# Python Project

## Diwali Sales Analysis

Minal Bhivgade

# Project Objective:

The Diwali Sale Analysis project aims to uncover key trends and actionable insights from customer purchase data during the Diwali season. By analyzing critical factors such as

- Customer Demographics
- Purchasing Patterns
- Product Preferences

This project provides valuable information to help businesses optimize their sales strategies, enhance customer targeting, and boost overall performance during the high-demand Diwali season.

Minal Bhivgade

# Steps of Data Analysis:

- Importing Libraries
- Loading Data
- Data Cleaning
- Exploratory Data Analysis (EDA)
- Finding Insights
- Recommendations

Minal Bhivgade

# Diwali Sale Analysis Project

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

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

df.shape

(11251, 15)

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 |

```
d.describe()
```

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

```
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.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  \
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]
```

```
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
```

```
df.dropna(inplace = True)
```

```
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
```

```
df.shape
```

```
(11239, 13)
```

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

df['Amount'].dtypes # to check data type use dtypes
```

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

```
df.columns # give list of all columns
```

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

```
# rename any column name
df.rename(columns={'Marital_Status':'Shadi'}) # to rename use dic key
as old name and value as new name which you want to give
```

```
       User_ID    Cust_name Product_ID Gender Age Group  Age  Shadi  \
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]
```

```
df.describe() # description of thr data in the Dataframme.
```

```
            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
```

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

```
                Age        Amount
count  11239.000000  11239.000000
mean      35.410357   9453.610553
std       12.753866   5222.355168
min       12.000000    188.000000
25%       27.000000   5443.000000
50%       33.000000   8109.000000
75%       43.000000  12675.000000
max       92.000000  23952.000000
```

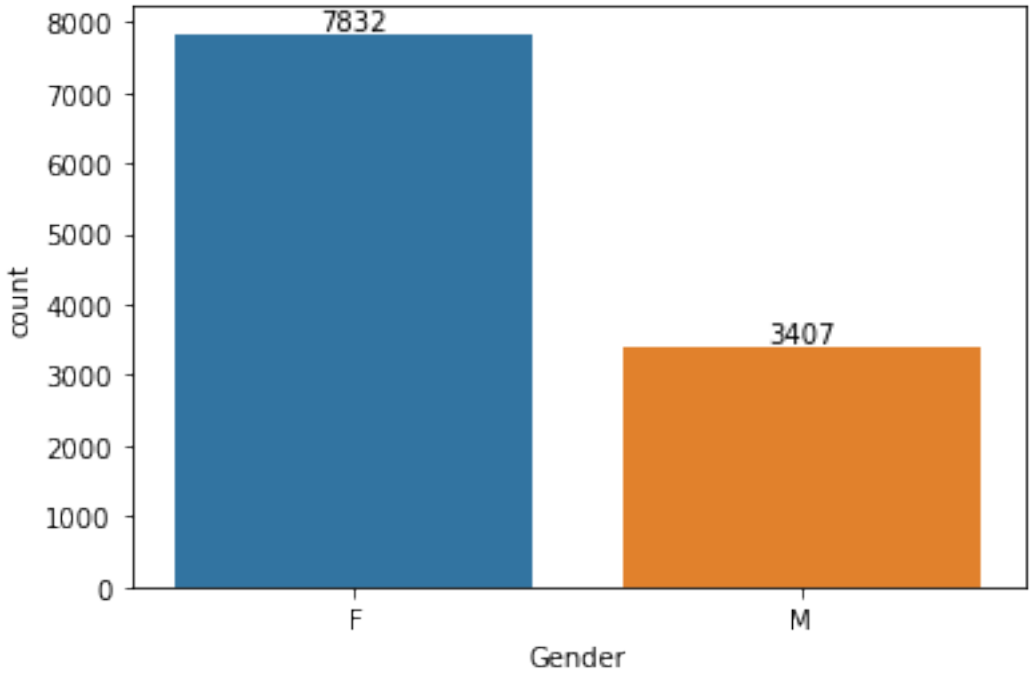
# Exploratory Data Analysis

```
df.columns
```

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

## Gender Distribution Analysis

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

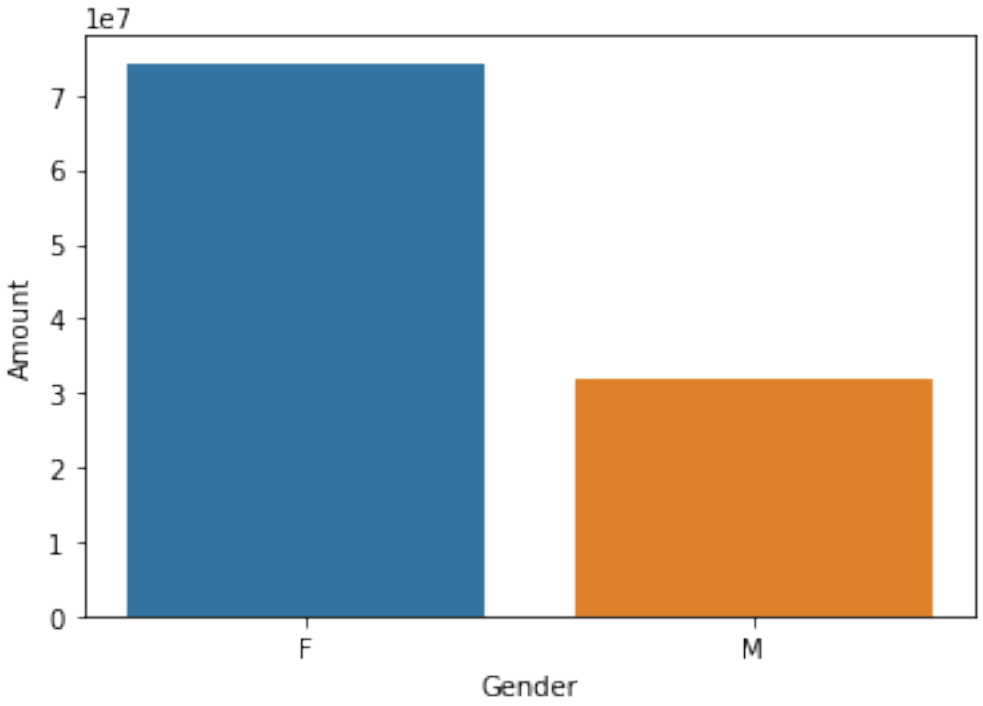



**Insight**:
The dataset reveals that **females (7832)** significantly outnumber **males (3407)**, making women the dominant customer demographic.

# Gender-wise Sales Analysis

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

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



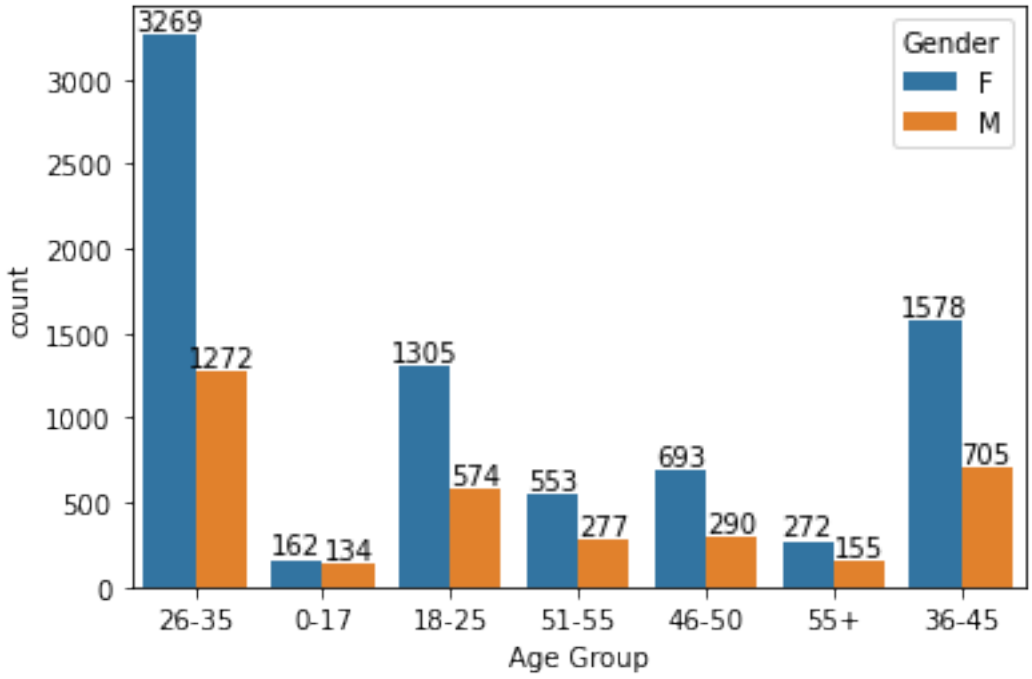

**Insights**:
The analysis shows that **females contribute the highest sales** in terms of total spending compared to males. This indicates that **women are the primary drivers of revenue** during the sales period.

```
df.columns
```

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

# Age Group-wise Sales Analysis by Gender

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

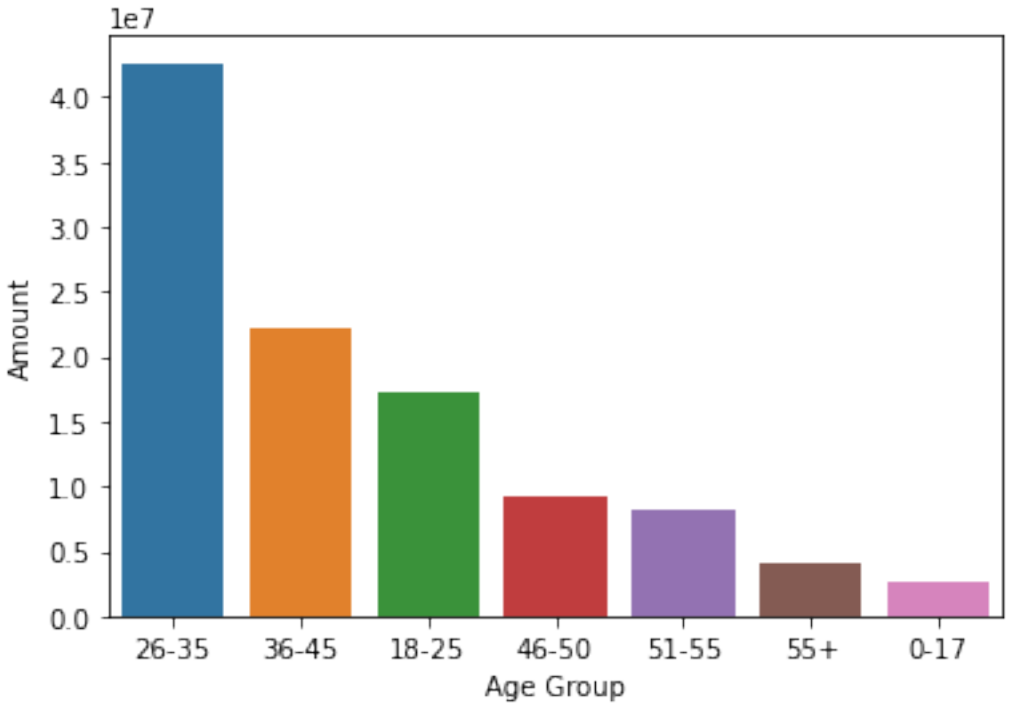



**Insights**:
The analysis reveals that in the **26-35 age group**, **females** contribute the highest sales compared to males, making this demographic the top spender.

# Age Group-wise Total Sales Analysis

```
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)
```

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




**Insights**:
The analysis shows that the **26-35 age group** contributes the highest total sales, followed by the **36-45 age group**, and then the **18-25 age group**. Sales from the **0-17 age group** are minimal.

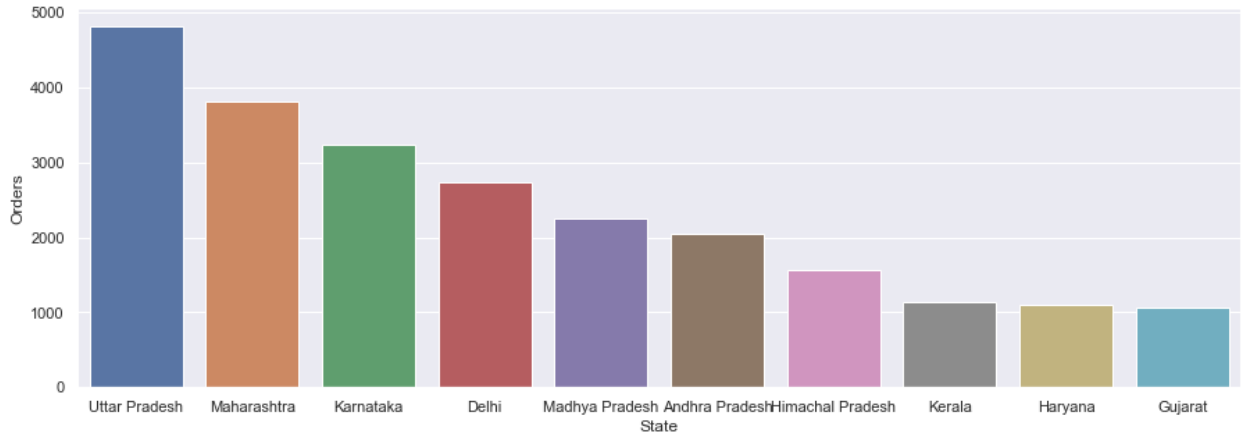
```
df.columns
```

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

# Top States by Total Orders

```
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(data =sales_state,x='State',y= 'Orders')
```

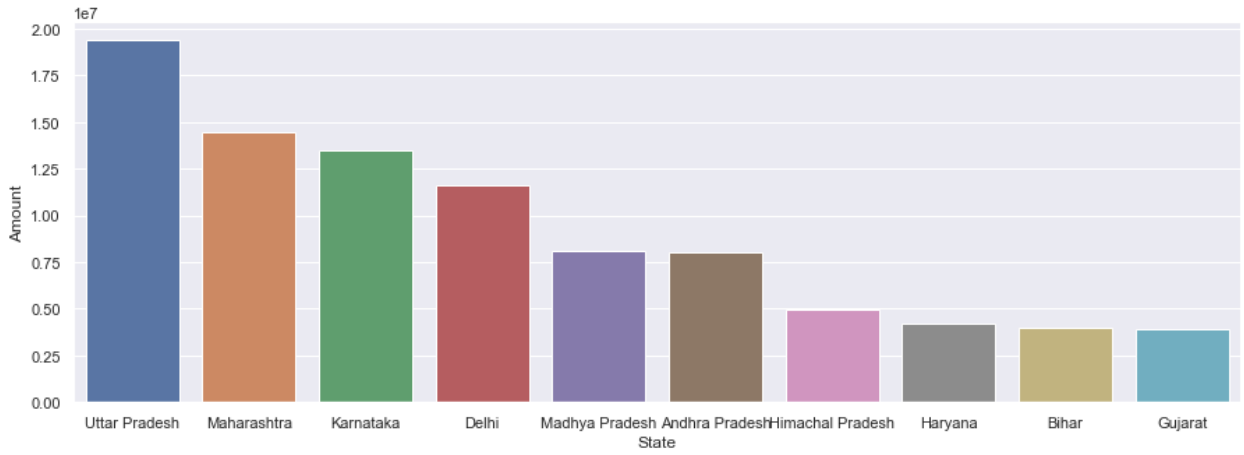
```
<AxesSubplot:xlabel='State', ylabel='Orders'>
```




**Insights**:
The analysis shows that **Uttar Pradesh** contributes the highest number of orders, followed by **Maharashtra**. These two states are the top performers in terms of order volume.

# Top States by Total Sales Amount

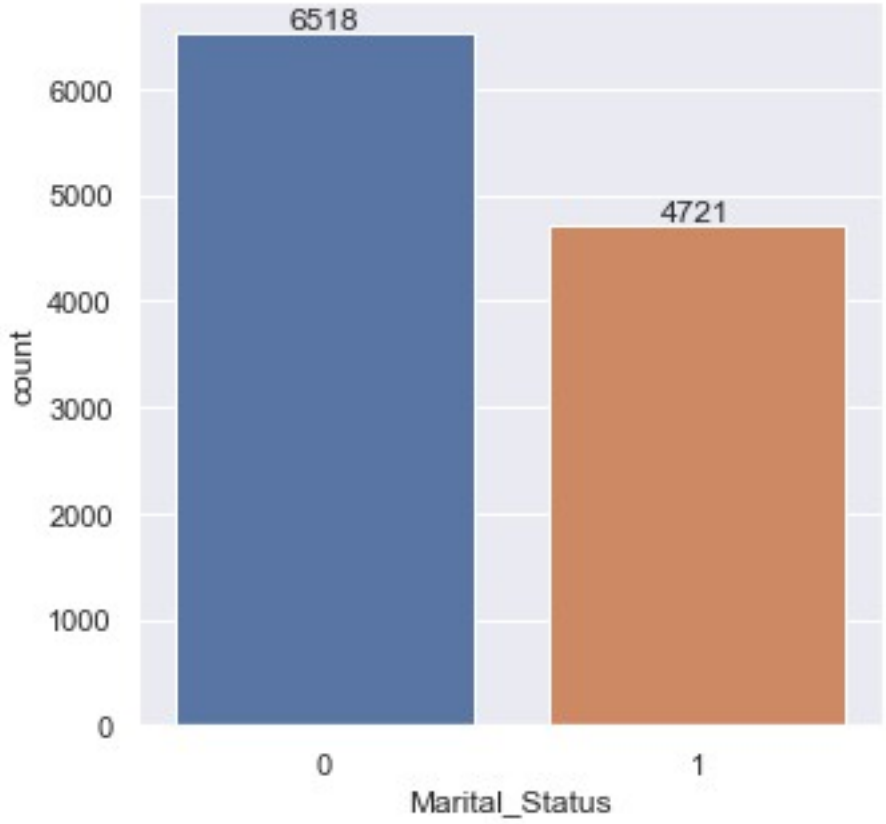
```
sales_state = df.groupby(['State'],as_index=False)
['Amount'].sum().sort_values(by ='Amount',ascending =False).head(10)

sns.set(rc={'figure.figsize': (15, 5)})

sns.barplot(data =sales_state,x='State',y= 'Amount')

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




**Insights**:
The analysis shows that **Uttar Pradesh** contributes the highest sales amount, followed by **Maharashtra** and **Karnataka**. These three states are the top performers in terms of total sales.

## Marital Status Distribution Analysis

```
mx = sns.countplot(x = 'Marital_Status',data = df)
sns.set(rc={'figure.figsize': (7, 5)})
for bars in mx.containers:
    mx.bar_label(bars)
```

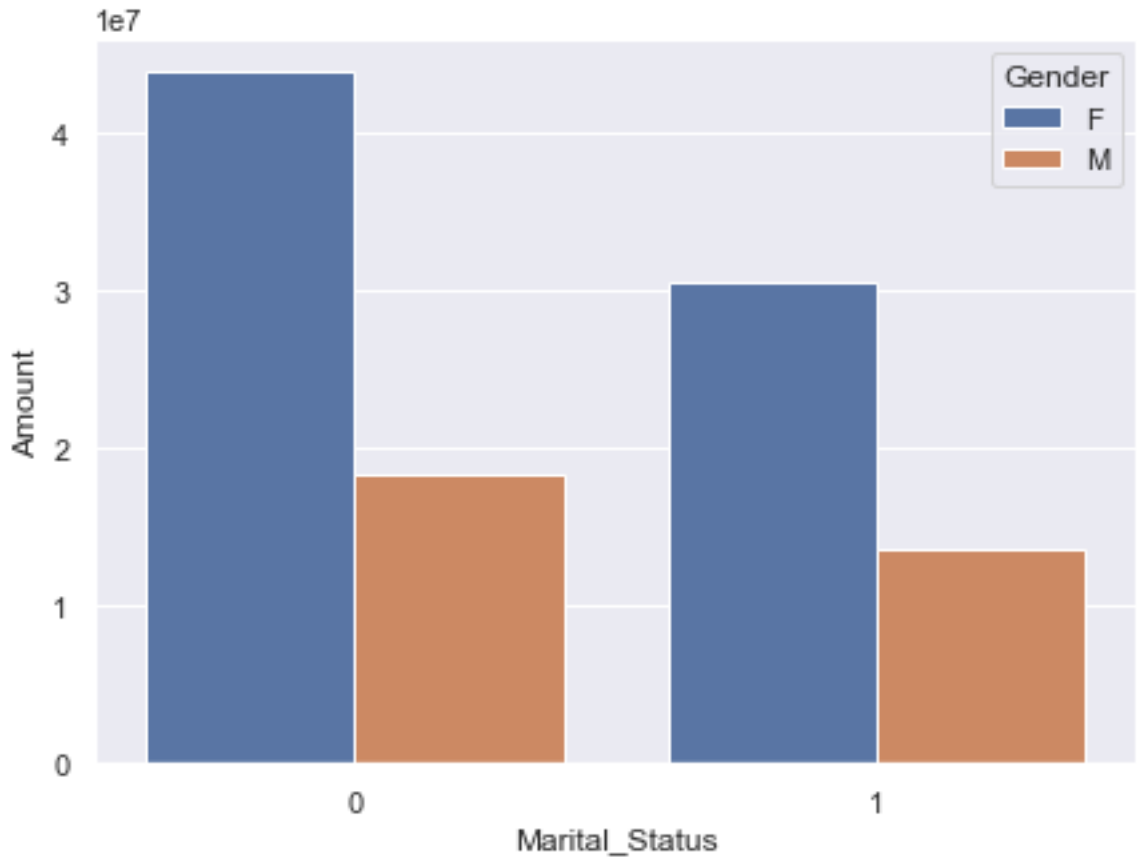



**Insights**:
The analysis shows that the majority of customers fall under the **"Married"** category, with very few customers in the **"Single"** category.

## Sales Analysis by Marital Status and Gender

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

sns.set(rc={'figure.figsize': (7, 5)})

sns.barplot(data =sales_state,x='Marital_Status',y= 'Amount',hue
='Gender')
```

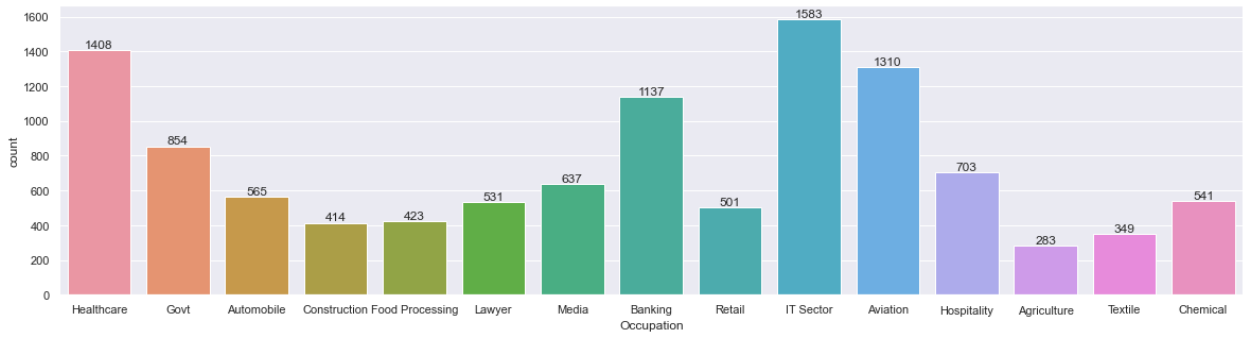
```
<AxesSubplot:xlabel='Marital_Status', ylabel='Amount'>
```




**Insights**:
From the analysis, we can see that most of the buyers are **married women**, contributing the highest sales compared to other gender-marital status combinations.

# Occupation Distribution Analysis

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

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



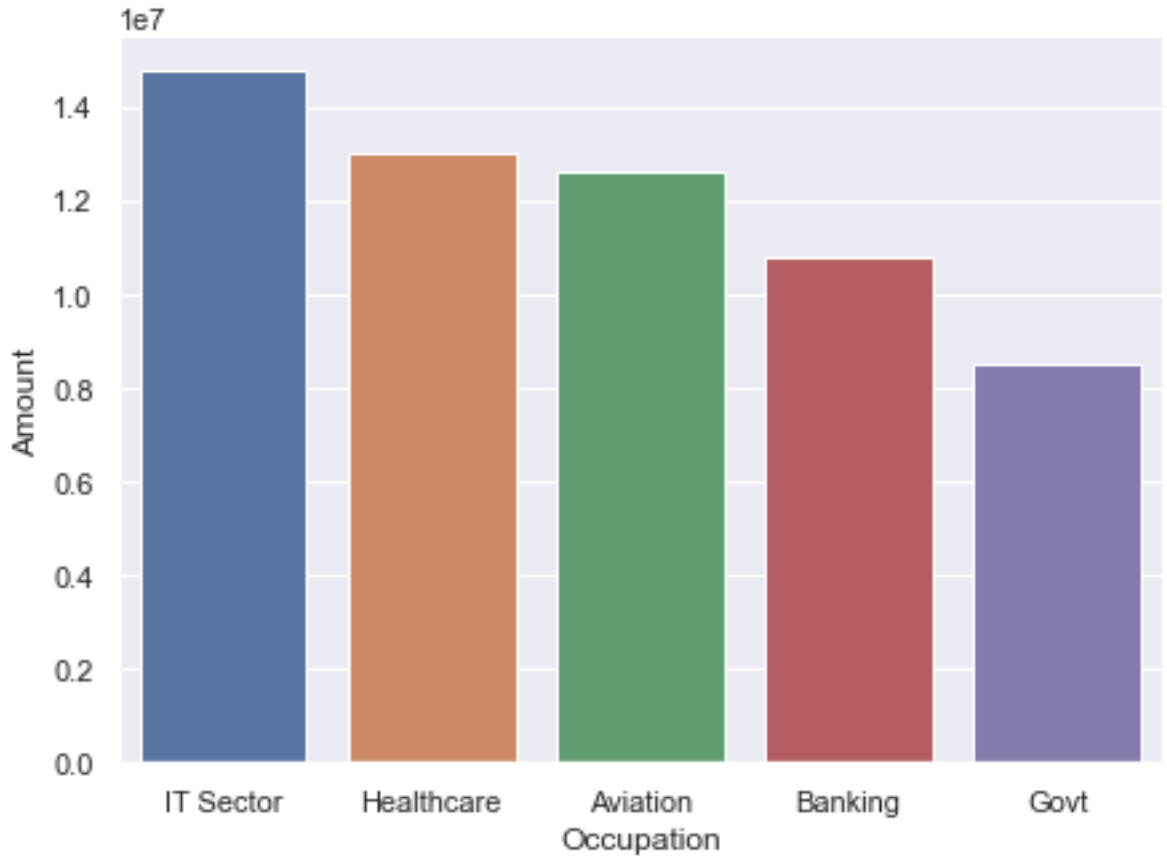
**Insights**:
The analysis reveals that the top occupations contributing to sales are:

- **IT**: 1583 individuals
- **Healthcare**: 1,418 individuals
- **Animation**: 1,310 individuals
- **Banking**: 1,137 individuals

# Sales Analysis by Occupation

```
sales_state = df.groupby(['Occupation'],as_index=False)
['Amount'].sum().sort_values(by ='Amount',ascending =False).head(5)

sns.set(rc={'figure.figsize': (7, 5)})

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

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

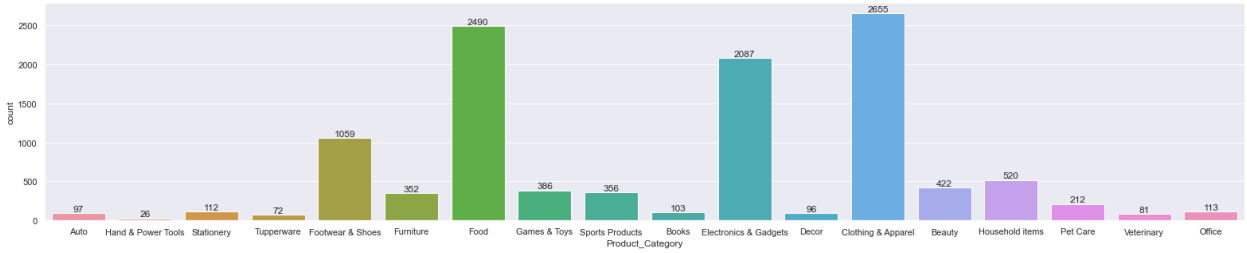



**Insights**:

- **Healthcare**: Leading the total sales amount, followed by other key occupations like **IT**, **Animation**, and **Banking**.

# Sales Analysis by Product Category

```
# Product Category
sns.set(rc={'figure.figsize': (2, 5)})
mx = sns.countplot(x = 'Product_Category',data = df)

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




## Insights:

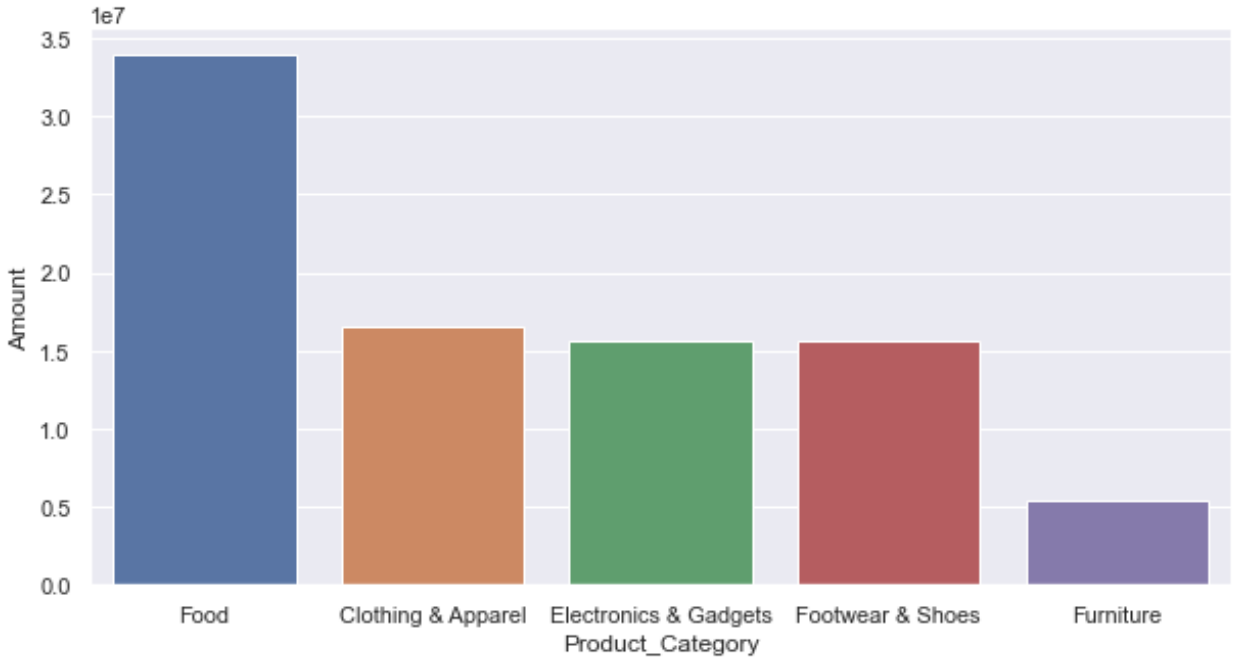
The analysis shows the following distribution of sales across product categories:

- **Clothing & Apparel**: 2,655 sales, the highest among all categories.
- **Food**: 2,490 sales.
- **Electronics & Gadgets**: 2,087 sales.

# Sales Analysis by Product Category (Amount)

```
sales_state = df.groupby(['Product_Category'],as_index=False)
['Amount'].sum().sort_values(by ='Amount',ascending =False).head(5)

sns.set(rc={'figure.figsize': (10, 5)})

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

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




## Insights:

The analysis shows that **Food** is the top-performing product category in terms of total sales amount, followed by other key categories such as **Clothing & Apparel** and **Electronics & Gadgets**.

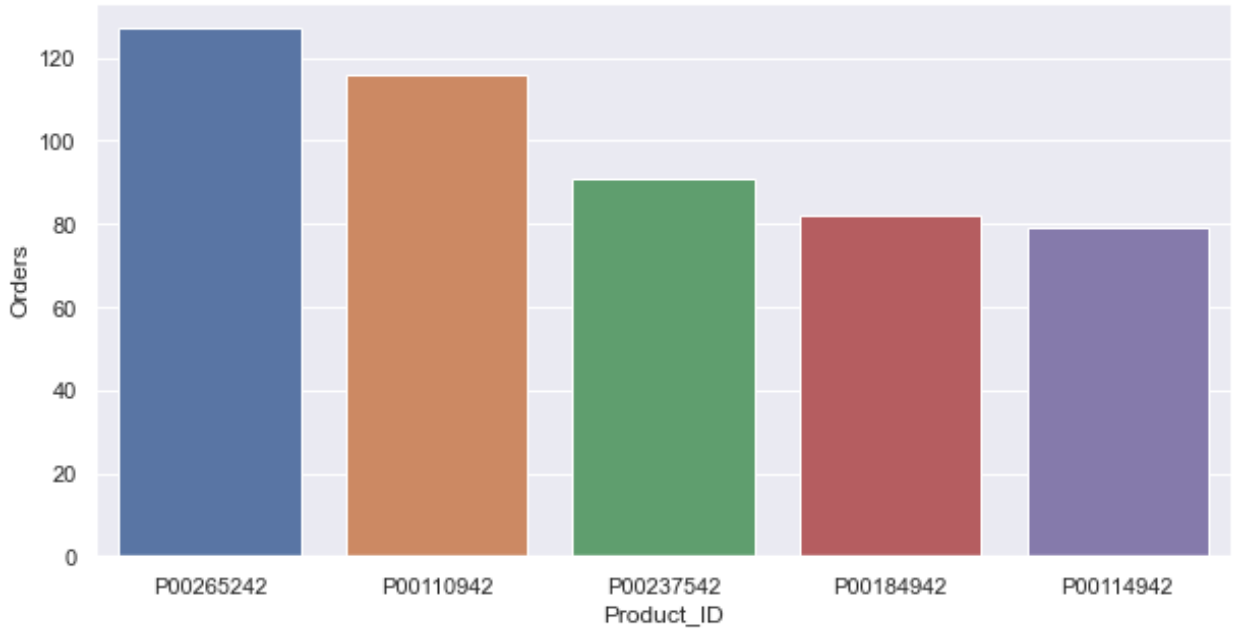
This indicates that food-related products drive the highest revenue during the Diwali sale period.

from the above graph we can see that most of sold product from food,Clothing and Electronics Gadgets

# Top 5 Products by Order Volume

```
sales_state = df.groupby(['Product_ID'],as_index=False)
['Orders'].sum().sort_values(by ='Orders',ascending =False).head(5)

sns.set(rc={'figure.figsize': (10, 5)})

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

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


120
100
80
60
40
20
0
Orders
P00265242
P00110942
P00237542
P00184942
P00114942
Product_ID

**Thank You**

# Recommendations for Optimizing Sales Strategy

- Target High-Performing Demographics: Prioritize marketing efforts towards women, especially those in the 26-35 age group, and married women, as they contribute significantly to total sales.Personalized discounts and loyalty programs can help retain this key audience.
- Strengthen Presence in Top States: Focus on Maharashtra, Uttar Pradesh, and Karnataka, which are the highest-performing states for both sales and order volume. Implement region-specific promotions to maximize sales in these areas
- Enhance Product Offering: Ensure high availability of top-selling products in Food, Clothing & Apparel, and Electronics & Gadgets. Heavily promote these categories through discounts and bundling to drive sales during peak periods.
- Leverage Industry-Specific Campaigns: Healthcare professionals, IT workers, and employees in Animation and Banking are significant buyers. Tailor marketing campaigns and discounts specifically for these sectors to boost sales.
- Utilize Data-Driven Strategies: Continuously monitor trends and adjust marketing strategies based on real-time sales data. Update inventory and product offerings to ensure alignment with market demands and consumer preferences.

Minal Bhivgade

# Thank You