```
# import python libraries

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

```
# import csv file
df = pd.read_csv('Diwali Sales Data.csv', encoding= 'unicode_escape')
```

```
df.shape #rows and column
```

```
(11251, 15)
```

```
df.head() #top 5 rows
```

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

Next steps: Generate code with df View recommended plots

Data Cleaning

```
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
```

```
#drop unrelated/blank columns
df.drop(['Status', 'unnamed1'], axis=1, inplace=True) #axis 1 = row and axis 0 = column
```

```
#check for null values
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
```

```
# drop/delete null values
df.dropna(inplace=True) # inplace = saves file change
```

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

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

```
dtype('int64')
```

```
df.columns
```

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

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

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

```
# describe() method returns mathmatical description of the data in the DataFrame (i.e. count, mean, std, etc)
df.describe()
```

	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

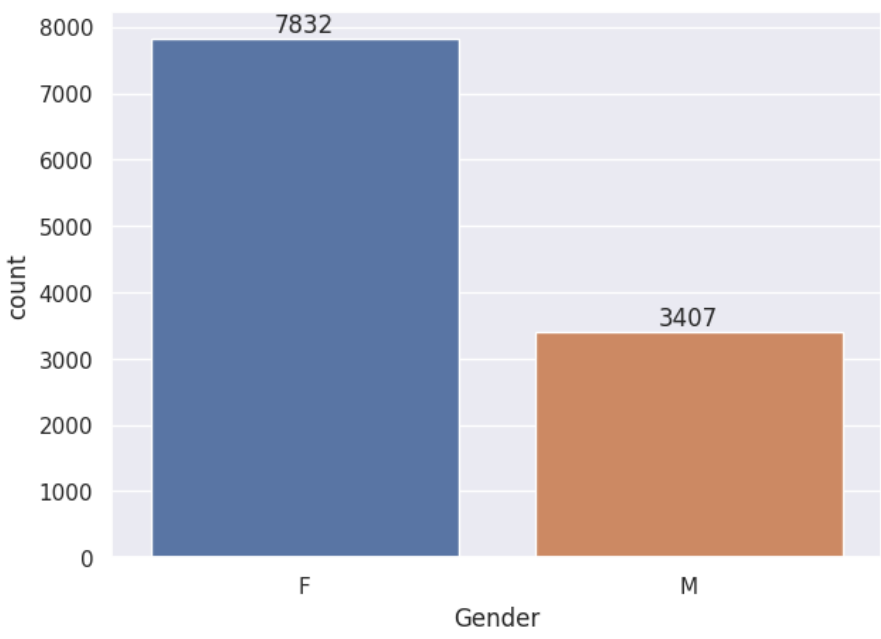
```
# use describe() for specific columns
df[['Age', 'Orders', 'Amount']].describe()
```

	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

Exploratory Data Analysis

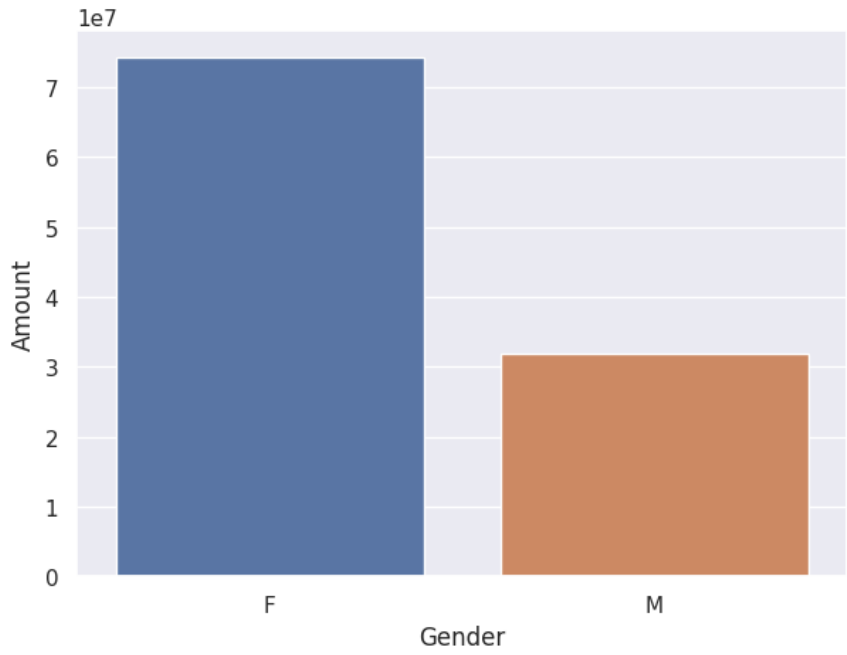
GENDER COLUMN

```
# plotting a bar chart for Gender and it's count

ax = sns.countplot(x = 'Gender',data = df,hue = 'Gender')

for bars in ax.containers: #for values 7832 F and 3407 M
    ax.bar_label(bars)
```



We see female counts are 7832 and male counts are 3407

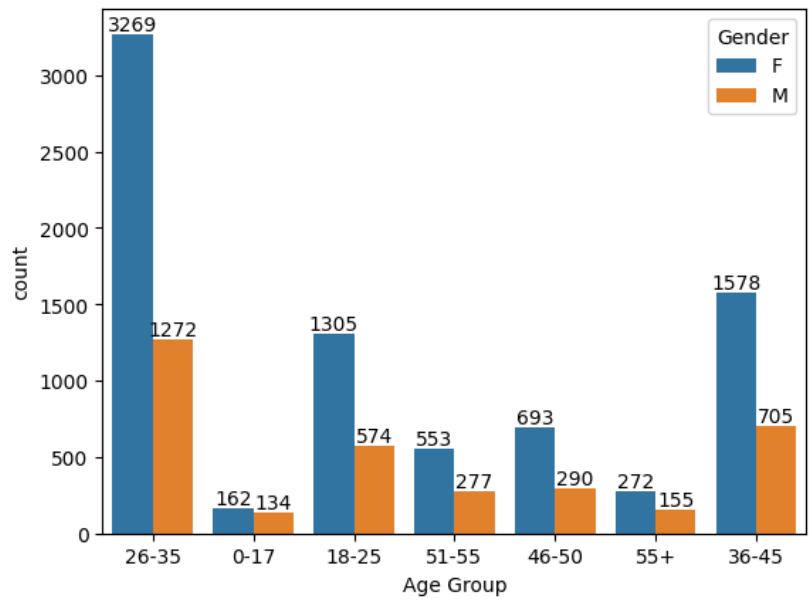
```
# plotting a bar chart for gender vs total amount

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,hue = 'Gender')
sns.set(rc={'figure.figsize':(8,3)})
```



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

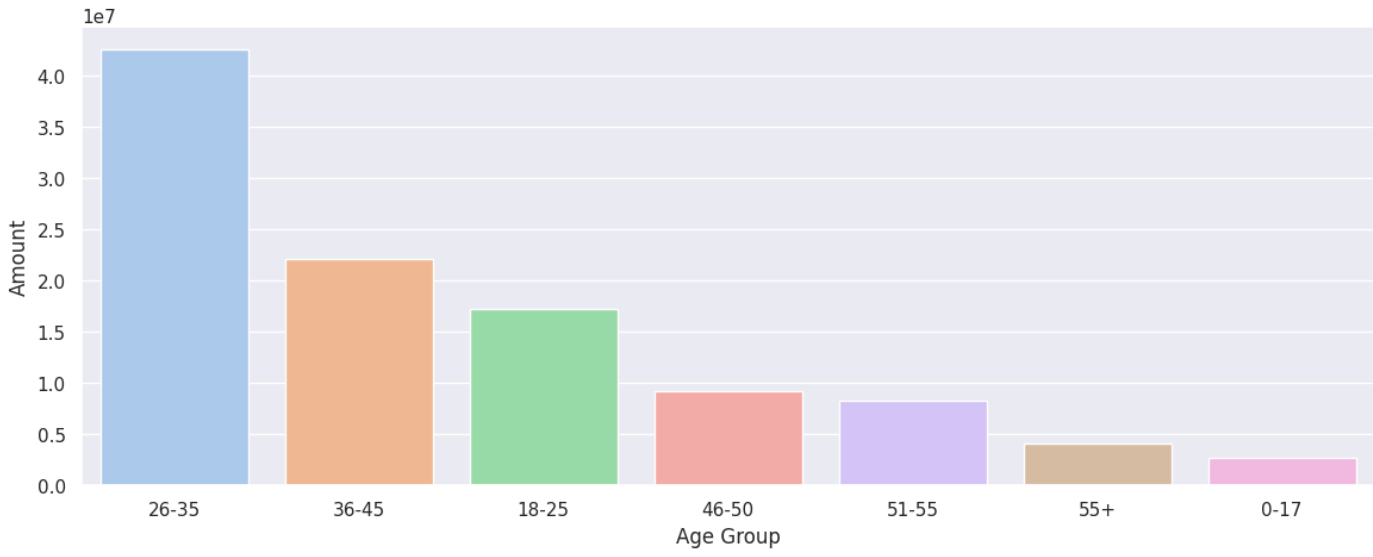
AGE COLUMN

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

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




```
# Total Amount vs Age Group

sales_age = df.groupby(['Age Group'], as_index=False)['Amount'].sum().sort_values(by='Amount', ascending=False)

custom_palette = sns.color_palette("pastel")
sns.barplot(x='Age Group', y='Amount', data=sales_age, palette=custom_palette)

plt.show()
```

```
<ipython-input-88-5e64bfcc14d0>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h

  sns.barplot(x='Age Group', y='Amount', data=sales_age, palette=custom_palette)
<ipython-input-88-5e64bfcc14d0>:6: UserWarning: The palette list has more values (10) than needed (7), which may not
  sns.barplot(x='Age Group', y='Amount', data=sales_age, palette=custom_palette)
```

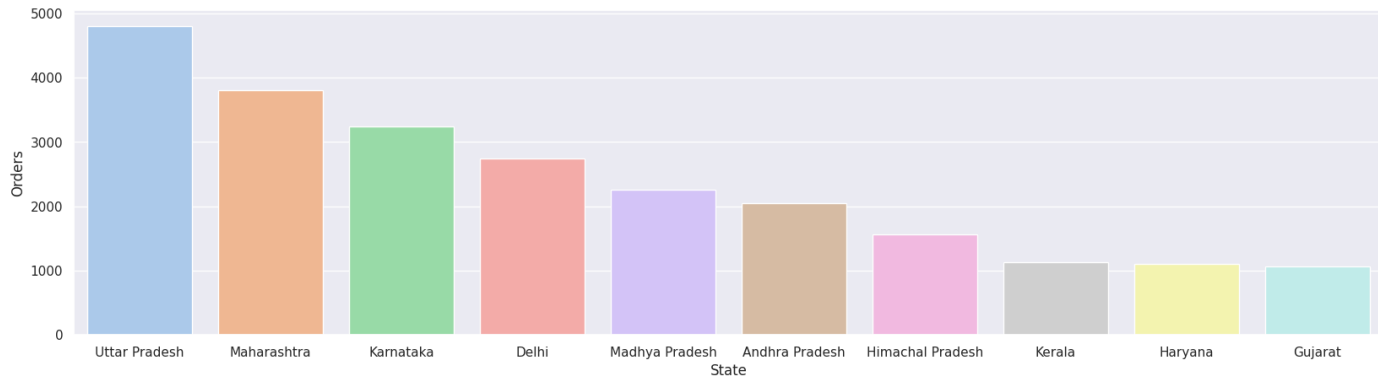


From above graphs we can see that most of the buyers are of age group between 26-35 yrs female

STATE COLUMN

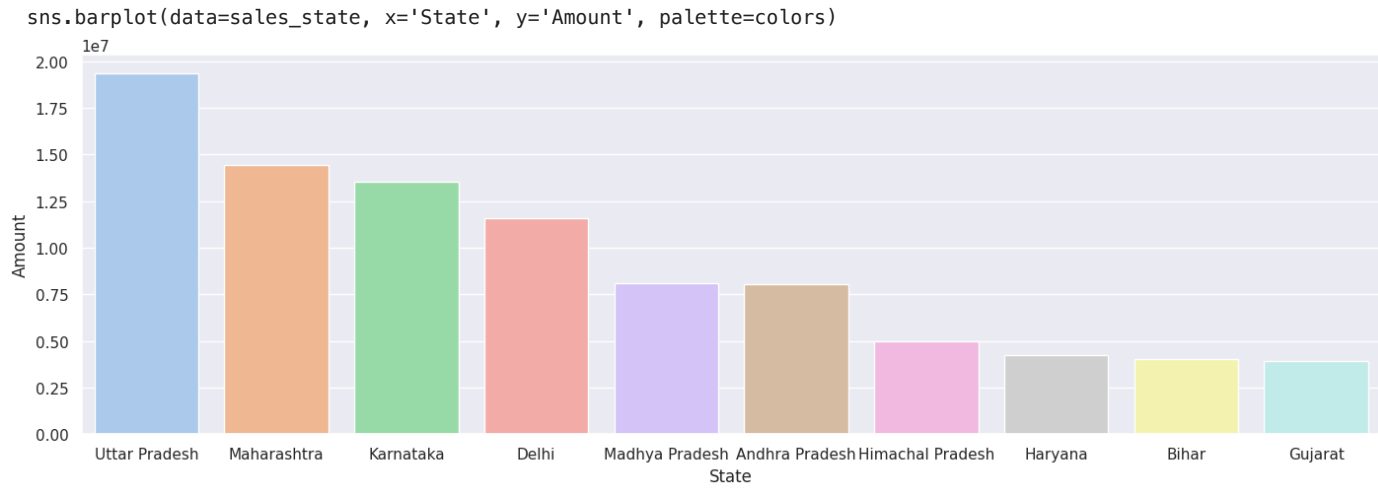
```
# total number of orders from top 10 states

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

colors = sns.color_palette("pastel", len(sales_state))

sns.set(rc={'figure.figsize':(20,5)})
sns.barplot(data=sales_state, x='State', y='Orders', palette=colors)
plt.show()
```

```
<ipython-input-91-db9c287ceec0>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h

  sns.barplot(data=sales_state, x='State', y='Orders', palette=colors)
```




```
# 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)
colors = sns.color_palette("pastel", len(sales_state))

sns.set(rc={'figure.figsize':(17,5)})
sns.barplot(data=sales_state, x='State', y='Amount', palette=colors)

plt.show()
```

```
<ipython-input-93-1c96ae6d1788>:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h

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

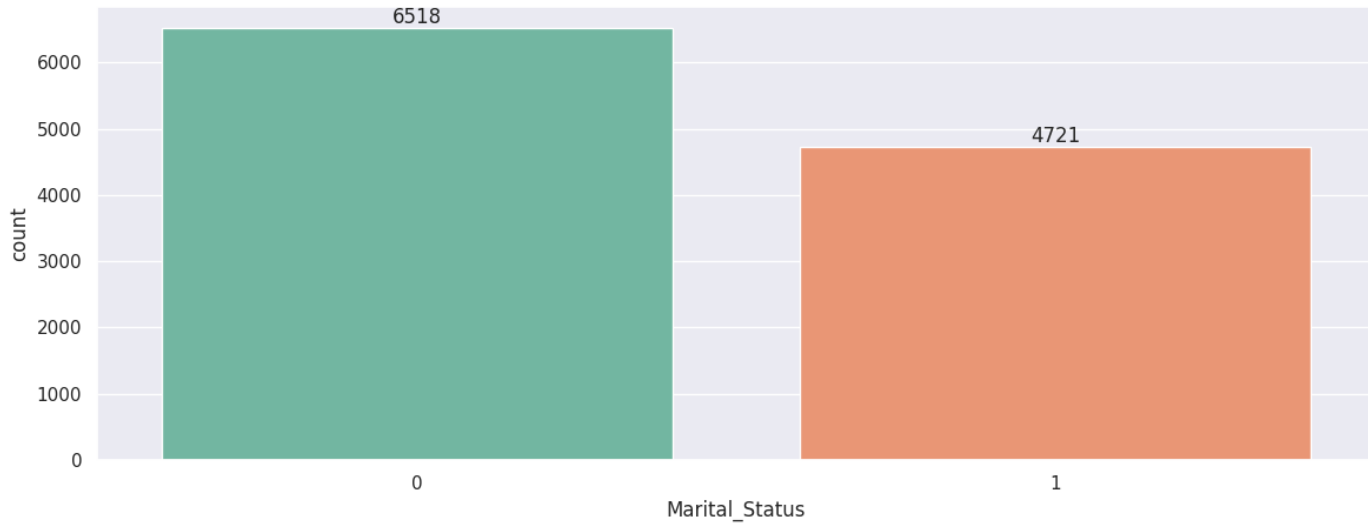


From above graphs we can see that most of the orders & total sales/amount are from Uttar Pradesh, Maharashtra and Karnataka respectively

MARITAL STATUS COLUMN

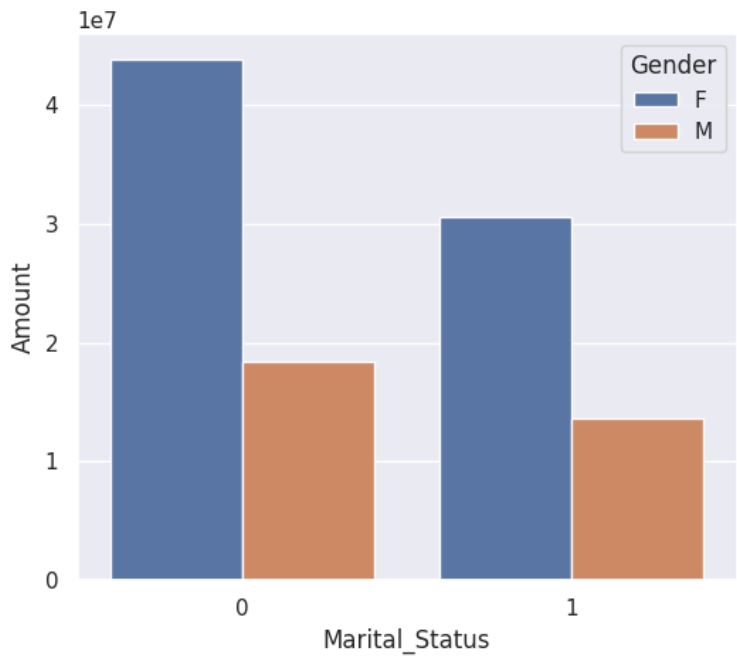
```
sns.set(rc={'figure.figsize':(14,5)})
ax = sns.countplot(data=df, x='Marital_Status', palette='Set2')

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

```
<ipython-input-94-a13a06137573>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h

  ax = sns.countplot(data=df, x='Marital_Status', palette='Set2')
```




```
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')
```

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

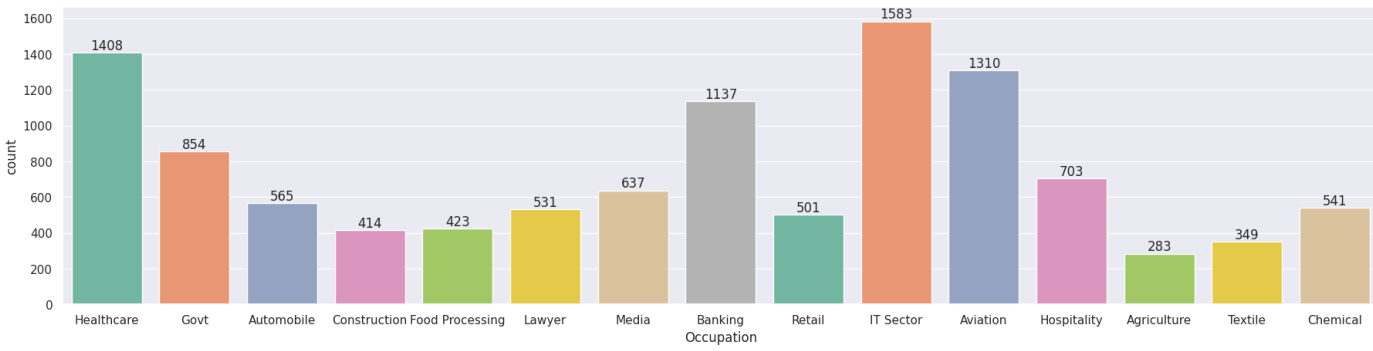


From above graphs we can see that most of the buyers are married (women) and they have high purchasing power

OCCUPATION COLUMN

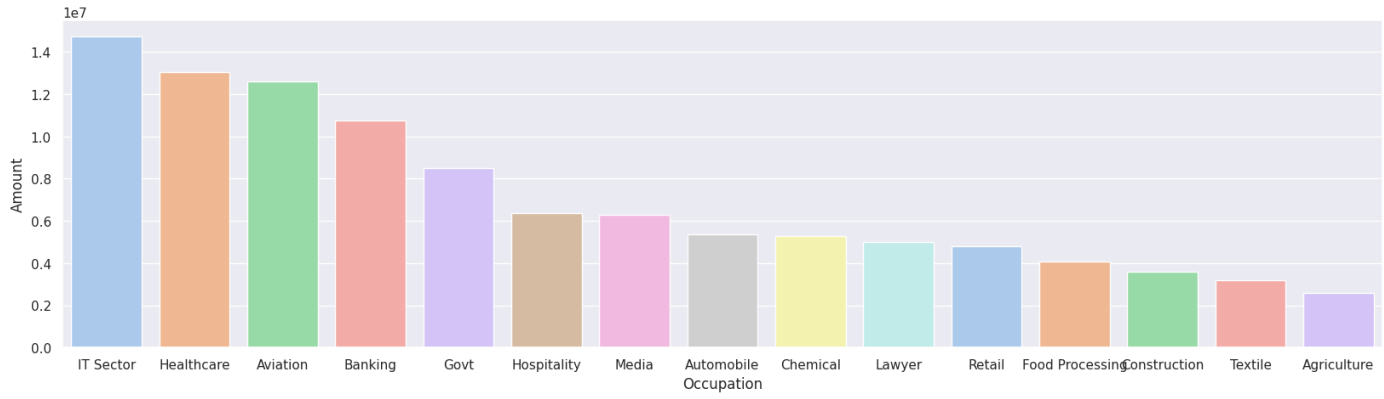
```
sns.set(rc={'figure.figsize':(22,5)})
ax = sns.countplot(data=df, x='Occupation', palette='Set2')  # You can change 'Set2' to any other palette

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

```
<ipython-input-95-a049eb38a316>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h

  ax = sns.countplot(data=df, x='Occupation', palette='Set2')  # You can change 'Set2' to any other palette
```




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

custom_palette = sns.color_palette("pastel")

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

plt.show()
```

```
<ipython-input-96-03f521e33b78>:6: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `l

  sns.barplot(data=sales_state, x='Occupation', y='Amount', palette=custom_palette)
<ipython-input-96-03f521e33b78>:6: UserWarning:
The palette list has fewer values (10) than needed (15) and will cycle, which may produce an uninterpretable plot.
  sns.barplot(data=sales_state, x='Occupation', y='Amount', palette=custom_palette)
```

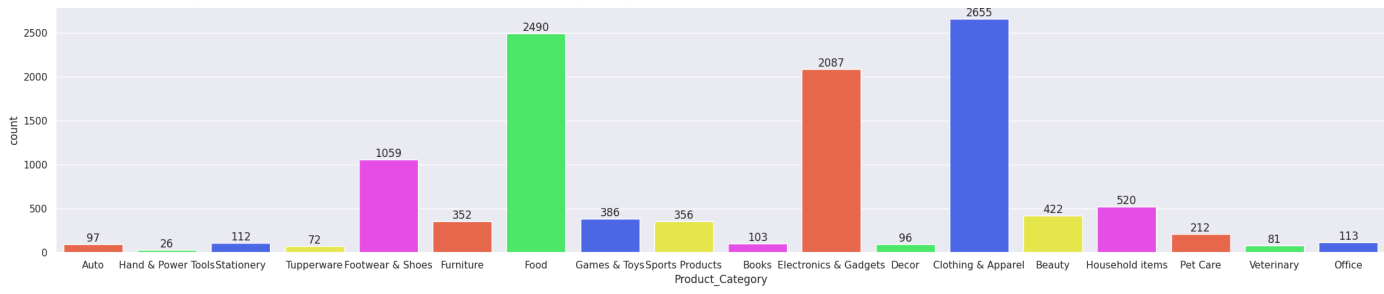


From above graphs we can see that most of the buyers are working in IT, Healthcare and Aviation sector

PRODUCT CATEGORY COLUMN

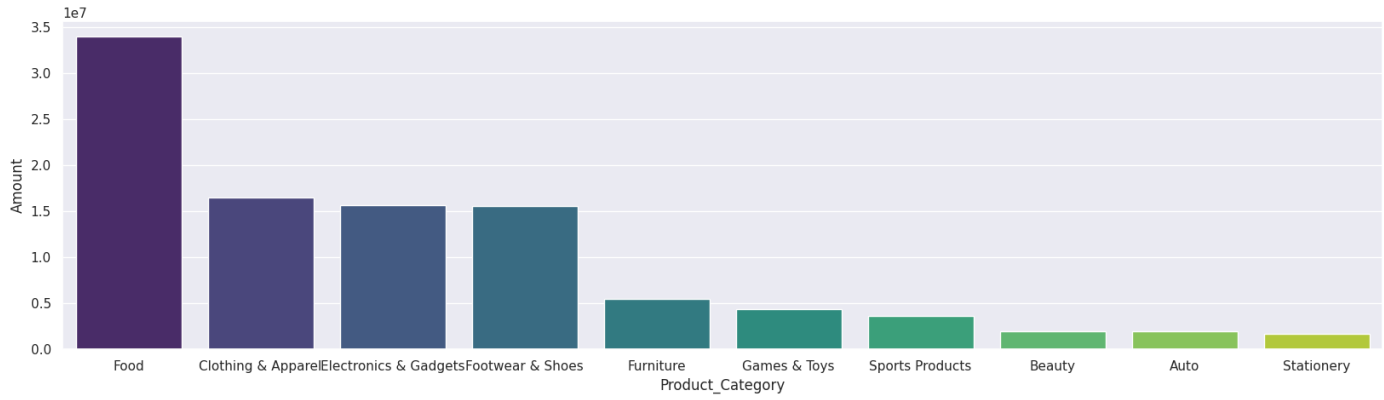
```
sns.set(rc={'figure.figsize':(27,5)})

# Define the color palette
colors = ["#FF5733", "#33FF57", "#3357FF", "#FFFF33", "#FF33FF"]

# Create the countplot with specified palette
ax = sns.countplot(data=df, x='Product_Category', palette=colors)

# Add labels to the bars
for bars in ax.containers:
    ax.bar_label(bars)
```

```
<ipython-input-97-33def8e319c7>:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `l

  ax = sns.countplot(data=df, x='Product_Category', palette=colors)
<ipython-input-97-33def8e319c7>:7: UserWarning:
The palette list has fewer values (5) than needed (18) and will cycle, which may produce an uninterpretable plot.
  ax = sns.countplot(data=df, x='Product_Category', palette=colors)
```




```
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', palette='viridis')  # You can choose any palette you like
plt.show()
```

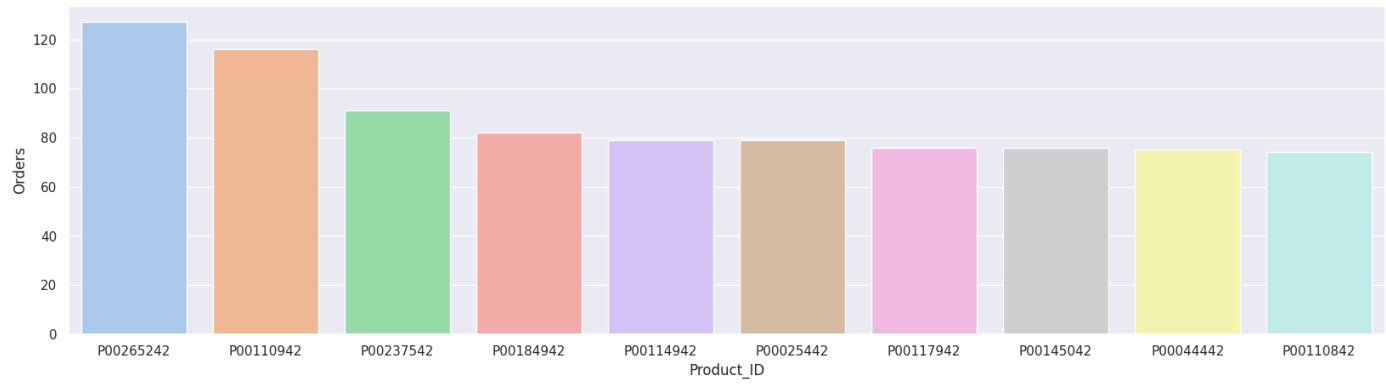
```
<ipython-input-98-251d052c4061>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `l

  sns.barplot(data=sales_state, x='Product_Category', y='Amount', palette='viridis')  # You can choose any palette y
```



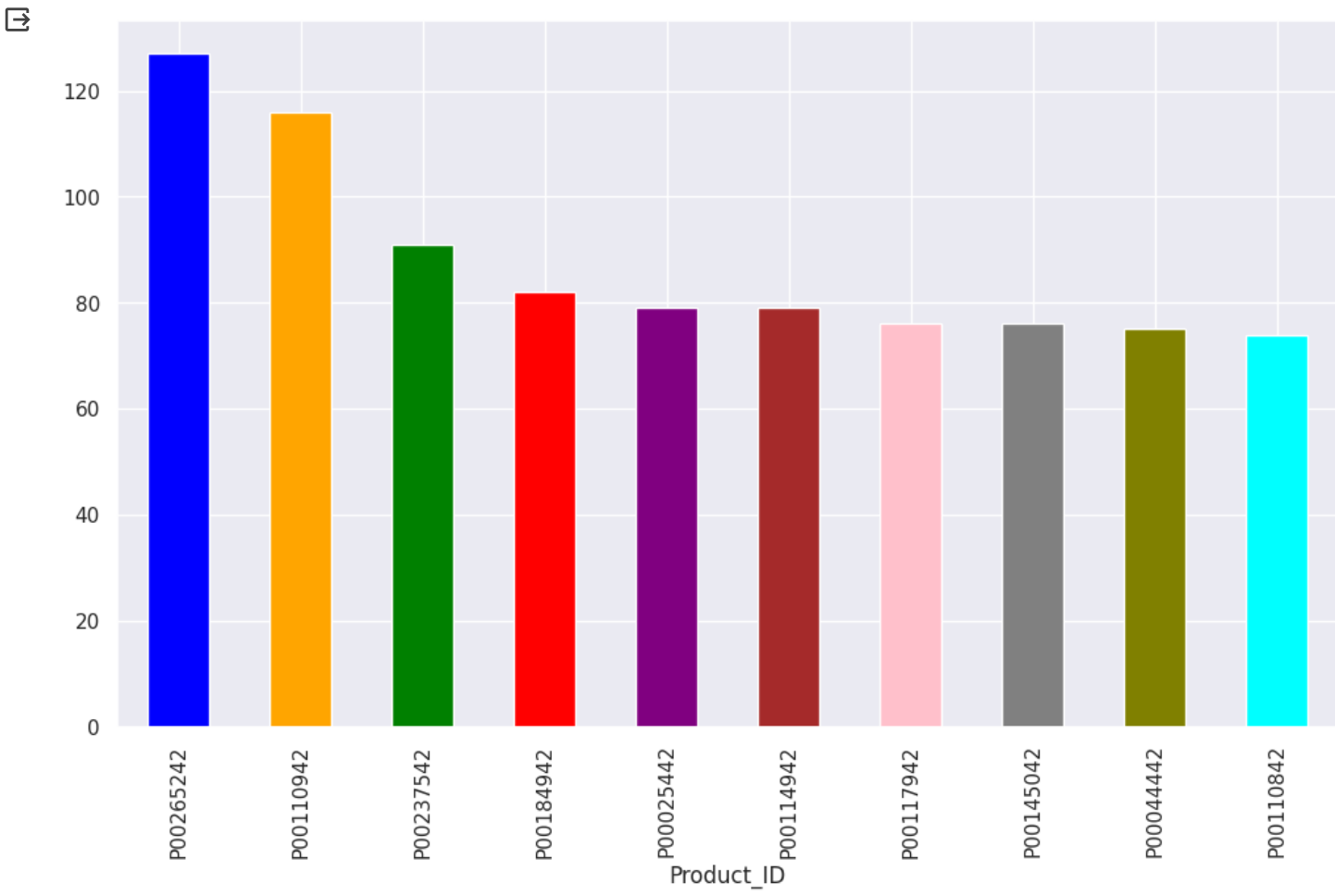
From above graphs we can see that most of the sold products are from Food, Clothing and Electronics category

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

colors = sns.color_palette('pastel')  # You can choose any Seaborn palette or specify custom colors

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

```
<ipython-input-99-445fc714b8c8>:8: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `h

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




```
fig1, ax1 = plt.subplots(figsize=(12,7))

# Getting the top 10 most sold products and their total orders
top_products = df.groupby('Product_ID')['Orders'].sum().nlargest(10).sort_values(ascending=False)

# Defining different colors for each bar
colors = ['blue', 'orange', 'green', 'red', 'purple', 'brown', 'pink', 'gray', 'olive', 'cyan']

# Plotting the top 10 most sold products
top_products.plot(kind='bar', color=colors)

plt.show()
```



CONCLUSION OF THE PROJECT :

According to the analyzed data it can be determined that Female individuals who are married and aged between 26 and 35 years and who work within the Information Technology, Healthcare, and Aviation sectors in the states of Uttar Pradesh, Maharashtra, and Karnataka, have a higher propensity to purchase products from the categories of Food, Clothing, and Electronics.

PROJECT LEARNING :

Executed data cleaning and manipulation procedures.

Conducted exploratory data analysis (EDA) utilizing the pandas, matplotlib, and seaborn libraries.

Enhanced the customer experience by identifying prospective customers among diverse states, occupations, genders, and age groups.

Boosted sales by identifying the highest selling product categories and products,