This project aims to analyse sales data during the Diwali festival season to uncover valuable insights that can drive business growth. The dataset includes attributes such as orders, productid, product category, occupation, marital status, and states. By leveraging advanced data analysis techniques and interactive visualizations, the project provides a comprehensive view of sales performance, customer behaviour, and market trends during Diwali.

# Diwali Sales Data Analysis

Python Analysis

Utkarsh Sharma
sharma.utkarsh2402@gmail.com

## Project Overview

This project is aims to analyse the sales of products with the different product categories done in different region and states by the people with different gender, occupation, marital status, etc.

## Project Objectives

The primary goal of this project is to analyse sales data during the Diwali festival season to uncover trends, patterns, and insights that can help businesses optimize their marketing strategies, improve customer satisfaction, and drive sales growth.

1. Product: Name or ID of the product sold.
2. Product Category: Category to which the product belongs (e.g., Electronics, Clothing, Food, etc.).
3. Gender: Gender of the customer (Male, Female, Other).
4. Occupation: Customer's occupation (e.g., IT, Healthcare, Education, etc.).
5. Marital Status: Marital status of the customer (Single, Married, Divorced, etc.).
6. States: The state from which the customer made the purchase (e.g., Maharashtra, Uttar Pradesh, Karnataka, etc.).
7. Region: Geographical region of the customer (e.g., North, South, East, West).
8. Age: Age of the customer.
9. Age Group: Age group of the customer (e.g., 18-25, 26-35, 36-45, etc.).

## Problem Statement

Sales Optimization
Customer Segmentation
Regional Analysis
Gender-Based Analysis
Age Group Analysis
These are the few key points which we are going to analyze in this project and with the help of visuals we need to find the all the actionable insights.

## Cleaning of the dataset

All the steps related to the cleaning of this data are as follows-
We need to do the data cleaning first; else it will affect our data analysis.
We need to remove all the null values, duplicate, and change the data types of the attributes. (if required)

1. **First, you need to import all the libraries that are being used in this project.**

```
[1]
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import statistics as st
import warnings
warnings.filterwarnings("ignore")
✓ 8.5s
```

2. **Then, you need to read the dataset by using the panda's library.**

```
[2]
df = pd.read_csv("Diwali Sales Data.csv",encoding='unicode_escape')
✓ 0.1s
```

3. **After this you need to see the head of the data.**

```
[7]
df.head(5)
```

| | User_ID | Cust_name | Product_ID | Gender | Age Group | Age | Marital_Status | State | Zone | Occupation |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1002903 | Sanskriti | P00125942 | F | 26-35 | 28 | 0 | Maharashtra | Western | Healthcare |
| 1 | 1000732 | Kartik | P00110942 | F | 26-35 | 35 | 1 | Andhra Pradesh | Southern | Govt |
| 2 | 1001990 | Bindu | P00118542 | F | 26-35 | 35 | 1 | Uttar Pradesh | Central | Automobile |
| 3 | 1001425 | Sudevi | P00237842 | M | 0-17 | 16 | 0 | Karnataka | Southern | Construction |
| 4 | 1000588 | Joni | P00057942 | M | 26-35 | 28 | 1 | Gujarat | Western | Food Processing |

4. **Then you need to remove the duplicate values from your dataset. And use in-place keyword to make the permanent changes in the dataset.**

```
[3]  df.drop_duplicates(inplace=True)
     ✓ 0.1s
```

| | User_ID | Cust_name | Product_ID | Gender | Age Group | Age | Marital_Status | State | Zone | Occupation |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1002903 | Sanskriti | P00125942 | F | 26-35 | 28 | 0 | Maharashtra | Western | Healthcare |
| 1 | 1000732 | Kartik | P00110942 | F | 26-35 | 35 | 1 | Andhra Pradesh | Southern | Govt |
| 2 | 1001990 | Bindu | P00118542 | F | 26-35 | 35 | 1 | Uttar Pradesh | Central | Automobile |
| 3 | 1001425 | Sudevi | P00237842 | M | 0-17 | 16 | 0 | Karnataka | Southern | Construction |
| 4 | 1000588 | Joni | P00057942 | M | 26-35 | 28 | 1 | Gujarat | Western | Food Processing |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 11246 | 1000695 | Manning | P00296942 | M | 18-25 | 19 | 1 | Maharashtra | Western | Chemical |
| 11247 | 1004089 | Reichenbach | P00171342 | M | 26-35 | 33 | 0 | Haryana | Northern | Healthcare |
| 11248 | 1001209 | Oshin | P00201342 | F | 36-45 | 40 | 0 | Madhya Pradesh | Central | Textile |
| 11249 | 1004023 | Noonan | P00059442 | M | 36-45 | 37 | 0 | Karnataka | Southern | Agriculture |
| 11250 | 1002744 | Brumley | P00281742 | F | 18-25 | 19 | 0 | Maharashtra | Western | Healthcare |

11243 rows × 15 columns

5. **After this, you need to see the info of the dataset. In which you find all the datatype, null values present in the dataset.**

```
[4]  df.info()
     ✓ 0.0s
```

```
<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
```

6. **Then, we need to drop those columns which is not required for the data analysis. As we are going to remove these two columns because these columns don't contain any values means these are totally null values.**

```
[5]  df.drop(['Status','unnamed1'],axis=1,inplace = True)
     ✓ 0.0s
```

7. **Then, we need to check weather our data contains any null values or not.**

```
[6]  df.isna().sum()
     ✓ 0.0s

...  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
```

8. **We found some null values in the "Amount" attribute now we need to fill or remove those null values.**

```
[7]  df.dropna(inplace=True)
     ✓ 0.0s
```

**This will remove all the null values present in the dataset.**

9. **Once you remove all the null values then you need to change the datatype of the column. (If required). In this case we need to change the datatype of the column "Amount" from 'float' to 'int'.**

```
[8]  df['Amount']= df['Amount'].astype('int')
     ✓ 0.0s
```

10. **Now at last we need to use this 5 pointer theory to see all the statistical data of this dataset.**

```
[9]  df.describe()
     ✓ 0.0s
```

| | 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 |

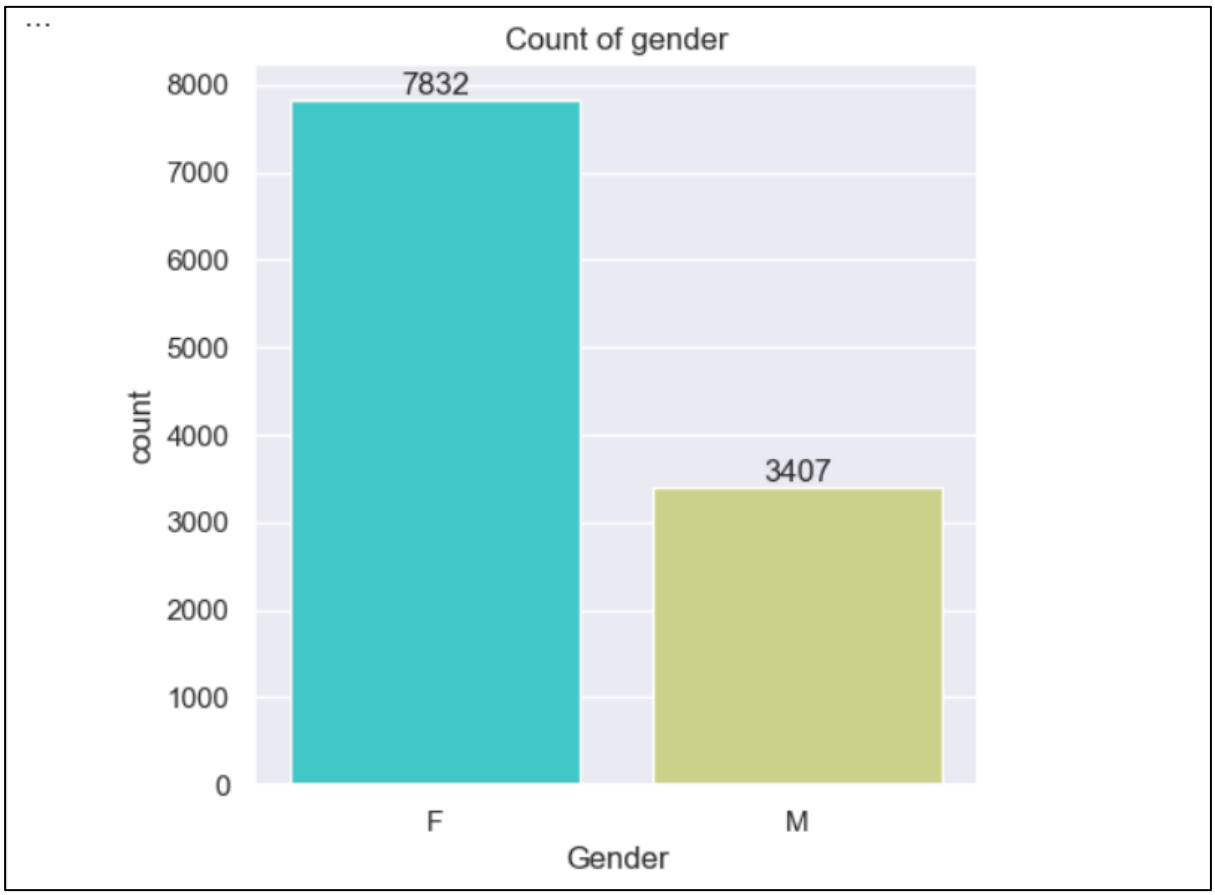
## Interpretation of the data

After the cleaning of the data, we need to analyze the data to get actionable insights.
We need to do the exploratory data analysis
All the data visuals are here as follows:

1. **No. of Gender in the dataset:** We need to see how many males and females are there in the dataset.

   **Code:** ax = sns.countplot(x=df['Gender'],palette='rainbow')
   sns.set(rc={'figure.figsize':(5,10)})
   for bar in ax.containers:
   ax.bar_label(bar)
   plt.title('Count of gender')
   plt.show()

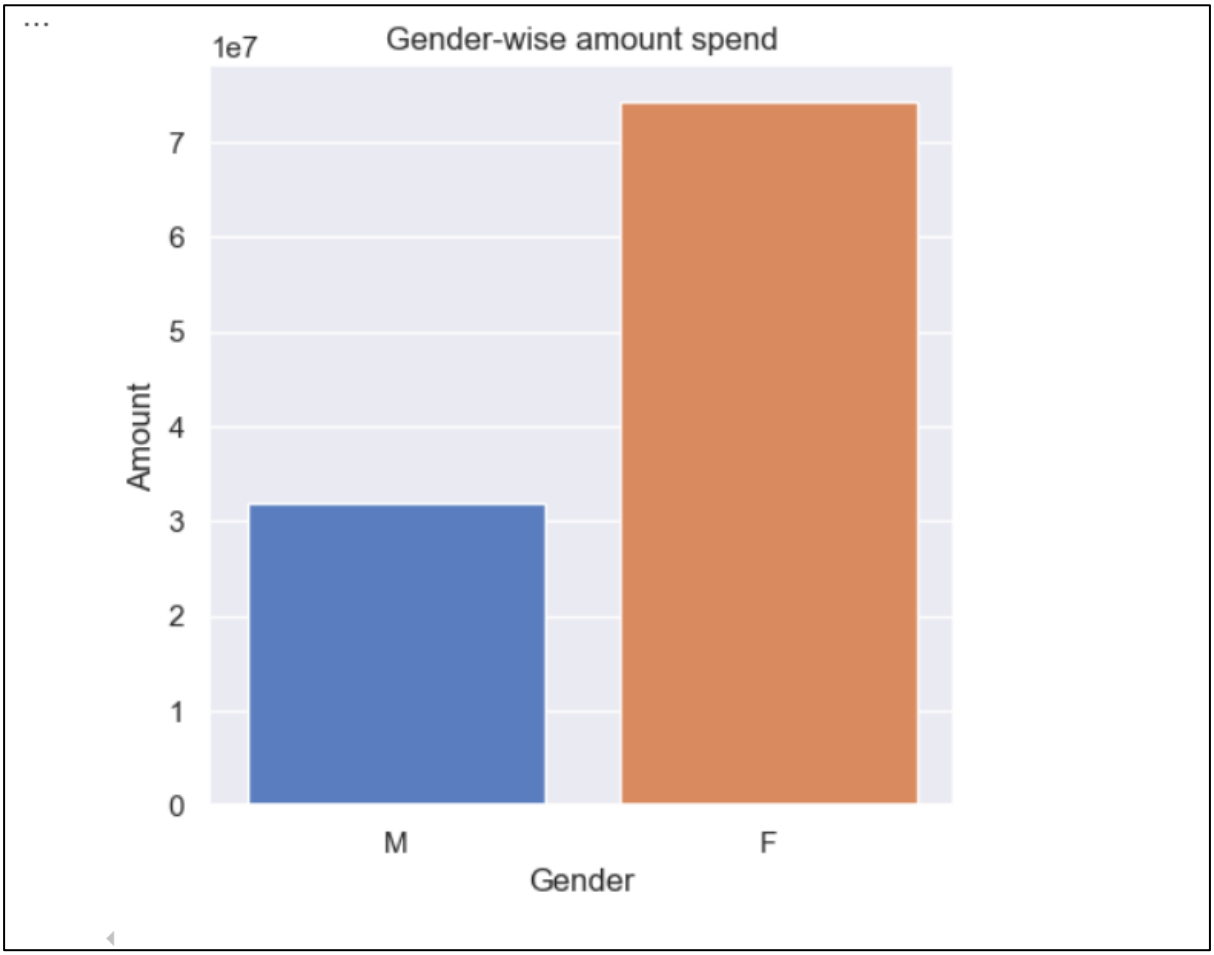
   **Output:**



2. **Amount spend by different gender:** We need to find how much amount spend by males and female on Diwali.

   **Code:** sorted = df.groupby(df['Gender'],as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=True)
   ax = sns.barplot(data=sorted,x='Gender',y='Amount',palette='muted')
   sns.set(rc={'figure.figsize':(5,5)})
   plt.title("Gender-wise amount spend")
   plt.show()

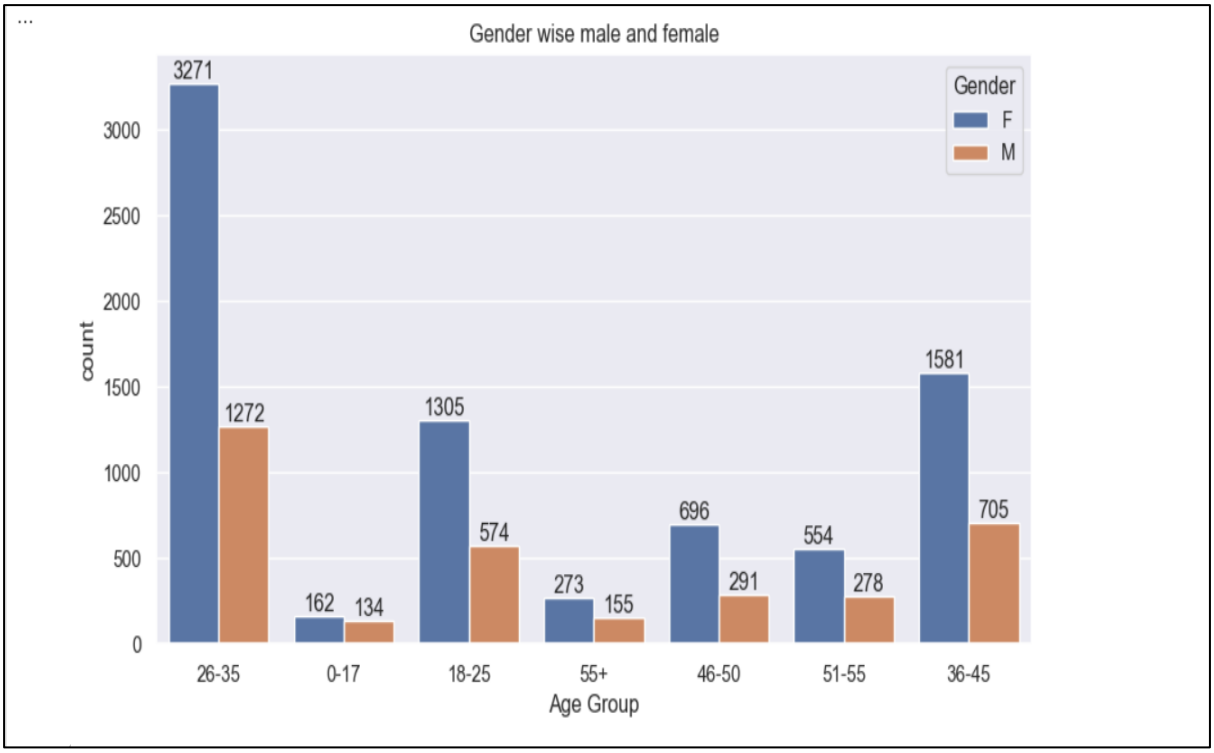
   **Output:**



3. **Age group-wise no. of males and females:** We need to find how many males and females are their according to the age group.

**Code:**
```
sorted = df.sort_values('Age Group')
ax = sns.countplot(data = sorted,x=df['Age Group'],hue=df['Gender'],palette='deep')
for bar in ax.containers:
    ax.bar_label(bar)
sns.set(rc={'figure.figsize':(10,5)})
plt.title("Gender wise male and female")
plt.show()
```

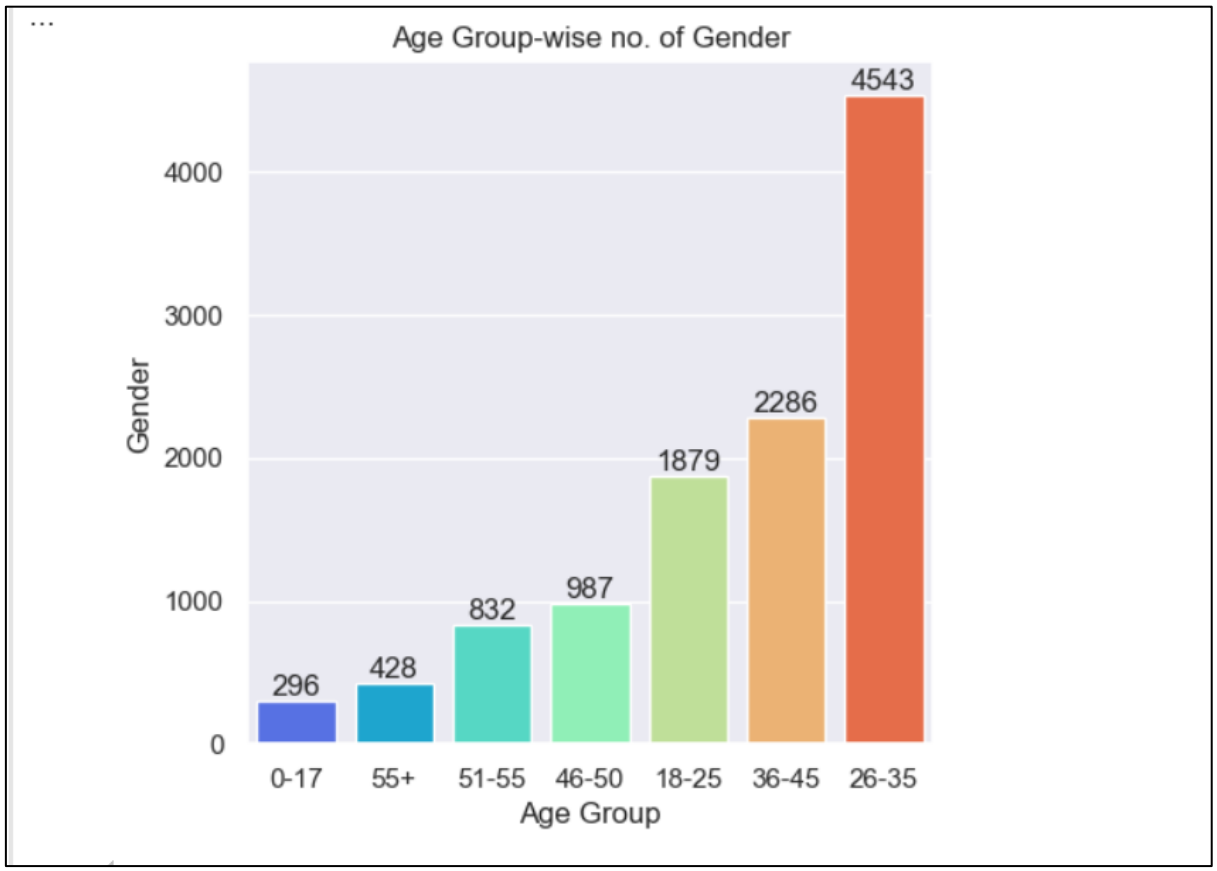
**Output:**



4. **Age Group wise no. of gender:** In this we need to find how many gender are there in each age group.

   **Code:** gen = df.groupby(df['Age Group'],as_index=False)['Gender'].count().sort_values(by='Gender',ascending=True)
   ax = sns.barplot(data=gen,x='Age Group',y='Gender',palette='rainbow')
   for i in ax.containers:
   ax.bar_label(i)
   plt.title("Age Group-wise no. of Gender")
   plt.show()

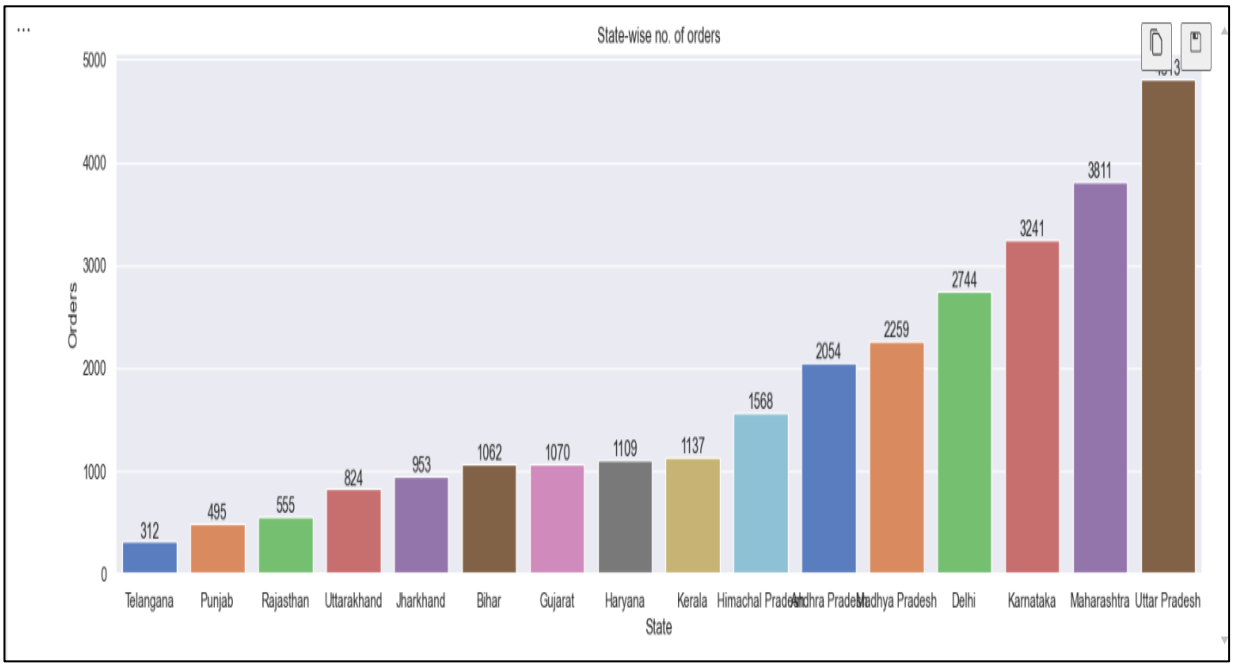
   **Output:**



5. **State-wise no. of order:** In this we need to see the number of orders according to the state.

   **Code:** sorted =
   df.groupby(df['State'],as_index=False)['Orders'].sum().sort_values(by='Orders', ascending=True)
   ax = sns.barplot(data=sorted,x='State',y='Orders',palette='muted')
   sns.set(rc={'figure.figsize':(20,5)})
   for i in ax.containers:
   ax.bar_label(i)
   plt.title("State-wise no. of orders")
   plt.show()

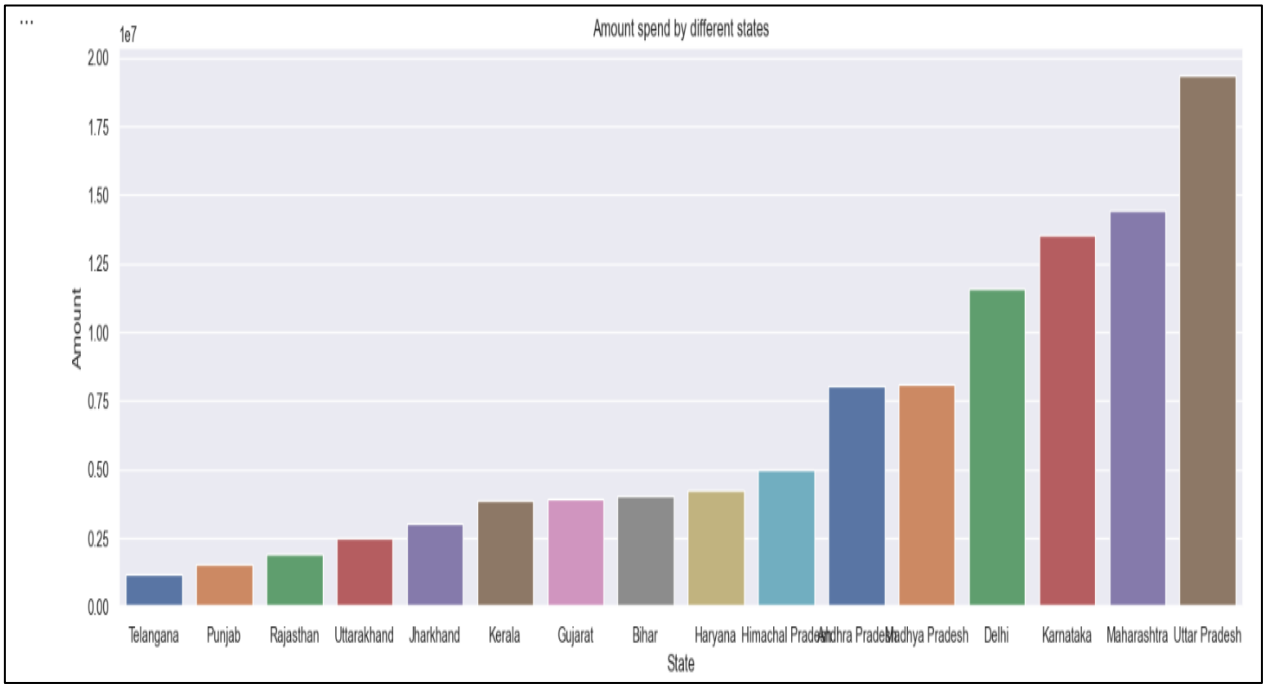
   **Output:**



6. **Amount spent by different state:** This visual show how much amount spend by different states.

   **Code:** sorted = df.groupby(df['State'],as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=True)
   ax = sns.barplot(data=sorted,x='State',y='Amount',palette='deep')
   sns.set(rc={'figure.figsize':(20,5)})
   plt.title("Amount spend by different states")
   plt.show()

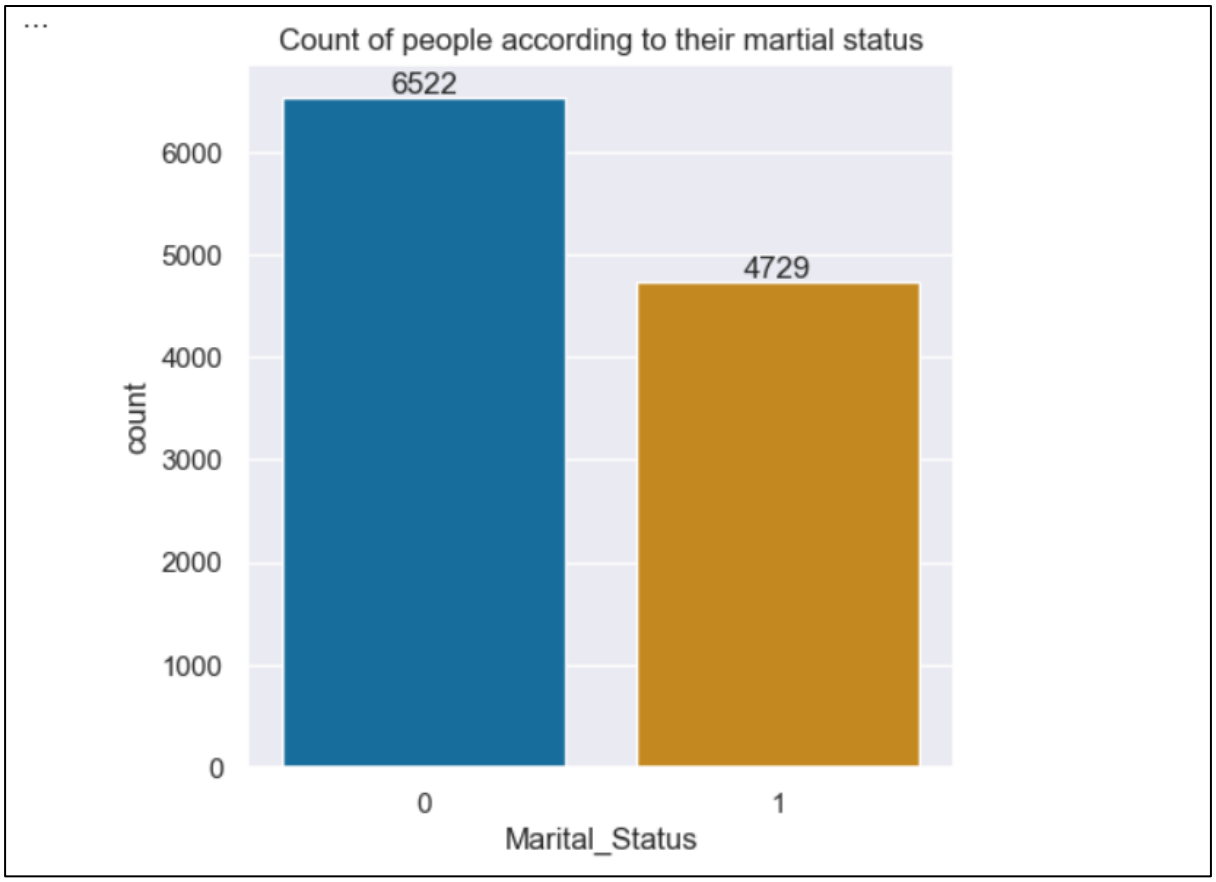
   **Output:**



7. **Count of people according to their marital status: This show how many married and unmarried people do shopping on Diwali.**

**Code:** ax=sns.countplot(data=df,x='Marital_Status',palette='colorblind')
for i in ax.containers:
  ax.bar_label(i)
plt.title("Count of people according to their martial status")
plt.show()

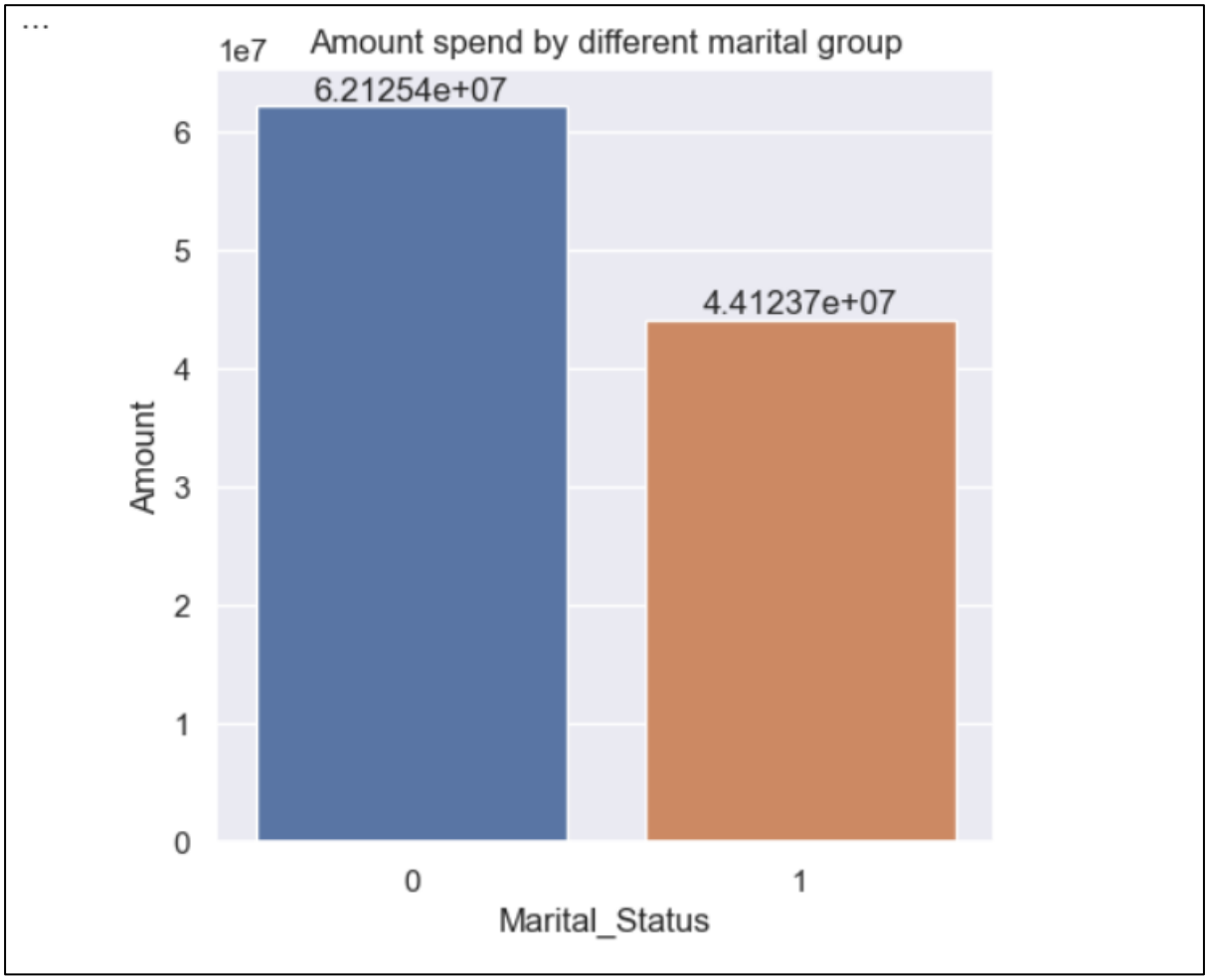
**Output:**



8. **Amount spend by different marital group: This bar chart shows how much amount spend by married and unmarried people.**

**Code:** ms = df.groupby(df['Marital_Status'],as_index=False)['Amount'].sum().sort_values(by ='Amount',ascending=True)
mss = sns.barplot(data=ms,x='Marital_Status',y='Amount',palette='deep')
for i in mss.containers:
  mss.bar_label(i)
sns.set(rc={'figure.figsize':(5,5)})
plt.title("Amount spend by different marital group")
plt.show()

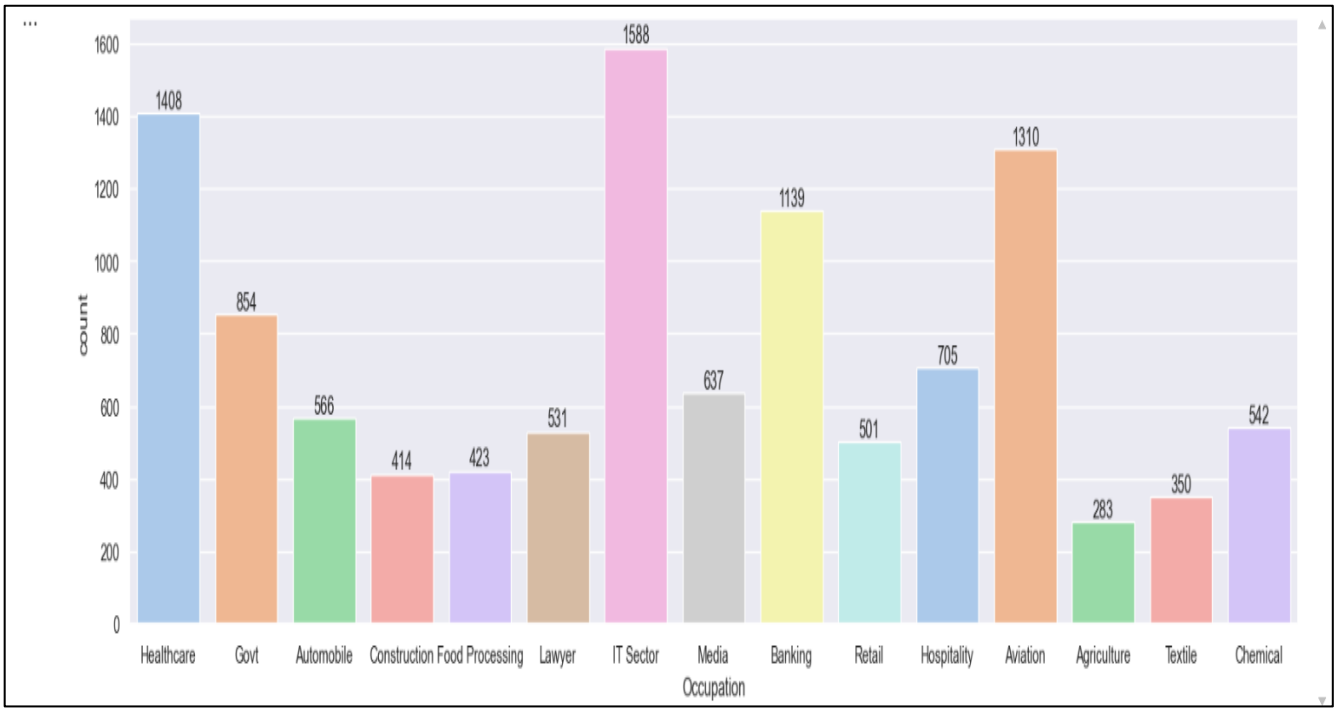
**Output:**



9. **No. of people in different Occupation:** This bar shows how many people belong to different occupation.

   **Code:** ax = sns.countplot(data = df, x = 'Occupation',palette='pastel')
   sns.set(rc={'figure.figsize':(20,5)})
   for i in ax.containers:
   ax.bar_label(i)

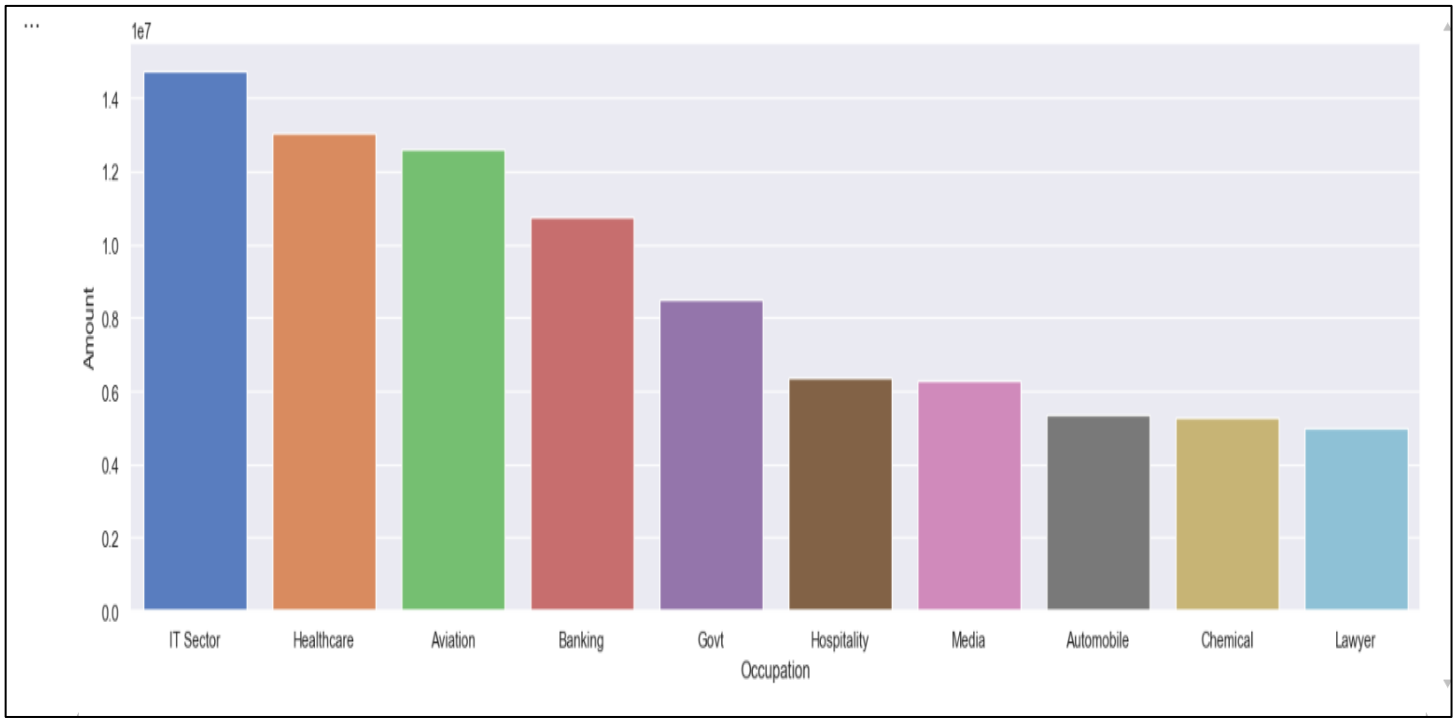
   **Output:**



10. **Amount spend by different occupation:** This shows how much amount spent by different occupation.

**Code:** occ = df.groupby(df['Occupation'],as_index=False)['Amount'].sum().sort_values(by='Amount',ascending=False).head(10)
sns.barplot(data = occ,x='Occupation',y="Amount",palette='muted')
sns.set(rc={'figure.figsize':(20,5)})

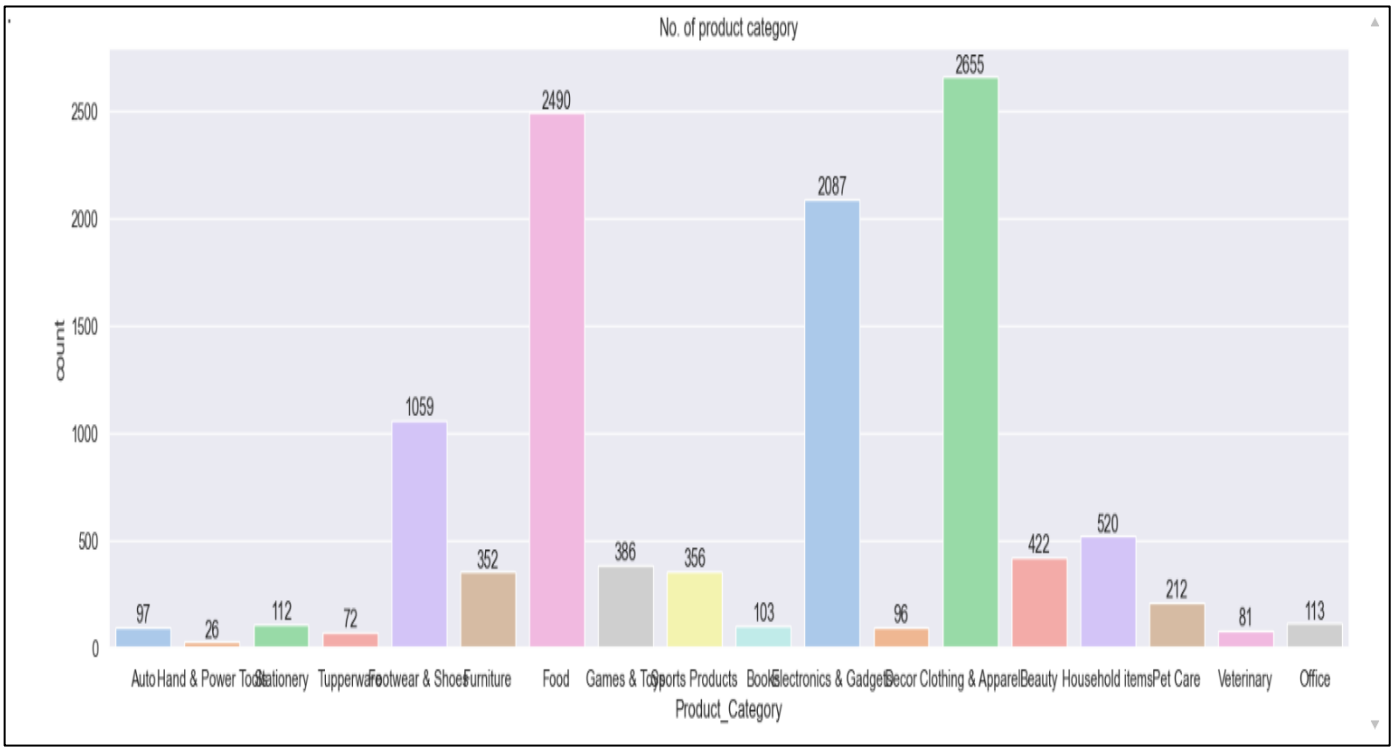
**Output:**



11. **No. of product category:** This shows how many products are there in different product categories.

**Code:**
```
ax = sns.countplot(data= df, x= df['Product_Category'],palette='pastel')
sns.set(rc={'figure.figsize':(20,5)})
for i in ax.containers:
   ax.bar_label(i)
plt.title("No. of product category")
plt.show()
```

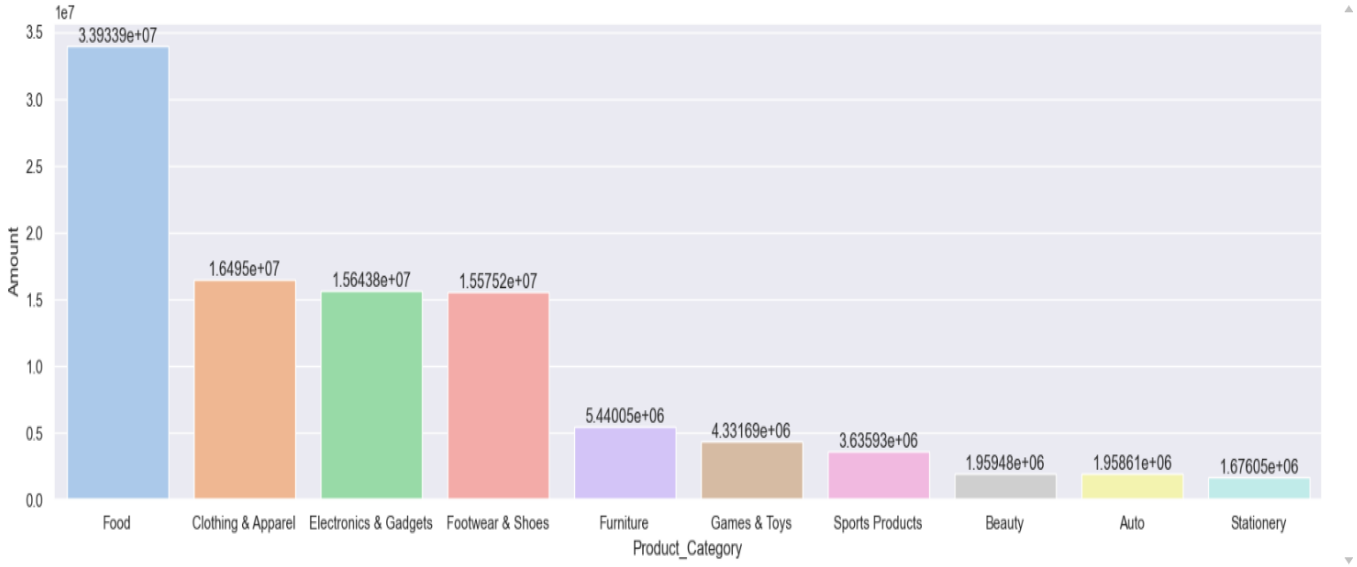
**Output:**



12. **Cost of products in product category:** This shows what are the total amount in different product categories**.**

**Code:** occ = df.groupby(df['Product_Category'],as_index=False)['Amount'].sum().sort_values(by= 'Amount',ascending=False).head(10)
ax = sns.barplot(data= occ,x='Product_Category',y="Amount", palette='pastel')
for i in ax.containers:
  ax.bar_label(i)

**Output:**



## Conclusion

The analysis of Diwali sales data reveals significant insights into consumer spending patterns across various demographics. The key findings include:

1. **Marital Status:** Married individuals tend to spend more on Diwali purchases compared to single individuals, indicating a trend of higher expenditure in households.
2. **Gender:** Males generally spend more than females during Diwali, though the gap may vary depending on the product categories.
3. **States:** States with higher GDPs, such as Maharashtra, Delhi, and Karnataka, show a higher average spend per consumer, reflecting regional economic disparities.
4. **Occupations:** Professionals and business owners exhibit higher spending habits, whereas students and retirees spend comparatively less.
   These insights can help businesses tailor their marketing strategies, optimize inventory management, and improve customer targeting during the Diwali season, maximizing sales and enhancing customer satisfaction.
5. **Product Categories:** This shows how many products are there in different product categories.