Diwali Sales Analysis Project

Import Required Libraries

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

dataset = pd.read_csv("Diwali Sales Data.csv", encoding=
'unicode_escape')

# Displying number of rows and columns
dataset.shape
```

```
(11251, 15)
```

```
# Displying above five rows data
dataset.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
```

```
dataset.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
```

```
# Checking Missing value
```

```
dataset.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
Status              11251
unnamed1            11251
dtype: int64
```

Above we have seemed Complete missing data in two column, so drop these column

```
dataset.drop(['Status','unnamed1'], axis = 1, inplace = True)    # 
dropping column existing in dataset

# After dropping displying missing data

dataset.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
```

```
# drop null value

dataset.dropna(inplace = True)

dataset.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
```

- change data type

```
dataset['Amount'] = dataset['Amount'].astype('int')

dataset['Amount'].dtypes
```

```
dtype('int32')

dataset.columns

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

- renaming column name

```
dataset.rename(columns= {'Marital_Status':'Shaadi'}).head()

   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

            State      Zone       Occupation Product_Category  Orders
Amount
0     Maharashtra   Western       Healthcare             Auto       1
23952
1  Andhra Pradesh  Southern             Govt             Auto       3
23934
2   Uttar Pradesh   Central       Automobile             Auto       3
23924
3       Karnataka  Southern     Construction             Auto       2
23912
4         Gujarat   Western  Food Processing             Auto       2
23877
```

- describe() method returns description of the data in the DataFrame (i.e. count, mean, std, etc)

```
dataset.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() method for specific columns

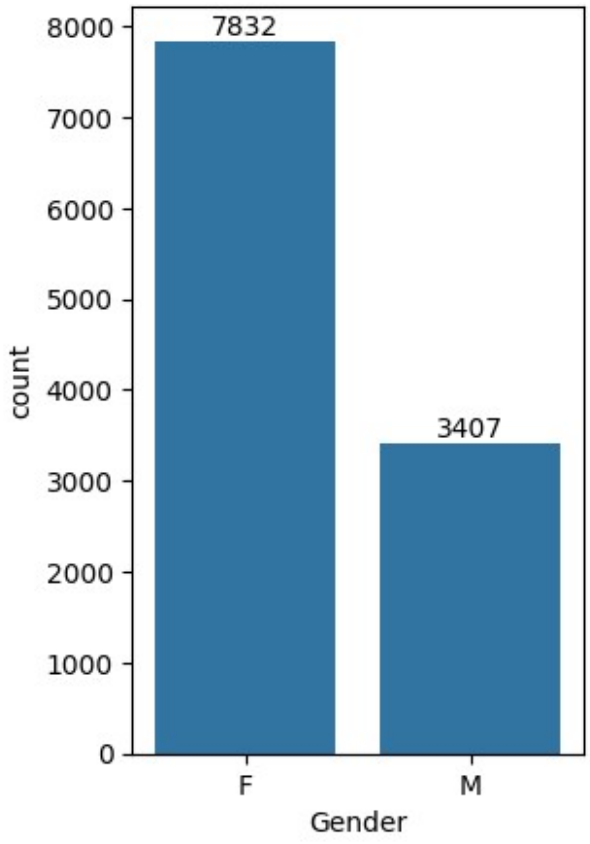
```
dataset[['Age', 'Orders', 'Amount']].describe()
```

```
                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
```

Exploratory Data Analysis

- plotting a bar chart for **Gender** and it's count

```
plt.figure(figsize = (3,5))

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

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



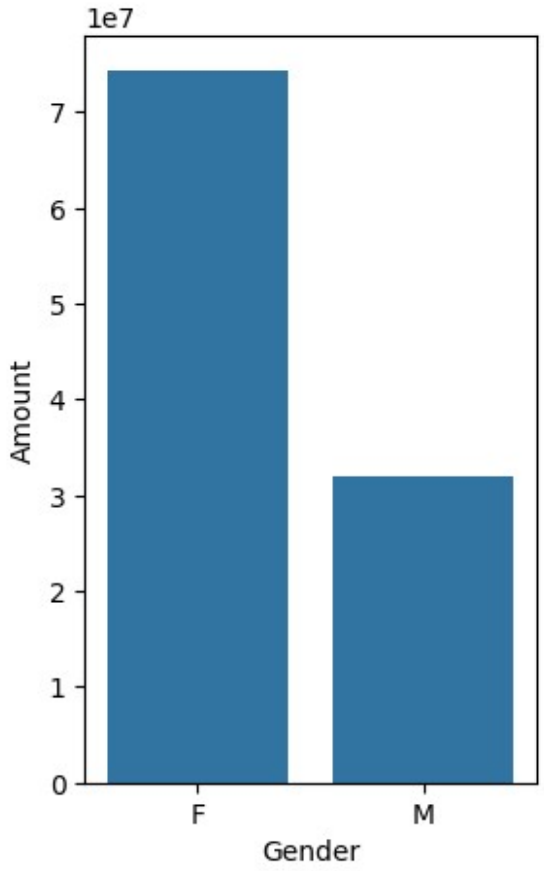
- plotting a bar chart for **gender vs total amount**

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

sales_gender
```

```
  Gender    Amount
0      F  74335853
1      M  31913276
```

```
plt.figure(figsize = (3,5))

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

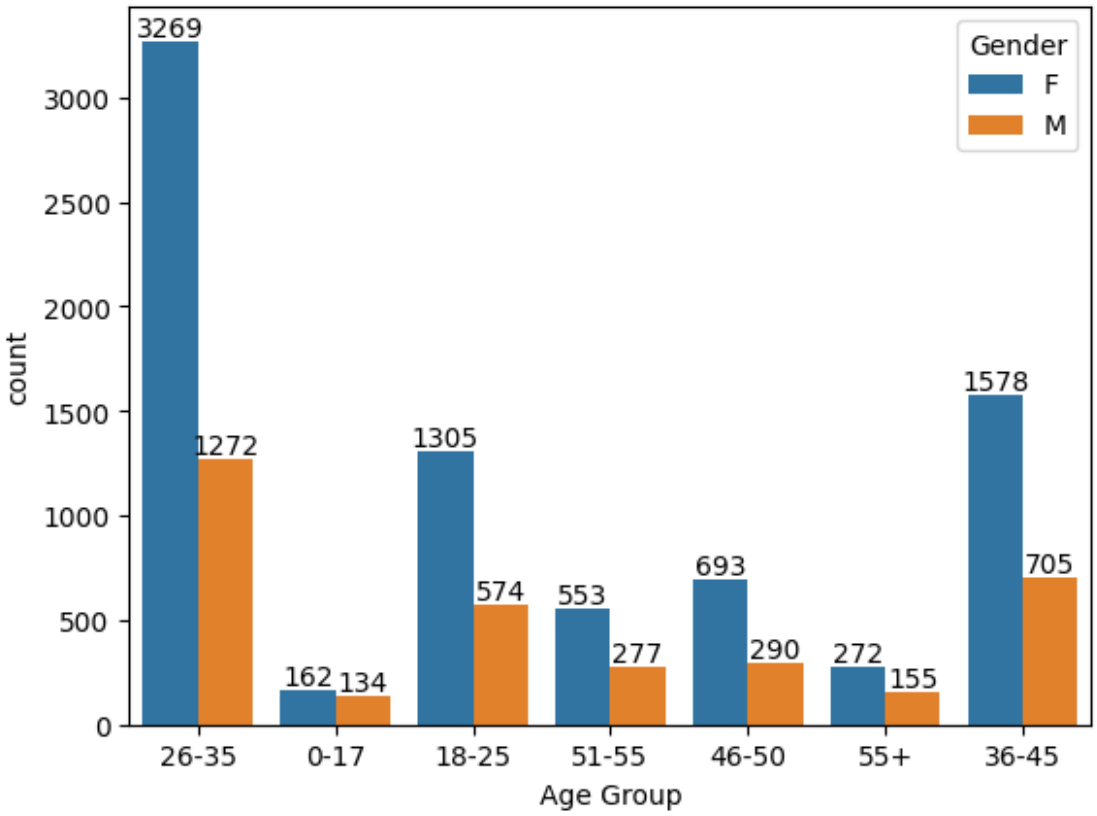
```
<Axes: xlabel='Gender', ylabel='Amount'>
```



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

- According to **Age Group** Counting Male & Female

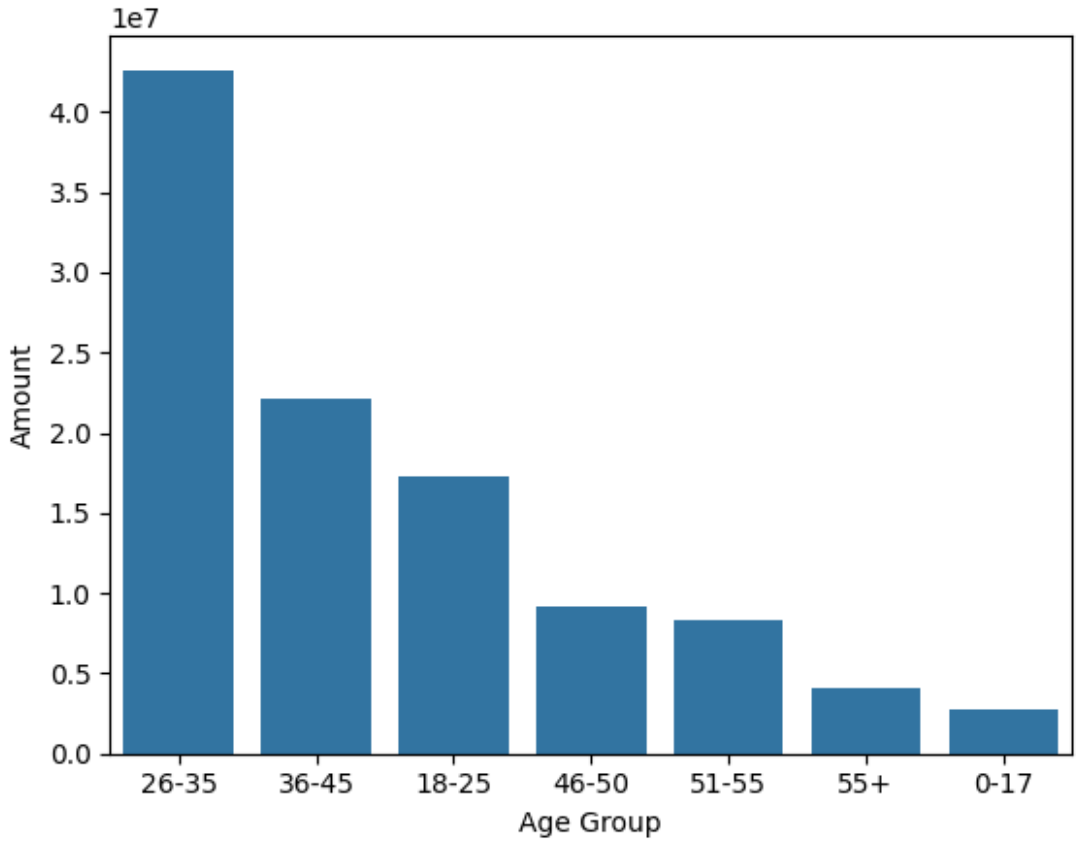
```
ax = sns.countplot(data = dataset, x = 'Age Group', hue = 'Gender')

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



- Total Amount vs Age Group

```
sales_age = dataset.groupby(['Age Group'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False)

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

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



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

- total number of orders from top 10 states

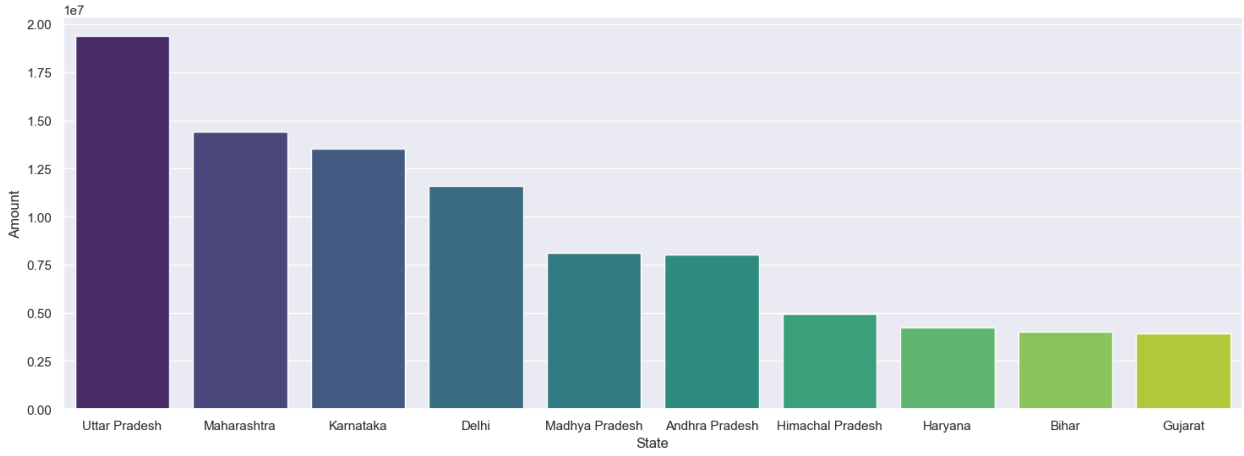
```
sales_state = dataset.groupby(['State'], as_index=False)
['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)

sns.set(rc = {'figure.figsize':(16,5)})

sns.barplot(data = sales_state, x = 'State', y = 'Orders', palette =
'viridis', hue = 'State')
```

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

- total amount/sales from top 10 states

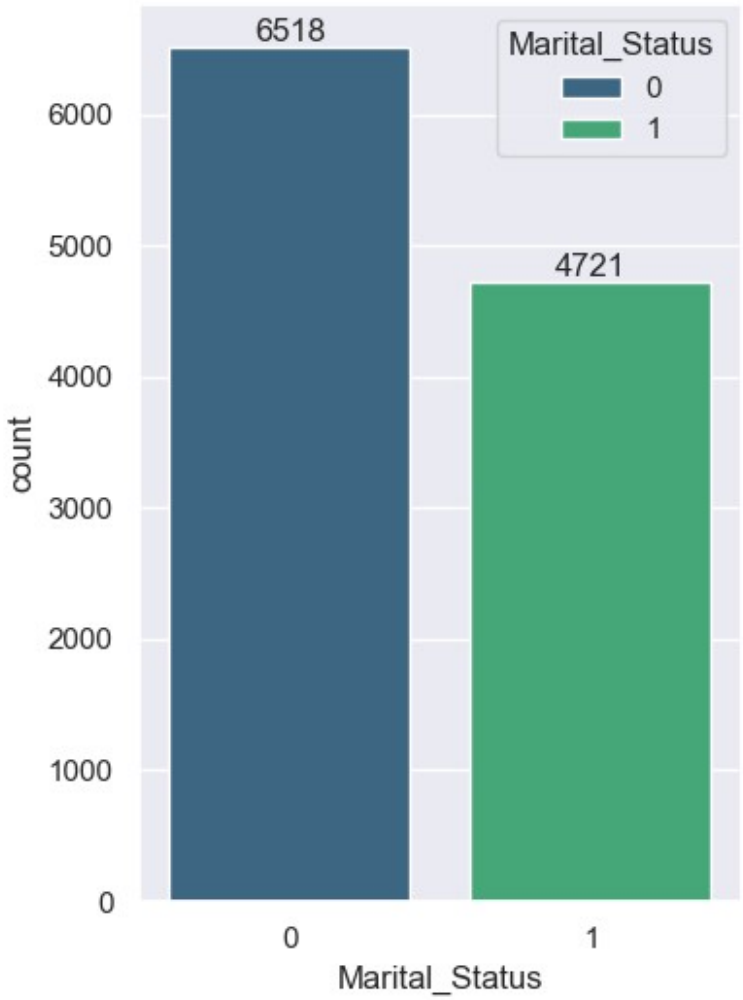
```
sales_state = dataset.groupby(['State'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)

sns.set(rc={'figure.figsize':(18,6)})

sns.barplot(data = sales_state, x = 'State',y= 'Amount', palette =
'viridis', hue = 'State')
```

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



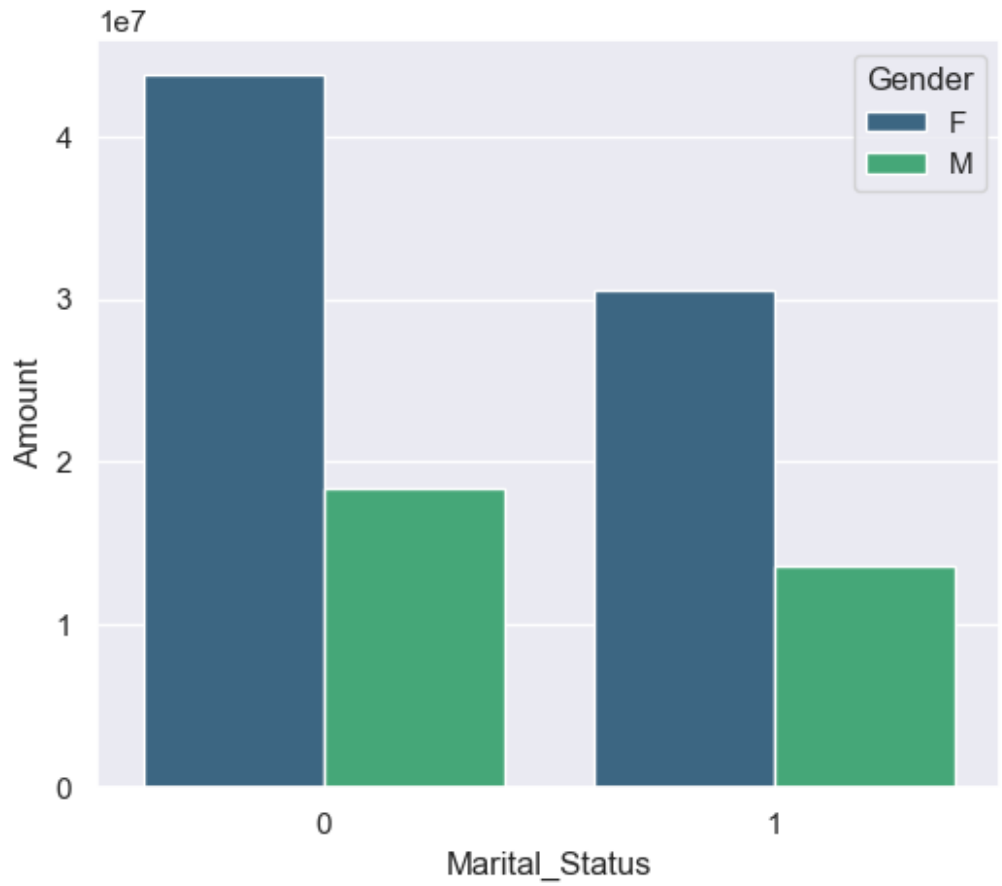
- Marital Status

```
ax = sns.countplot(data = dataset, x = 'Marital_Status',hue =
'Marital_Status', palette = 'viridis')

sns.set(rc={'figure.figsize':(4,6)})

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




```
sales_state = dataset.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_state, x = 'Marital_Status',y= 'Amount',
hue='Gender', palette = 'viridis')
```

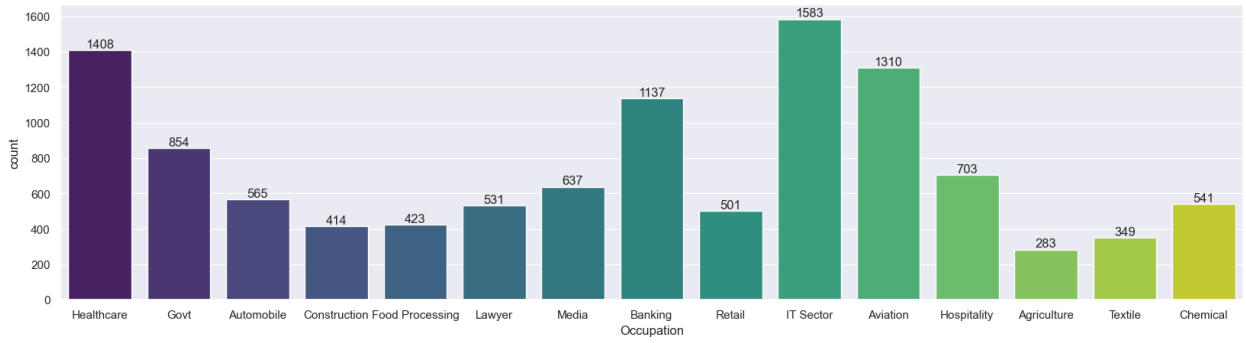
```
<Axes: xlabel='Marital_Status', ylabel='Amount'>
```



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

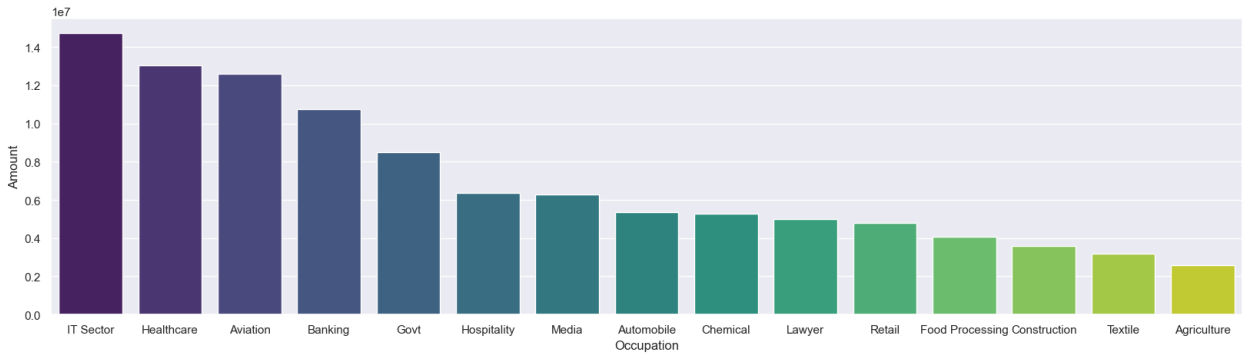
- Occupation Count

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

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



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

sns.set(rc = {'figure.figsize':(20,5)})

sns.barplot(data = sales_state, x = 'Occupation', y = 'Amount',
palette = 'viridis', hue = 'Occupation')
```

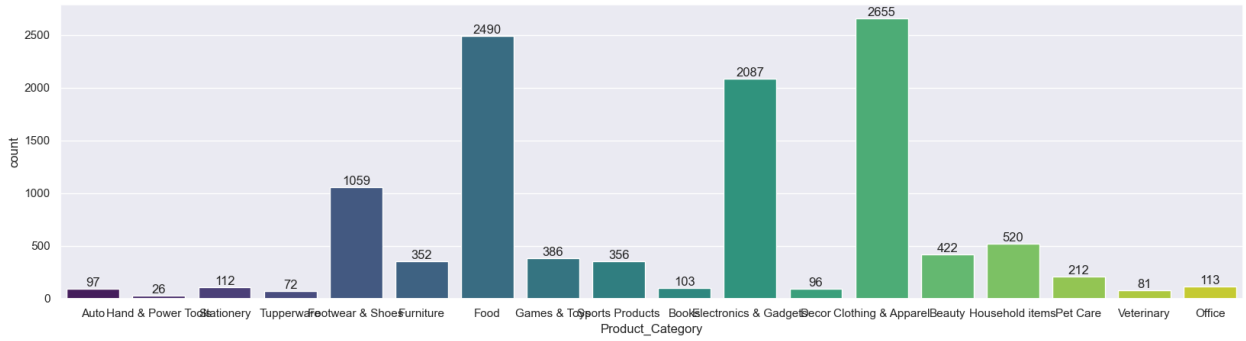
```
<Axes: xlabel='Occupation', ylabel='Amount'>
```



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

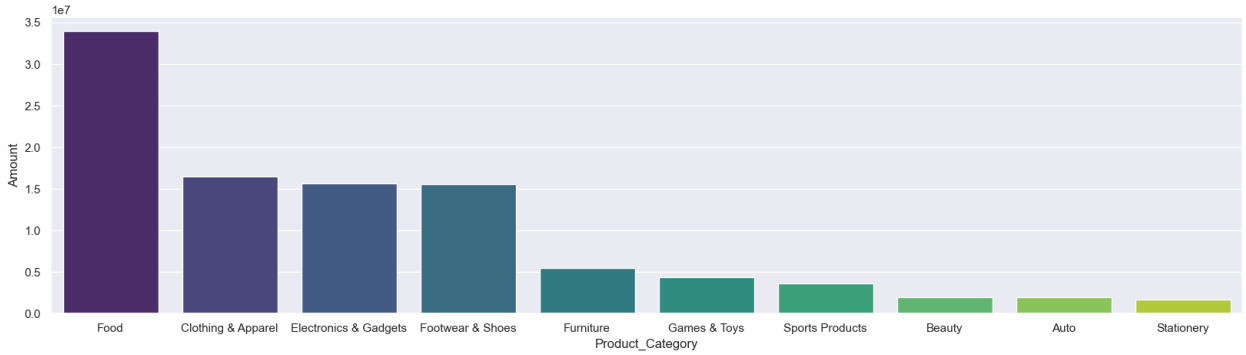
- Product Category

```
# Ignore futurewarnings

import warnings
warnings.filterwarnings("ignore")

sns.set(rc={'figure.figsize':(20,5)})

ax = sns.countplot(data = dataset, x = 'Product_Category', palette =
'viridis')

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



```
sales_state = dataset.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_state, x = 'Product_Category',y= 'Amount',
hue = 'Product_Category', palette = 'viridis')
```

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

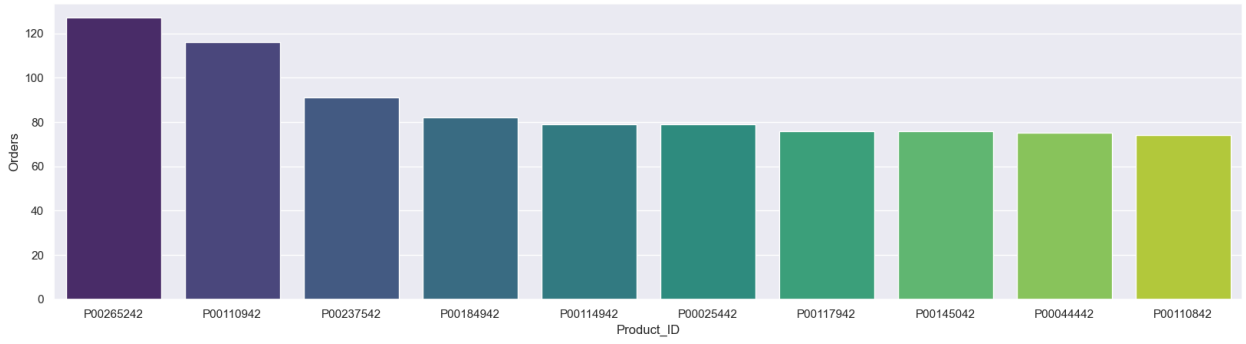


From above graphs we can see that most of the sold products are from Food, Clothing and Electronics category

- According to Product_Id Orders

```
sales_state = dataset.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_state, x = 'Product_ID',y= 'Orders', palette
= 'viridis')
```

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



- top 10 most sold products (same thing as above)

```
fig1, ax1 = plt.subplots(figsize=(12,7))

dataset.groupby('Product_ID')
```

```
['Orders'].sum().nlargest(10).sort_values(ascending=False).plot(kind='
bar')
```

```
<Axes: xlabel='Product_ID'>
```