```python
import pandas as pd
import numpy as np
from scipy import stats
from sklearn.preprocessing import MinMaxScaler, OneHotEncoder
from sklearn.feature_selection import VarianceThreshold

df = pd.read_csv('/content/Cars_India_dataset.csv')

df.head()
```

{"type":"dataframe","variable_name":"df"}

STEP-2:

DATA CLEANING AND PRE-PROCESSING TASK

```python
import pandas as pd

# Data Cleaning Tasks

# 1. Handling Missing Values
# Check for missing values
missing_values = df.isnull().sum()
print("Missing Values:")
print(missing_values)


# Drop rows with missing values in specified columns
columns_to_check = ['Displacement cc', 'Emission Type', 'Fuel Tank Capacity','Transmission','No_of_Cylinders','Maker']
df.dropna(subset=columns_to_check, inplace=True)

# Drop the specified columns
columns_to_drop = ['Turning Radius', 'Boot Space','Model']
df.drop(columns=columns_to_drop, inplace=True)

# Confirm the changes
print(df.head())
missing_values = df.isnull().sum()

print(missing_values)
```

```
Missing Values:
ID                      0
Model                   0
Maker                   0
Type                    0
Seats                   0
Displacement cc         1
Length mm               0
Width                   0
```

```
Height                   0
Wheelbase                0
No_of_Cylinders          1
Fuel                     0
Engine Type              0
Transmission             1
Front Brake              0
Rear Brake               0
Drive                    0
Turning Radius          19
Fuel Tank Capacity       1
Boot Space              15
Fuel Efficiency          0
Emission Type            1
Tyre Size                0
Variants                 0
NCAP Rating              0
dtype: int64
   ID       Maker            Type  Seats  Displacement cc  Length mm
Width  \
0  24      Nissan     Sports Car      2           3799.0       4710
1895
1  40    Mahindra             SUV      6           2000.0       4662
1917
2   2  Volkswagen           Sedan      5           1498.0       4561
1752
3   9       Honda  Compact Sedan      5           1199.0       3995
1695
4   8       Honda  Compact Sedan      5           1199.0       3995
1695

   Height  Wheelbase  No_of_Cylinders  ... Transmission Front Brake  \
0    1370       2780              6.0  ...  6-Speed DCT        Disc
1    1857       2750              4.0  ...         6 MT        Disc
2    1507       2651              4.0  ...  7-Speed DSG        Disc
3    1501       2470              4.0  ...          CVT        Disc
4    1501       2470              4.0  ...         5 MT        Disc

  Rear Brake Drive Fuel Tank Capacity Fuel Efficiency  Emission
Type  \
0       Disc  4 WD               74.0           10.16         Euro 6

1       Disc   2WD               57.0           18.60          BS VI

2       Drum   2WD               45.0           18.67          BS VI

3       Drum   2WD               35.0           18.30          BS VI

4       Drum   2WD               35.0           18.60          BS VI
```

```
    Tyre Size Variants NCAP Rating
0   285/35/20        1  Not Tested
1  255/16 R18        1   Not Rated
2  205/55 R16        1  Not Tested
3  175/65 R15        2           4
4  175/65 R15        3           4

[5 rows x 22 columns]
ID                     0
Maker                  0
Type                   0
Seats                  0
Displacement cc        1
Length mm              0
Width                  0
Height                 0
Wheelbase              0
No_of_Cylinders        1
Fuel                   0
Engine Type            0
Transmission           1
Front Brake            0
Rear Brake             0
Drive                  0
Fuel Tank Capacity     1
Fuel Efficiency        0
Emission Type          1
Tyre Size              0
Variants               0
NCAP Rating            0
dtype: int64
```

```
df.head()
```

{"type":"dataframe","variable_name":"df"}

```
data = df
# 2. Checking for Duplicates
duplicate_rows = data.duplicated()
print("\nDuplicate Rows:")
print(df[duplicate_rows])
# df = data.drop_duplicates()
```

```
Duplicate Rows:
Empty DataFrame
Columns: [ID, Maker, Type, Seats, Displacement cc, Length mm, Width,
Height, Wheelbase, No_of_Cylinders, Fuel, Engine Type, Transmission,
Front Brake, Rear Brake, Drive, Fuel Tank Capacity, Fuel Efficiency,
```

```
Emission Type, Tyre Size, Variants, NCAP Rating]
Index: []

[0 rows x 22 columns]
```

```python
# Data Preprocessing Tasks
# 1. Normalization/Scaling
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
data[['Length mm', 'Width', 'Height']] =
scaler.fit_transform(data[['Length mm', 'Width', 'Height']])

data.head()
```

{"type":"dataframe","variable_name":"data"}

```python
# 3. One-Hot Encoding
data = pd.get_dummies(data, columns=['Fuel','Type','Engine
Type','Transmission','Front Brake','Rear Brake','Drive','Tyre
Size','NCAP Rating',"Emission Type"])
```

STEP - 3

REGRESSION

```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.impute import SimpleImputer

# Step 3: Impute missing values
imputer = SimpleImputer()
X_imputed = imputer.fit_transform(X)

# Step 4: Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_imputed, y,
test_size=0.2, random_state=42)

# Step 5: Initialize the Decision Tree Classifier
model = DecisionTreeClassifier(random_state=42)

# Step 6: Train the model on the training data
model.fit(X_train, y_train)

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```python
# Classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
ID                          0
Maker                       0
Seats                       0
Displacement cc             1
Length mm                   0
                           ..
NCAP Rating_6               0
NCAP Rating_Not Rated       0
NCAP Rating_Not Tested      0
Emission Type_BS VI         0
Emission Type_Euro 6        0
Length: 150, dtype: int64
Accuracy: 0.90625

Classification Report:
              precision    recall  f1-score   support

     Citroen       0.00      0.00      0.00         1
       Honda       1.00      0.50      0.67         2
     Hyundai       0.91      0.91      0.91        11
         Kia       1.00      1.00      1.00         6
    Mahindra       0.75      1.00      0.86         3
      Nissan       1.00      1.00      1.00         2
     Renault       1.00      1.00      1.00         1
        Tata       1.00      1.00      1.00         2
      Toyota       1.00      1.00      1.00         3
  Volkswagen       0.50      1.00      0.67         1

    accuracy                           0.91        32
   macro avg       0.82      0.84      0.81        32
weighted avg       0.90      0.91      0.89        32


/usr/local/lib/python3.10/dist-packages/sklearn/metrics/
_classification.py:1344: UndefinedMetricWarning: Precision and F-score
are ill-defined and being set to 0.0 in labels with no predicted
samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classificatio
n.py:1344: UndefinedMetricWarning: Precision and F-score are ill-
defined and being set to 0.0 in labels with no predicted samples. Use
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classificatio
n.py:1344: UndefinedMetricWarning: Precision and F-score are ill-
defined and being set to 0.0 in labels with no predicted samples. Use
```

```
`zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

1.Outlier Detection

```
from scipy.stats import zscore
import pandas as pd
import numpy as np

df=pd.read_csv('Car Sales.xlsx - car_data.csv')
df.head()


# Select numerical features, excluding the target column
numerical_features = df.select_dtypes(include=np.number).columns[:-1]

# Calculate Z-scores for numerical features only
z_scores = np.abs(zscore(df[numerical_features]))

# Define threshold
threshold = 3

# Detect outliers
outliers_zscore = (z_scores > threshold).any(axis=1)

# Display outliers
print("Z-Score Method Outliers:\n", df[outliers_zscore])
```

```
Z-Score Method Outliers:
               Car_id        Date Customer Name  Gender  Annual Income  \
8      C_CND_000009    1/2/2022         Naomi    Male         815000
48     C_CND_000049    1/3/2022     Valentine    Male        4060000
122    C_CND_000123    1/9/2022        Benott    Male        3000000
170    C_CND_000171   1/13/2022        Jordan    Male          13500
269    C_CND_000270   1/27/2022       Destiny  Female         825000
...             ...         ...           ...     ...            ...
23711  C_CND_023712  12/29/2023        Gilian    Male         900000
23762  C_CND_023763  12/29/2023          Loan    Male        1234500
23769  C_CND_023770  12/29/2023        Shyrel    Male        4111000
23809  C_CND_023810  12/29/2023         Louen    Male        2065000
23839  C_CND_023840  12/30/2023         Sofia  Female         555000

                                     Dealer_Name     Company  \
8                           Rabun Used Car Sales   Chevrolet
48                                 Race Car Help      Nissan
122                       Clay Johnson Auto Sales    Cadillac
170                Scrivener Performance Engineering     Lincoln
269                               Pars Auto Sales      Toyota
...                                          ...         ...
23711                                Capitol KIA    Cadillac
23762  Progressive Shippers Cooperative Association No     Lincoln
23769                              Suburban Ford  Volkswagen
23809                                 U-Haul CO     Lincoln
23839                              Suburban Ford    Cadillac

             Model                    Engine Transmission       Color  \
8           Malibu          Overhead Camshaft       Manual  Pale White
48          Altima  DoubleÂ Overhead Camshaft         Auto  Pale White
122       Eldorado  DoubleÂ Overhead Camshaft         Auto  Pale White
170    Continental          Overhead Camshaft       Manual  Pale White
269         Tacoma          Overhead Camshaft       Manual  Pale White
...            ...                        ...          ...         ...
23711     Eldorado          Overhead Camshaft       Manual       Black
23762  Continental          Overhead Camshaft       Manual         Red
23769          GTI  DoubleÂ Overhead Camshaft         Auto  Pale White
23809  Continental          Overhead Camshaft       Manual         Red
23839       Catera  DoubleÂ Overhead Camshaft         Auto  Pale White

       Price ($)  Dealer_No  Body Style    Phone Dealer_Region
8          82000  85257-3102    Hardtop  7194857         Pasco
48         20000  78758-7841  Hatchback  7117432        Austin
122        31000  78758-7841  Passenger  8668755        Austin
170        82000  38701-8047  Passenger  6642461    Greenville
269        82000  38701-8047  Hatchback  7848361    Greenville
...          ...         ...        ...      ...           ...
23711      85000  38701-8047  Passenger  7788669    Greenville
23762      82450  53546-9427  Passenger  7468114        Austin
23769      20100  53546-9427        SUV  8363552        Austin
23809      82500  78758-7841  Passenger  6406323        Aurora
23839      75000  53546-9427  Hatchback  7752902    Janesville

[680 rows x 16 columns]
```

Double-click (or enter) to edit

```
from sklearn.ensemble import IsolationForest
```

```
# Initialize Isolation Forest
iso_forest = IsolationForest(contamination=0.1, random_state=42)

# Select numerical features, excluding the target column and 'Car_id'
numerical_features = df.select_dtypes(include=np.number).columns.difference(['outlier_iso'])

# Fit and predict using only numerical features
df['outlier_iso'] = iso_forest.fit_predict(df[numerical_features])

# Display outliers
print("Isolation Forest Outliers:\n", df[df['outlier_iso'] == -1])
```

```
Isolation Forest Outliers:
             Car_id        Date Customer Name Gender  Annual Income  \
8      C_CND_000009    1/2/2022         Naomi   Male         815000
21     C_CND_000022    1/2/2022        Joshua   Male        2500000
37     C_CND_000038    1/3/2022        Haylee   Male          13500
48     C_CND_000049    1/3/2022     Valentine   Male        4060000
65     C_CND_000066    1/4/2022      Annaelle   Male        1650000
...             ...         ...           ...    ...            ...
23885  C_CND_023886  12/31/2023      Jeremias   Male        1890000
23890  C_CND_023891  12/31/2023       Joaquin   Male        2450000
23891  C_CND_023892  12/31/2023     Annabelle   Male        2340000
23895  C_CND_023896  12/31/2023          Sima   Male         965000
23899  C_CND_023900  12/31/2023          Yuna   Male          13500

                                     Dealer_Name    Company  \
8                          Rabun Used Car Sales  Chevrolet
21                                Classic Chevy   Infiniti
37                     Gartner Buick Hyundai Saab      Buick
48                                Race Car Help     Nissan
65                          Star Enterprises Inc      Buick
...                                          ...        ...
23885  Progressive Shippers Cooperative Association No       Ford
23890                            Saab-Belle Dodge      Dodge
23891                Ryder Truck Rental and Leasing  Chevrolet
23895  Progressive Shippers Cooperative Association No    Mercury
23899                                  U-Haul CO      Buick

             Model                     Engine Transmission       Color  \
8           Malibu           Overhead Camshaft       Manual  Pale White
21             I30  DoubleÂ Overhead Camshaft         Auto       Black
37     Park Avenue  DoubleÂ Overhead Camshaft         Auto       Black
48          Altima  DoubleÂ Overhead Camshaft         Auto  Pale White
65     Park Avenue  DoubleÂ Overhead Camshaft         Auto       Black
...            ...                        ...          ...         ...
23885       Ranger           Overhead Camshaft       Manual       Black
23890   Ram Pickup           Overhead Camshaft       Manual  Pale White
23891     Corvette  DoubleÂ Overhead Camshaft         Auto  Pale White
23895        Sable           Overhead Camshaft       Manual         Red
23899  Park Avenue  DoubleÂ Overhead Camshaft         Auto  Pale White

       Price ($)  Dealer_No  Body Style    Phone Dealer_Region  outlier_iso
8          82000  85257-3102    Hardtop  7194857         Pasco           -1
21         21000  85257-3102    Hardtop  6183219        Austin           -1
37         61000  38701-8047  Hatchback  7438037    Greenville           -1
48         20000  78758-7841  Hatchback  7117432        Austin           -1
65         61000  99301-3882  Hatchback  8380613         Pasco           -1
...          ...         ...        ...      ...           ...          ...
23885      18000  53546-9427    Hardtop  6009530    Janesville           -1
23890      20001  60504-7114    Hardtop  6172324        Aurora           -1
23891      46000  06457-3834        SUV  6435802    Middletown           -1
23895      61000  53546-9427      Sedan  8439821    Middletown           -1
23899      62000  78758-7841  Hatchback  8384785        Aurora           -1

[2391 rows x 17 columns]
```

```
from sklearn.neighbors import LocalOutlierFactor
import pandas as pd

# Initialize Local Outlier Factor
lof = LocalOutlierFactor(n_neighbors=20, contamination=0.1)

# Select only numerical features for outlier detection
numerical_features = df.select_dtypes(include=['number'])  # Select numerical columns

# Fit and predict on numerical features only
outliers_lof = lof.fit_predict(numerical_features)

# Add LOF results to the dataframe
df['outlier_lof'] = outliers_lof

# Display outliers
print("Local Outlier Factor Outliers:\n", df[df['outlier_lof'] == -1])
```

```
Local Outlier Factor Outliers:
             Car_id        Date Customer Name  Gender  Annual Income  \
7      C_CND_000008    1/2/2022        Graham    Male          13500
8      C_CND_000009    1/2/2022         Naomi    Male         815000
11     C_CND_000012    1/2/2022        Amar'E    Male          13500
28     C_CND_000029    1/2/2022        Sloane    Male          13500
30     C_CND_000031    1/2/2022        Sophia    Male         210000
...             ...         ...           ...     ...            ...
23839  C_CND_023840  12/30/2023         Sofia  Female         555000
23846  C_CND_023847  12/30/2023        Sylvia  Female         925000
23873  C_CND_023874  12/31/2023       Gabriel    Male          13500
23881  C_CND_023882  12/31/2023         Vicky    Male         843000
23899  C_CND_023900  12/31/2023          Yuna    Male          13500

                  Dealer_Name     Company       Model  \
7                  U-Haul CO  Mitsubishi      Galant
8      Rabun Used Car Sales   Chevrolet      Malibu
11              Race Car Help      Nissan  Pathfinder
28              Race Car Help    Chrysler         LHS
30          Saab-Belle Dodge  Mitsubishi      3000GT
...                       ...         ...         ...
23839          Suburban Ford    Cadillac      Catera
23846          Race Car Help  Oldsmobile      Aurora
23873       Saab-Belle Dodge      Subaru     Outback
23881  Star Enterprises Inc       Lexus       LS400
23899               U-Haul CO       Buick  Park Avenue

                          Engine Transmission       Color  Price ($)  \
7      DoubleÂ Overhead Camshaft         Auto  Pale White      42000
8              Overhead Camshaft       Manual  Pale White      82000
11     DoubleÂ Overhead Camshaft         Auto  Pale White      46000
28             Overhead Camshaft       Manual  Pale White      41000
30             Overhead Camshaft       Manual  Pale White      20000
...                          ...          ...         ...        ...
23839  DoubleÂ Overhead Camshaft         Auto  Pale White      75000
23846          Overhead Camshaft       Manual         Red      71000
23873          Overhead Camshaft       Manual         Red      49000
23881          Overhead Camshaft       Manual       Black      69001
23899  DoubleÂ Overhead Camshaft         Auto  Pale White      62000

        Dealer_No  Body Style    Phone Dealer_Region  outlier_iso  outlier_lof
7      78758-7841  Passenger  6206512        Austin            1           -1
8      85257-3102    Hardtop  7194857         Pasco           -1           -1
11     78758-7841    Hardtop  7288103         Pasco            1           -1
28     78758-7841  Hatchback  6292720    Janesville            1           -1
30     60504-7114      Sedan  8847858        Austin            1           -1
...           ...        ...      ...           ...          ...          ...
23839  53546-9427  Hatchback  7752902    Janesville           -1           -1
23846  78758-7841  Passenger  7265067        Austin           -1           -1
23873  60504-7114        SUV  7090003        Aurora            1           -1
23881  99301-3882      Sedan  7011127         Pasco           -1           -1
23899  78758-7841  Hatchback  8384785        Aurora           -1           -1

[2391 rows x 18 columns]
```

```python
import seaborn as sns
import matplotlib.pyplot as plt

# Visualize each feature using boxplots
# Calculate the number of rows and columns for subplots dynamically
num_cols = len(df.columns[:-1])  # Number of features to plot
num_rows = (num_cols + 1) // 2  # Calculate rows, ensuring enough space

plt.figure(figsize=(12, 8))
for i, column in enumerate(df.columns[:-1]):  # Exclude the target and outlier columns
    plt.subplot(num_rows, 2, i + 1)  # Use calculated rows and columns
    sns.boxplot(y=df[column], color="lightblue")
    plt.title(f'Boxplot of {column}')

plt.tight_layout()
plt.show()
```

...

2. Text Mining

```
import pandas as pd
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from collections import Counter

# Download necessary NLTK data
nltk.download('punkt')
nltk.download('stopwords')
# Download the 'punkt_tab' data package for sentence tokenization
nltk.download('punkt_tab')

# Load dataset (replace 'your_dataset.csv' with your actual dataset)
df = pd.read_csv('Car Sales.xlsx - car_data.csv')

# Assuming 'text_column' is the column containing text data
text_data = df['Dealer_Region'].tolist()

# Tokenize and preprocess text
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    tokens = word_tokenize(text.lower())
    return [token for token in tokens if token.isalnum() and token not in stop_words]

preprocessed_data = [preprocess_text(text) for text in text_data]

# Count word frequencies
all_words = [word for text in preprocessed_data for word in text]
word_freq = Counter(all_words)

# Print top 10 most common words
print("Top 10 most common words:")
for word, count in word_freq.most_common(10):
    print(f"{word}: {count}")

# Basic sentiment analysis (you may need to install TextBlob)
from textblob import TextBlob

def get_sentiment(text):
    return TextBlob(text).sentiment.polarity

# Assuming 'text_column' is the correct column name
# Replace 'text_column' with the actual column name if it's different
df['sentiment'] = df['Dealer_Region'].apply(get_sentiment)

print("\nAverage sentiment score:", df['sentiment'].mean())

# You can add more advanced text mining techniques here, such as:
# - Topic modeling (e.g., using Latent Dirichlet Allocation)
# - Named Entity Recognition
# - Text classification
# - Word embeddings (e.g., Word2Vec, GloVe)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Package punkt_tab is already up-to-date!
Top 10 most common words:
austin: 4135
janesville: 3821
scottsdale: 3433
pasco: 3131
aurora: 3130
middletown: 3128
greenville: 3128

Average sentiment score: 0.0
```

```python
# import python libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# import csv file
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/DAV
Project/Diwali Sales Data.csv', encoding= 'unicode_escape')

from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```python
df.shape
```

```
(11251, 15)
```

```python
df.head(20)
```

```
    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
5   1000588       Joni  P00057942      M     26-35   28
1
6   1001132       Balk  P00018042      F     18-25   25
1
7   1002092   Shivangi  P00273442      F       55+   61
0
8   1003224     Kushal  P00205642      M     26-35   35
0
9   1003650      Ginny  P00031142      F     26-35   26
1
10  1003829   Harshita  P00200842      M     26-35   34
0
11  1000214   Kargatis  P00119142      F     18-25   20
0
12  1004035     Elijah  P00080342      F     18-25   20
1
13  1001680    Vasudev  P00324942      M     26-35   26
1
```

```
14  1003858       Cano  P00293742      M     46-50   46
1
15  1000813     Lauren  P00289942      F     18-25   24
0
16  1005447        Amy  P00275642      F     46-50   48
1
17  1001193       Mick  P00004842      F     26-35   29
0
18  1001883    Praneet  P00029842      M     51-55   54
1
19  1001883    Praneet  P00029842      M     51-55   54
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
5   Himachal Pradesh  Northern  Food Processing             Auto
1
6      Uttar Pradesh   Central           Lawyer             Auto
4
7        Maharashtra   Western        IT Sector             Auto
1
8      Uttar Pradesh   Central             Govt             Auto
2
9     Andhra Pradesh  Southern            Media             Auto
4
10             Delhi   Central          Banking             Auto
1
11    Andhra Pradesh  Southern           Retail             Auto
2
12    Andhra Pradesh  Southern        IT Sector             Auto
2
13    Andhra Pradesh  Southern       Automobile             Auto
4
14    Madhya Pradesh   Central      Hospitality             Auto
3
15    Andhra Pradesh  Southern             Govt             Auto
2
16    Andhra Pradesh  Southern        IT Sector             Auto
3
```

```
17    Andhra Pradesh  Southern        Aviation             Auto
1
18     Uttar Pradesh   Central     Hospitality             Auto
1
19     Uttar Pradesh   Central     Hospitality             Auto
1

      Amount  Status  unnamed1
0   23952.00     NaN       NaN
1   23934.00     NaN       NaN
2   23924.00     NaN       NaN
3   23912.00     NaN       NaN
4   23877.00     NaN       NaN
5   23877.00     NaN       NaN
6   23841.00     NaN       NaN
7        NaN     NaN       NaN
8   23809.00     NaN       NaN
9   23799.99     NaN       NaN
10  23770.00     NaN       NaN
11  23752.00     NaN       NaN
12  23730.00     NaN       NaN
13  23718.00     NaN       NaN
14       NaN     NaN       NaN
15  23664.00     NaN       NaN
16       NaN     NaN       NaN
17  23619.00     NaN       NaN
18  23568.00     NaN       NaN
19  23568.00     NaN       NaN
```

```
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)
```

```
#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 null values
df.dropna(inplace=True)
```

```
# 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')
```

```
df
```

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

       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
```

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

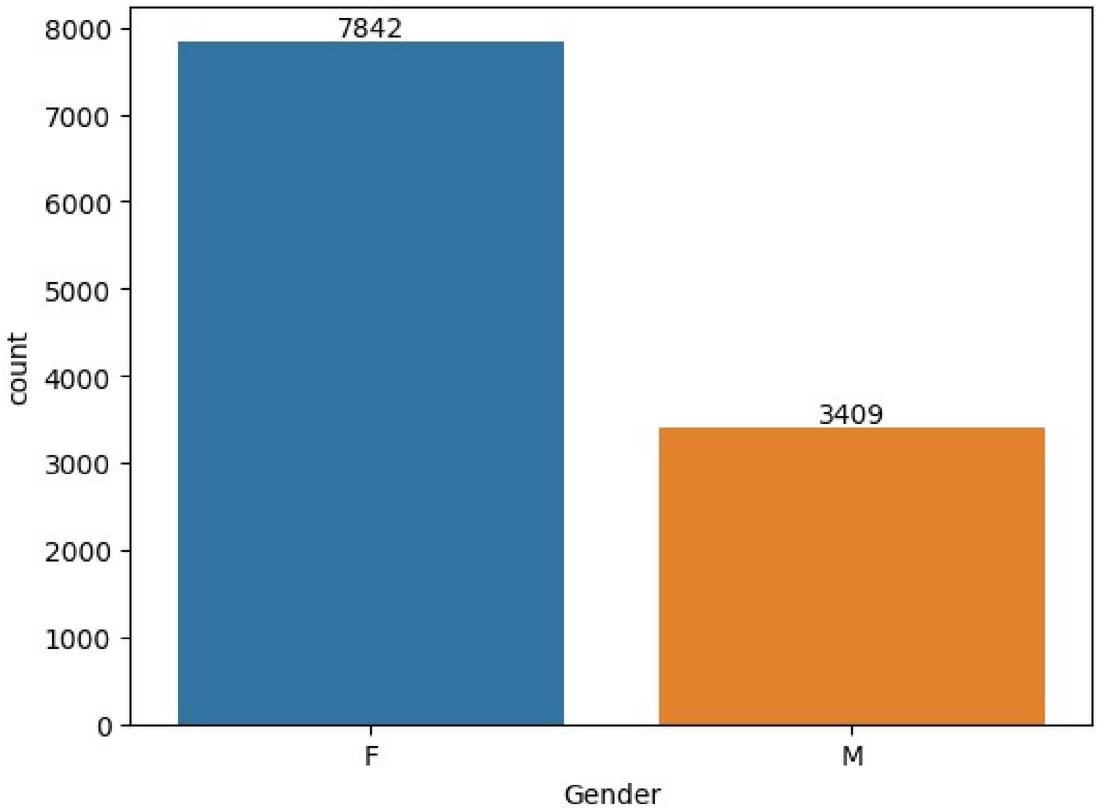
	Age	Orders	Amount
count	11251.000000	11251.000000	11239.000000
mean	35.421207	2.489290	9453.610858
std	12.754122	1.115047	5222.355869
min	12.000000	1.000000	188.000000
25%	27.000000	1.500000	5443.000000
50%	33.000000	2.000000	8109.000000
75%	43.000000	3.000000	12675.000000
max	92.000000	4.000000	23952.000000

Exploratory Data Analysis

Gender

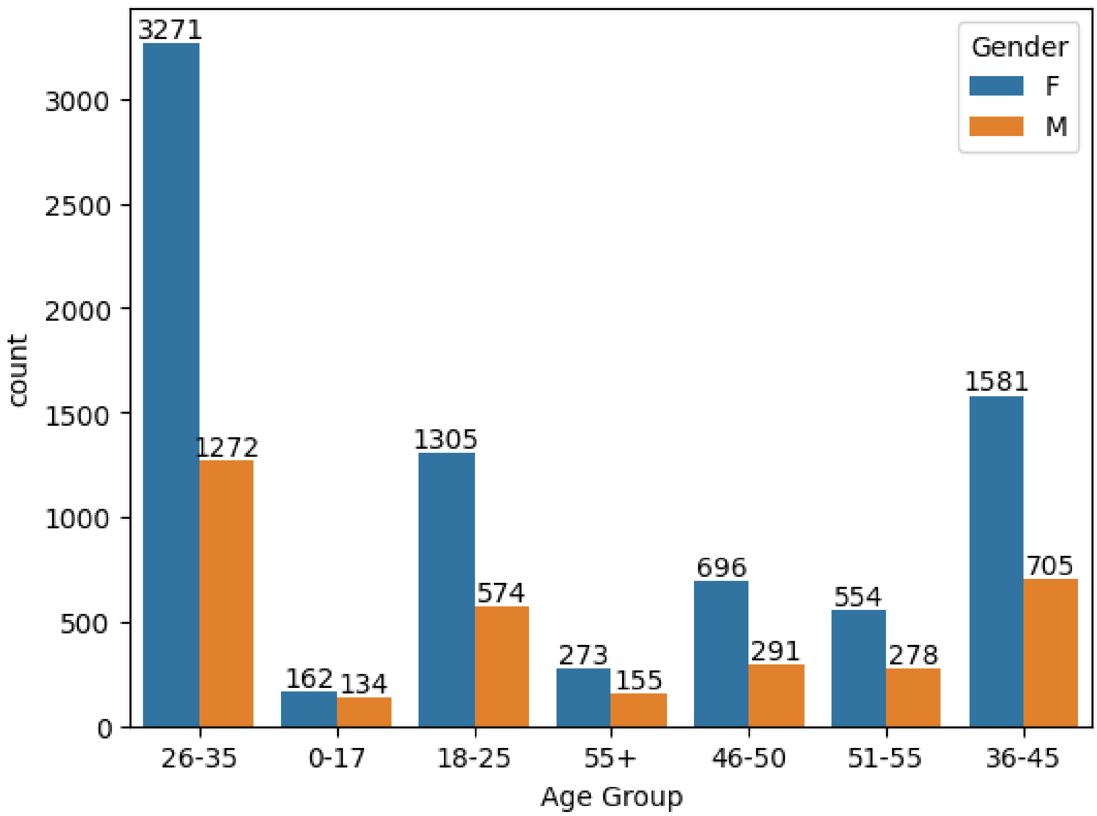
```
# plotting a bar chart for Gender and it's count

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

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



Ques1: What is the distribution of purchasing power among different genders, and how does it compare between males and females?

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

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

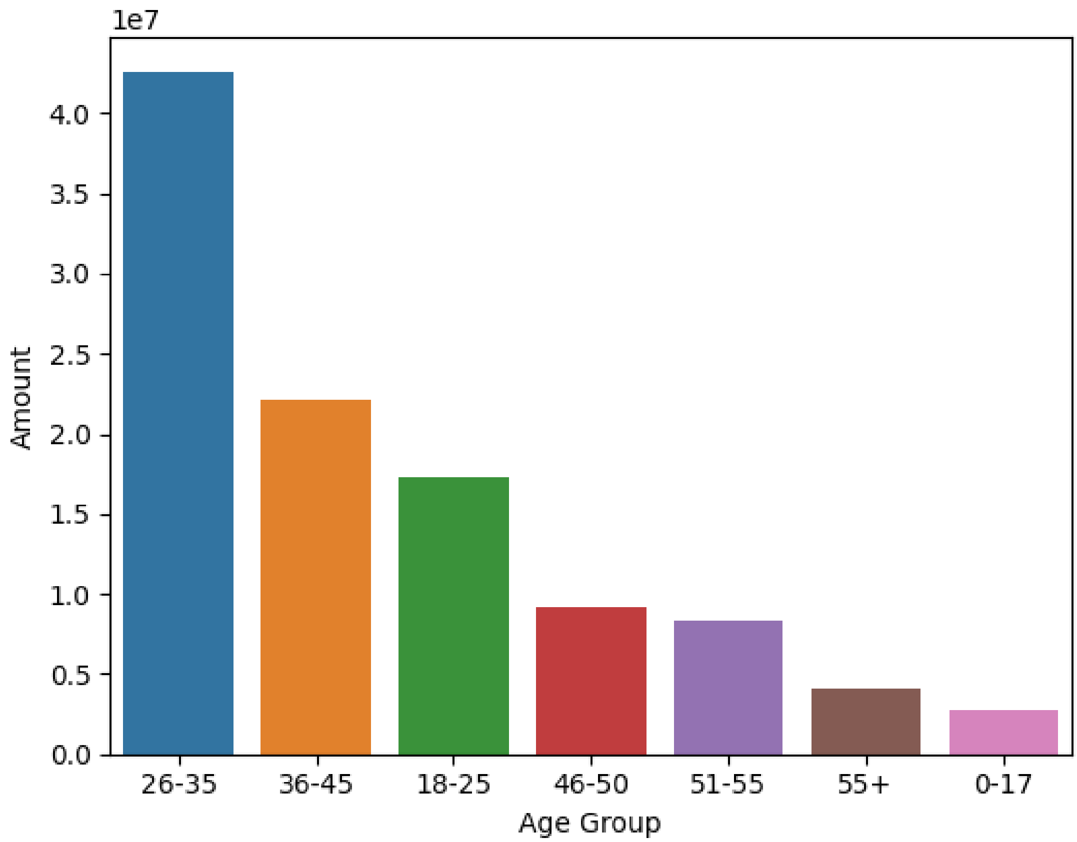


From above graphs we can see that most of the buyers are females and even the purchasing power of females are greater than men

Age

Ques2: Most and least buyers are in which age group?

```
# 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)

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

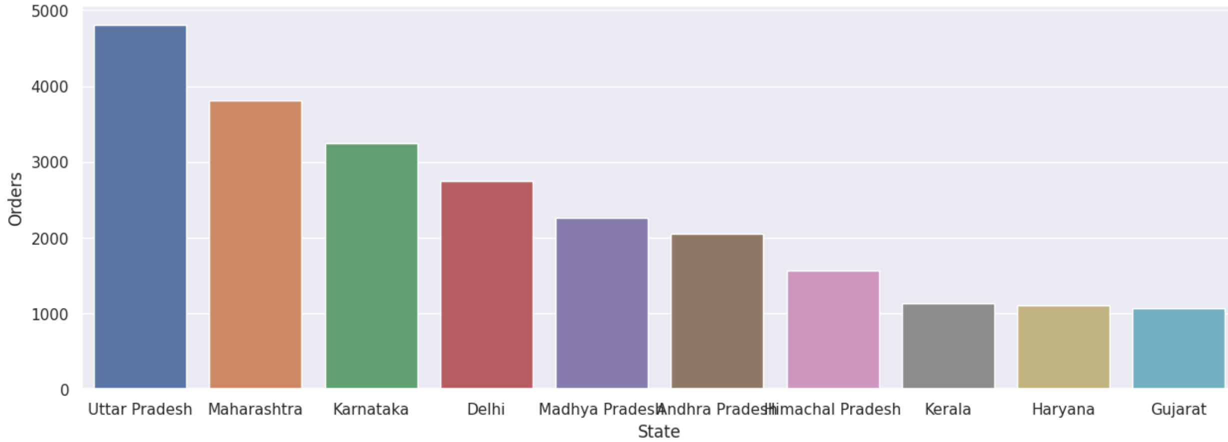


From above graphs we can see that most of the buyers are of age group between 26-35 yrs and least are of age group between 0-17 yrs

State

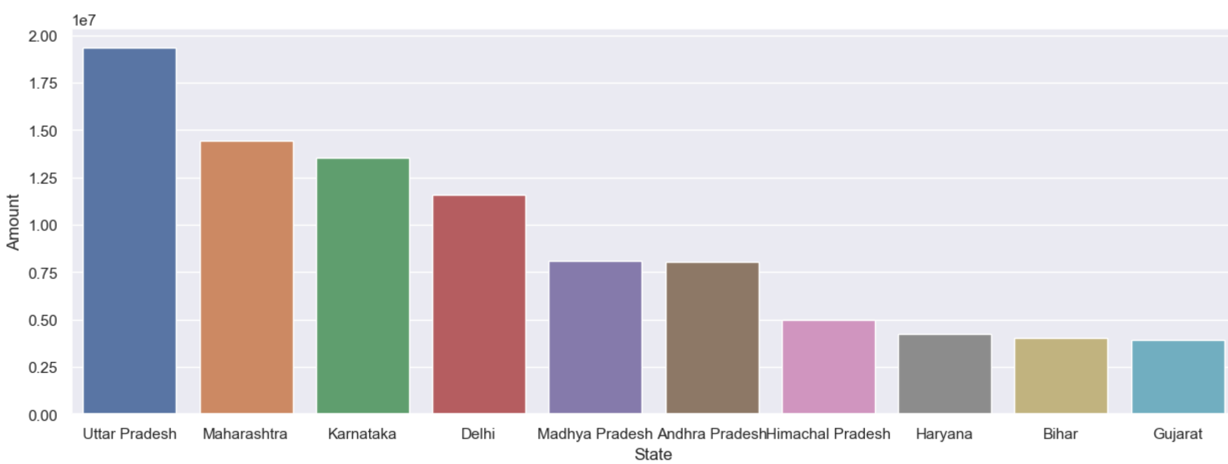
Ques3: List the Top 10 States having most purchases

```
# 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')

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



Ques3: What are the key regions driving the highest number of orders and total sales?

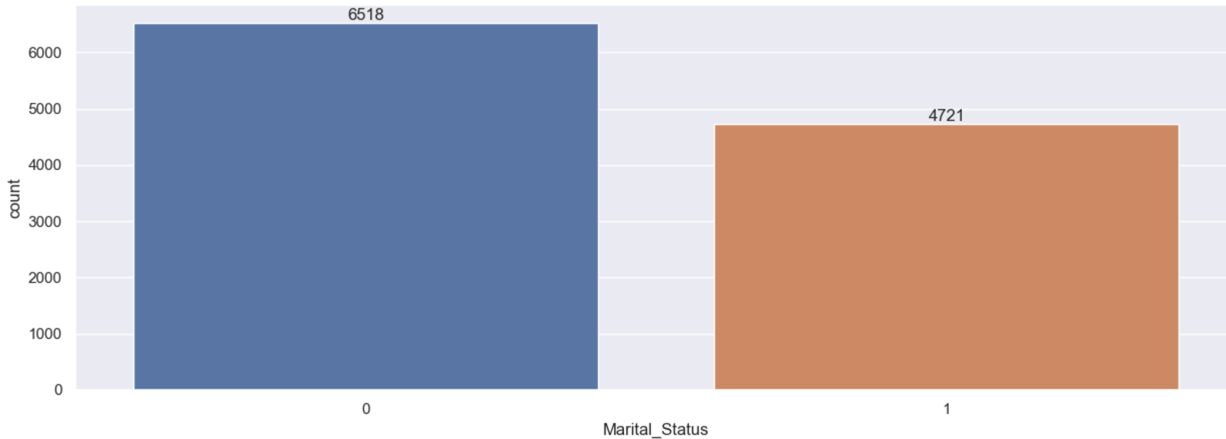
```
# total amount/sales from top 10 states

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')

<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

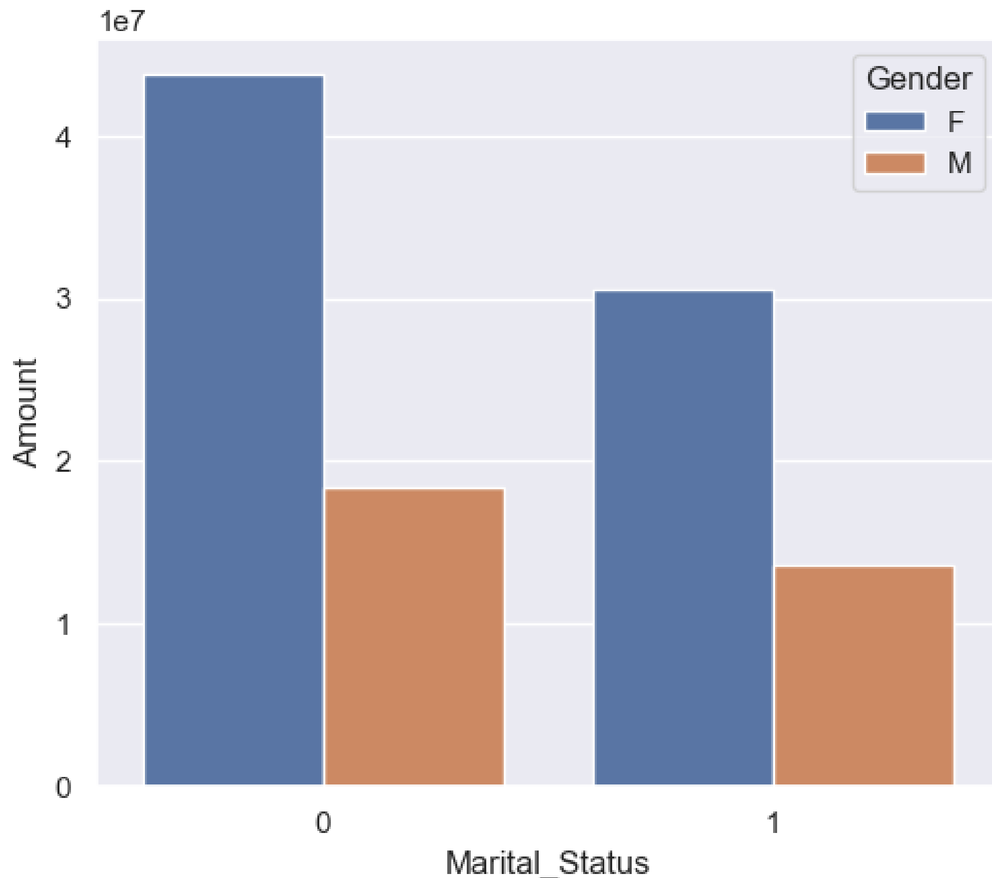
Marital Status

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

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



Ques4: "What is the purchasing behavior and power of married women in the dataset?

```
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'>
```

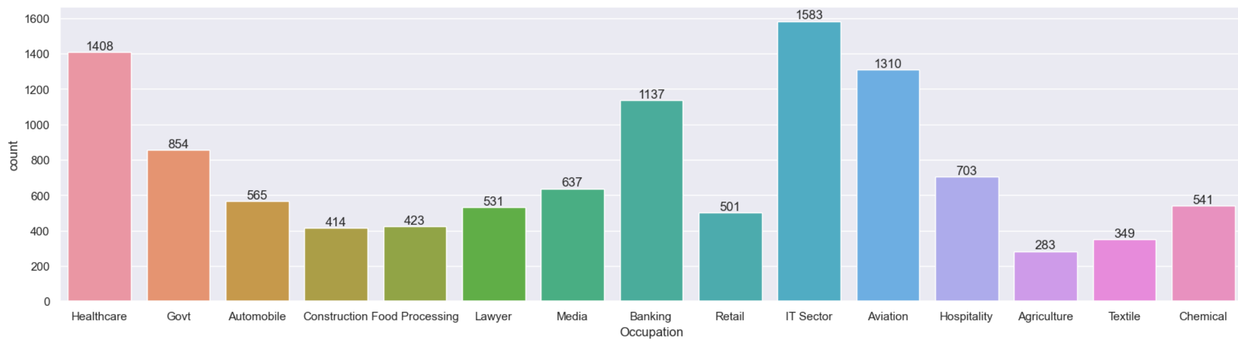


From above graphs we can see that most of the buyers are married (women) and they have high purchasing power

Occupation

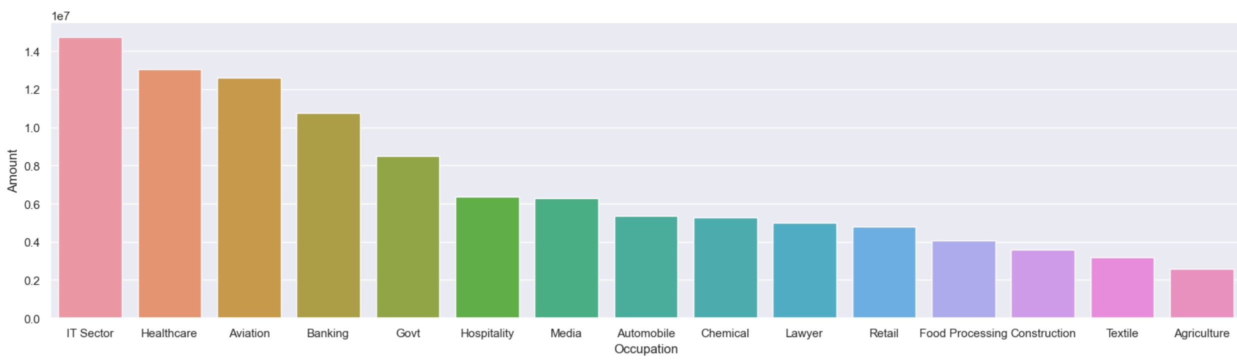
Ques5: "What is the occupational distribution of buyers in the dataset, and how does it vary across different sectors?

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

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




```
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')
```

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

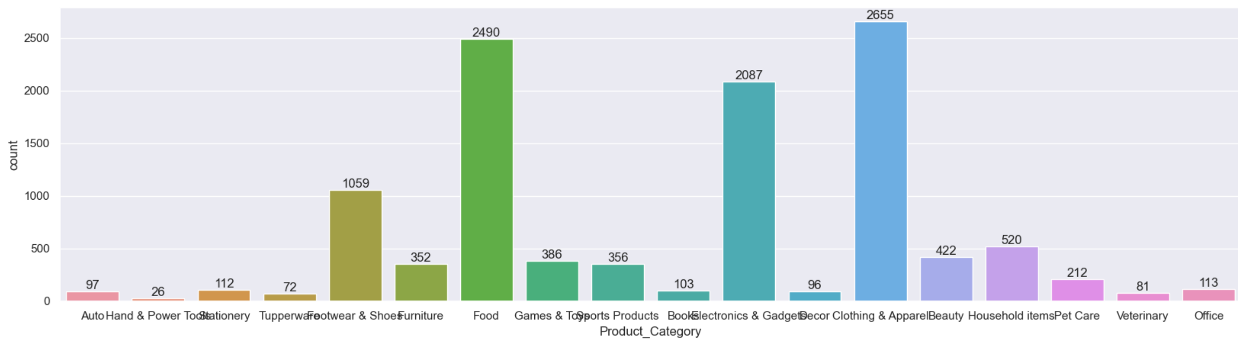


From above graphs we can see that most of the buyers are working in IT, Healthcare and Aviation sector

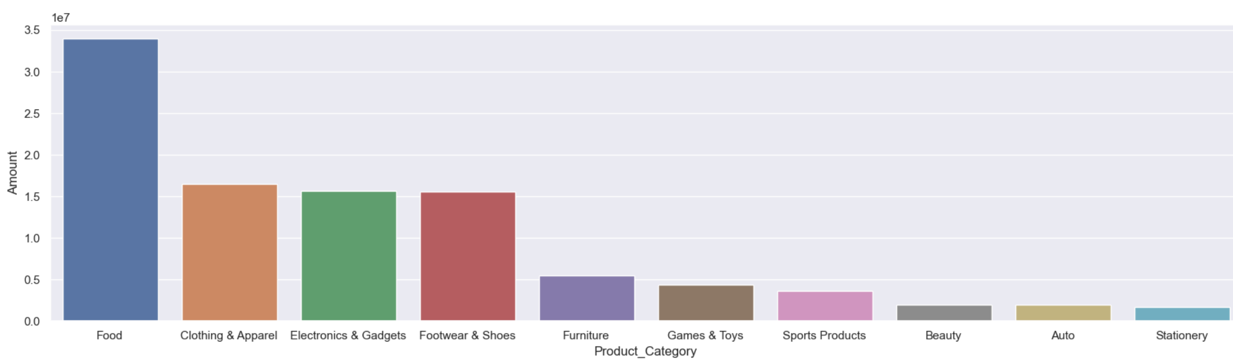
Product Category

Ques 7: What is the distribution of sold products across various categories?

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

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



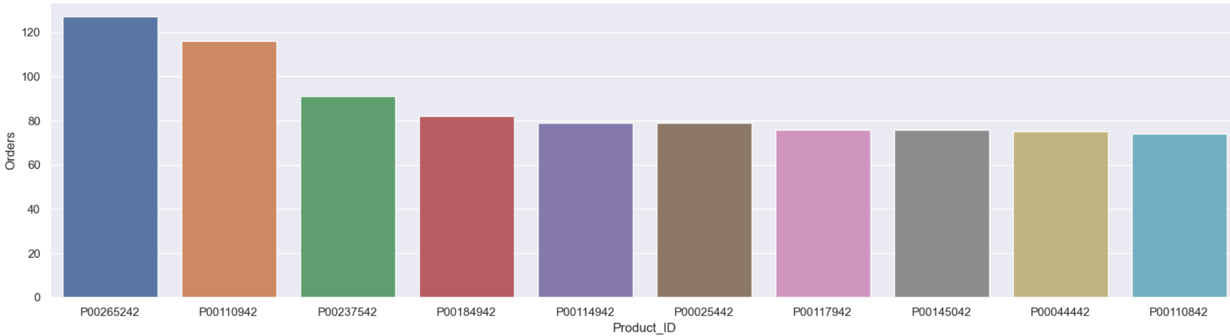

```
sales_state = df.groupby(['Product_Category'], as_index=False)
['Amount'].sum().sort_values(by='Amount', ascending=False).head(10)

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

<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

Which Product ID have maximum number of orders?

```
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')

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



120
100
80
60
40
20
0
Orders
P00265242
P00110942
P00237542
P00184942
P00114942
P00025442
P00117942
P00145042
P00044442
P00110842
Product_ID