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

df = pd.read_csv("historical_automobile_sales.csv")

df
```

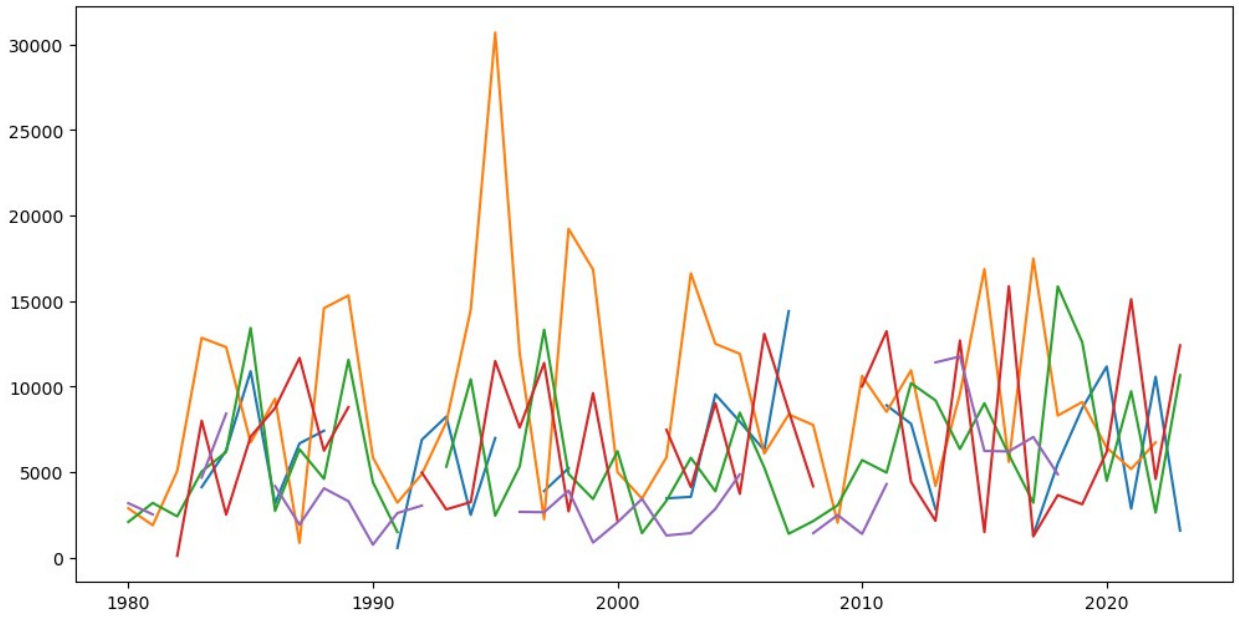
```
           Date  Year Month  Recession  Consumer_Confidence  \
0     1/31/1980  1980   Jan          1                108.24
1     2/29/1980  1980   Feb          1                 98.75
2     3/31/1980  1980   Mar          1                107.48
3     4/30/1980  1980   Apr          1                115.01
4     5/31/1980  1980   May          1                 98.72
..          ...   ...   ...        ...                   ...
523   8/31/2023  2023   Aug          0                103.36
524   9/30/2023  2023   Sep          0                101.55
525  10/31/2023  2023   Oct          0                124.66
526  11/30/2023  2023   Nov          0                 97.09
527  12/31/2023  2023   Dec          0                 95.92

     Seasonality_Weight      Price  Advertising_Expenditure
Competition  \
0                  0.50  27483.571                     1558
7
1                  0.75  24308.678                     3048
4
2                  0.20  28238.443                     3137
3
3                  1.00  32615.149                     1653
7
4                  0.20  23829.233                     1319
4
..                  ...        ...                      ...           .
..
523                0.25  27852.993                     1793
6
524                0.07  21183.704                     1028
5
525                0.12  15975.589                     1148
9
526                0.25  16862.288                     4850
5
527                0.34  25240.425                     2319
3

        GDP  Growth_Rate  unemployment_rate  Automobile_Sales  \
0    60.223     0.010000                5.4             456.0
1    45.986    -0.309594                4.8             555.9
```

```
2    35.141    -0.308614                3.4              620.0
3    45.673     0.230596                4.2              702.8
4    52.997     0.138197                5.3              770.4
..      ...          ...                ...                ...
523  57.169     0.764155                2.6             1579.6
524  59.315     0.036180                2.5             1123.4
525  19.472    -2.046169                2.5             1685.9
526  27.904     0.302179                2.9             2124.6
527  13.518    -1.064211                2.1             3538.5
```

```
        Vehicle_Type        City
0      Supperminicar     Georgia
1      Supperminicar    New York
2    Mediumfamilycar    New York
3      Supperminicar    Illinois
4    Smallfamiliycar  California
..               ...         ...
523     Executivecar    New York
524  Smallfamiliycar  California
525           Sports  California
526  Smallfamiliycar     Georgia
527  Smallfamiliycar     Georgia
```

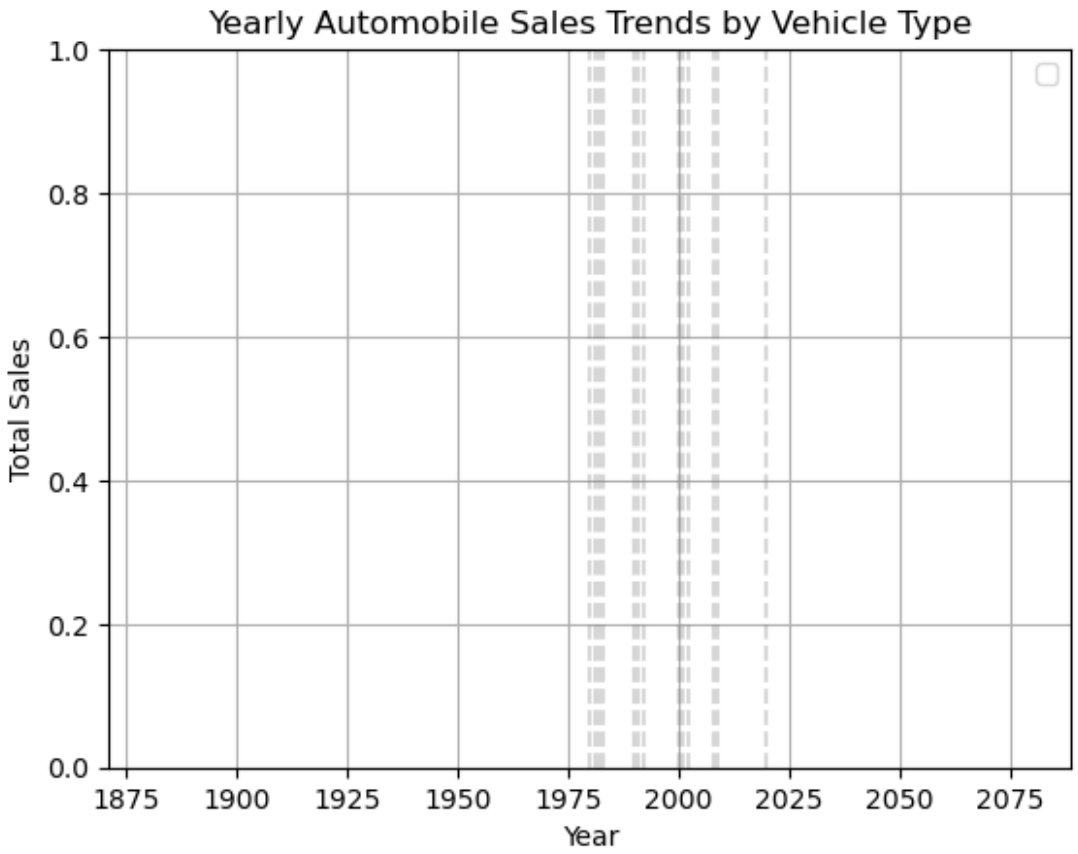
```
[528 rows x 15 columns]
```

```
df['Year'] = pd.to_numeric(df['Year'])

#Develop a Line chart using the functionality of pandas to show how.
automobile sales fluctuate from year to year
sales_trends = df.groupby(['Year', 'Vehicle_Type'])
['Automobile_Sales'].sum().unstack()

plt.figure(figsize=(12, 6))
for vehicle_type in sales_trends.columns:
    plt.plot(sales_trends.index, sales_trends[vehicle_type],
label=vehicle_type)
```




```
#Plot different lines for categories of vehicle type and analyse the
trend to answer the question Is there a noticeable difference in sales
trends between different vehicle types during recession periods?
# Highlighting recession periods
recession_years = df[df['Recession'] == 1]['Year'].unique()
for year in recession_years:
    plt.axvline(x=year, color='gray', alpha=0.3, linestyle='--')

plt.title("Yearly Automobile Sales Trends by Vehicle Type")
plt.xlabel("Year")
plt.ylabel("Total Sales")
plt.legend()
plt.grid()
plt.show()
```

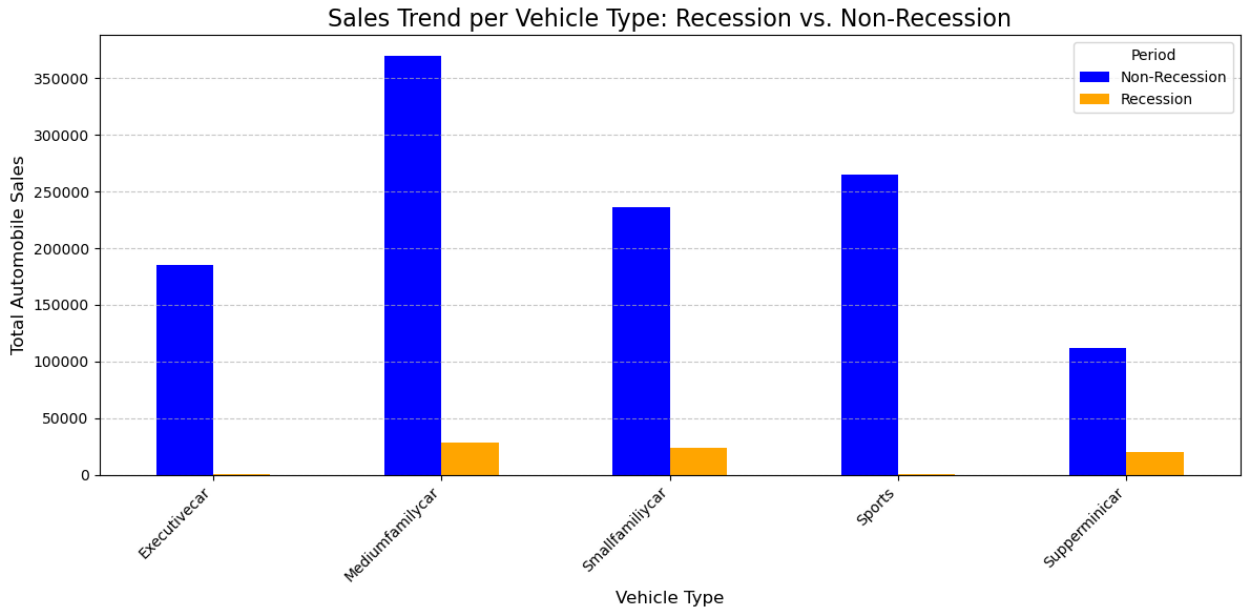
```
No artists with labels found to put in legend.  Note that artists
whose label start with an underscore are ignored when legend() is
called with no argument.
```



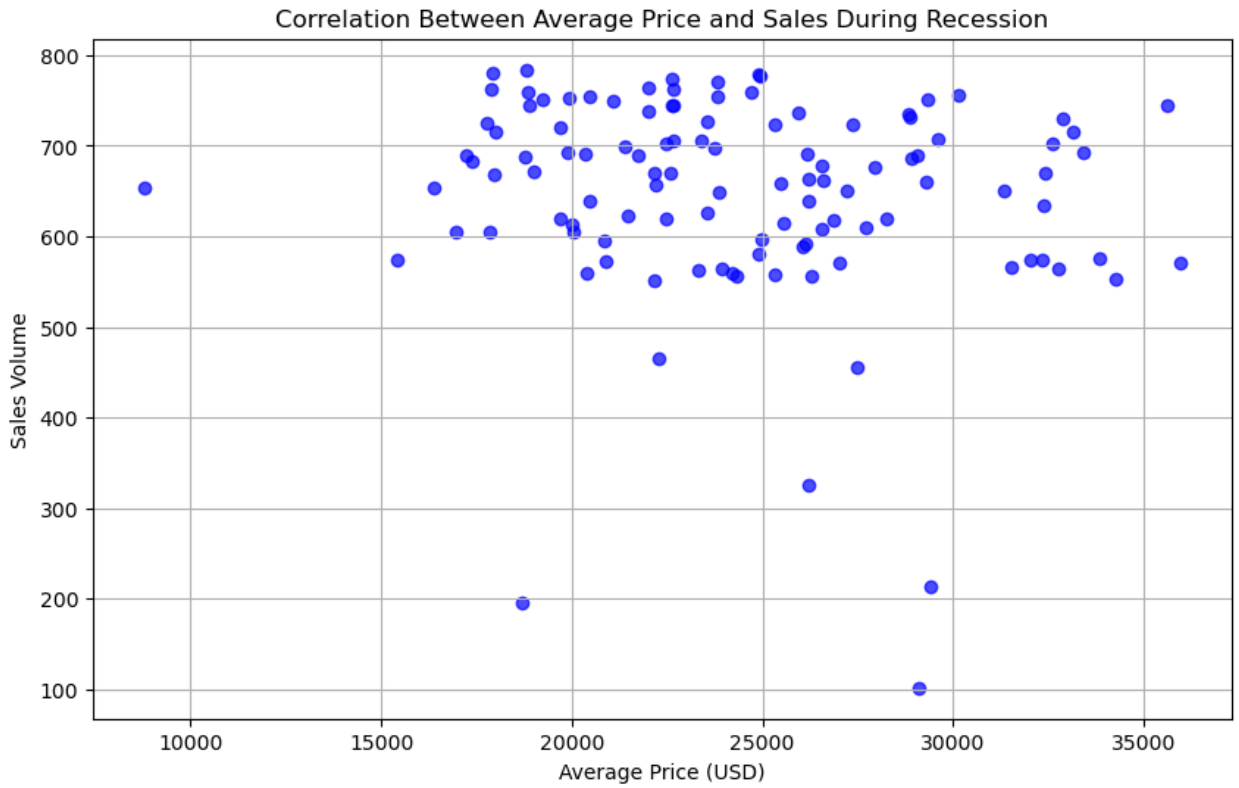

```
#Use the Matplotlib Library to create a visualization to compare the
sales trend per vehicle type for a recession period with a non-
recession period.

# Group data by Vehicle_Type and Recession, summing up
Automobile_Sales
sales_summary = df.groupby(['Vehicle_Type', 'Recession'])
['Automobile_Sales'].sum().unstack()

# Create the bar chart
plt.figure(figsize=(12, 6))
sales_summary.plot(kind='bar', figsize=(12, 6), color=['blue',
'orange'])

# Customize the plot
plt.title("Sales Trend per Vehicle Type: Recession vs. Non-Recession",
fontsize=16)
plt.xlabel("Vehicle Type", fontsize=12)
plt.ylabel("Total Automobile Sales", fontsize=12)
plt.xticks(rotation=45, ha='right', fontsize=10)
plt.legend(['Non-Recession', 'Recession'], title="Period")
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
```

```
# Display the plot
plt.show()
```

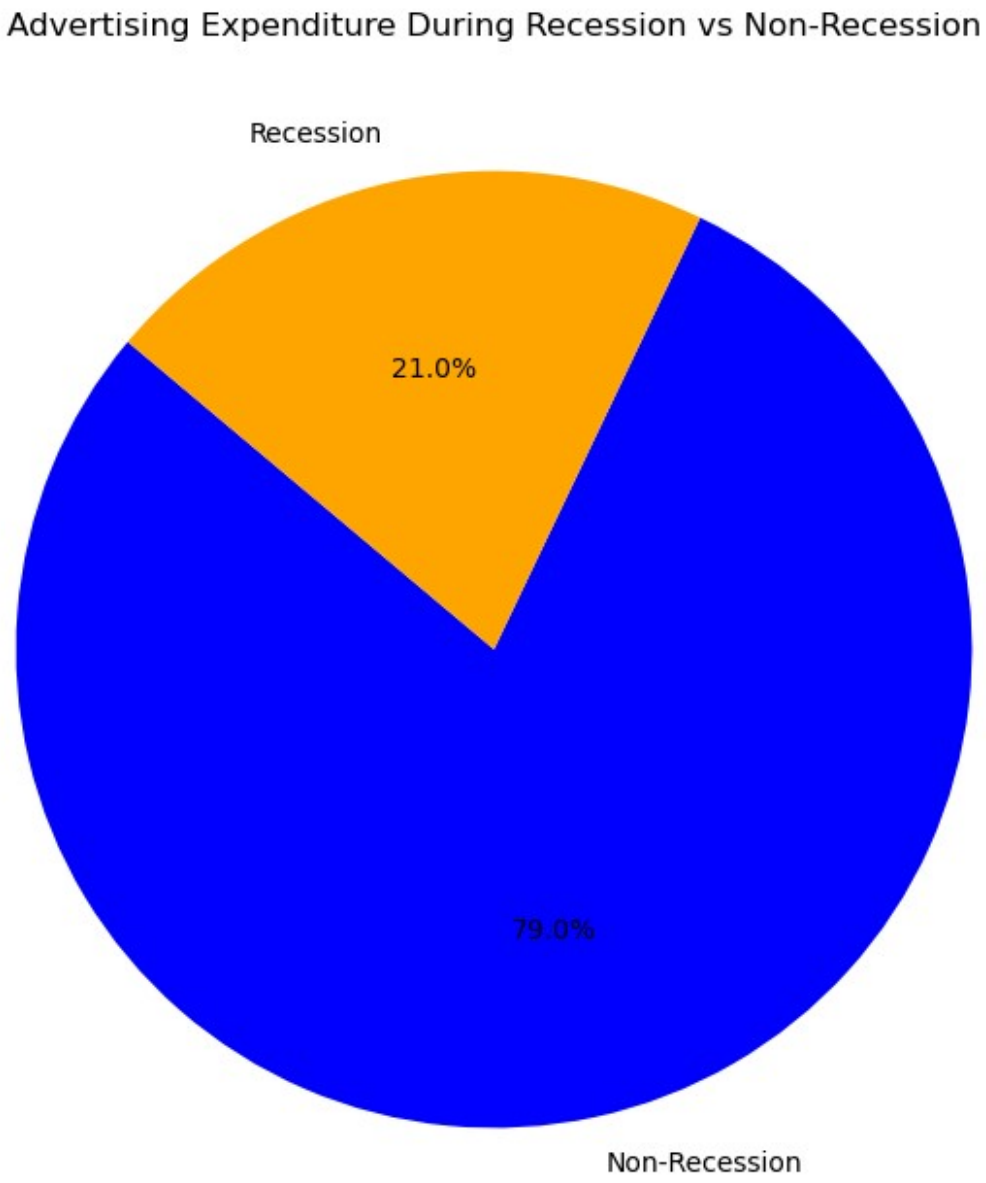
```
<Figure size 1200x600 with 0 Axes>
```




```
#Use the functionality of Matplotlib to develop a scatter plot to
identify the correlation between average vehicle price relate to the
sales volume during recessions.
recession_data = df[df['Recession'] == 1]

plt.figure(figsize=(10, 6))
plt.scatter(recession_data['Price'],
recession_data['Automobile_Sales'], alpha=0.7, c='blue')
plt.title("Correlation Between Average Price and Sales During
Recession")
plt.xlabel("Average Price (USD)")
plt.ylabel("Sales Volume")
plt.grid()
plt.show()
```

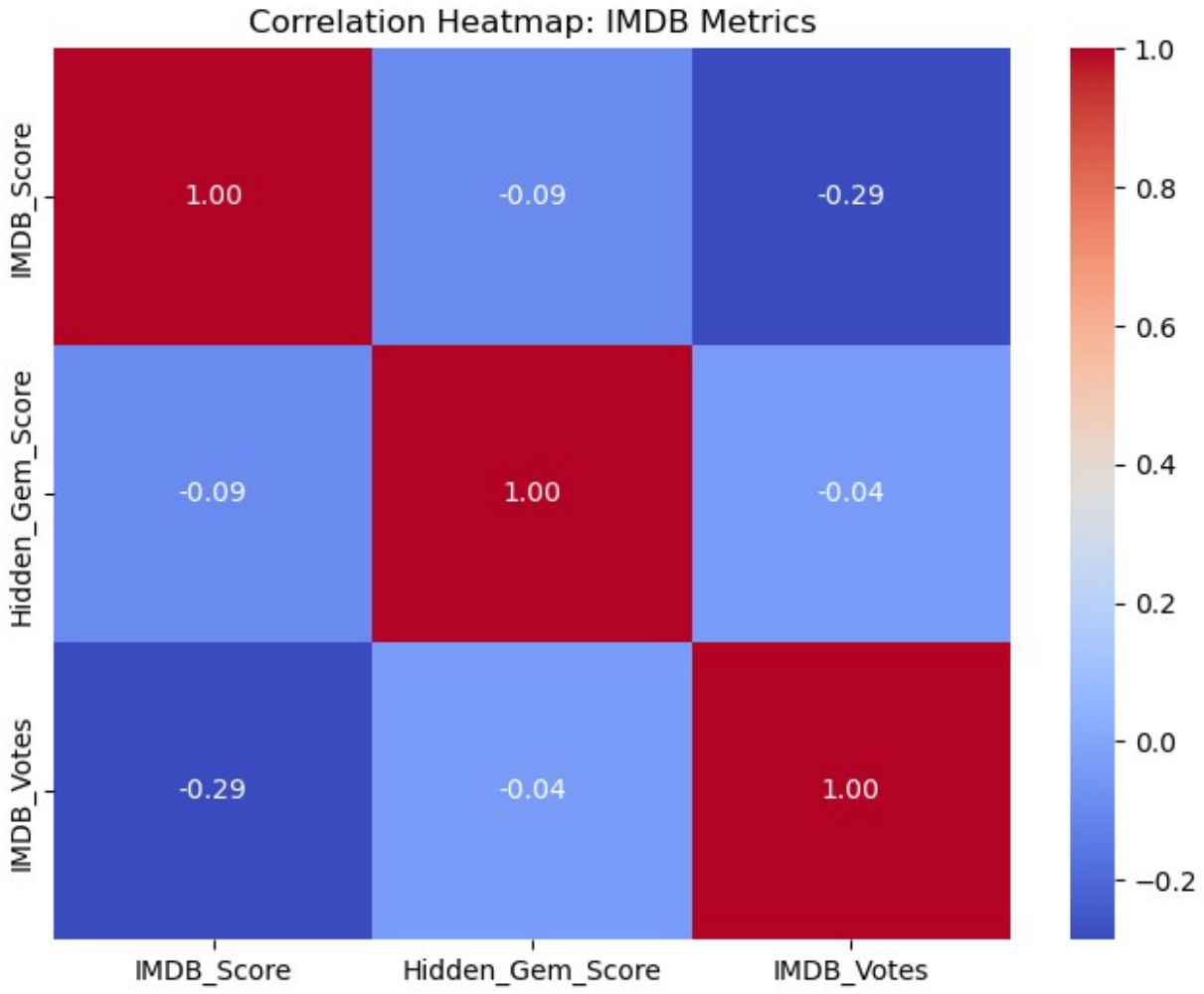



```
#Create a pie chart to display the portion of advertising expenditure
of. Automotives during recession and non-recession periods.
ad_expenditure = df.groupby('Recession')
['Advertising_Expenditure'].sum()
plt.figure(figsize=(8, 8))
plt.pie(ad_expenditure, labels=["Non-Recession", "Recession"],
autopct='%1.1f%%', startangle=140, colors=["blue", "orange"])
plt.title("Advertising Expenditure During Recession vs Non-Recession")
plt.show()
```



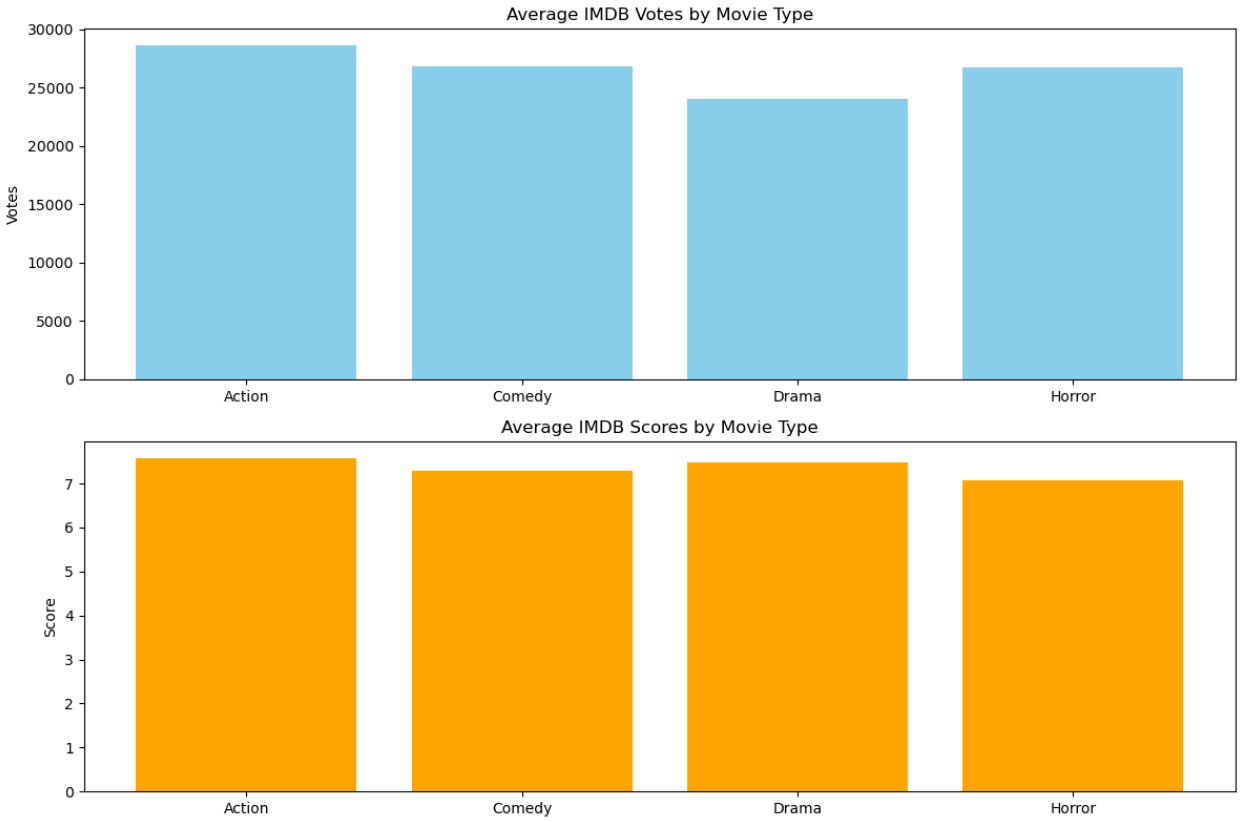

```
#Create a heatmap to understand correlation between IMDB Score,Hidden
Gem Score and IMDB Votes.
movie_data = pd.DataFrame({
    "IMDB_Score": np.random.uniform(5, 10, 100),
    "Hidden_Gem_Score": np.random.uniform(50, 100, 100),
    "IMDB_Votes": np.random.randint(1000, 50000, 100)
})

# Computing correlation and plotting heatmap
corr_matrix = movie_data.corr()
plt.figure(figsize=(8, 6))
```

```
sns.heatmap(corr_matrix, annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Correlation Heatmap: IMDB Metrics")
plt.show()
```




```
#Plot lines for categories of every movie type and analyse how they
have received IMDB Votes. Create a subplot to compare the same
categories with IMDB Score.
movie_data['Movie_Type'] = np.random.choice(["Action", "Comedy",
"Drama", "Horror"], 100)
votes_by_type = movie_data.groupby("Movie_Type")["IMDB_Votes"].mean()
scores_by_type = movie_data.groupby("Movie_Type")["IMDB_Score"].mean()

fig, ax = plt.subplots(2, 1, figsize=(12, 8))
ax[0].bar(votes_by_type.index, votes_by_type.values, color="skyblue")
ax[0].set_title("Average IMDB Votes by Movie Type")
ax[0].set_ylabel("Votes")
ax[1].bar(scores_by_type.index, scores_by_type.values, color="orange")
```

```
ax[1].set_title("Average IMDB Scores by Movie Type")
ax[1].set_ylabel("Score")
plt.tight_layout()
plt.show()
```




```
#Create 2 bar plots to understand movies and web series by languages
in which they have been made.
movie_data['Language'] = np.random.choice(["English", "Hindi",
"Spanish", "French"], 100)
movie_data['Category'] = np.random.choice(["Movie", "Web Series"],
100)

movies_by_language = movie_data[movie_data["Category"] ==
"Movie"].groupby("Language").size()
web_series_by_language = movie_data[movie_data["Category"] == "Web
Series"].groupby("Language").size()

fig, axes = plt.subplots(1, 2, figsize=(16, 6))
axes[0].bar(movies_by_language.index, movies_by_language.values,
color="green")
axes[0].set_title("Movies by Language")
axes[0].set_ylabel("Count")
axes[1].bar(web_series_by_language.index,
web_series_by_language.values, color="purple")
```

```
axes[1].set_title("Web Series by Language")
axes[1].set_ylabel("Count")
plt.tight_layout()
plt.show()
```