Diwali-analysis-data-science-project

[42]:
```python
# import python libraries

import numpy as np #mathematical use
import pandas as pd #use for dataframe means tables
import matplotlib.pyplot as plt # visualizing data
import seaborn as sns #matplotlib and seaborn is use for making charts and␣
↪graphs
```

[43]:
```python
# import csv file
df = pd.read_csv('Diwali Sales Data.csv', encoding= 'unicode_escape')
df
```

[43]:
```
         1002903    Cust_name Product_ID Gender Age Group  Age  \
0     1000732.0   Sanskriti  P00125942      F     26-35  28
1     1001990.0      Kartik  P00110942      F     26-35  35
2     1001425.0       Bindu  P00118542      F     26-35  35
3     1000588.0      Sudevi  P00237842      M      0-17  16
4     1000588.0        Joni  P00057942      M     26-35  28
…           …           …          …     …         …  …
11246 1004089.0    Manning  P00296942      M     18-25  19
11247 1001209.0 Reichenbach P00171342      M     26-35  33
11248 1004023.0      Oshin  P00201342      F     36-45  40
11249 1002744.0     Noonan  P00059442      M     36-45  37
11250       NaN     Brumley  P00281742      F     18-25  19

      Marital_Status         State      Zone      Occupation \
0                   0    Maharashtra Western      Healthcare
1                  1 Andhra Pradesh Southern            Govt
2                   1 Uttar Pradesh Central      Automobile
3                   0      Karnataka Southern    Construction
4                   1        Gujarat Western Food Processing
…                 …            …       …            …
11246 1     Maharashtra Western    Chemical 11247   0
      Haryana Northern Healthcare
11248  0  Madhya  Pradesh  Central  Textile  11249  0  Karnataka
Southern Agriculture
11250               0     Maharashtra Western      Healthcare
      Product_Category Orders Amount Status unnamed1
0              Auto    1 23952.0   NaN   NaN

1              Auto    3 23934.0   NaN   NaN
```

```
2              Auto   3 23924.0  NaN   NaN
3              Auto   2 23912.0  NaN   NaN
4              Auto   2 23877.0  NaN   NaN
     …             …     …      …     …       …
11246         Office   4     370.0 NaN   NaN
11247         Veterinary    3     367.0 NaN   NaN
11248         Office   4     213.0 NaN   NaN
11249         Office   3     206.0 NaN   NaN
11250         Office   3     188.0 NaN   NaN
     [11251 rows x 15 columns]
```

[44]:
```
df.shape
```

[44]:
```
(11251, 15)
```

[45]:
```
df.head()
```

[45]:
```
  1002903 Cust_name Product_ID Gender Age Group Age Marital_Status \
0 1000732.0 Sanskriti P00125942     F     26-35 28              0
 1 1001990.0 Kartik P00110942 F 26-35 35                          5
 2 1001425.0 Bindu P00118542 F 26-35 35                           5
 3 1000588.0 Sudevi P00237842 M 0-17 16 04 1000588.0 Joni P00057942 M
 26-35 28                                                         5
       State Zone Occupation Product_Category Orders \          5
 0 Maharashtra Western Healthcare Auto                            5
 1 Andhra Pradesh Southern Govt Auto                              5


 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
```

[46]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11251 entries, 0 to 11250
```

```
Data columns (total 15 columns):
 # Column              Non-Null Count
                       Dtype
--- ------             -------------- ----
                       -
 0 1002903             11250      non-null
                       float64
 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(4), int64(3), object(8)
memory usage: 1.3+ MB
```

[47]:

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

[48]:

```
#check for null values
pd.isnull(df).sum()
```

[48]:

```
1002903            1
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
```

[49]:
```
# drop null values
df.dropna(inplace=True) #dropna delete null values
```

[50]:
```
#initialize list of lists
data_test=[['Anushka',11],['Kahna',15],['Keshav',],['Lalita',16]]

#create the pandas dataframe using list
df_test = pd.DataFrame(data_test,columns=['Name','Age'])

df_test
```

[50]:

	Name	Age
0	Anushka	11.0
1	Kahna	15.0
2	Keshav	NaN
3	Lalita	16.0

[51]:
```
# change data type
df['Amount'] = df['Amount'].astype('int')
```

[52]:
```
df['Amount'].dtypes
```

[52]:
```
dtype('int64')
```

[53]:
```
df.columns
```

[53]:
```
Index(['1002903', 'Cust_name', 'Product_ID', 'Gender', 'Age Group',
'Age',
  'Marital_Status', 'State', 'Zone', 'Occupation', 'Product_Category',
          'Orders', 'Amount'],
         dtype='object')
```

[54]:
```
#Remove spaces
df.columns = df.columns.str.strip()
```

[55]:
```
#rename column
df.rename(columns= {'Marital_Status':'Shaadi'})
```

[55]:
```
        1002903    Cust_name Product_ID Gender Age Group  Age  Shaadi  \
0     1000732.0   Sanskriti  P00125942      F     26-35  28       0
```

```
1     1001990.0     Kartik P00110942     F     26-35 35       1
2     1001425.0      Bindu P00118542     F     26-35 35       1
3     1000588.0     Sudevi P00237842     M      0-17 16       0
4     1000588.0       Joni P00057942     M     26-35 28       1
…          …          …        …    …        … …      …
11245 1000695.0  Bertelson P00057442   F     26-35 31   1
11246 1004089.0  Manning P00296942     M     18-25 19   1
11247 1001209.0 Reichenbach P00171342 M     26-35 33   0
11248 1004023.0  Oshin P00201342  F     36-45 40   0
11249 1002744.0  Noonan P00059442 M     36-45 37   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
…              …       …           …           …     …
11245          Delhi Central Aviation   Office     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
         Amount
0        23952
1        23934
2        23924
3        23912
4        23877
…        …
11245    381
11246    370
11247    367
11248    213
11249    206
```

```
[11238 rows x 13 columns]
```

[56]:
```
# describe() method returns description of the data in the DataFrame (i.e.
↪count, mean, std, etc)
df.describe()
```

[56]:
```
             1002903           Age Marital_Status       Orders       Amount
count 1.123800e+04 11238.000000 11238.000000         11238.000000
                                 11238.000000
mean 1.003004e+06    35.411817       0.420093    2.489589
                                                 9454.435042
std   1.716159e+03    12.753494       0.493595    1.115006
                                                 5221.855974
min   1.000001e+06    12.000000       0.000000    1.000000  206.000000
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%   1.004430e+06    43.000000       1.000000    3.000000
                                                 12676.000000
max   1.006040e+06    92.000000       1.000000    4.000000
                                                 23952.000000
```

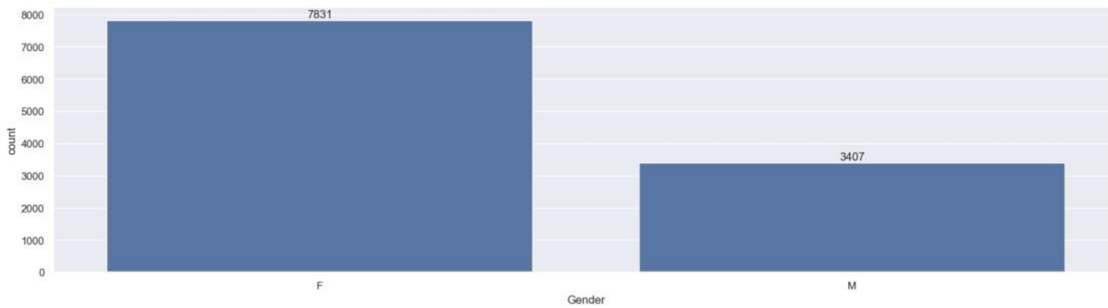
[57]:
```
# use describe() for specific columns
df[['Age', 'Orders',
'Amount']].describe()
```

[57]:
```
Age   Orders      Amount count 11238.000000
11238.000000 11238.000000 mean   35.411817
2.489589 9454.435042 std   12.753494
1.115006 5221.855974
min       12.000000     1.000000 206.000000
25%       27.000000     2.000000
                        5443.000000
50%       33.000000     2.000000
                        8109.000000
75%       43.000000     3.000000
                        12676.000000
max       92.000000     4.000000
                        23952.000000
```

1 Exploratory Data Analysis

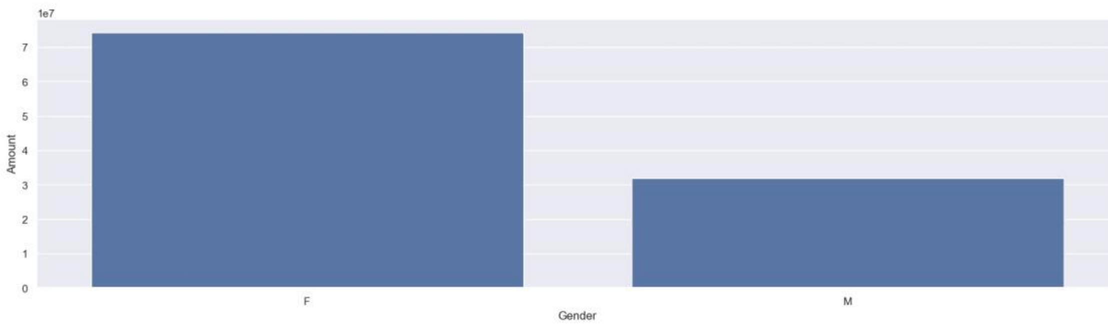
1.0.1 Gender

[58]:
```
# 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)
```

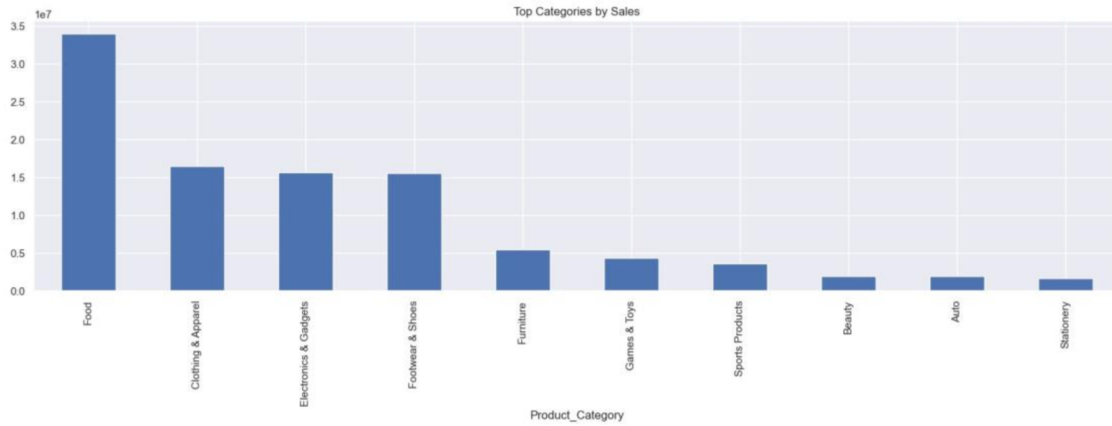


[59]:
```
# plotting a bar chart for gender vs total amount

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)
```

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



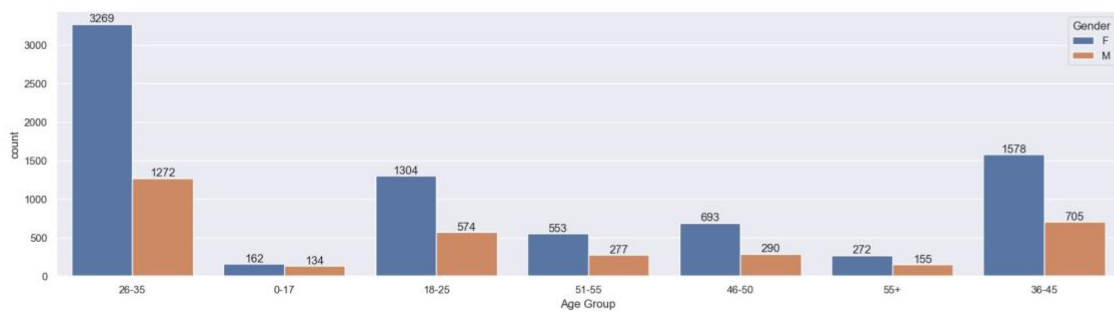
```
[60]: # Category vs Amount
df.groupby('Product_Category')['Amount'].sum().sort_values(ascending=False).
  ↪head(10).plot(kind='bar')
plt.title("Top Categories by Sales")
plt.show()
```



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

1.0.2 Age

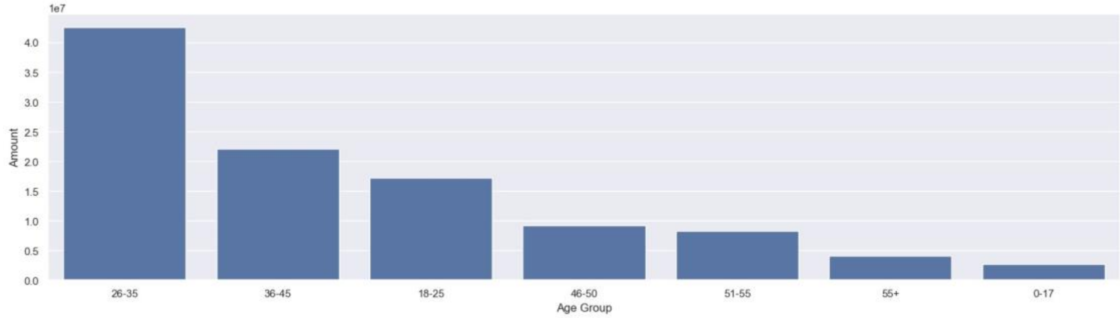
```
[61]: ax = sns.countplot(data = df, x = 'Age Group', hue = 'Gender')

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




```
[62]: # 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)
```

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

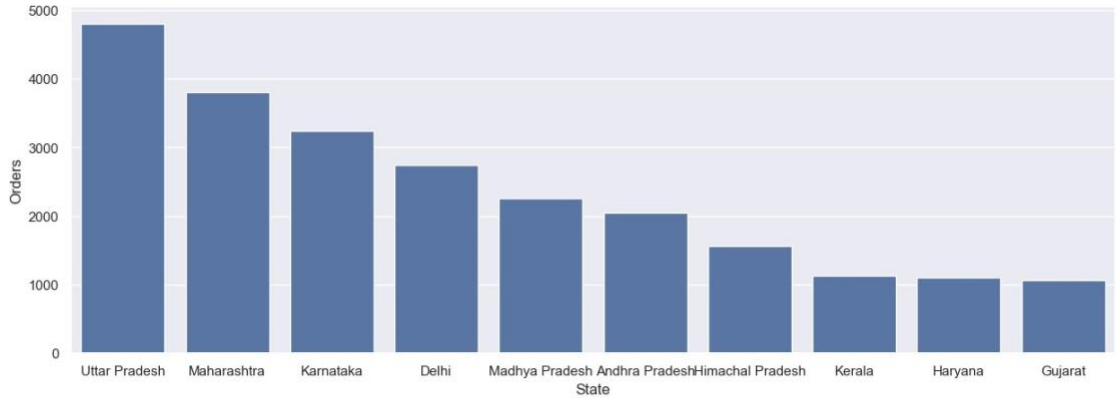


From above graphs we can see that most of the buyers are of age group between 26-35 yrs female

1.0.3 State

[63]:
```
# 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':(15,5)})
sns.barplot(data = sales_state, x = 'State',y= 'Orders')
```

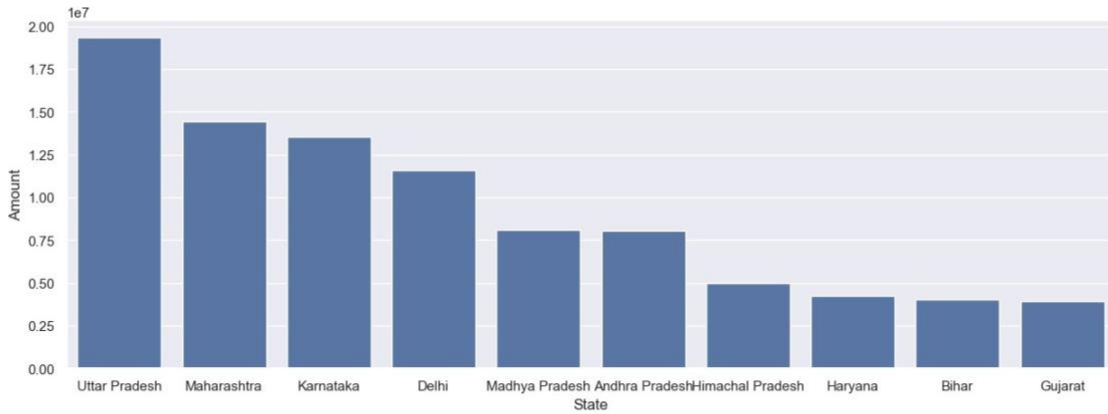
[63]: <Axes: xlabel='State', ylabel='Orders'>



[64]:
```
# 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':(15,5)})
```

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

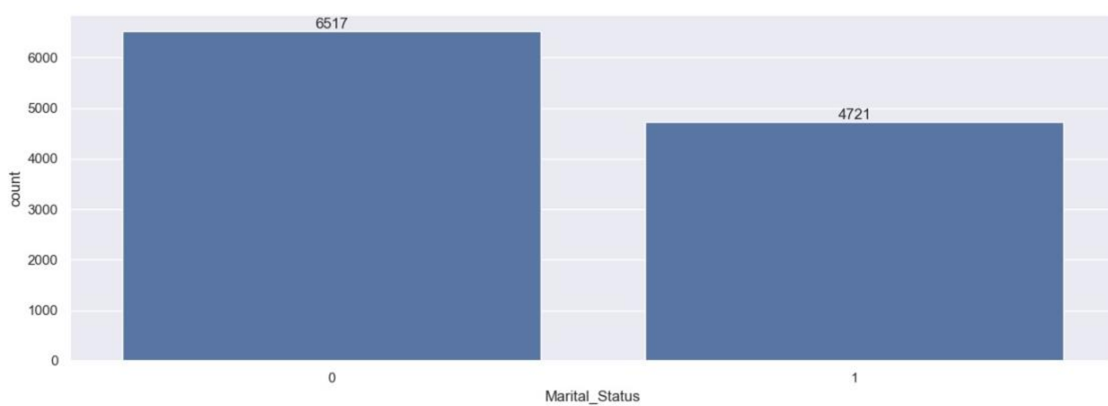
```
[64]: <Axes: xlabel='State', ylabel='Amount'>
```



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

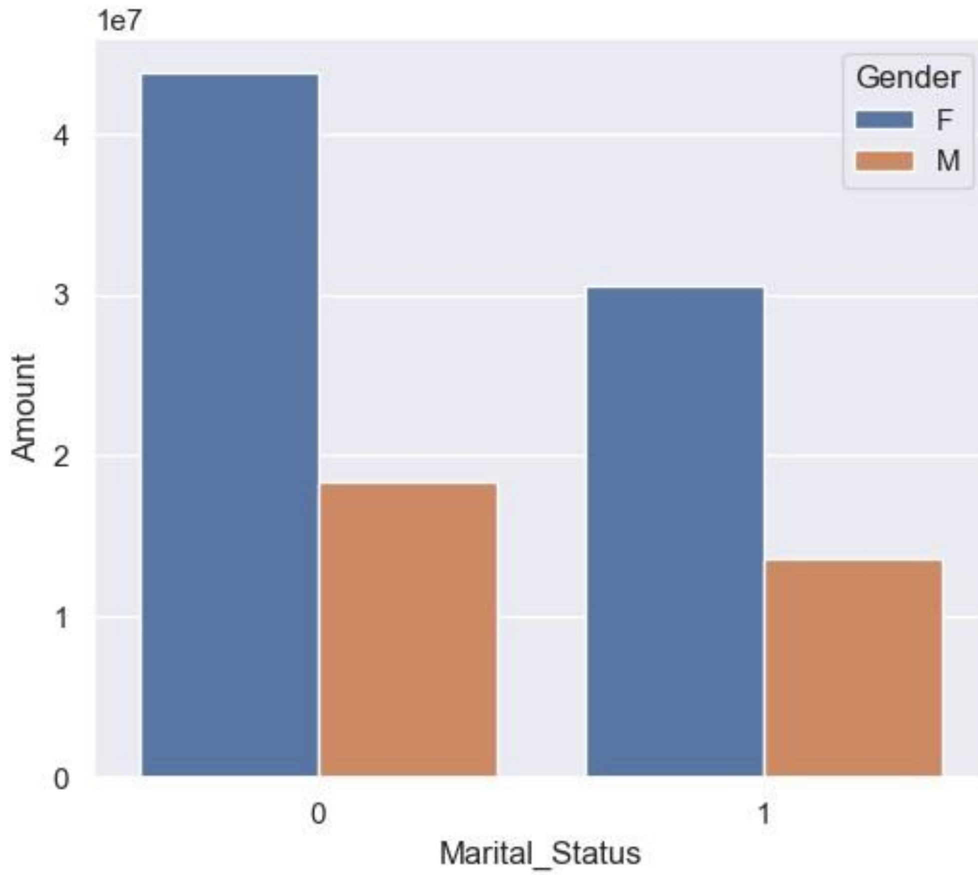
1.0.4 Marital Status

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

sns.set(rc={'figure.figsize':(7,5)})
for bars in ax.containers:
    ax.bar_label(bars)
```




```
[66]: sales_state = df.groupby(['Marital_Status', 'Gender'],␣
  ↪as_index=False)['Amount'].sum().sort_values(by='Amount',
  ascending=False)

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

[66]: <Axes: xlabel='Marital_Status', ylabel='Amount'>

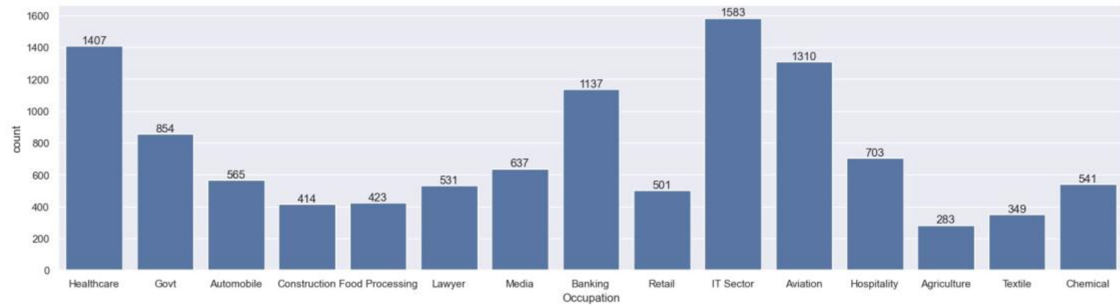


From above graphs we can see that most of the buyers are married (women) and they have high purchasing power

1.0.5 Occupation

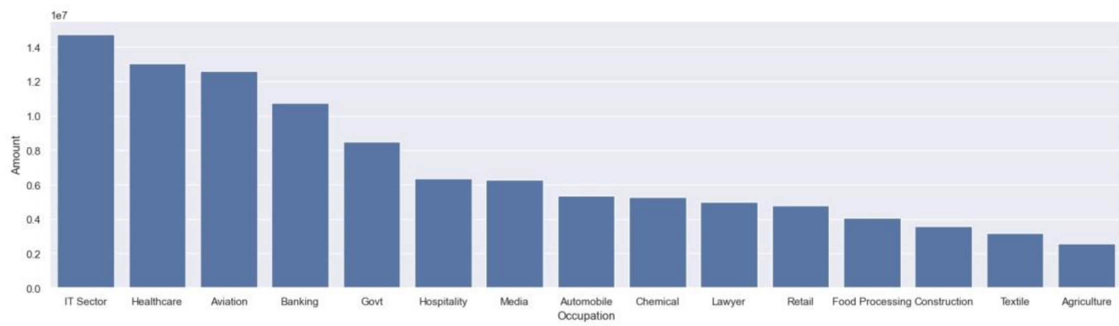
[67]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Occupation')

for bars in ax.containers:
```

```
    ax.bar_label(bars)
```



[68]:
```
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')
```

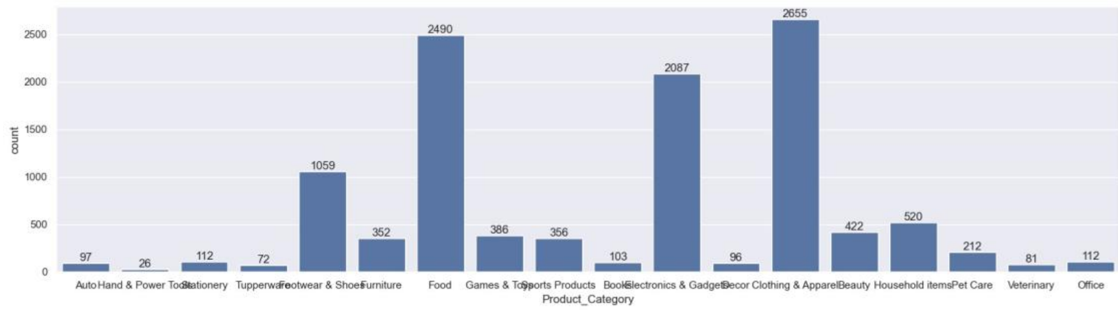
[68]: <Axes: xlabel='Occupation', ylabel='Amount'>



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

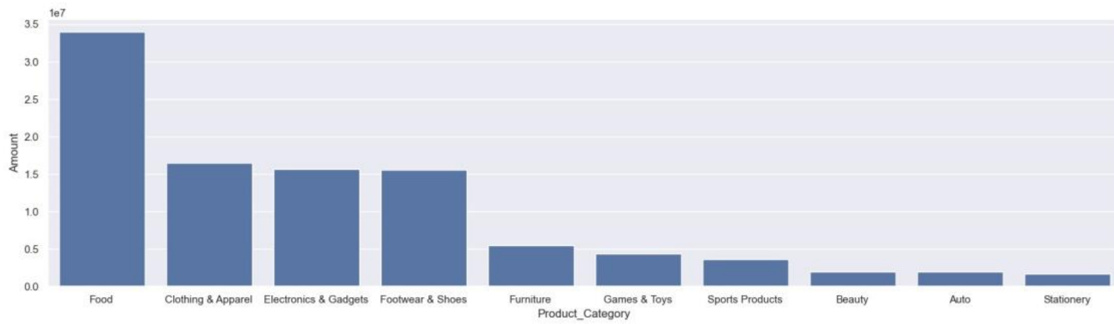
1.0.6 Product Category

[69]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = sns.countplot(data = df, x = 'Product_Category')

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




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

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

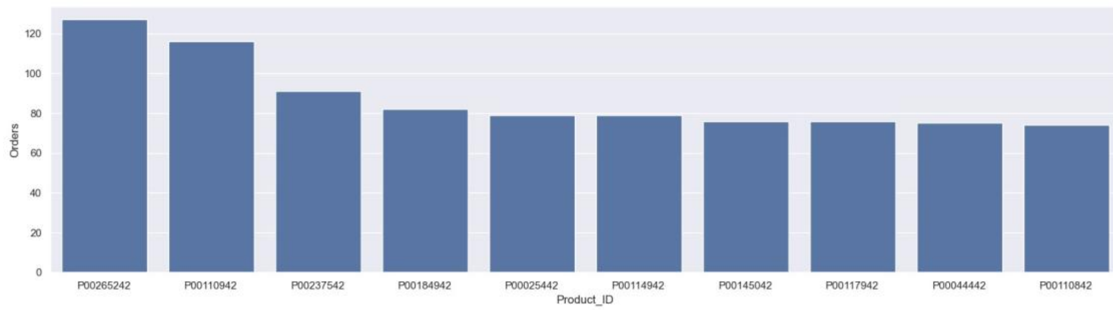
```
[70]: <Axes: xlabel='Product_Category', ylabel='Amount'>
```



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

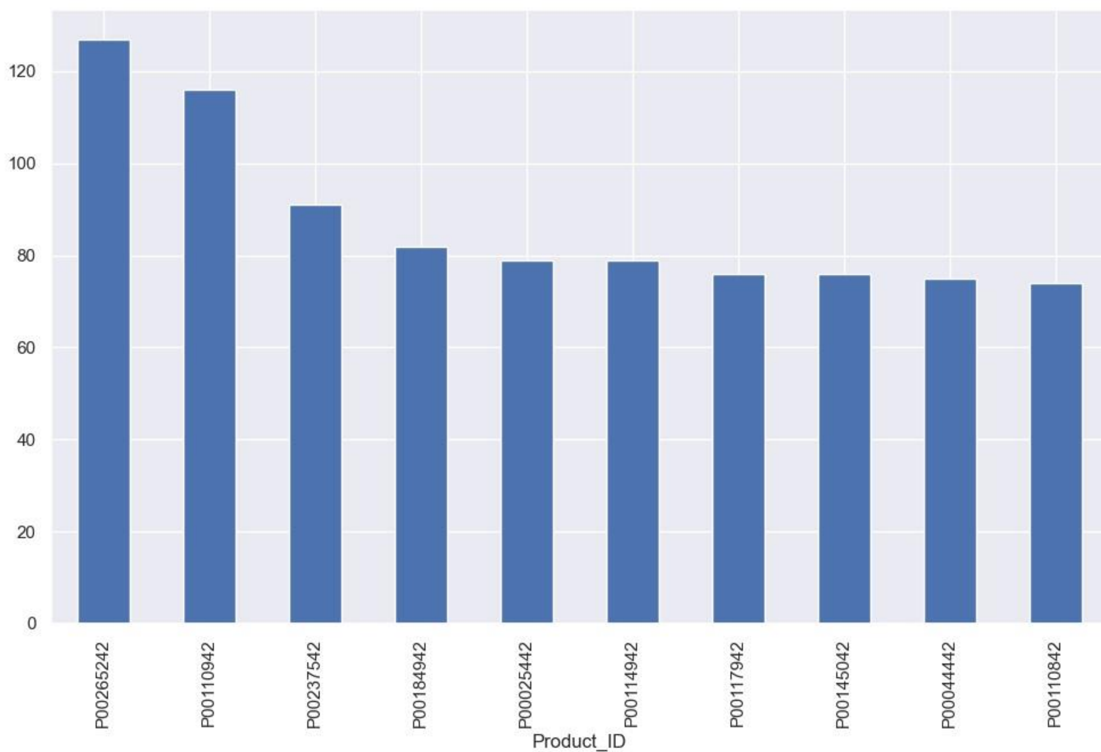
```
[71]: sales_state = df.groupby(['Product_ID'],
       as_index=False)['Orders'].sum(). ↪sort_values(by='Orders',
       ascending=False).head(10)

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

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



[72]:
```
# 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')
```

[72]: <Axes: xlabel='Product_ID'>



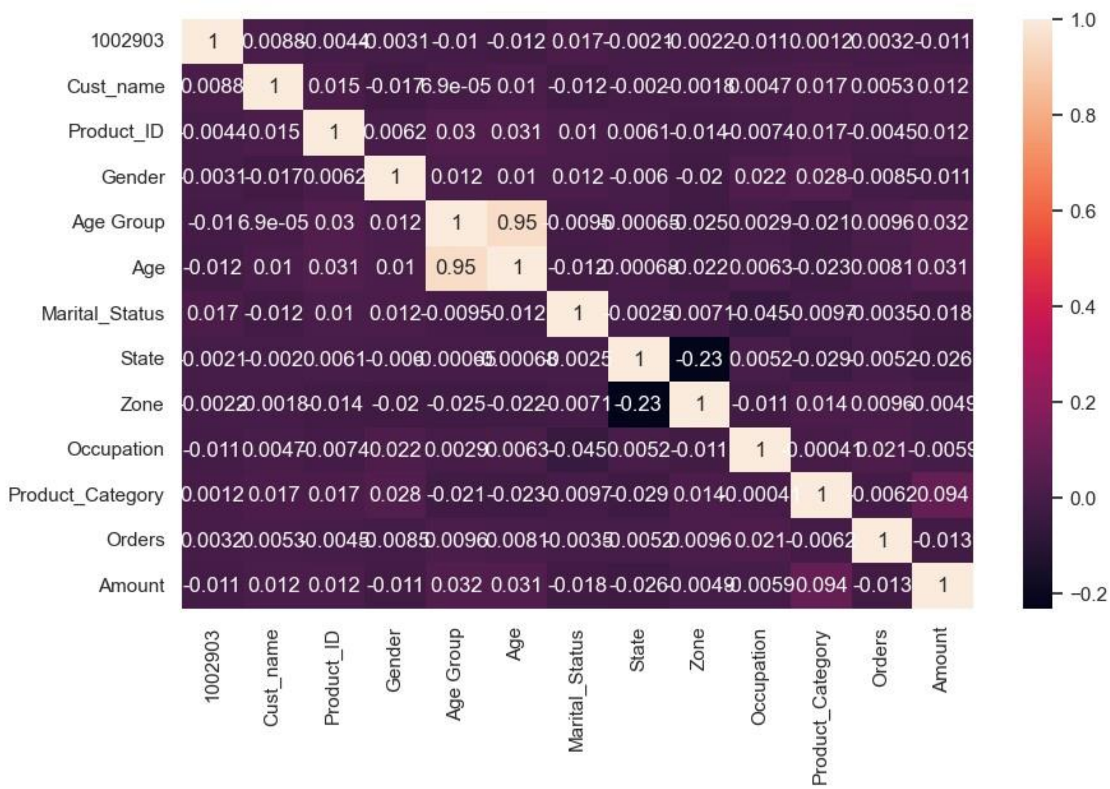
Encoding Categorical Data

[73]:
```
from sklearn .preprocessing import LabelEncoder
le = LabelEncoder()
for col in df.select_dtypes(include='object'):
    df[col] = le.fit_transform(df[col])

df.head()
```

[73]:
```
   1002903 Cust_name Product_ID Gender Age Group Age Marital_Status \
0  1000732.0      991   774   0     2 28  0
1  1001990.0      596   644   0     2 35  1
2  1001425.0      150   714   0     2 35  1
3  1000588.0      1102  1507  1     0 16  0
4  1000588.0      574   387   1     2 28  1

   State  Zone  Occupation  Product_Category Orders Amount
0      10    4      8      0     1  23952
1      0     3      7      0     3  23934
2      14    0      1      0     3  23924
3      7     3      5      0     2  23912
4      3     4      6      0     2  23877
```

Feature ENGINEERING

[74]:
```
# Example: If Qty column exists
# df['Total'] = df['Amount'] * df['Qty']

corr = df.corr()
plt.figure(figsize=(10,6))
sns.heatmap(corr, annot=True)
plt.show()
```



Machine Learning Predict Sales

[75]:
```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score

# Split data
x = df.drop("Amount", axis=1)
y = df["Amount"]

x_train, x_test, y_train, y_test  = train_test_split(x, y, test_size=0.2,
 ↪random_state=42)

# Train model
model = LinearRegression()
model.fit(x_train, y_train)

# Predict
pred = model.predict(x_test)

# Evaluate
print("MAE:", mean_absolute_error(y_test, pred))
```

```
print("R2 Score:", r2_score(y_test, pred))
```

```
MAE: 4200.949470638929
R2 Score: 0.005413966964511l3
```

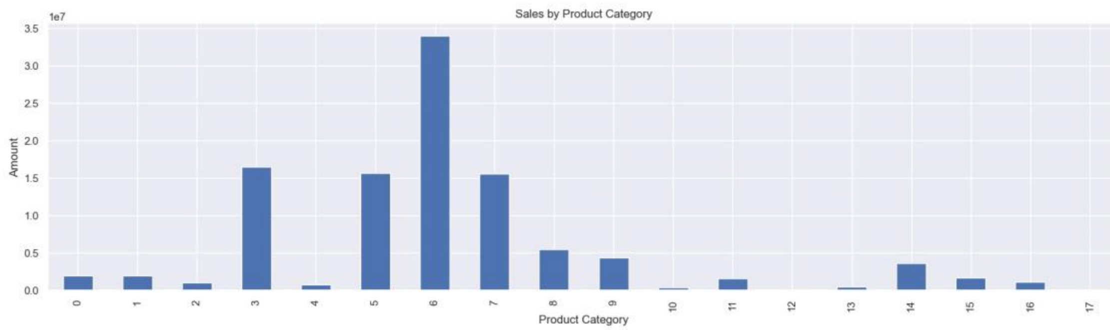
CUSTOMER SEGMENTATION (K-MEANS)

[78]:
```
# Import required libraries
from sklearn.cluster import KMeans
import seaborn as sns
import matplotlib.pyplot as plt

# Select variables for clustering
seg_data = df[['Amount', 'Age']]

# Create KMeans model
kmeans = KMeans(n_clusters=4, random_state=42)

# Fit and predict clusters
df['Cluster'] = kmeans.fit_predict(seg_data)

# View first few rows
print(df[['Amount', 'Age', 'Cluster']].head())

# Visualize clusters
sns.scatterplot(x='Age',  ='Amount', hue='Cluster', data=df, palette='Set2')
plt.title("Customer Segmentation")
plt.show()
```

	Amount	Age	Cluster
0	23952	28	2
1	23934	35	2
2	23924	35	2
3	23912	16	2
4	23877	28	2

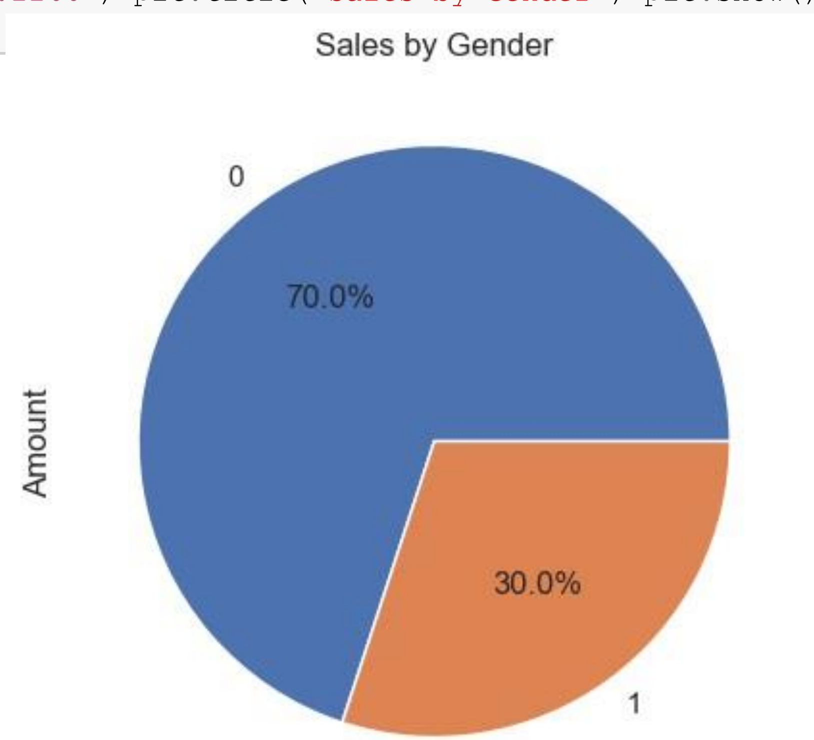
Total Sales by Product Category

[88]:
```python
import matplotlib.pyplot as plt

df.groupby("Product_Category")["Amount"].sum().plot(kind="bar")
plt.title("Sales by Product Category")
plt.xlabel("Product Category")
plt.ylabel("Amount")
plt.show()
```



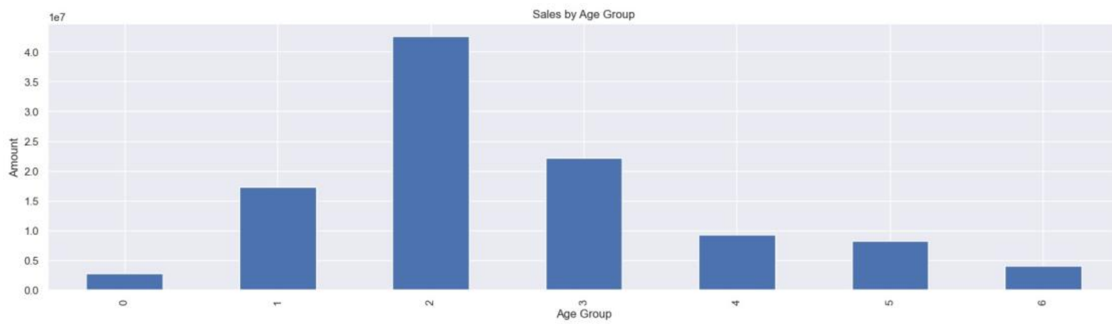
Sales by Gender

[89]:
```python
            df.groupby("Gender")["Amount"].sum().plot(kind="pie",
autopct="%1.1f%%") plt.title("Sales by Gender") plt.show()
```



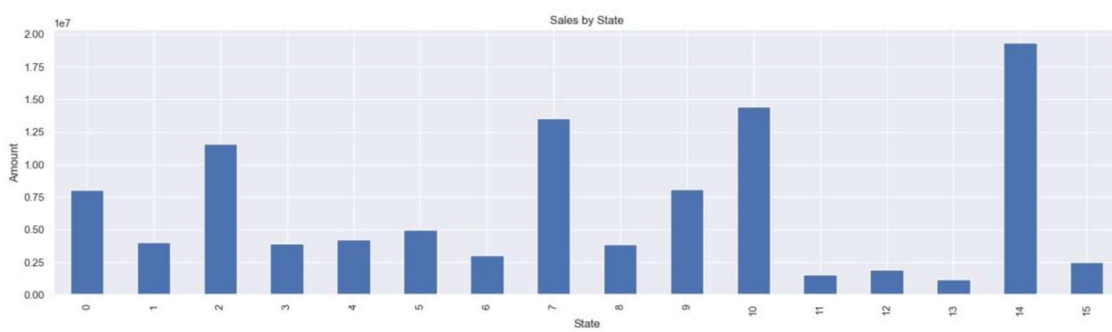
Sales by Age Group

[90]:
```
df.groupby("Age Group")["Amount"].sum().plot(kind="bar")
plt.title("Sales by Age Group")
plt.xlabel("Age Group")
plt.ylabel("Amount")
plt.show()
```



Sales by State

[91]:
```
df.groupby("State")["Amount"].sum().plot(kind="bar")
plt.title("Sales by State")
plt.xlabel("State")
plt.ylabel("Amount")
plt.show()
```



1.1 Conclusion:

1.1.1

Females contributed the highest to overall sales → Women purchased more frequently and spent more money compared to men. Targeting female customers can increase future sales.

2 Most buyers were from the age group 26–35 years → This group is the most active in festive shopping. Marketing campaigns should focus on young working professionals.

3 Maharashtra, Karnataka & Uttar Pradesh generated maximum revenue → These states show strong purchase behavior. Region-based offers & ad campaigns can further increase sales.

4 Top selling product categories were Clothing, Electronics & Household items → Most customers prefer spending on lifestyle & home improvement items during Diwali. These segments should be prioritized for inventory & promotion.

5 High-value customers contributed significantly to sales → Focusing on repeat premium buyers can increase retention & overall revenue. Overall Summary Diwali festival has a strong positive influence on sales. Female customers aged 26–35, especially from Maharashtra, Karnataka & UP, majorly drive the revenue. Clothing, Electronics & Household products remain the most profitable categories.

Thank you!