Diwali Sales Analysis

(Python Project)

1. Introduction

This project focuses on analyzing sales data using Python to understand customer behavior and purchasing patterns. The objective of the project was to clean and analyze raw sales data, extract meaningful insights, and support business decision-making related to customer targeting, sales improvement, and inventory planning.

2. Tools and Technologies Used

Programming Language: Python

Libraries:

- Pandas – data cleaning and manipulation
- Matplotlib & Seaborn – data visualization

Environment: Jupyter Notebook

3. Data Cleaning and Preparation

The dataset was first cleaned to ensure accuracy and consistency. This included handling missing values, correcting data types, removing unnecessary columns, and standardizing categorical variables. Pandas was used extensively to manipulate and prepare the data for analysis.

Figure 1: Python code for data cleaning and preprocessing.

```
[3]: df = pd.read_csv("C:/Users/AHK/Desktop/Python Projects/Diwali sales/Dataset/Diwali Sales Data.csv", encoding= 'unicode_escape')
     print(df.shape)

     (11251, 15)

[4]: df.head()
```

[4]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnamed
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.0	NaN	Na
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.0	NaN	Na
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.0	NaN	Na
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2	23912.0	NaN	Na
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2	23877.0	NaN	Na

```
[5]: df.columns.tolist

[5]: <bound method IndexOpsMixin.tolist of Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
            'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
            'Orders', 'Amount', 'Status', 'unnamed1'],
           dtype='object')>
```

[8]:
```
#droping the blank columns/unrelated columns
df.drop(['Status','unnamed1'], axis=1, inplace = True)
```

[9]:
```
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
```

[11]:
```
#to check the null values
pd.isnull(df).sum()
```

[11]:
```
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
```

[12]:
```
df.shape
```

[12]: (11251, 13)

[13]:
```
#to delete the null values
df.dropna(inplace=True)
```

[17]:
```
pd.isnull(df).sum()
```

[17]:
```
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
```

[14]:
```
df.shape
```

[14]: (11239, 13)

[15]:
```
df['Amount'] = df['Amount'].astype('int')
```

[16]:
```
df['Amount'].dtypes
```

[16]: dtype('int64')

[22]:
```
df.rename(columns={'Marital_Status':'Shaadi'})
#it wont be saved because not used inplace= True. now it created new table/dataframe, wont effect the orginal one
```

[22]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Shaadi	State	Zone	Occupation	Product_Category	Orders	Amount
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924

4. Exploratory Data Analysis (EDA)

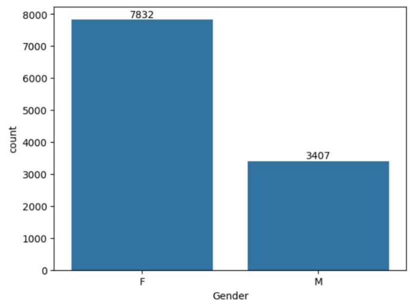
Exploratory Data Analysis was performed to understand data distribution and identify trends. Visualizations were created using Matplotlib and Seaborn to analyze sales performance across different dimensions such as:

- Gender
- Age groups
- States
- Occupation
- Product categories

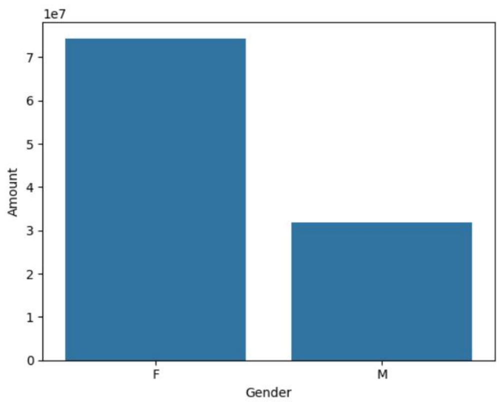
Gender-wise Sales Analysis

This analysis highlights differences in purchasing behavior between male and female customers.

```
[37]: ax=sns.countplot(data = df,x='Gender')

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



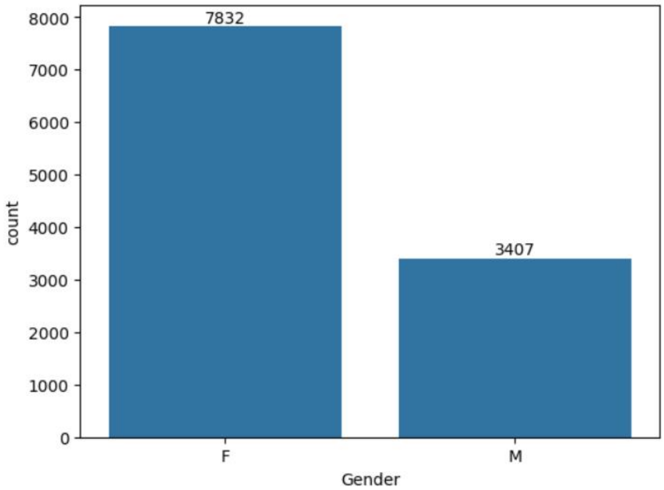

```
[52]: sales_gen = df.groupby(['Gender'],as_index=False).agg({'Amount':'sum'}).sort_values(by='Amount', ascending=False)

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

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



Age Group Analysis

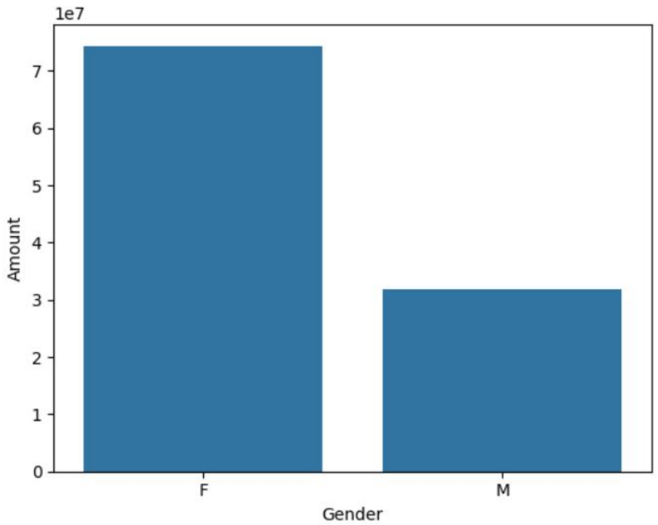
Sales performance was analyzed across different age groups to understand which segment contributes the most to revenue.

[37]:
```
ax=sns.countplot(data = df,x='Gender')

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

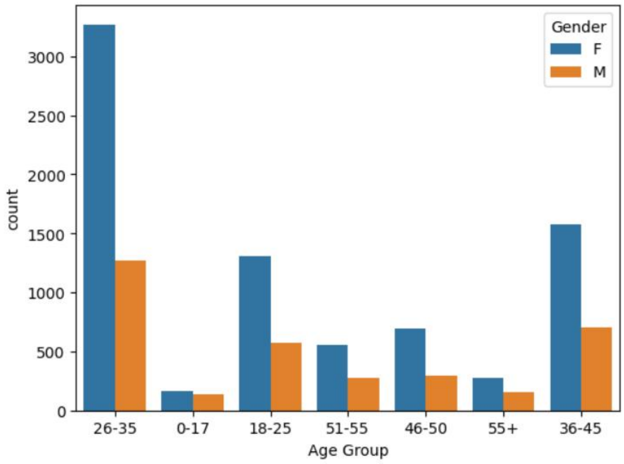


[52]:
```
sales_gen = df.groupby(['Gender'],as_index=False).agg({'Amount':'sum'}).sort_values(by='Amount', ascending=False)

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

[52]: <Axes: xlabel='Gender', ylabel='Amount'>

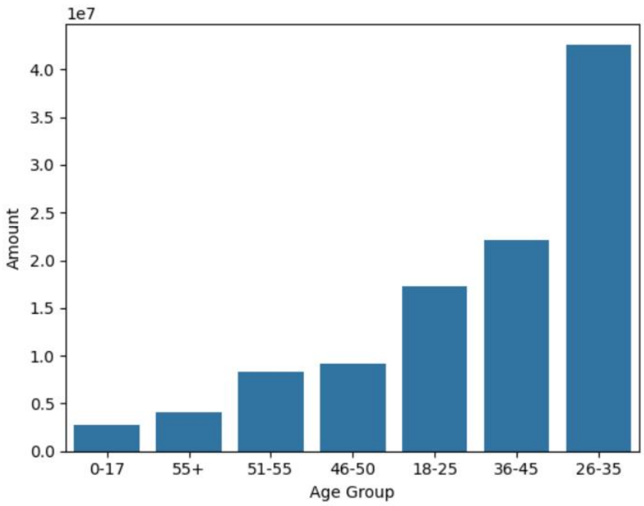


[61]:
```
sns.countplot(data=df,x='Age Group',hue='Gender')
plt.show()
```



[73]:
```
sales_age= df.groupby(['Age Group'],as_index=False).agg({'Amount':'sum'}).sort_values(by='Amount')

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

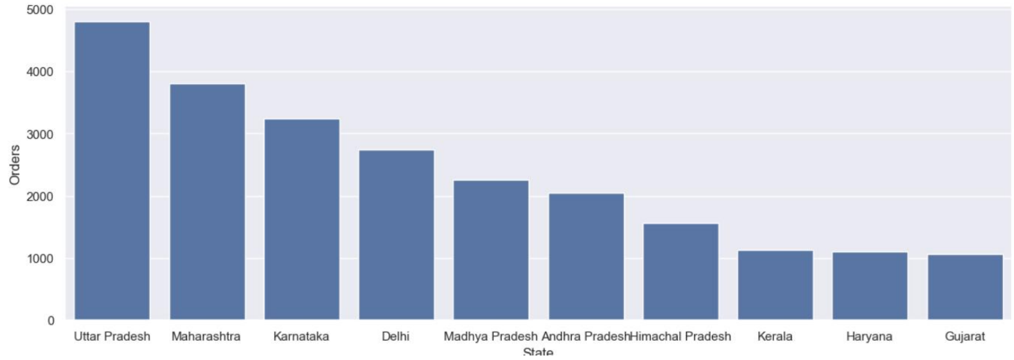
[73]: <Axes: xlabel='Age Group', ylabel='Amount'>



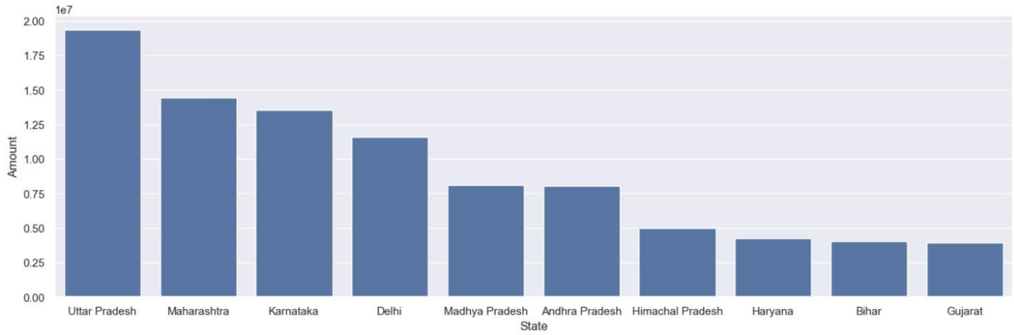
State-wise and Occupation-wise Analysis

State-wise and occupation-wise analysis helped identify potential customer regions and professions contributing significantly to sales.

```
[89]: #total no.of orders from top 10 states

      Orders_state= df.groupby('State').agg({'Orders':'sum'}).sort_values(by='Orders',ascending=False).head(10)
      sns.set(rc={'figure.figsize':(15,5)})
      sns.barplot(data = Orders_state, x='State',y='Orders')

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




```
Total sales from to 10 states

[95]: Amount_states= df.groupby('State').agg({'Amount':'sum'}).sort_values(by='Amount',ascending=False).head(10)
      sns.set(rc={'figure.figsize':(17,5)})
      sns.barplot(data=Amount_states, x='State', y='Amount')

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



From above graphs we can see that most of the orders and total sales/amount are from uttarpradesh,maharastra and karnataka respectively

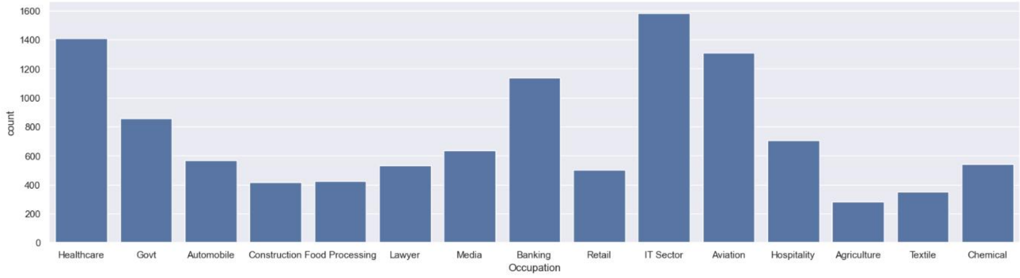
Occupation

[115]:
```
df.columns
```

[115]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

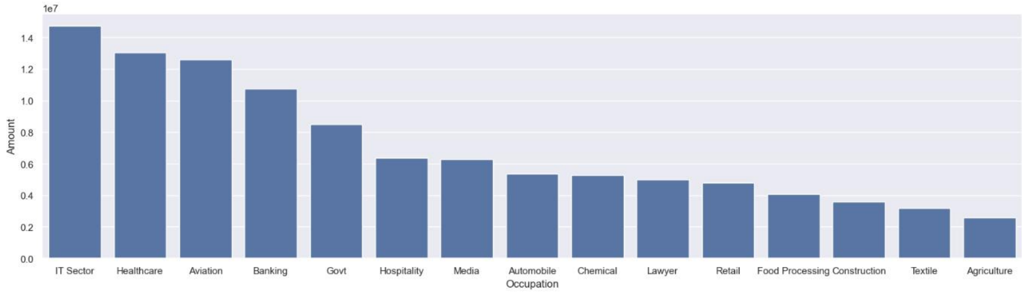
[119]:
```
sns.set(rc={'figure.figsize':(20,5)})
sns.countplot(data=df,x='Occupation')
```

[119]: <Axes: xlabel='Occupation', ylabel='count'>



[122]:
```
#which occupation people purchased more

occ_sales=df.groupby('Occupation').agg({'Amount':'sum'}).sort_values(by='Amount',ascending=False)

sns.barplot(data=occ_sales,x='Occupation',y='Amount')
```

[122]: <Axes: xlabel='Occupation', ylabel='Amount'>

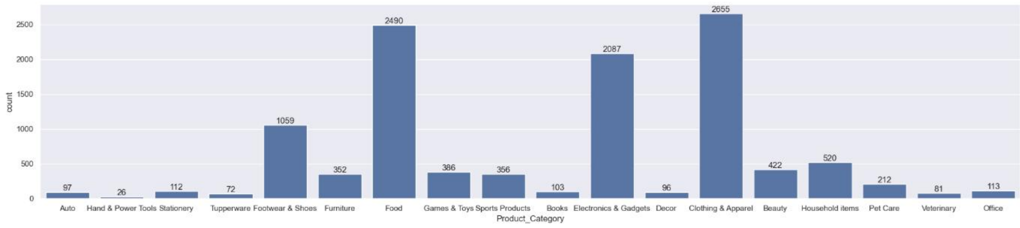


From the above graph, we can see that most of the buyers from IT sector, Healthcare and Aviation sector.

Product Category Analysis

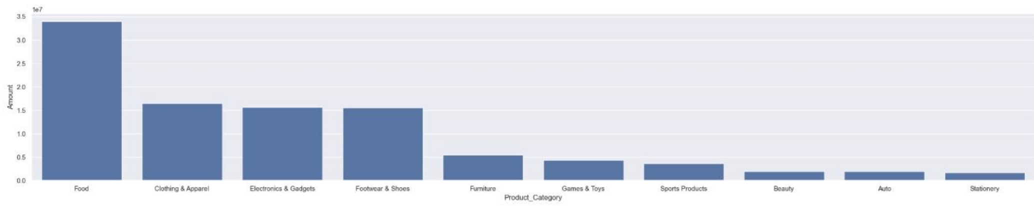
This analysis identified the most selling product categories and products, which can help in inventory planning and demand forecasting.

```
[129]: sns.set(rc={'figure.figsize':(25,5)})
ax=sns.countplot(data=df,x='Product_Category')

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




```
[130]: #which product category got more sales/amount

procat_sales= df.groupby('Product_Category').agg({'Amount':'sum'}).sort_values(by='Amount',ascending=False).head(10)
sns.set(rc={'figure.figsize':(30,5)})
sns.barplot(data=procat_sales,x='Product_Category',y='Amount')
```

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



From above graph, we can see that most of the sold products are from Food,clothing& apparel and Electronics&Gadgets

EDA helped in identifying patterns, outliers, and relationships between customer demographics and purchasing behavior.

5. Key Insights and Findings

- Identified potential customers across different **states, occupations, genders, and age groups**, helping improve customer targeting strategies.
- Analyzed **most selling product categories and products**, which can support better inventory planning.
- Observed purchasing trends that can help businesses align supply with customer demand.
- Insights from the analysis can be used to enhance **customer experience and sales performance**.

6. Business Impact

The insights derived from this project can help businesses:

- Improve customer experience through targeted marketing
- Increase sales by focusing on high-performing products
- Optimize inventory planning to meet demand efficiently
- Make data-driven decisions using customer behavior analysis

7. Conclusion

This project provided hands-on experience in data cleaning, exploratory data analysis, and data visualization using Python. It demonstrated how raw data can be transformed into actionable insights to support business growth. Overall, the project strengthened practical skills in Python, data analysis, and analytical thinking.