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

```
df = pd.read_csv('/content/drive/MyDrive/Diwali Sales Data.csv', encoding='unicode_escape')
```

```
df.head(5)
```

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

Next steps: Generate code with df New interactive sheet

```
df.shape
```

```
(11251, 15)
```

```
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
```

```
df.describe()
```

	User_ID	Age	Marital_Status	Orders	Amount	Status	unnamed1
count	1.125100e+04	11251.000000	11251.000000	11251.000000	11239.000000	0.0	0.0
mean	1.003004e+06	35.421207	0.420318	2.489290	9453.610858	NaN	NaN
std	1.716125e+03	12.754122	0.493632	1.115047	5222.355869	NaN	NaN
min	1.000001e+06	12.000000	0.000000	1.000000	188.000000	NaN	NaN
25%	1.001492e+06	27.000000	0.000000	1.500000	5443.000000	NaN	NaN
50%	1.003065e+06	33.000000	0.000000	2.000000	8109.000000	NaN	NaN
75%	1.004430e+06	43.000000	1.000000	3.000000	12675.000000	NaN	NaN
max	1.006040e+06	92.000000	1.000000	4.000000	23952.000000	NaN	NaN

```
df.drop(['Status', 'unnamed1'], axis=1, inplace=True)
```

```
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
```

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

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders
0	False	False	False	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...	...	...	...
11246	False	False	False	False	False	False	False	False	False	False	False	False
11247	False	False	False	False	False	False	False	False	False	False	False	False
11248	False	False	False	False	False	False	False	False	False	False	False	False
11249	False	False	False	False	False	False	False	False	False	False	False	False
11250	False	False	False	False	False	False	False	False	False	False	False	False

```
df.isnull().sum()
```

	0
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)
```

```
# drop null values
df.dropna(inplace=True)
```

```
df.shape
```

```
(11239, 13)
```

```
data_test = [['kapil',23], ['sonu',22],['manish' ]]
df_test = pd.DataFrame(data_test, columns=['Name','Age'])
df_test
```

	Name	Age
0	kapil	23.0
1	sonu	22.0
2	manish	NaN

Next steps: Generate code with `df_test` New interactive sheet

```
df_test.dropna()
```

	Name	Age
0	kapil	23.0
1	sonu	22.0

```
df_test
```

	Name	Age
0	kapil	23.0
1	sonu	22.0
2	manish	NaN

Next steps: Generate code with `df_test` New interactive sheet

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

```
df['Amount'].dtype
```

```
dtype('int64')
```

```
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 column
df.rename(columns={'Marital_Status':'Shaadi'})
```

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Shaadi	State	Zone	Occupation	Product_Category	Orde
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	
...	...	...	...	...	...	...	...	...	...	...	...	
11246	1000695	Manning	P00296942	M	18-25	19	1	Maharashtra	Western	Chemical	Office	
11247	1004089	Reichenbach	P00171342	M	26-35	33	0	Haryana	Northern	Healthcare	Veterinary	
11248	1001209	Oshin	P00201342	F	36-45	40	0	Madhya Pradesh	Central	Textile	Office	
11249	1004023	Noonan	P00059442	M	36-45	37	0	Karnataka	Southern	Agriculture	Office	

```
df.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
```

```
# describe() method return description of the data in 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 columns
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

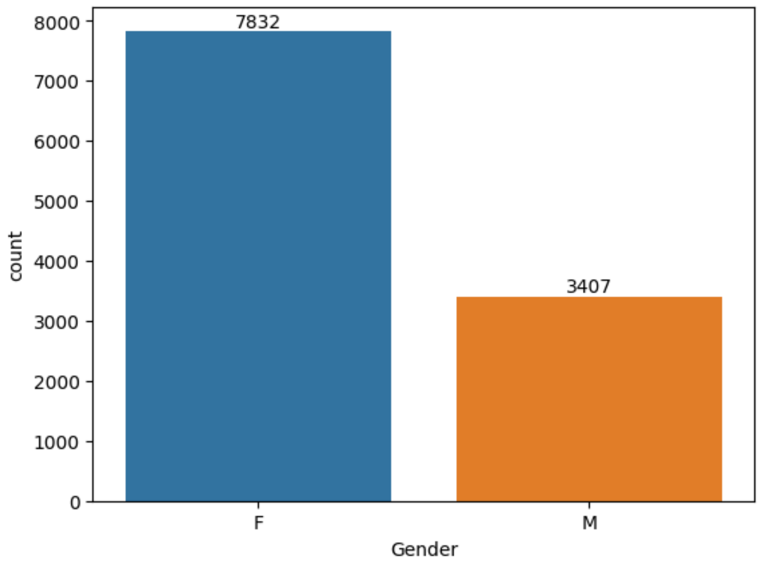
Expolaratory Data analysis

Gender

```
df.columns
```

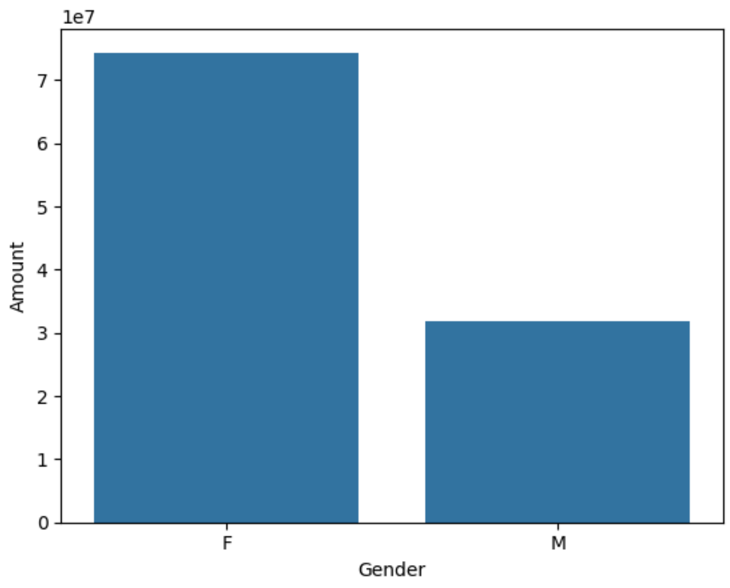
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

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



```
sales_gen = df.groupby(['Gender'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
sns.barplot(x='Gender', y='Amount', data=sales_gen)
```

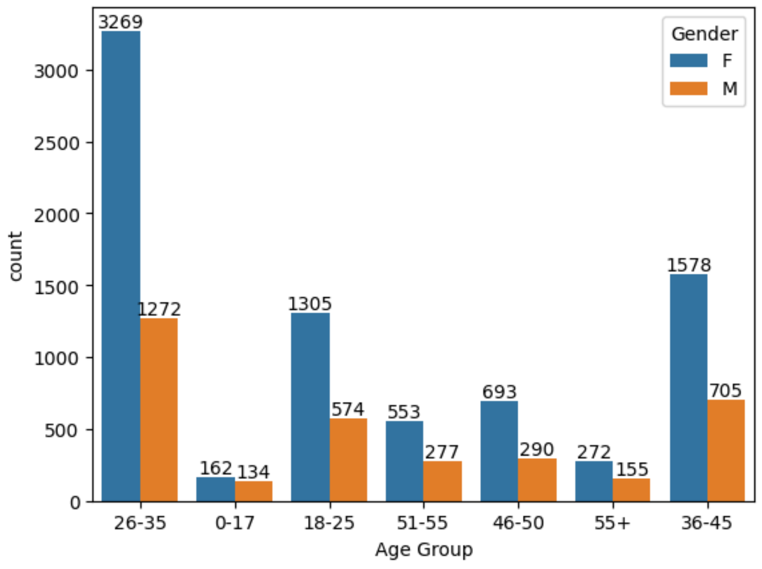
```
<Axes: xlabel='Gender', ylabel='Amount'>
```



from above graphs we can see that most of the buyers are female and even the power of purchasing power of the females are the greater then men

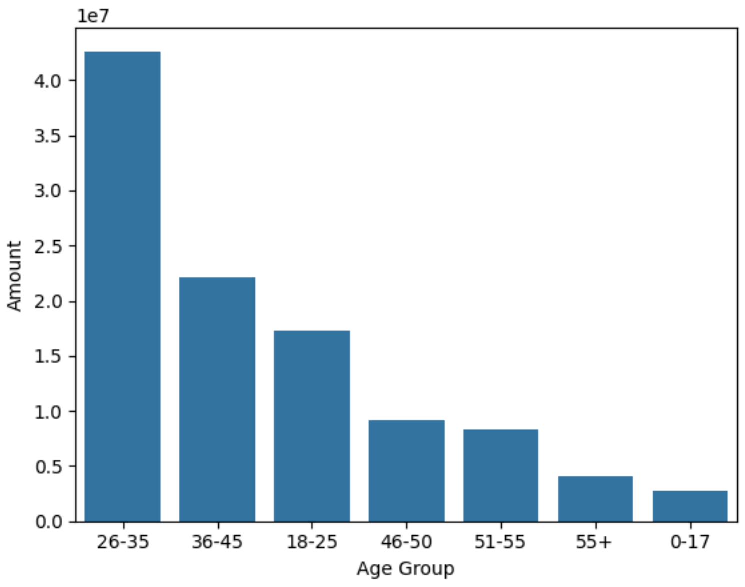
Age

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



```
# total amount vs age group
sales_gen = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
sns.barplot(x='Age Group', y='Amount', data=sales_gen)
```

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



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

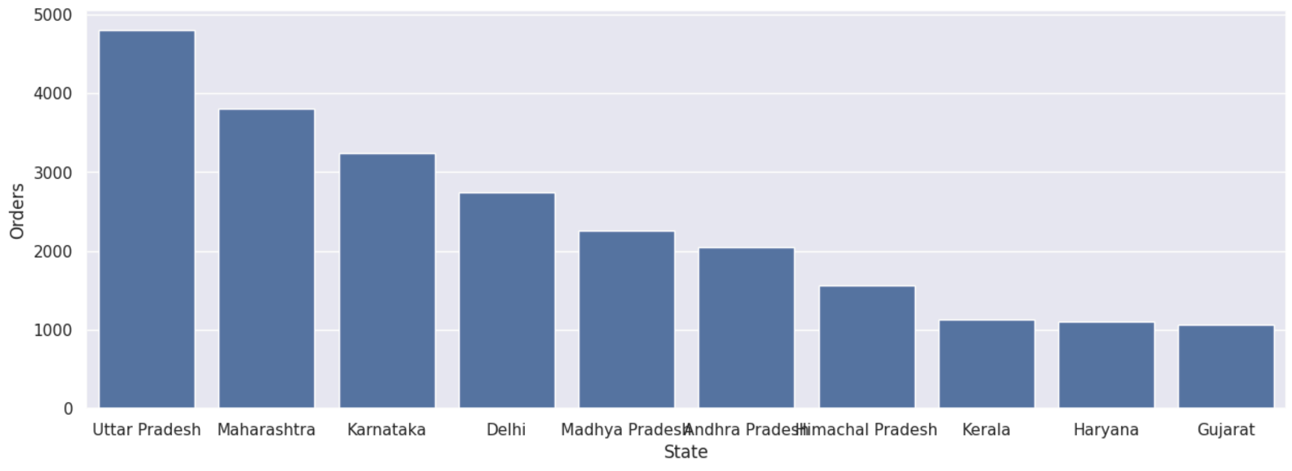
state

```
df.columns
```

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

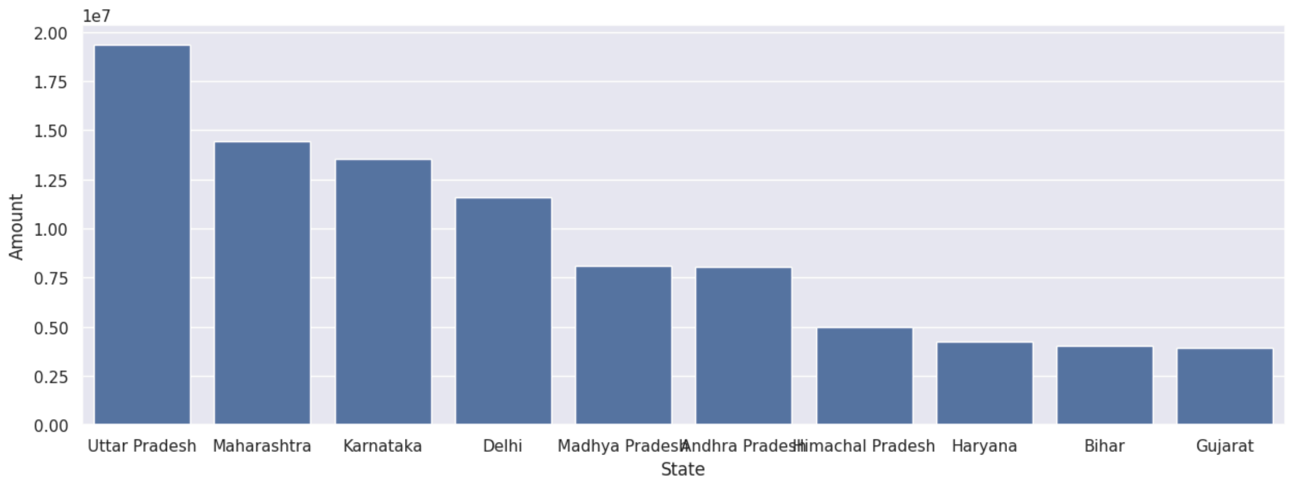
```
# total number of order from top 10 state
sales_state = df.groupby(['State'], as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=False).head(10)
sns.set(rc={'figure.figsize':(15,5)})
sns.barplot(x='State', y='Orders', data=sales_state)
```

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



```
#total amount/sales top 10 state
sales_state = 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(x='State', y='Amount', data=sales_state)
```

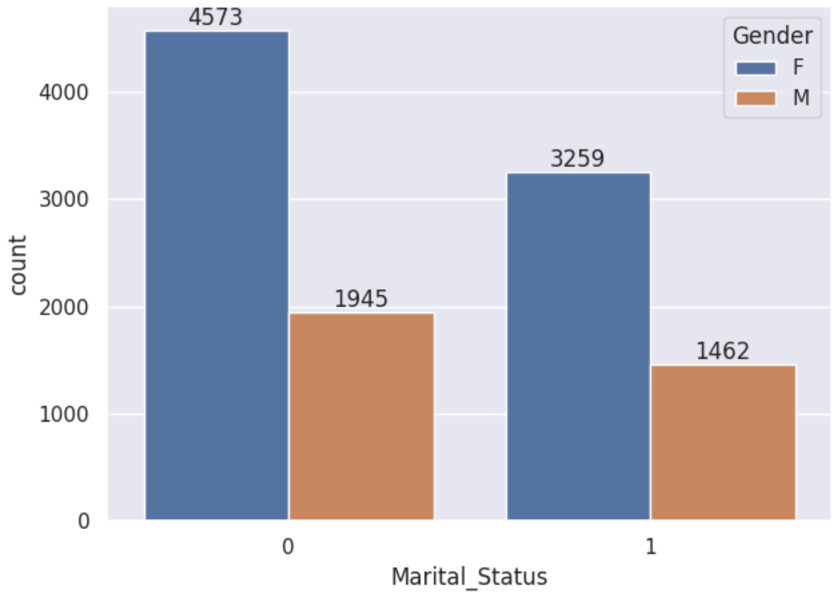
```
<Axes: xlabel='State', ylabel='Amount'>
```



from above graph we can see that unexpectedly most of the orders are from uttar pradesh maharashtra and karnataka respectively but total sales/amount from up karnataka and then maharashtra

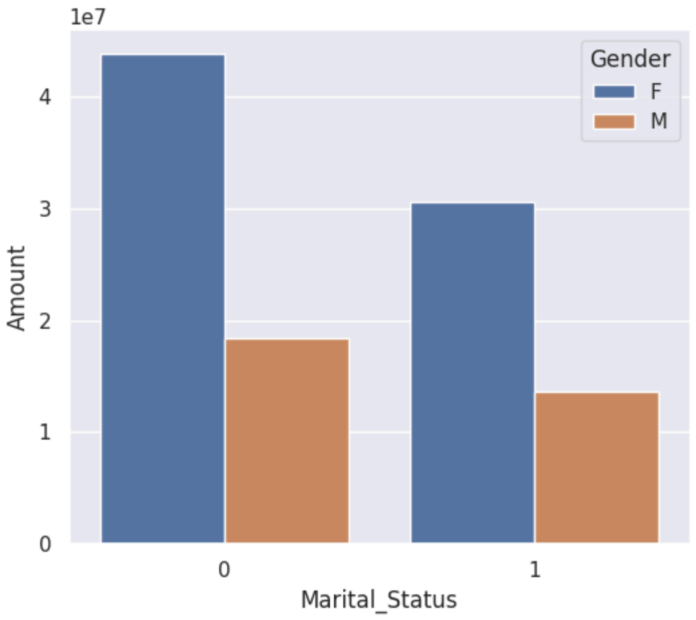
Marital Status

```
ax = sns.countplot(data=df, x='Marital_Status',hue = 'Gender')
sns.set(rc={'figure.figsize':(7,5)})
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=Fal
sns.set(rc={'figure.figsize':(6,5)})
sns.barplot(data=sales_state, x='Marital_Status', y='Amount', hue='Gender')
```

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

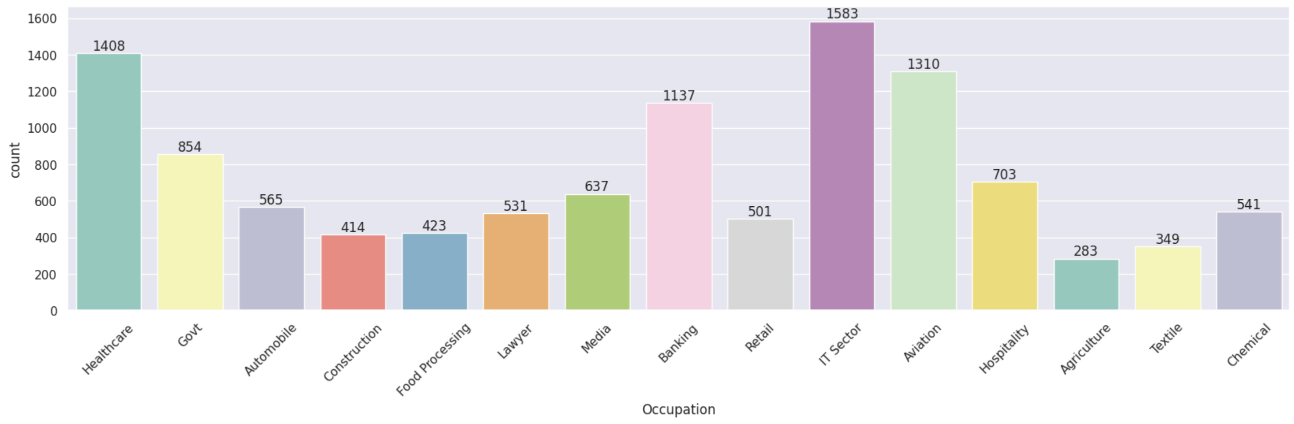


from above graphs we can see that most of the buyers are married (women) and they have purchasing power

occupation

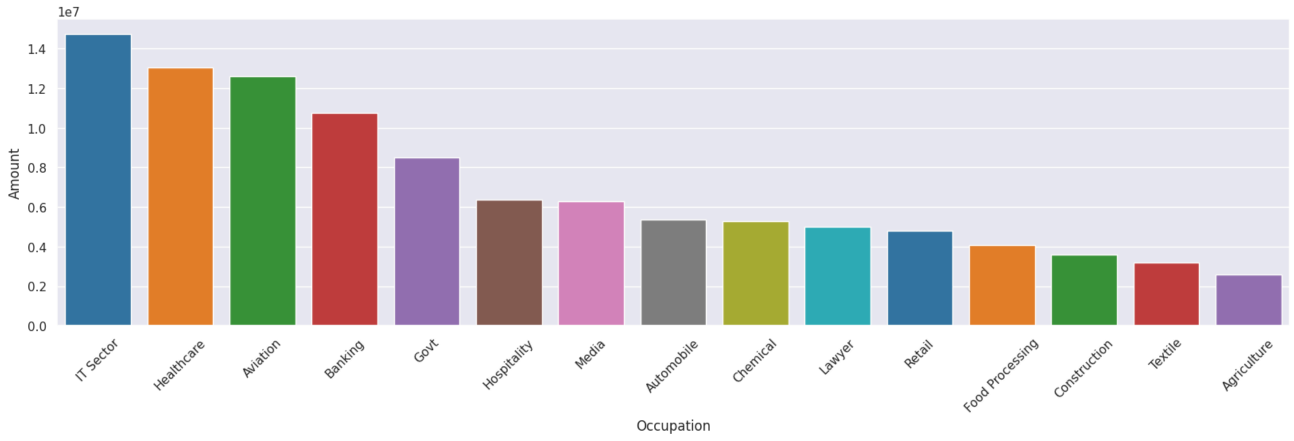
```
# Set figure size
sns.set(rc={'figure.figsize':(20,5)})

# Countplot with colorful bars
ax = sns.countplot(data=df, x='Occupation', palette="Set3")

# Add labels on top of bars
for bars in ax.containers:
    ax.bar_label(bars)

plt.xticks(rotation=45)
plt.show()
```

```
/tmp/ipython-input-3654508151.py:5: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and se

  ax = sns.countplot(data=df, x='Occupation', palette="Set3")
```



```
# Grouping occupation-wise total sales amount
sales_state = df.groupby(['Occupation'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)
sns.set(rc={'figure.figsize':(20,5)})

# Colorful barplot
sns.barplot(
    data=sales_state,
    x='Occupation',
    y='Amount',
    palette='tab10'   #palette change karne se alag-alag colors aayenge
)

plt.xticks(rotation=45)  # x-labels readable banane ke liye
plt.show()
```

```
/tmp/ipython-input-1092369730.py:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and se

  sns.barplot(
```

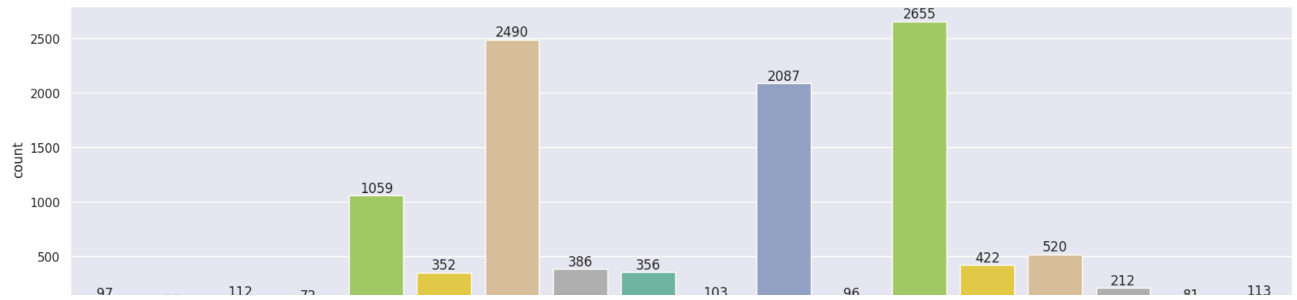


from above graphs we can see that most of the buyers are working in it ,Aviation and healthcare sector

Product category

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

ax = sns.countplot(data=df, x='Product_Category', palette="Set2")

# Add count labels
for bars in ax.containers:
    ax.bar_label(bars)

plt.xticks(rotation=45)
plt.show()
```

```
/tmp/ipython-input-2892381369.py:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and se

  ax = sns.countplot(data=df, x='Product_Category', palette="Set2")
```


count