Data Engineer INTERN at HACKVEDA LIMITED

AUTHOR : BANDANA PRAKASH
TASK 1 : Insights-Driven_sales
PURPOSE : to analyze sales transactions and derive actionable insights that can enhance sales strategies.
Specifically, the dataset contains information about customers, their demographics, purchase behaviors, and transaction details.

Here are the key objectives:

Customer Analysis: Understand customer demographics, including age, gender, marital status, and location.
Sales Performance: Analyze sales data to identify trends in orders and revenue generation.
Insight Generation: Generate insights that can inform marketing strategies and improve customer targeting.
Data Visualization: Use visual tools to present findings clearly, making it easier to interpret data trends.
Overall, the goal is to leverage this data for informed decision-making that drives sales growth and enhances customer engagement.

Steps Involved:

Import Libraries: Load essential Python libraries like numpy, pandas, matplotlib, and seaborn for data manipulation and visualization.
Load Dataset: Import the sales dataset (Insights-Driven_sales-main.csv) into a Pandas DataFrame.
Check Dataset Dimensions: Verify the shape of the dataset to understand its size (rows and columns).
Explore Data: Inspect the dataset structure, including column names and sample records, to understand its contents.
Statistical Summary: Analyze statistical metrics (count, mean, min, max, etc.) for numerical columns like Age, Orders, and Amount.
Data Cleaning: Handle missing values or anomalies in the dataset (if required).
Data Visualization: Use tools like Matplotlib and Seaborn to create graphs and charts for better insights into sales trends and customer behavior.
Generate Insights: Derive actionable insights such as top-performing products, customer demographics, and purchasing patterns.

```
# import python libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt # visualizing data
%matplotlib inline
import seaborn as sns


# import csv file
df = pd.read_csv('Insights-Driven_sales-main.csv', encoding= 'unicode_escape')


df.shape
```

```
(11251, 15)
```

```
df.head()
```

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category
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

```
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
```

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

```
#check for null values
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             12
dtype: int64
```

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

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

```
df['Amount'].dtypes
```

```
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	Ord
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	
11250	1002744	Brumley	P00281742	F	18-25	19	0	Maharashtra	Western	Healthcare	Office	

```
# describe() method returns 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

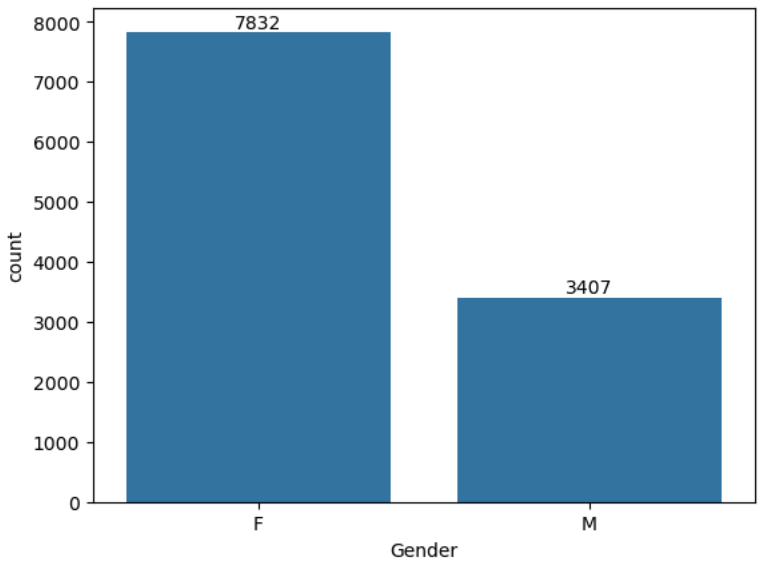
```
# use describe() for specific columns
df[['Age', 'Orders', 'Amount']].describe()
```

	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

Exploratory Data Analysis

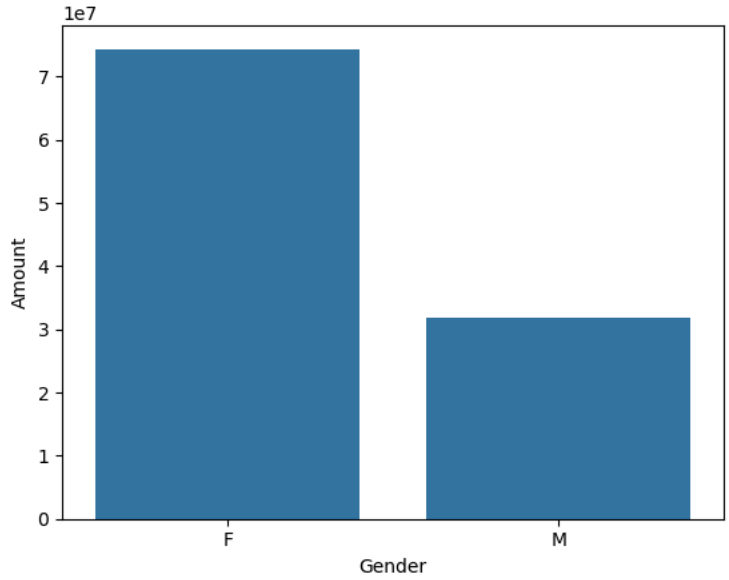
Gender

```
# plotting a bar chart for Gender and it's count

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

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




```
# plotting a bar chart for gender vs total amount

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)
```

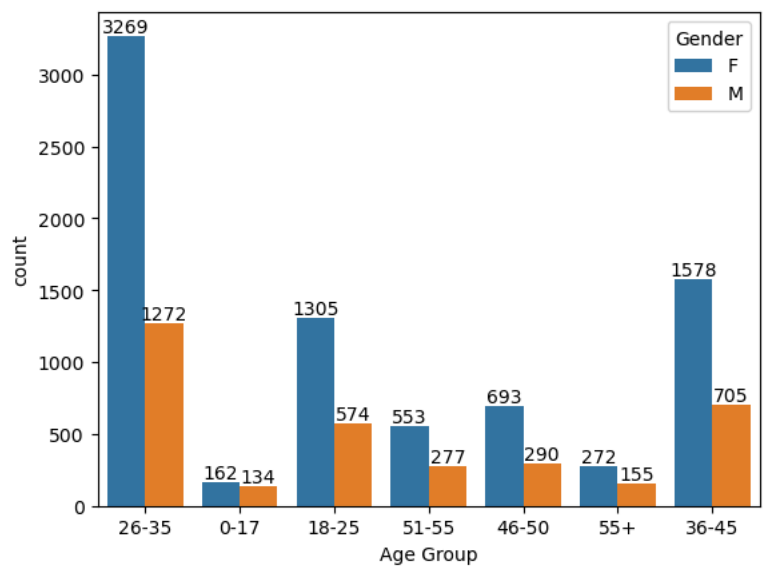
```
<Axes: xlabel='Gender', ylabel='Amount'>
```



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

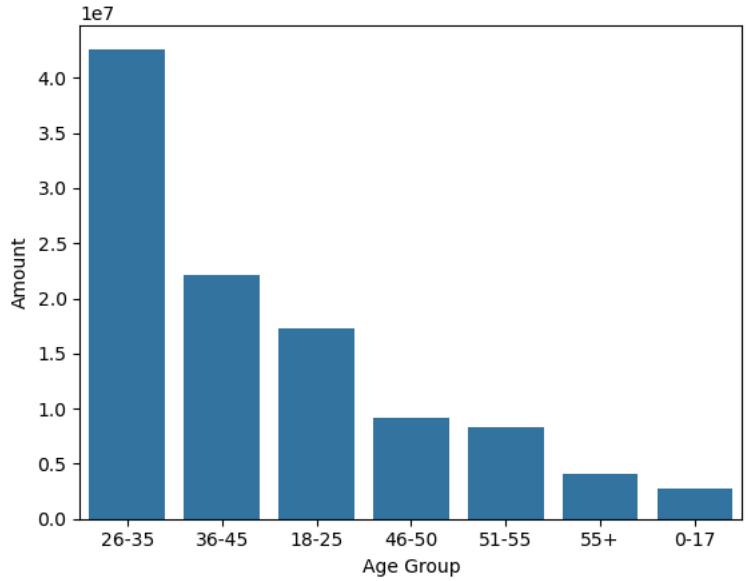
Age

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

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




```
# 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)
```

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

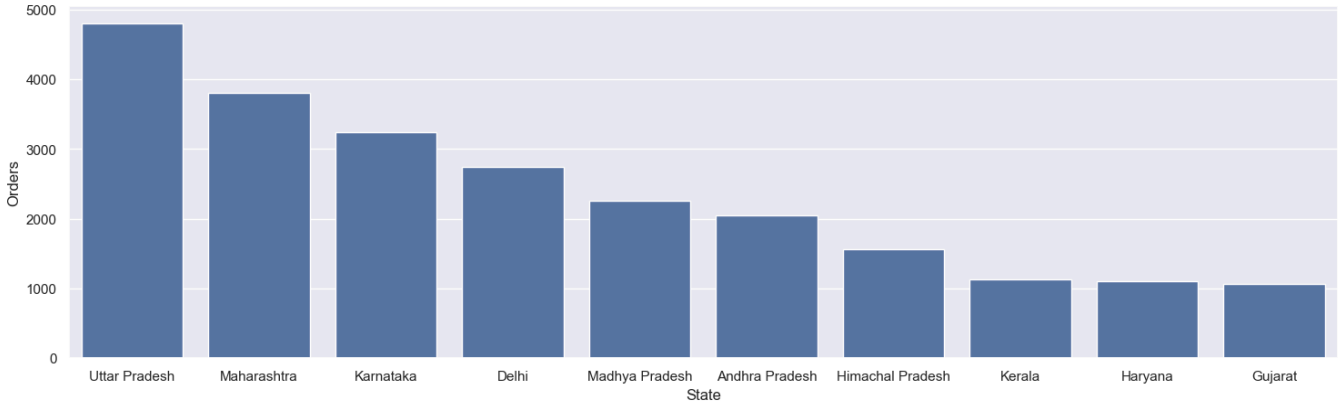


From above graphs we can see that most of the buyers are of age group between 26-35 yrs female

State

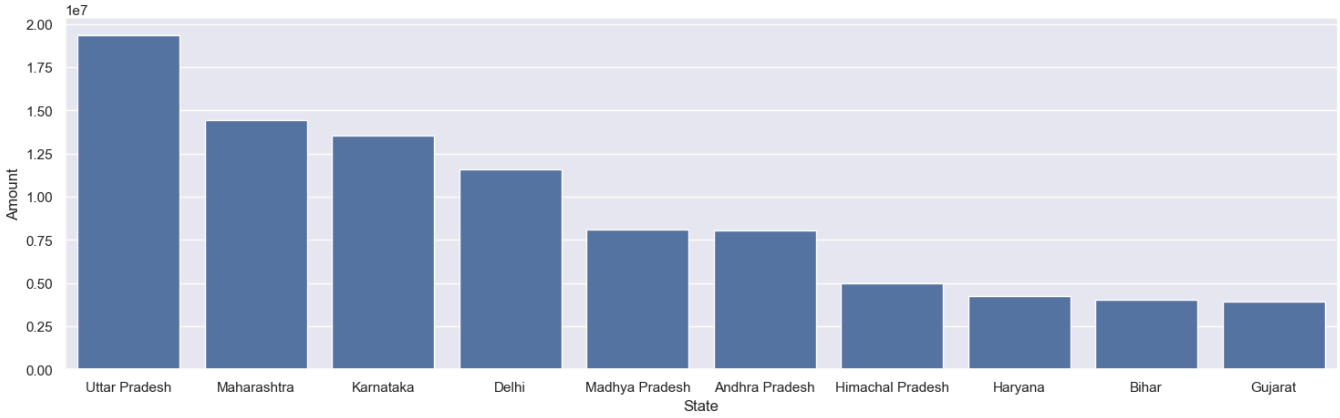
```
# total number of orders from top 10 states

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

sns.set(rc={'figure.figsize':(18,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Orders')
```

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




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

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

sns.set(rc={'figure.figsize':(18,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Amount')
```

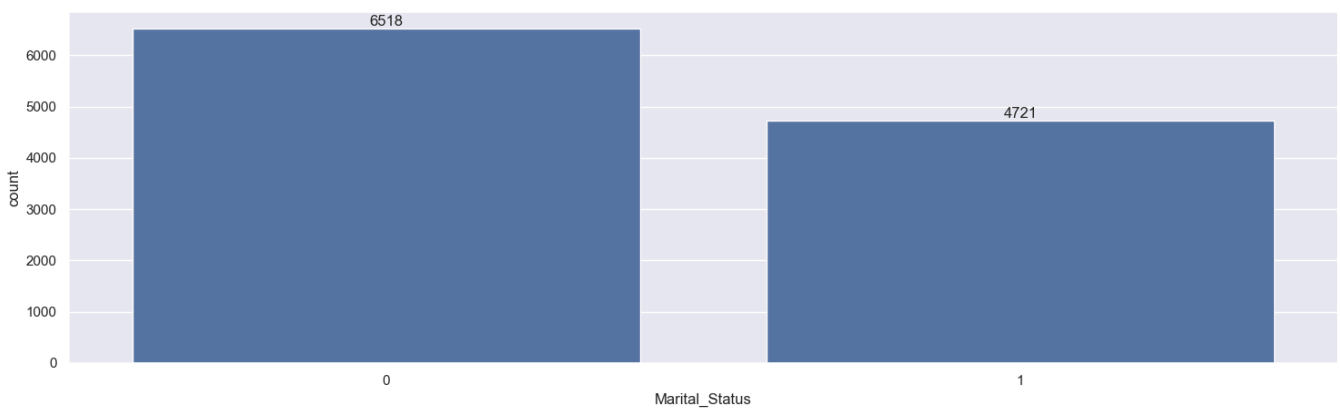
```
<Axes: xlabel='State', ylabel='Amount'>
```



From above graphs we can see that most of the orders & total sales/amount are from Uttar Pradesh, Maharashtra and Karnataka respectively

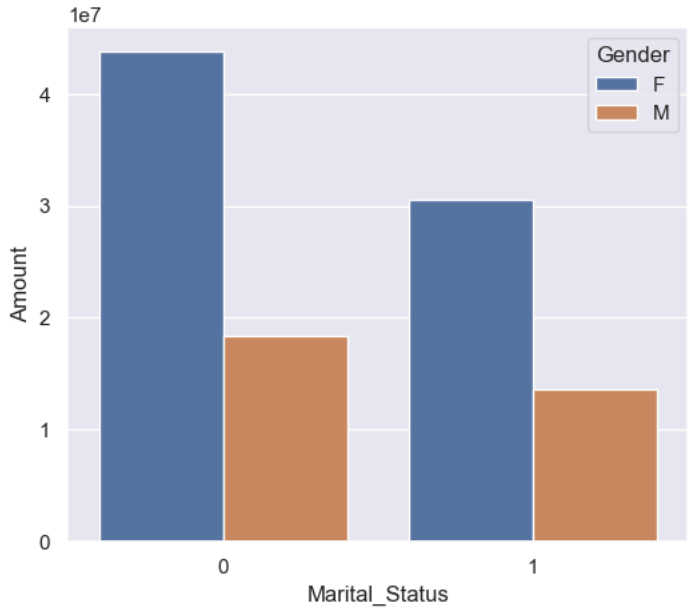
Marital Status

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

sns.set(rc={'figure.figsize':(7,3)})
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=Fa

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

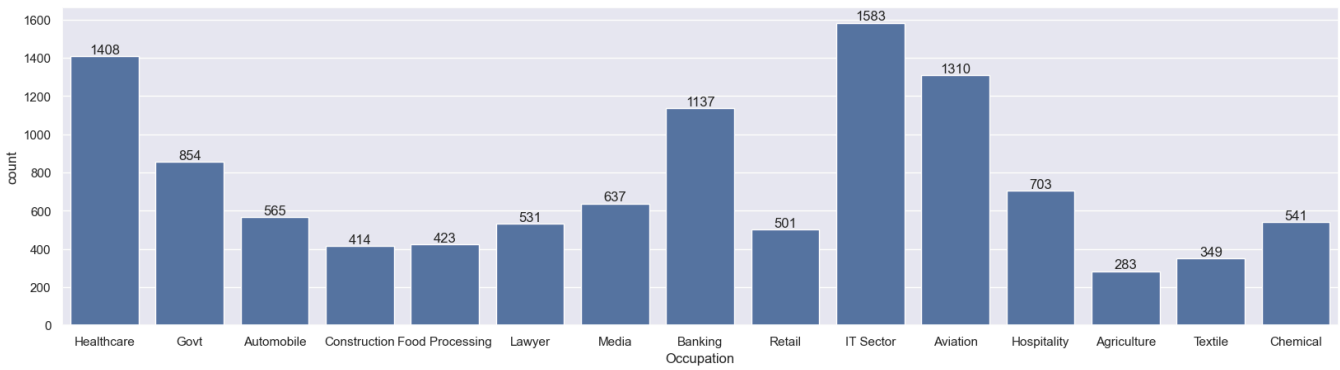
```
<Axes: xlabel='Marital_Status', ylabel='Amount'>
```



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

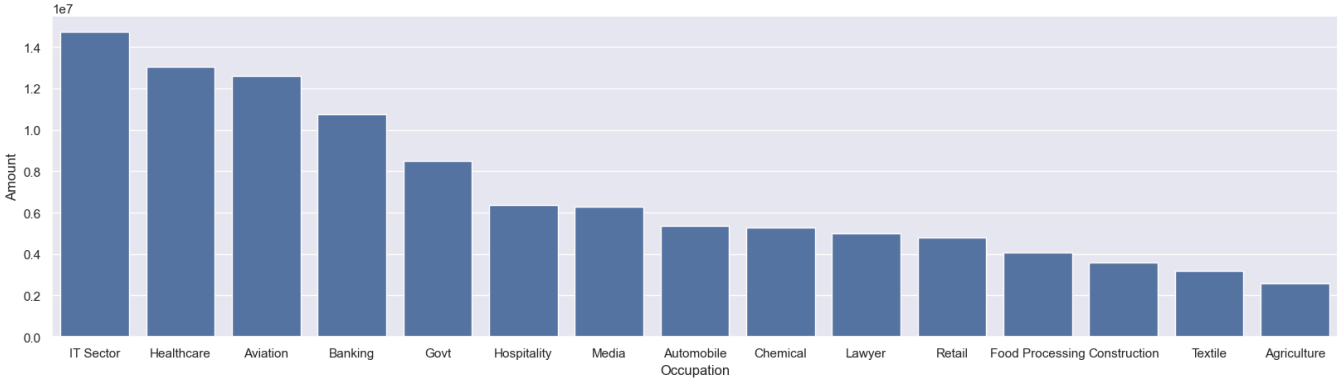
Occupation

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

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




```
sales_state = 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_state, x = 'Occupation',y= 'Amount')
```

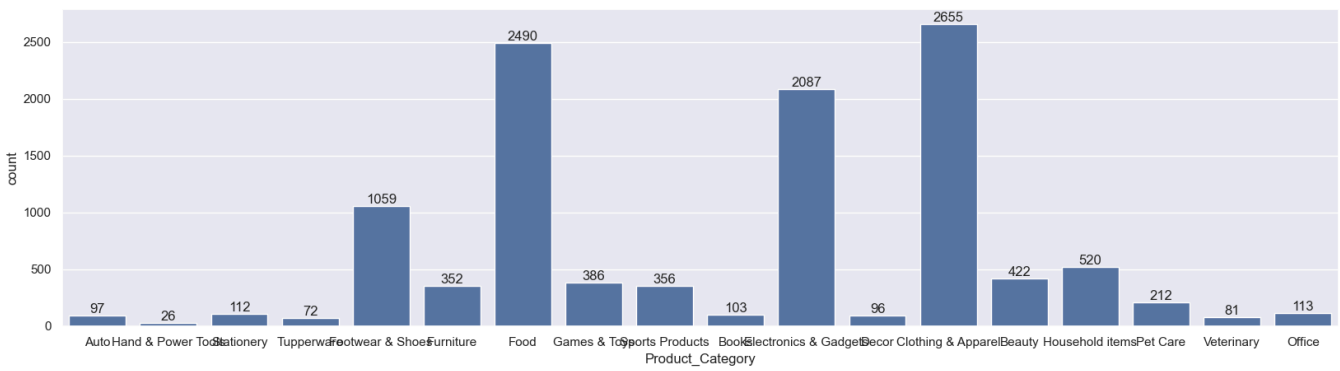
```
<Axes: xlabel='Occupation', ylabel='Amount'>
```



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

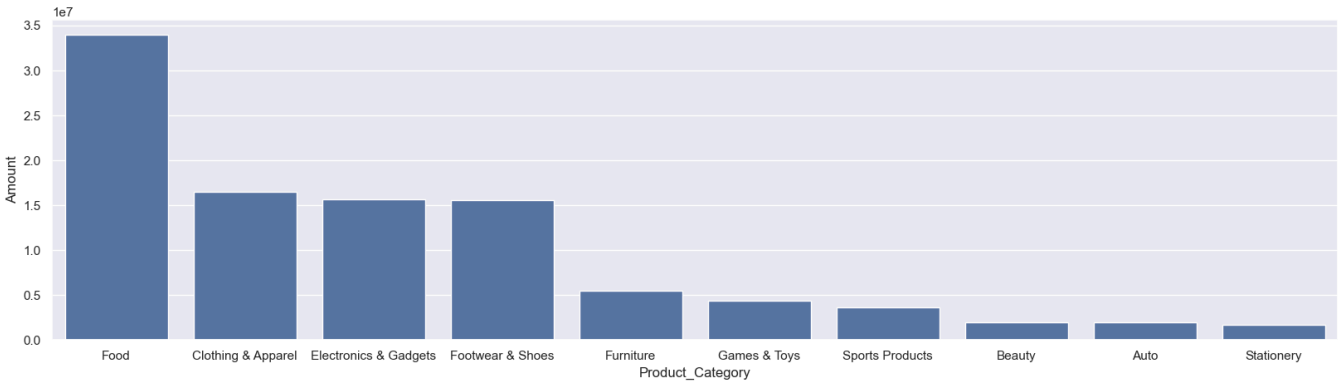
Product Category

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

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




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

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data = sales_state, x = 'Product_Category',y= 'Amount')
```

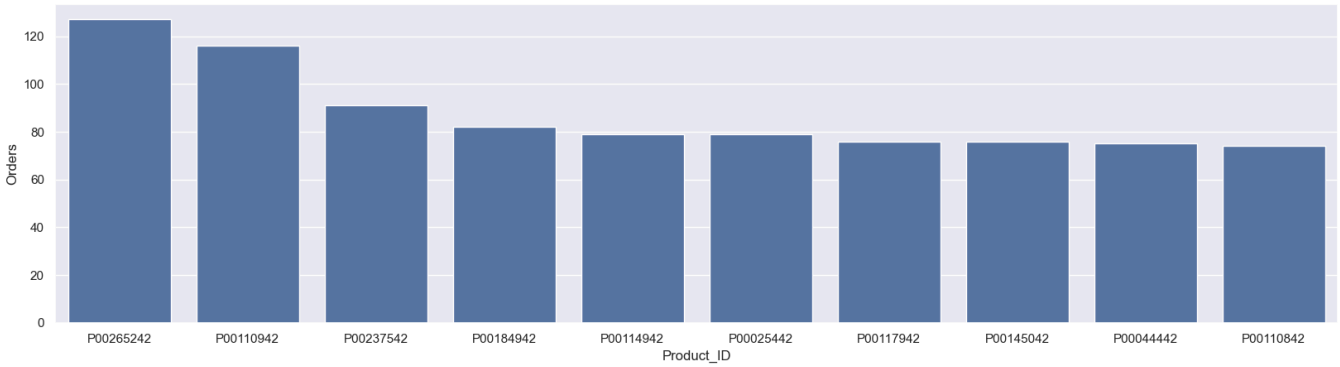
```
<Axes: xlabel='Product_Category', ylabel='Amount'>
```



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

```
sales_state = 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_state, x = 'Product_ID',y= 'Orders')
```

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




```
# top 10 most sold products (same thing as above)

fig1, ax1 = plt.subplots(figsize=(12,7))
df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=False).plot(kind='bar')
```

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