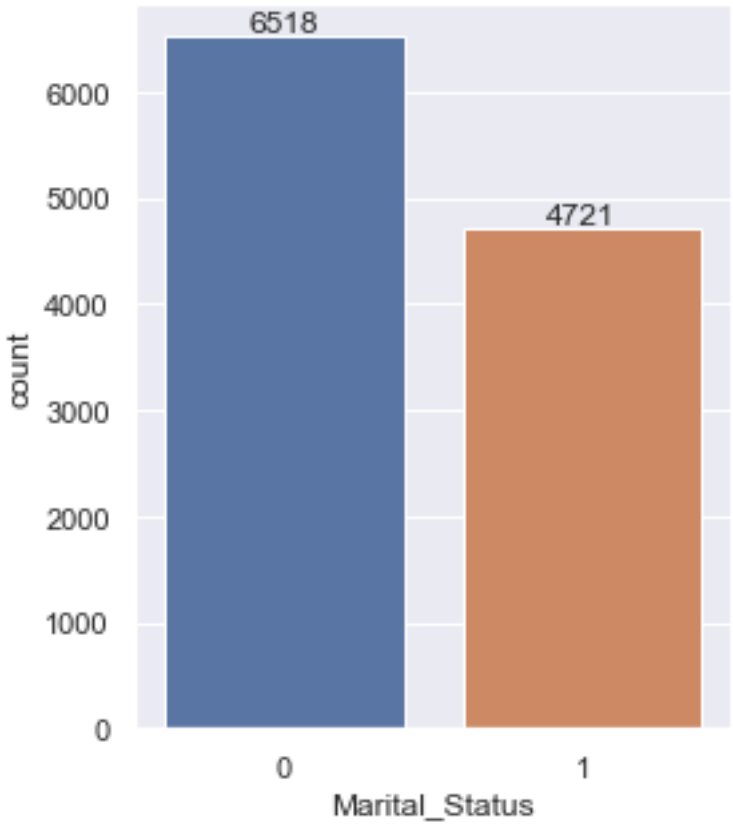
```
sns.set(rc={'figure.figsize' :(20,10)})
ax = sns.barplot(x = 'State', y = 'Amount', data=sales_amount)

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



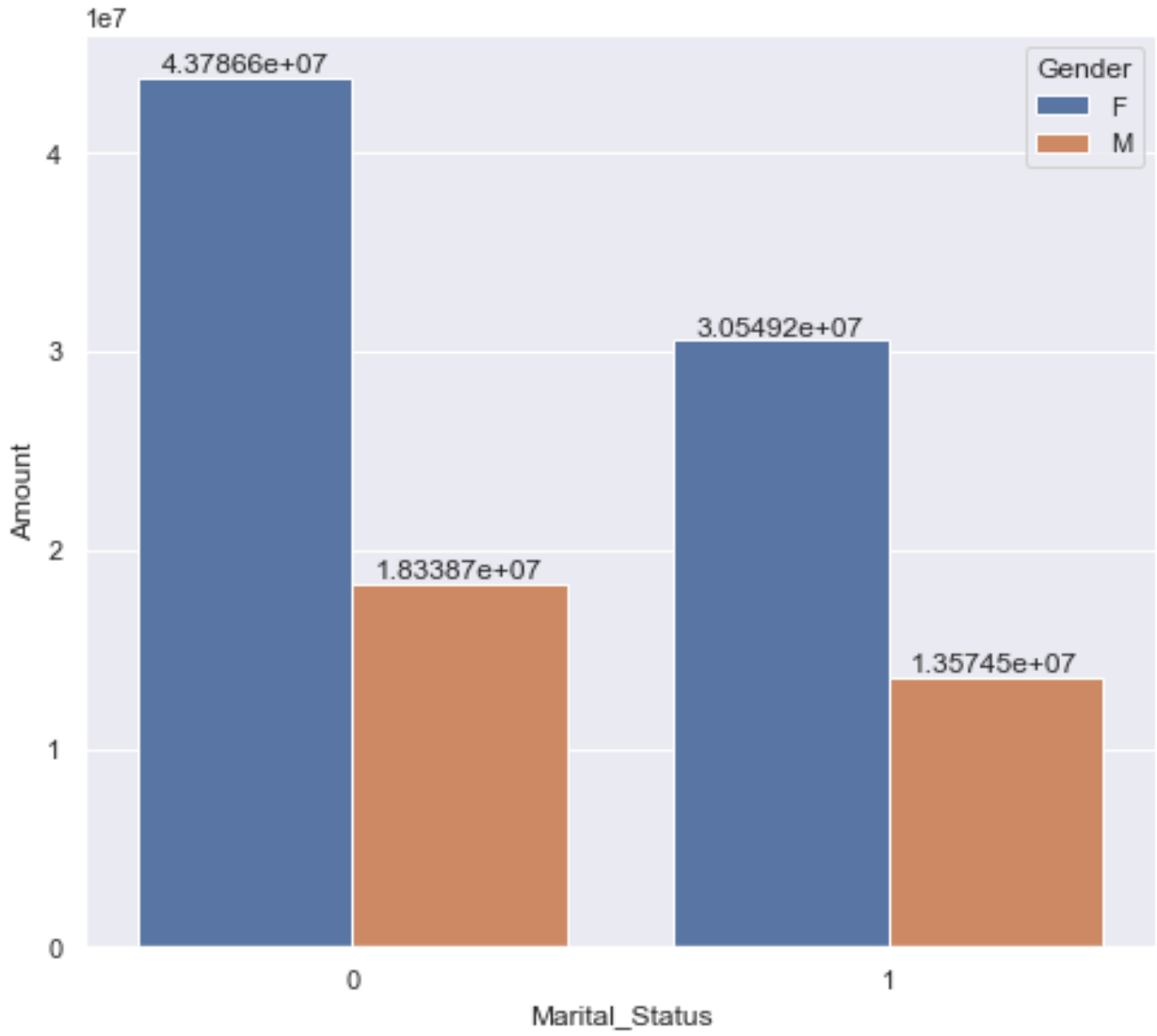
From above graph we can seee that most of the orders and sales/amount are from Uttar Pradesh, Maharashtra and Karnataka respectively

Marial Status

```
df.columns

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

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

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




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

sns.set(rc={'figure.figsize': (8,7)})
ax = sns.barplot(data = sales_state , x= 'Marital_Status', y =
'Amount', hue='Gender')

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



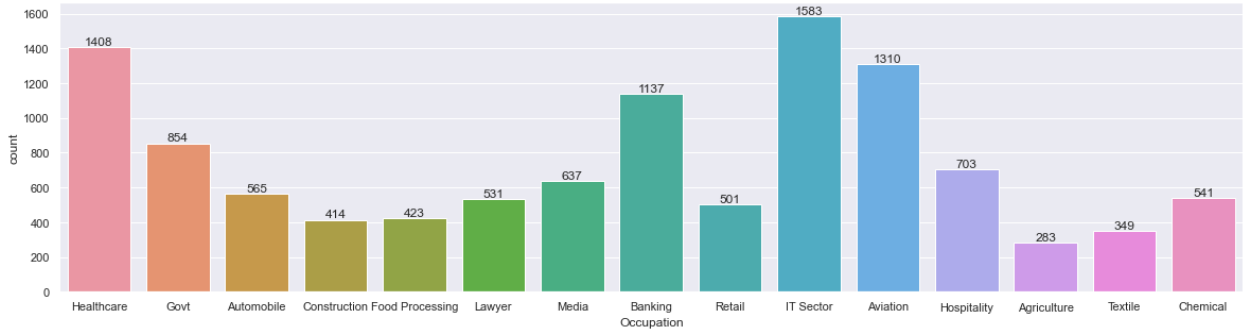
From above graph we can see that most of the buyers are married(women) and they have high purchasing power

Occupation

```
df.columns
```

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

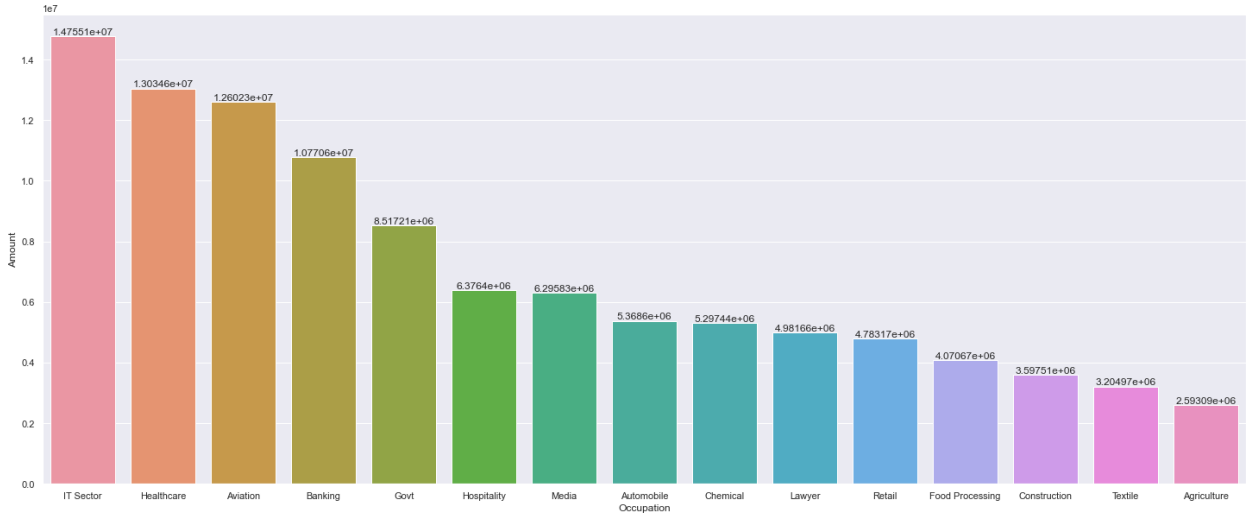
```
sns.set(rc={'figure.figsize': (20,5)})
ax = sns.countplot(data=df, x = 'Occupation')

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




```
## total sales/amount vs occupation

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

	Occupation	Amount
10	IT Sector	14755079
8	Healthcare	13034586
2	Aviation	12602298
3	Banking	10770610
7	Govt	8517212
9	Hospitality	6376405
12	Media	6295832
1	Automobile	5368596
4	Chemical	5297436
11	Lawyer	4981665
13	Retail	4783170
6	Food Processing	4070670
5	Construction	3597511
14	Textile	3204972
0	Agriculture	2593087

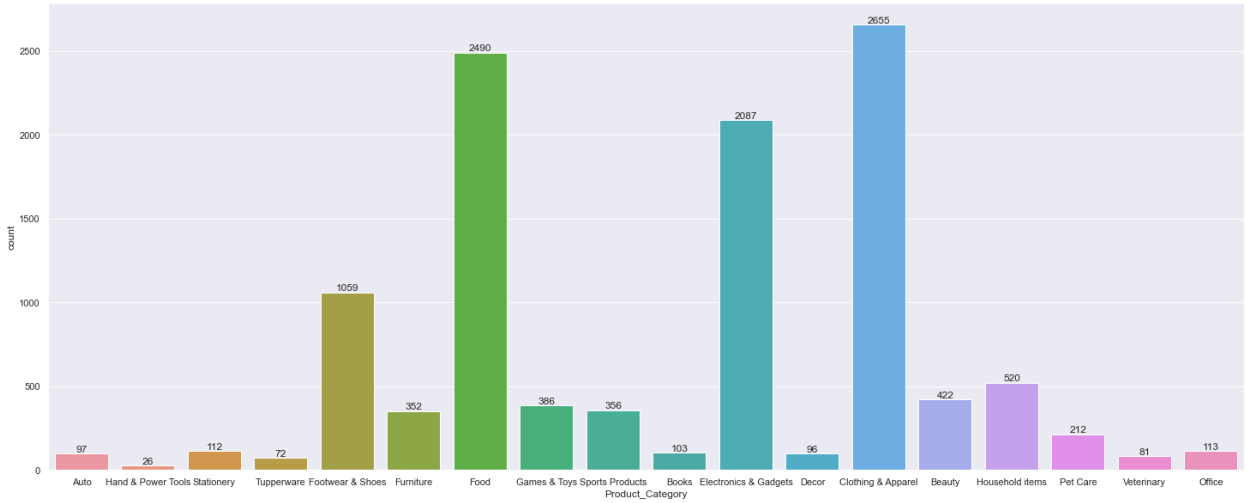
```
sns.set(rc={'figure.figsize': (25, 10)})
ax = sns.barplot(data = sales_occupation, x='Occupation', y =
'Amount')
for bars in ax.containers:
    ax.bar_label(bars)
```



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

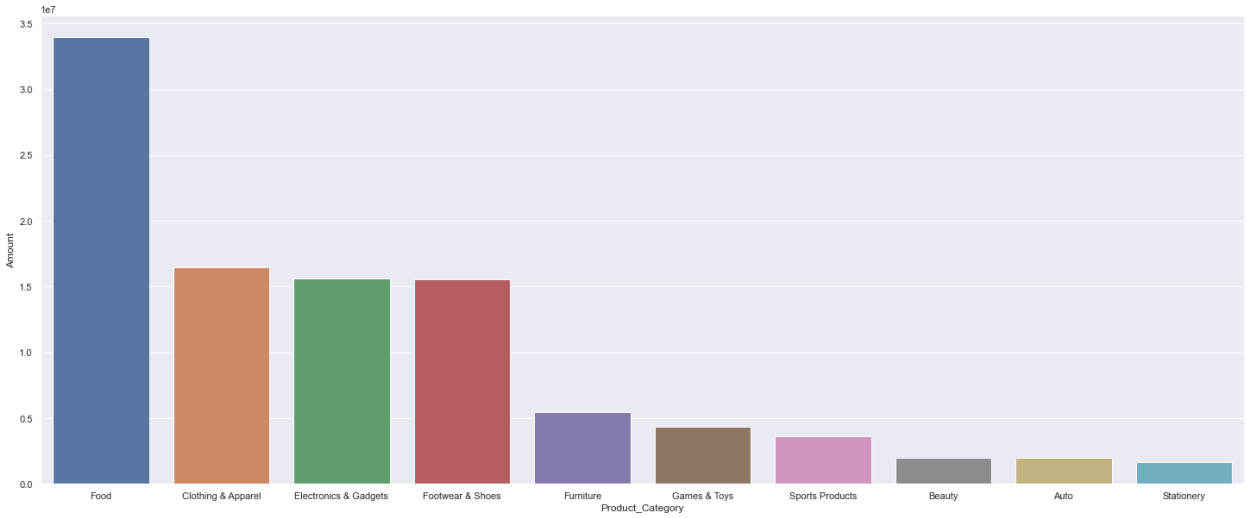
Product Category`

```
df.columns

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

sns.set(rc={'figure.figsize': (25,10)})
ax = sns.countplot(data=df, x ='Product_Category')

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




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

sns.barplot(data= sales_product, x = 'Product_Category' , y =
'Amount')
```

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



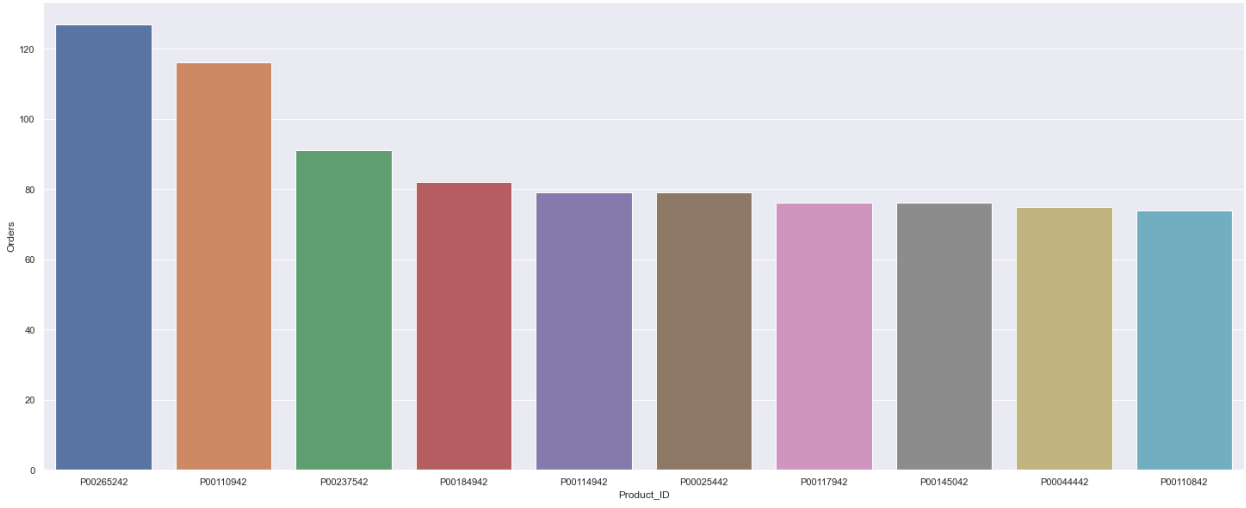
From above graph we can see that most of the sold product are from Food , Cloting and apparel & Electronics category

```
# find top selling product

sales_order = df.groupby('Product_ID', as_index=False)
['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)
sales_order
```

```
     Product_ID  Orders
1679  P00265242     127
644   P00110942     116
1504  P00237542      91
1146  P00184942      82
679   P00114942      79
171   P00025442      79
708   P00117942      76
888   P00145042      76
298   P00044442      75
643   P00110842      74
```

```
sns.barplot(data=sales_order, x = 'Product_ID', y = 'Orders')
```

```
<AxesSubplot:xlabel='Product_ID', ylabel='Orders'>
```



Conclusion:

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

Project on Github: https://github.com/SachinBorse009