diwali-sales-analysis

August 11, 2023

1 Importing Data and Library

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

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

2 General Overview Of Data

[3]:
```
df.shape # shape of dataset
```

[3]:
```
(11251, 15)
```

[4]:
```
df.head(5)  # top 5 values present in dataframe
```

[4]:

	User_ID	Cust_name	Product_ID	Gender	Age Group	Age	Marital_Status	State	Zone	Occupation	Product_Category	Orders	Amount	Status	unnamed1
0	1002903	Sanskriti	P00125942	F	26-35	28	0	Maharashtra	Western	Healthcare	Auto	1	23952.0	NaN	NaN
1	1000732	Kartik	P00110942	F	26-35	35	1	Andhra Pradesh	Southern	Govt	Auto	3	23934.0	NaN	NaN
2	1001990	Bindu	P00118542	F	26-35	35	1	Uttar Pradesh	Central	Automobile	Auto	3	23924.0	NaN	NaN
3	1001425	Sudevi	P00237842	M	0-17	16	0	Karnataka	Southern	Construction	Auto	2			
4	1000588	Joni	P00057942	M	26-35	28	1	Gujarat	Western	Food Processing	Auto	2			

```
3  23912.0     NaN       NaN
4  23877.0     NaN       NaN
```

[5]:
```
df.tail()   # bottom 5 values present in dataframe
```

[5]:
```
       User_ID    Cust_name Product_ID Gender Age Group  Age  Marital_Status  \
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  Amount  \
11246     Maharashtra   Western     Chemical           Office       4   370.0
11247         Haryana  Northern   Healthcare       Veterinary       3   367.0
11248  Madhya Pradesh   Central      Textile           Office       4   213.0
11249       Karnataka  Southern  Agriculture           Office       3   206.0
11250     Maharashtra   Western   Healthcare           Office       3   188.0

       Status  unnamed1
11246     NaN       NaN
11247     NaN       NaN
11248     NaN       NaN
11249     NaN       NaN
11250     NaN       NaN
```

[6]:
```
df.info()  # provide basic information about the dataframe
```

```
<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
```

[7]:
```
df.describe()  # provides basic statistical measures
```

[7]:

	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	

	Status	unnamed1
count	0.0	0.0
mean	NaN	NaN
std	NaN	NaN
min	NaN	NaN
25%	NaN	NaN
50%	NaN	NaN
75%	NaN	NaN
max	NaN	NaN

3 Data Cleaning

Droping column with null values or no values ei; No use.

[8]:
```
df.drop(['Status','unnamed1'],axis=1,inplace=True)
```

[9]:
```
df              # prints dataframe
```

[9]:

	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	
…	…	…	…	…	…	…	…	
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      23952.0
1      23934.0
2      23924.0
3      23912.0
4      23877.0
…          …
11246    370.0
11247    367.0
11248    213.0
11249    206.0
11250    188.0

[11251 rows x 13 columns]
```

[10]:
```
df.isnull() # gives either value is null or not.  false ---> value exist  True
↪---> no value or null
```

[10]:
```
       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]
```

Finding null values and Droping them.

[11]:
```
df.isnull().sum()    # summ of all the null values in the particular column.
```

[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.dropna(how='any',inplace=True)  # how='any' ---> default   removes value␣
 ↪which is null (1 or more null).
```

[13]:
```
df.shape  # previously before data cleaning was ---> (11251, 15)
```

[13]:
```
(11239, 13)
```

4 Data Transformation

[14]:
```
df['Amount']=df['Amount'].astype(int)     # Amount is in Float ----> into Int
```

[15]:
```
df.describe()
```

[15]:

	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

[16]:
```
df[['Age','Orders','Amount']].describe()  # no need for UserId and Marital␣
↪Status so removing them
```

[16]:

	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

5 Exploratory Data Analysis

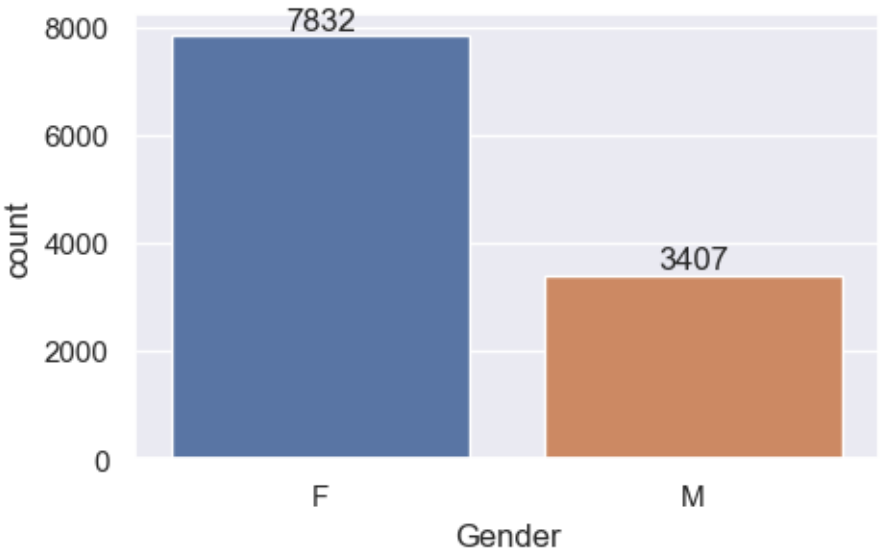
5.0.1 Gender

[17]:
```
df.columns  # to know the Column Values
```

[17]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

[19]:
```
ax=sns.countplot(x='Gender',data=df) # Creating countplot of Gender
sns.set(rc={'figure.figsize':(5,3)})
```

```
for bars in ax.containers:                  # Adding Data-Labels to data
    ax.bar_label(bars)
```



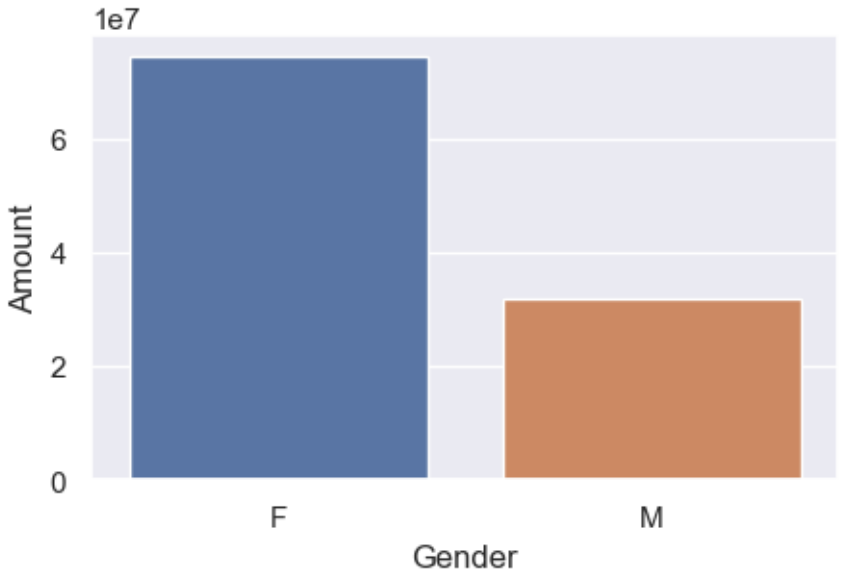
5.0.2 Gender WRT Amount

[20]:
```
sls_gen = df.groupby(["Gender"],as_index=False)['Amount'].sum().
 ↪sort_values(by='Amount',ascending=False)
sls_gen    # Creating dataframe of Gender wrt Amount spent
```

[20]:

	Gender	Amount
0	F	74335853
1	M	31913276

[21]:
```
sns.barplot(x='Gender',y='Amount',data=sls_gen)  # Creating barplot of Gender␣
 ↪wrt Amount spent
sns.set(rc={'figure.figsize':(5,3)})
```



5.0.3 Insight 1:

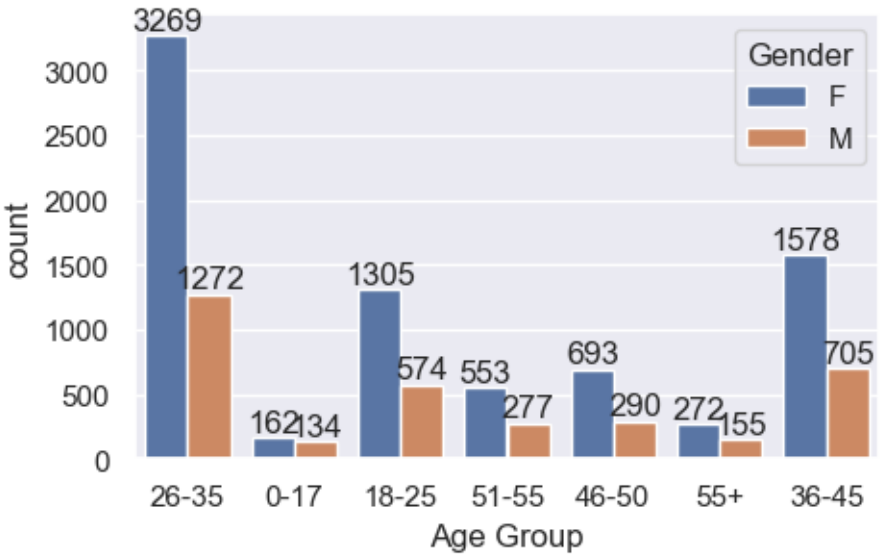
Female shoppers not only exhibit a higher order count but also contribute significantly more to total sales compared to their male counterparts during Diwali sales.

5.0.4 AGE

[22]:
```
df.columns
```

[22]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

[24]:
```
ax=sns.countplot(x='Age Group',data=df,hue='Gender')
for i in ax.containers:
    ax.bar_label(i)
```



5.0.5 Age WRT Amount

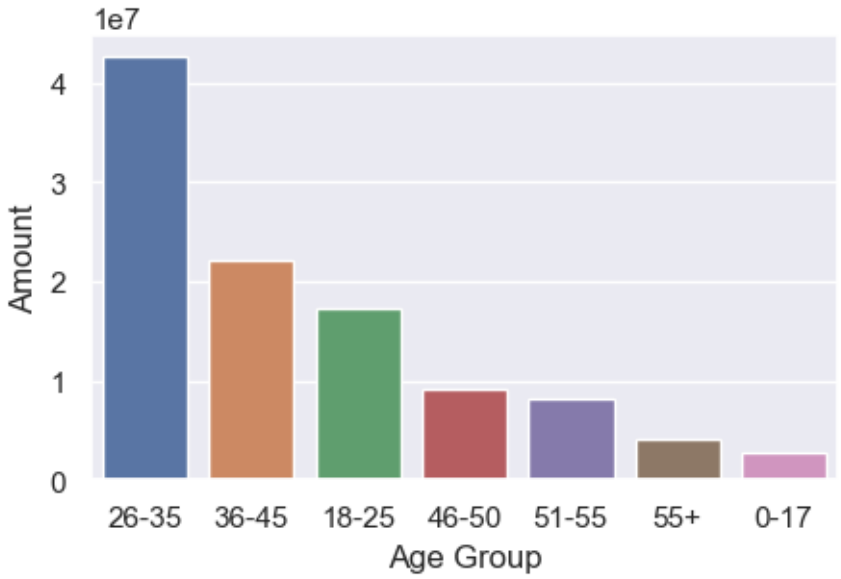
[25]:
```
age_amt = df.groupby(['Age Group'], as_index=False)['Amount'].sum().
↪sort_values(by='Amount', ascending=False)
age_amt.reset_index(drop=True,inplace=True)
age_amt
```

[25]:

	Age Group	Amount
0	26-35	42613442
1	36-45	22144994
2	18-25	17240732
3	46-50	9207844
4	51-55	8261477
5	55+	4080987
6	0-17	2699653

[26]:
```
sns.barplot(x='Age Group',y='Amount',data=age_amt)
```

[26]:
```
<Axes: xlabel='Age Group', ylabel='Amount'>
```



5.0.6 Insight 2:

The prime purchasing power lies within the age group of 26 to 45 for both females and males. This age segment exhibits the highest order counts.

5.0.7 States WRT Orders

[27]:
```
df.columns
```

[27]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

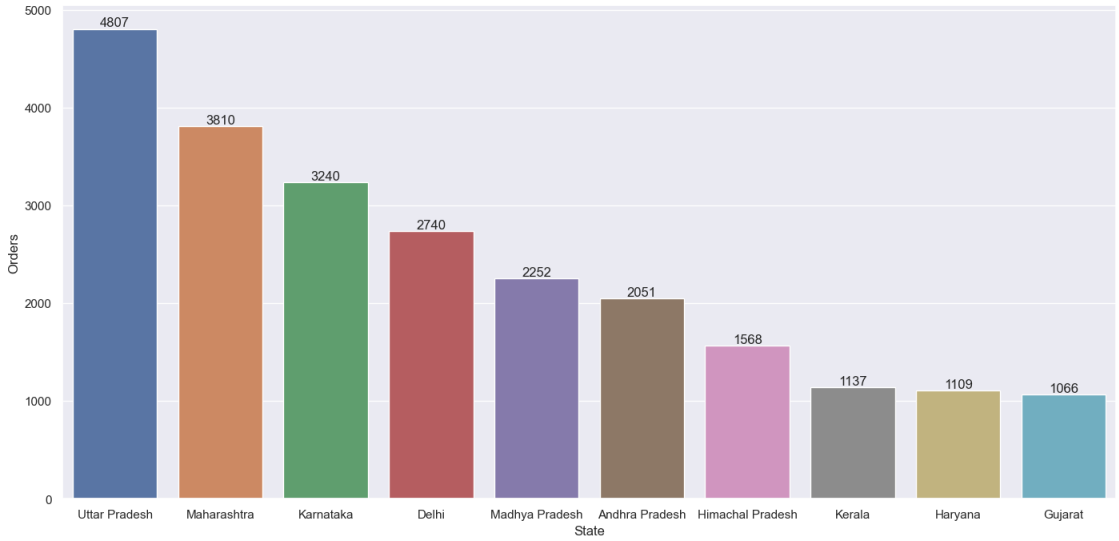
[28]:
```
state_ord = df.groupby(['State'],as_index=False)['Orders'].sum().
↪sort_values(by='Orders',ascending=False).head(10)
state_ord.reset_index(drop=True,inplace=True)
state_ord
```

[28]:

	State	Orders
0	Uttar Pradesh	4807
1	Maharashtra	3810
2	Karnataka	3240
3	Delhi	2740
4	Madhya Pradesh	2252
5	Andhra Pradesh	2051
6	Himachal Pradesh	1568

```
7            Kerala    1137
8           Haryana    1109
9           Gujarat    1066
```

[30]:
```
ax=sns.barplot(x='State',y='Orders',data=state_ord)
sns.set(rc={'figure.figsize':(17,8)})
for i in ax.containers:
    ax.bar_label(i)
```



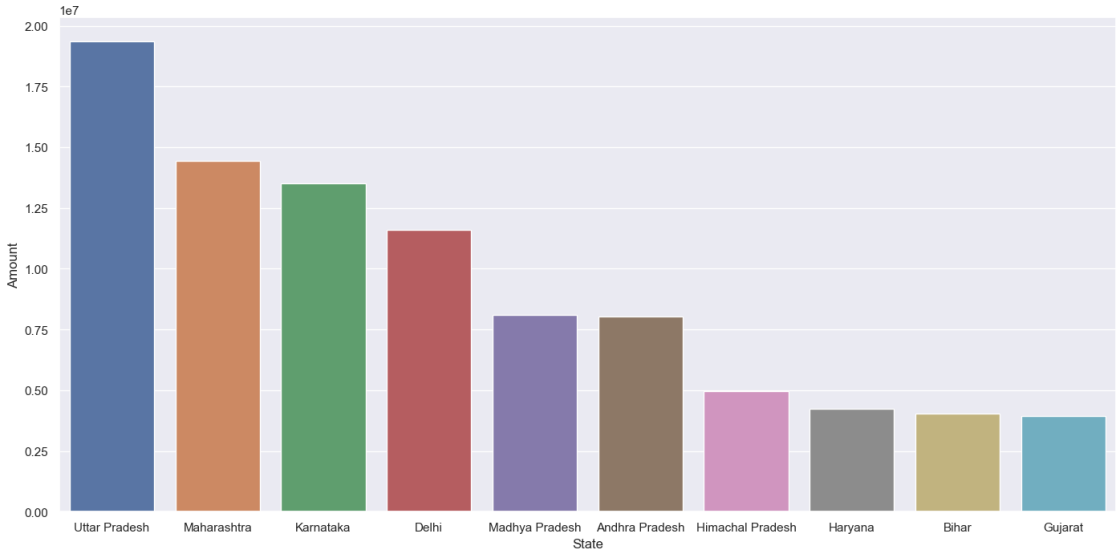
5.0.8 State WRT Amount

[31]:
```
sts_amt=df.groupby(['State'],as_index=False)['Amount'].sum().
  ↪sort_values(by='Amount',ascending=False)
sts_amt=sts_amt.head(10)
sts_amt.reset_index(drop=True,inplace=True)
sts_amt
```

[31]:
```
              State    Amount
0     Uttar Pradesh  19374968
1       Maharashtra  14427543
2         Karnataka  13523540
3             Delhi  11603818
4    Madhya Pradesh   8101142
5    Andhra Pradesh   8037146
6  Himachal Pradesh   4963368
7           Haryana   4220175
8             Bihar   4022757
9           Gujarat   3946082
```

[32]:
```
sns.barplot(x='State',y='Amount',data=sts_amt)
```

[32]:
```
<Axes: xlabel='State', ylabel='Amount'>
```

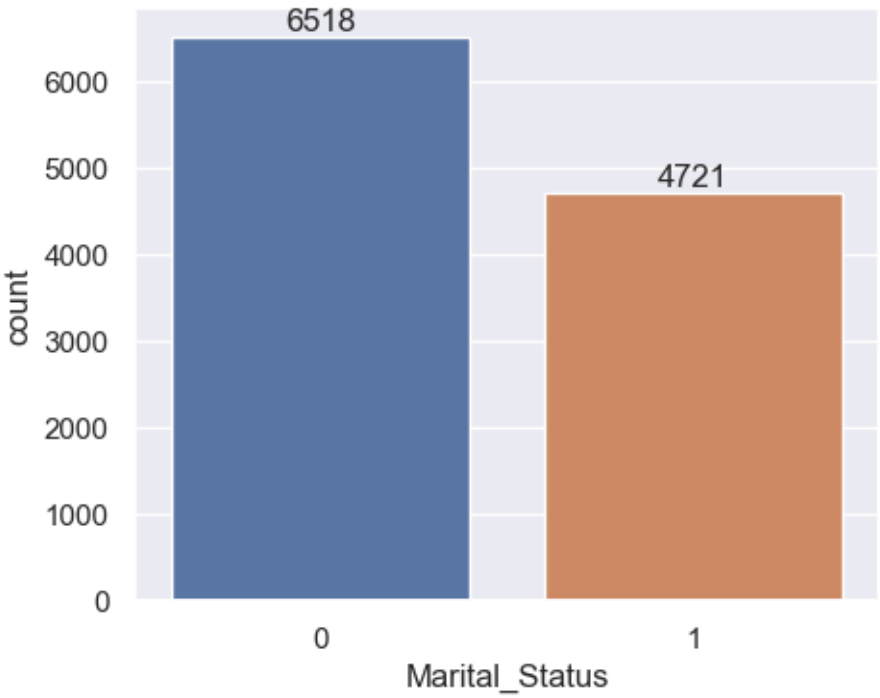


5.0.9 Insight 3:

Uttar Pradesh, Maharashtra, Karnataka, and Delhi lead in both order frequency and total expenditure, indicating their pivotal role in our Diwali sales success.

5.0.10 Marital_Status

[35]:
```
ax=sns.countplot(x='Marital_Status',data=df)
sns.set(rc={'figure.figsize':(5,4)})
for i in ax.containers:
    ax.bar_label(i)
```



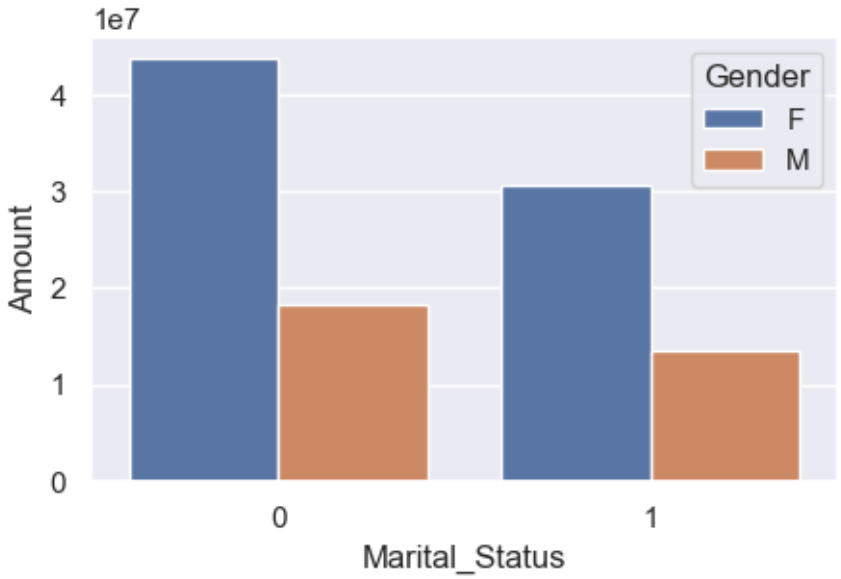
5.0.11 Marital Status WRT Amount —> Gender

[36]:
```
mar_amt=df.groupby(['Marital_Status','Gender'],as_index=False)['Amount'].sum().
  ↪sort_values(by='Amount',ascending=False)
mar_amt.reset_index(drop=True,inplace=True)
mar_amt
```

[36]:

	Marital_Status	Gender	Amount
0	0	F	43786646
1	1	F	30549207
2	0	M	18338738
3	1	M	13574538

[41]:
```
sns.barplot(x='Marital_Status',y='Amount',data=mar_amt,hue='Gender')
sns.set(rc={'figure.figsize':(5,3)})
```



5.0.12 Insight 4:

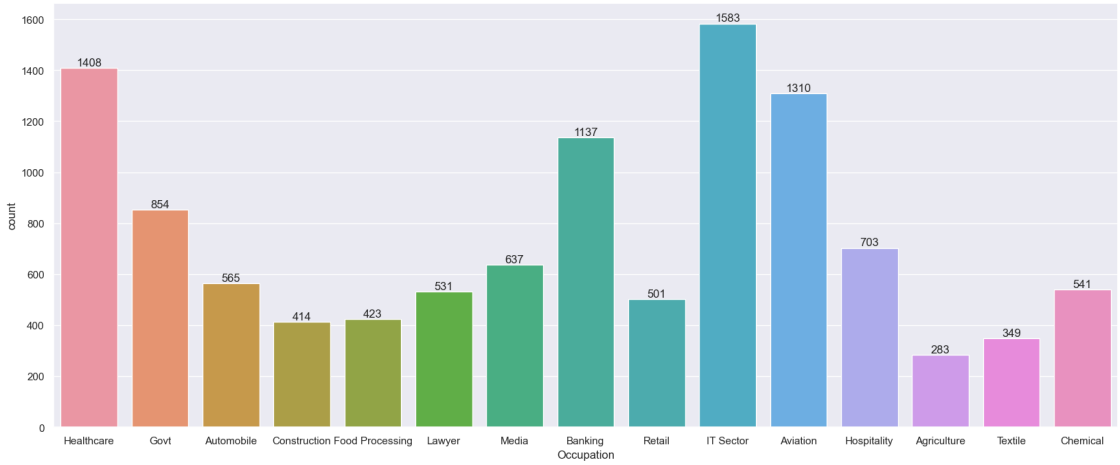
Among our customer base, bachelors exhibit higher spending tendencies than those who are married. Notably, regardless of marital status, females lead in spending, emphasizing their integral role in driving sales growth.

5.0.13 Occupation

[42]:
```
df.columns
```

[42]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

[44]:
```
ax=sns.countplot(x='Occupation',data=df)
sns.set(rc={'figure.figsize':(20,8)})
for i in ax.containers:
    ax.bar_label(i)
```



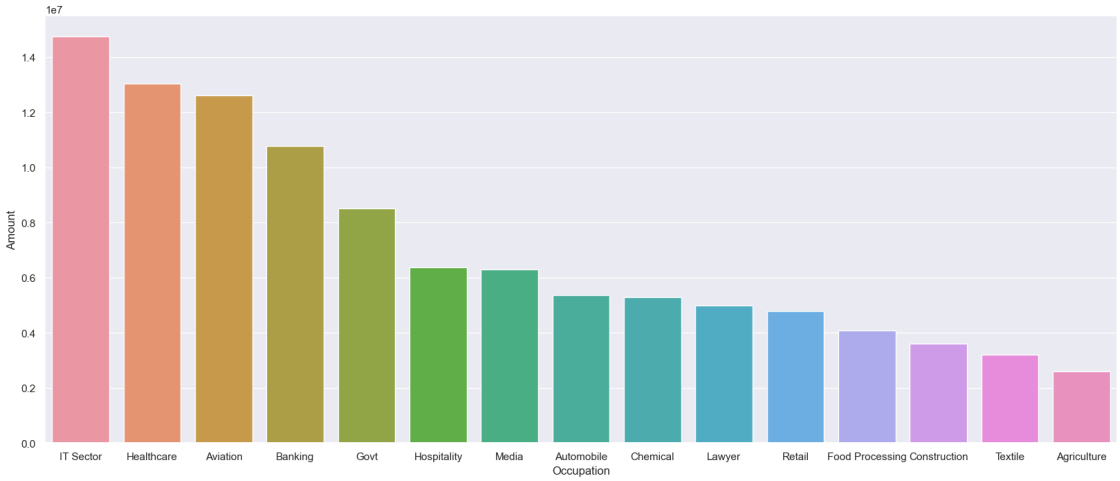
5.0.14 Occupation WRT Amount

[45]:
```
Occ_amt=df.groupby(['Occupation'],as_index=False)['Amount'].sum().
 ↪sort_values(by='Amount',ascending=False)
Occ_amt.reset_index(drop=True,inplace=True)
Occ_amt
```

[45]:

	Occupation	Amount
0	IT Sector	14755079
1	Healthcare	13034586
2	Aviation	12602298
3	Banking	10770610
4	Govt	8517212
5	Hospitality	6376405
6	Media	6295832
7	Automobile	5368596
8	Chemical	5297436
9	Lawyer	4981665
10	Retail	4783170
11	Food Processing	4070670
12	Construction	3597511
13	Textile	3204972
14	Agriculture	2593087

[46]:
```
sns.barplot(x='Occupation',y='Amount',data=Occ_amt)
sns.set(rc={'figure.figsize':(20,8)})
```

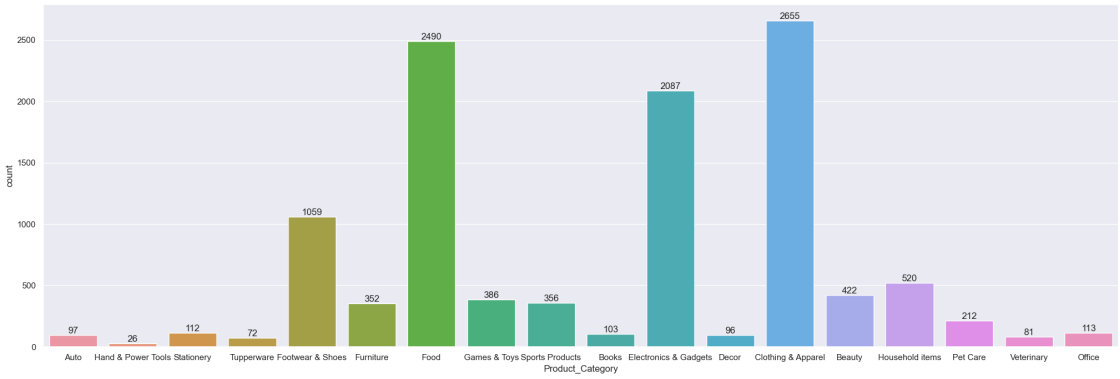


5.0.15 Insight 5:

Notable consumption is observed among customers employed in the IT sector, Healthcare, Aviation, and Banking industries, indicating these sectors as the leading contributors to our sales.

5.0.16 Product_Category

[48]:
```
ax=sns.countplot(x='Product_Category',data=df)
sns.set(rc={'figure.figsize':(25,8)})
for i in ax.containers:
    ax.bar_label(i)
```



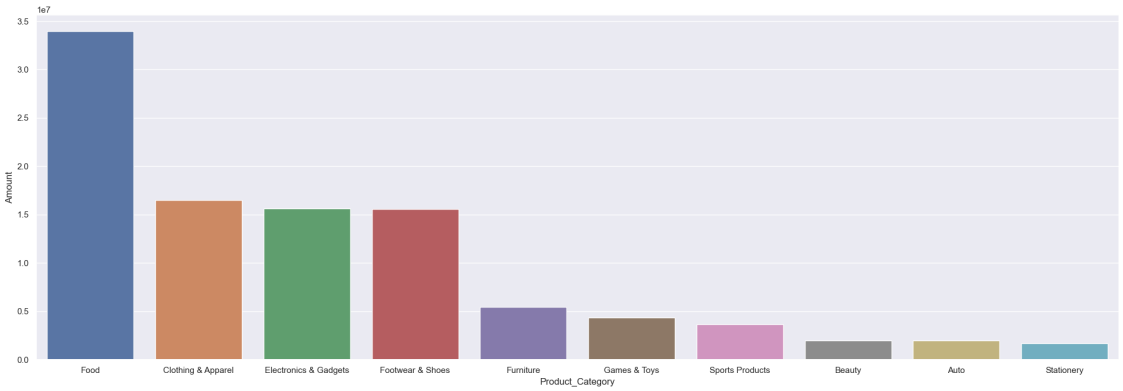
5.0.17 Product Category WRT Amount

[49]:
```
pro_amt=df.groupby(['Product_Category'],as_index=False)['Amount'].sum().
  ↪sort_values(by='Amount',ascending=False).head(10)
pro_amt.reset_index(drop=True,inplace=True)
pro_amt
```

[49]:

	Product_Category	Amount
0	Food	33933883
1	Clothing & Apparel	16495019
2	Electronics & Gadgets	15643846
3	Footwear & Shoes	15575209
4	Furniture	5440051
5	Games & Toys	4331694
6	Sports Products	3635933
7	Beauty	1959484
8	Auto	1958609
9	Stationery	1676051

[50]:
```
sns.barplot(x='Product_Category',y='Amount',data=pro_amt)
sns.set(rc={'figure.figsize':(19,7)})
plt.show()
```



5.0.18 Insight 6:

Within the market, heightened demand is observed for product categories such as Food, Clothing & Apparel, Electronics & Gadgets, and Footwear & Shoes, underscoring their prominence among consumers.

5.0.19 Zone WRT Orders

[51]:
```
df.columns
```

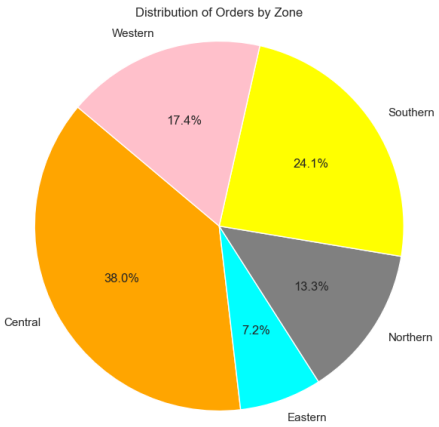
[51]:
```
Index(['User_ID', 'Cust_name', 'Product_ID', 'Gender', 'Age Group', 'Age',
       'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
       'Orders', 'Amount'],
      dtype='object')
```

[52]:
```
# Group by 'Zone' and calculate the sum of orders
zone_orders = df.groupby('Zone')['Orders'].sum()
zone_orders
```

[52]:
```
Zone
Central     10623
Eastern      2015
Northern     3727
Southern     6740
Western      4876
Name: Orders, dtype: int64
```

[53]:
```
colors = ['orange','cyan','gray','yellow', 'pink']
plt.pie(zone_orders, labels=zone_orders.index, autopct='%1.1f%%',␣
 ↪startangle=140,colors=colors)
plt.title('Distribution of Orders by Zone')
plt.axis('equal')  # Equal aspect ratio ensures that the pie is drawn as a␣
 ↪circle.

plt.show()
```



5.0.20 Insight 7:

An analysis of the pie chart reveals that the Central and Southern zones harbor a significant proportion of our customer base, indicating their substantial contribution to our market presence.

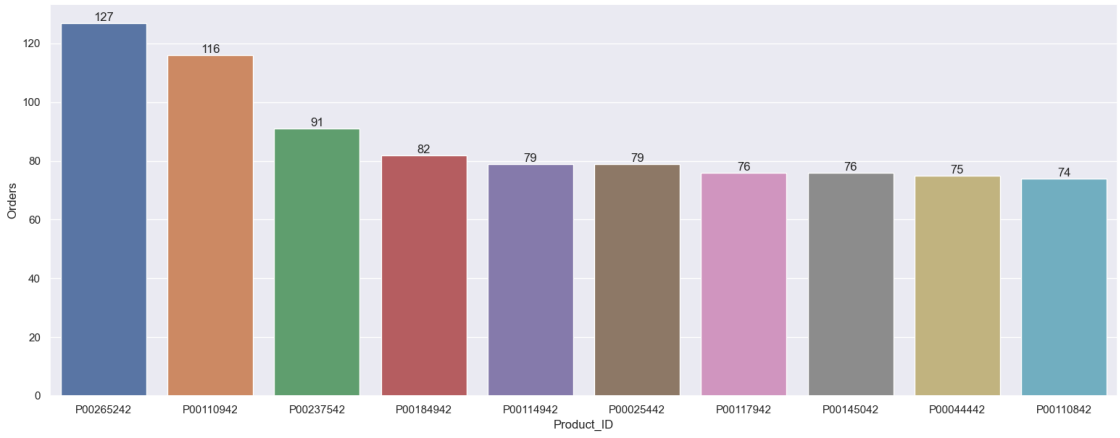
5.0.21 Product ID WRT Orders

[54]:
```
prod_ord = df.groupby(['Product_ID'],as_index=False)['Orders'].sum().
↪sort_values(by='Orders',ascending=False).head(10)
prod_ord.reset_index(drop=True,inplace=True)
prod_ord
```

[54]:

	Product_ID	Orders
0	P00265242	127
1	P00110942	116
2	P00237542	91
3	P00184942	82
4	P00114942	79
5	P00025442	79
6	P00117942	76
7	P00145042	76
8	P00044442	75
9	P00110842	74

[55]:
```
ax=sns.barplot(data=prod_ord,x='Product_ID',y='Orders')
sns.set(rc={'figure.figsize':(10,5)})
for i in ax.containers:
    ax.bar_label(i)
```



5.0.22 Insight 8:

Product IDs such as P00265242, P00110942, and P00237542 stand out with higher order counts, reflecting their strong customer appeal.

6 Conclusion

- Females contribute significantly to both order count and expenditure, emphasizing the need for gender-targeted marketing strategies.

- The age group of 26 to 45 emerges as the prime spending segment for both genders.

- States like Uttar Pradesh, Maharashtra, Karnataka, and Delhi are key drivers of both order frequency and total expenditure.

- Industries such as IT, Healthcare, Aviation, and Banking show substantial consumer engagement, suggesting avenues for focused marketing efforts.

- Product categories including Food, Clothing, Electronics, and Footwear demonstrate higher demand compared to other categories.

- The Central and Southern zones host a considerable customer base, showcasing their importance in our market presence.

6.1 Recommendations:

1. **Segmented Marketing:** Tailor marketing for females and males in the 26-45 age group.
2. **Regional Amplification:** Focus efforts on high-demand states like Uttar Pradesh, Maharashtra, Karnataka, and Delhi.
3. **Industry Partnerships:** Collaborate with IT, Healthcare, Aviation, and Banking sectors for targeted offers.
4. **Product Expansion:** Diversify offerings within popular categories like Food, Clothing, Electronics, and Footwear.
5. **Zone-specific Approach:** Customize strategies for the Central and Southern zones.
6. **Enhanced Engagement:** Implement loyalty programs, personalized services, and feedback channels.
7. **Data-Driven Adaptation:** Continuously analyze sales data for real-time adjustments.
8. **Social Media Impact:** Leverage platforms for showcasing products and customer engagement.
9. **Collaborative Alignment:** Foster teamwork among departments for cohesive strategies.

Implementing these recommendations can elevate the store's performance and customer satisfaction during Diwali sales and beyond.

7 Let's Connect

7.0.1 Linkedin : https://www.linkedin.com/in/yash-sonkhiya/

7.0.2 Portfolio : https://www.datascienceportfol.io/YashSonkhiya

7.0.3 GitHub : https://github.com/ysonkhiya122