44517 - 03
Team Phoenix

CruiseControl: Steering Through Automotive Market Analysis

Anirudh Gunde,
Sahithi Kasarapu,
Sai Chaitanya Vishnu Kanth Inavolu,
Sri Venkata Satya Sai Charan Teja Tallam

April 25, 2024

School of Computer Science and Information Systems
Northwest Missouri State University

1 Project Idea

Utilizing the "Vehicle Sales and Market Trends Dataset," the project aims to:

- Conduct comprehensive market analysis to identify trends and patterns in the automotive industry.
- Create predictive models to forecast vehicle prices based on factors like condition, mileage, and market trends.
- Explore factors influencing pricing strategies and consumer preferences.
- Generate business insights for automotive industry professionals, dealerships, and financial institutions.
- Conduct comparative analysis across different vehicle types, makes, and models.
- Investigate the impact of geographical location on sales trends and pricing.
- Identify variations and inconsistencies in sales transactions for additional investigation.

2 Tools & Technologies

- **Apache Spark:** For scalable data processing and analysis.
- **Jupyter Notebook:** For developing the code.
- **Power BI:** For visualizing the data.
- **Additional Resources:** Pandas.

3 Architecture Diagram

Figure 1: Architecture Diagram

4 Architecture Summary

1. The dataset containing vehicle sales and market trends serves as the input.
2. Data processing tasks, utilizing Apache PySpark is performed to clean, transform, and analyze the dataset.
3. Predictive modeling techniques are applied to develop models for estimating vehicle prices and predicting market trends.
4. The processed data and model outputs are visualized using Power BI, facilitating easy interpretation and decision-making.

5 Goals to investigate

1. Year-wise, identify the top-selling vehicle for each manufacturer.
2. Calculate the average selling price for each car type or manufacturer.
3. Determine the total revenue generated for each make over the years.
4. Count the number of vehicles sold for each body type.
5. Average Odometer Reading by Vehicle Make.
6. Top Selling models for each make/manufacturer.
7. Identify the Top 5 States with the Highest Number of Vehicle Sales.
8. Identify the Average MMR Value by Make and Model.

6 Project Description

The dataset comprises comprehensive information on used vehicles, encompassing various attributes crucial for understanding the automotive market dynamics. Each entry includes details such as the vehicle's manufacturing year, make, model, trim, body type, transmission type, Vehicle Identification Number (VIN), state of registration, condition rating, odometer reading, exterior and interior colors, seller type, Manheim Market Report (MMR) value, actual selling price, and sale date. With such rich and diverse data, this dataset facilitates in-depth analysis of the automotive industry, allowing exploration of trends, patterns, and correlations across different vehicle types, manufacturers, geographical regions, and market conditions. Insights gleaned from this dataset can aid stakeholders, including automotive industry professionals, dealerships, financial institutions, and researchers, in making informed decisions regarding pricing strategies, inventory management, market positioning, and consumer preferences. Additionally, the dataset presents an opportunity to develop predictive models for forecasting vehicle prices, understanding sales trends, and identifying factors influencing purchasing behaviors in the used car market.

6.1 Choosing Dataset

For this project, we selected a comprehensive dataset containing information on used vehicles, sourced from Kaggle. The dataset encompasses a wide range of attributes essential for analyzing trends and patterns in the automotive market.

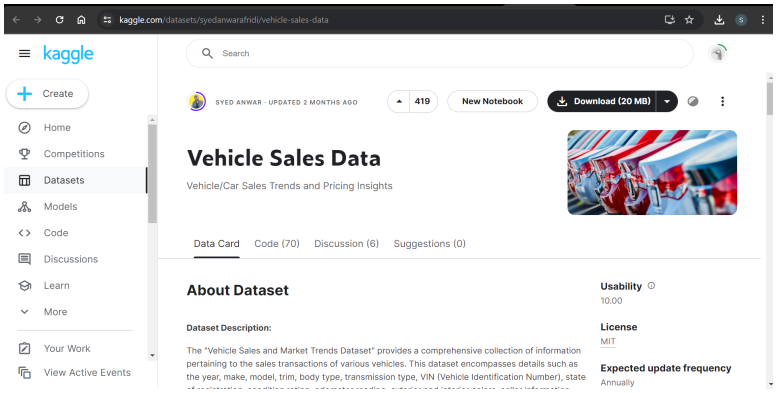

Figure 2: Dataset

6.2 Cleaning Dataset

Prior to analysis, the dataset underwent thorough cleaning to ensure data integrity and consistency which involves removing the null values and also missing values have been handled by replacing them with the mean values of their respective columns, ensuring that the dataset remains complete and suitable for analysis and addressing outliers. By cleaning the dataset, we aimed to prepare a reliable and accurate dataset for subsequent analysis.

```
For Object types

object_columns = ['transmission', 'body','model', 'make','color','interior','saledate','vin']
# Replace missing values in these columns with the most frequent value of each column
for col in object_columns:
    most_frequent = df[col].mode()[0]  # mode() returns a Series, [0] gets the most frequent element
    df[col] = df[col].fillna(most_frequent)  # Direct assignment avoids the FutureWarning

print(df)

-----------------------------------------------------------------------------------------------------
For value types

# Calculate the mean price
mean_price = df['condition'].mean()

# Use recommended approach to avoid chained assignment
df['condition'] = df['condition'].fillna(mean_price)
```

Figure 3: Commands used to clean data

```
transmission    65352
body            13195
condition       11820
trim            10651
model           10399
make            10301
color             749
interior          749
odometer           94
mmr                38
sellingprice       12
saledate           12
vin                 4
year                0
state               0
seller              0
dtype: int64
```

(a) Before Cleaning

```
trim            0
year            0
make            0
model           0
body            0
transmission    0
vin             0
state           0
condition       0
odometer        0
color           0
interior        0
seller          0
mmr             0
sellingprice    0
saledate        0
dtype: int64
```

(b) After Cleaning

Figure 4: Comparison of data before and after cleaning

6.3 Specifying Objectives

Clear objectives were outlined to guide our analysis and exploration of the dataset. These objectives included identifying trends in vehicle sales, Calculate the average selling price ,Identify the top States with the Highest number of vehicle Sales in the automotive industry and detailed information of goals is mentioned in the section 5

6.4 Working on Queries

To achieve our objectives, we formulated a series of queries tailored to extract relevant information from the dataset. These queries ranged from basic descriptive statistics to more complex analytical queries aimed at uncovering insights into sales trends, pricing strategies, and market dynamics.

6.5 Visualization of Query Outputs

The outputs from our queries were visualized using appropriate visualization techniques such as bar charts, line graphs, and scatter plots. Visualization played a crucial role in facilitating the interpretation of results and communicating key findings to stakeholders effectively.

7 Results

The following section outlines the objectives and corresponding queries for each objective, along with the results of each query and the visualization of the output using the Matplotlib library and Power BI.

Question-1

Query

```
import pandas as pd
import time

# Record the start time
start_time = time.time()

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by year, make, and model, then count the total sales for each group
sales_counts = df.groupby(['year', 'make', 'model']).size().reset_index(name='TotalSales')

# Record the end time
end_time = time.time()

# Calculate processingtime
processingtime = end_time - start_time
print("Processing Time: {:.4f} seconds".format(processingtime))

# Display the top-selling vehicles for each manufacturer year-wise in the desired format
print(sales_counts)

# Save the result to a CSV file
sales_counts.to_csv('Q1.csv', index=False)
```

Output

```
Top-selling vehicles for each manufacturer year-wise:
      year       make      model  TotalSales
0     1982       Ford     Altima           2
1     1983       Ford     Altima           1
2     1984       Ford     Altima           4
3     1984  chevrolet   corvette           1
4     1985       Ford     Altima           8
...    ...        ...        ...         ...
5608  2015      Volvo        V60          83
5609  2015      Volvo       XC60          78
5610  2015      Volvo       XC70          25
5611  2015  chevrolet       capt           3
5612  2015      smart     fortwo           3

[5613 rows x 4 columns]

Execution started at: Wed Apr 24 14:29:31 2024
Execution ended at: Wed Apr 24 14:29:36 2024
Execution time: 4.409718751907349 seconds
```

Visualization code

```
# Graph Code for 1st Question

```

```
3
4  import pandas as pd
5  import matplotlib.pyplot as plt
6
7  # Read the data into a DataFrame
8  # Replace 'your_data.csv' with the actual path to your data file
9  df = pd.read_csv('output.csv')
10
11 # Reset index to avoid ambiguity
12 top_selling_vehicles = df.groupby(['year', 'make']).apply(lambda x:
       x.loc[x['sellingprice'].idxmax()]).reset_index(drop=True)
13
14 # Count the number of top-selling vehicles for each manufacturer
       year-wise
15 top_selling_counts = top_selling_vehicles.groupby(['year', 'make'])
       .size().unstack(fill_value=0)
16
17 # Plotting
18 top_selling_counts.plot(kind='bar', stacked=True, figsize=(12, 6))
19 plt.title('Top Selling Vehicles by Manufacturer (Year-wise)')
20 plt.xlabel('Year')
21 plt.ylabel('Number of Top Selling Vehicles')
22 plt.legend(title='Manufacturer', bbox_to_anchor=(1.05, 1), loc='
       upper left')
23 plt.tight_layout()
24 plt.show()
```

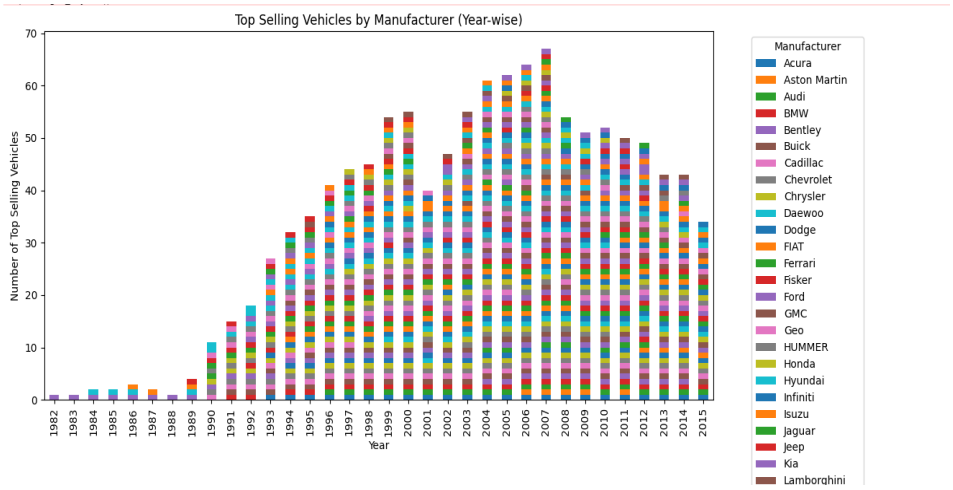

Figure 5: Visualization Image using Mat-Plot Library

We also used POWER BI to represent the same information using different chart

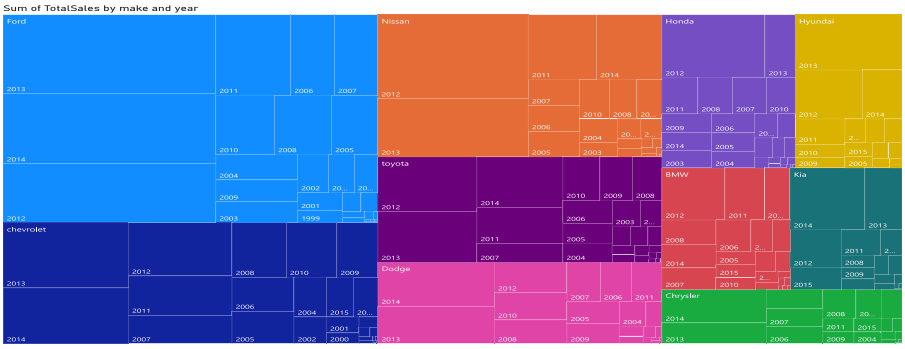

Figure 6: Visualization Image using POWER BI

Question-2

Query

```python
# Question-2

import pandas as pd
import time

# Record start time
start_time = time.time()

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by car type (model) and calculate the average selling price
average_price_by_model = df.groupby('model')['sellingprice'].mean()

# Group the data by manufacturer (make) and calculate the average selling price
average_price_by_manufacturer = df.groupby('make')['sellingprice'].mean()

# Reset index and rename columns for average selling price by manufacturer
average_price_by_manufacturer = average_price_by_manufacturer.reset_index()
average_price_by_manufacturer.columns = ['Make', 'AvgSellingPrice']

# Record end time
end_time = time.time()

# Calculate processing time
processing_time = end_time - start_time

```

```
print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
print("Processing Time:", processing_time, "seconds")

print(average_price_by_manufacturer)

average_price_by_manufacturer.to_csv('Q2.csv', index=False)
```

Output

```
Start Time: 2024-04-24 14:41:27
End Time: 2024-04-24 14:41:31
Processing Time: 3.8979320526123047 seconds
            Make  AvgSellingPrice
0          Acura     14017.268260
1   Aston Martin     54812.000000
2           Audi     19915.432782
3            BMW     21441.895748
4        Bentley     74367.672414
..           ...              ...
91        subaru      3710.416667
92        suzuki      4810.000000
93        toyota      7339.105263
94    volkswagen      6145.833333
95            vw     13672.916667

[96 rows x 2 columns]
```

Visualization code

```
# Graph Code for 2nd Question

import pandas as pd
import matplotlib.pyplot as plt

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by car type (model) and calculate the average
    selling price
average_price_by_model = df.groupby('model')['sellingprice'].mean()

# Group the data by manufacturer (make) and calculate the average
    selling price
average_price_by_manufacturer = df.groupby('make')['sellingprice'].
    mean()

# Plotting
plt.figure(figsize=(10, 25))

# Plotting average selling price by manufacturer (make)
plt.subplot(2, 1, 1)
average_price_by_manufacturer.plot(kind='barh', color='lightgreen')
      # Use 'barh' for horizontal bar plot
plt.title('Average Selling Price by Manufacturer (Make)')
plt.xlabel('Average Selling Price')
```

```
plt.ylabel('Manufacturer (Make)')
plt.tight_layout()

plt.show()
```

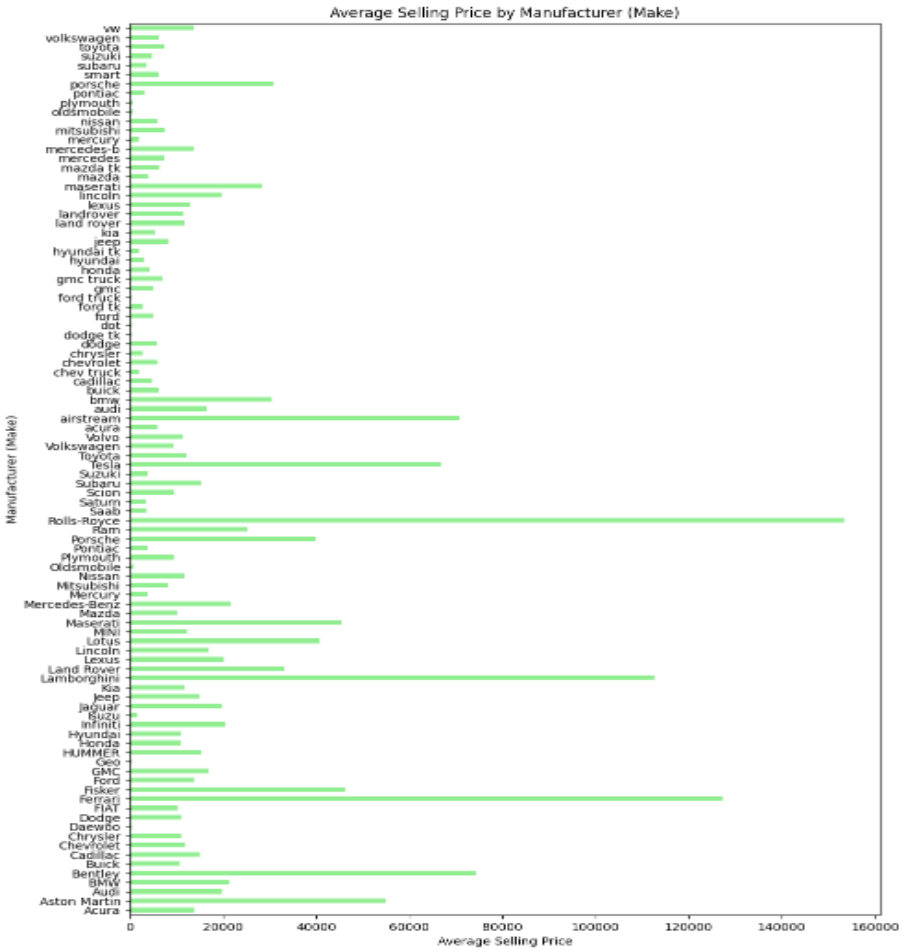

Figure 7: Visualization Image using Mat-plot Library

We also used POWER BI to represent the same information using different chart

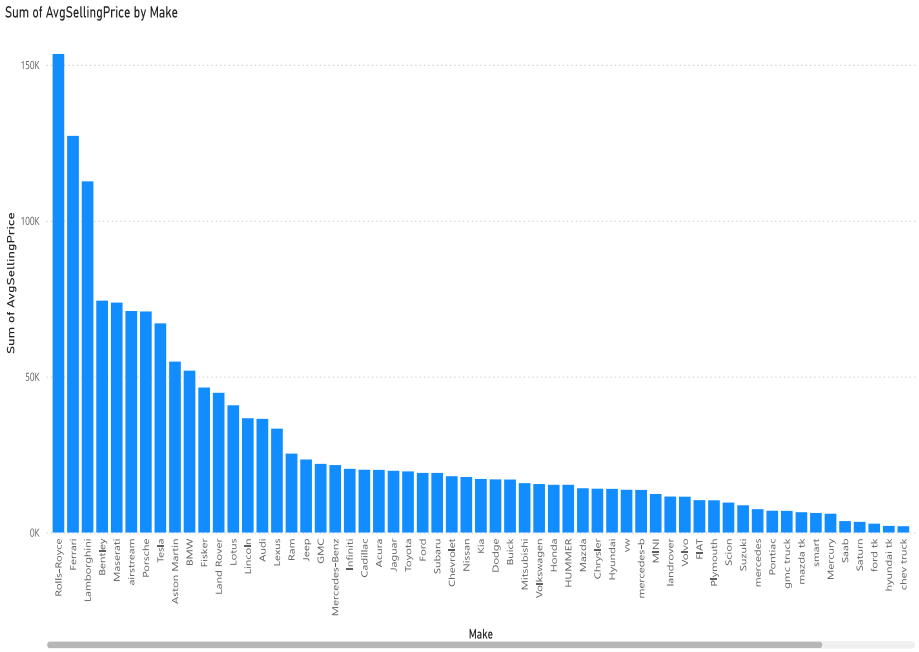

Figure 8: Visualization Image using POWER BI

Question-3

Query

```
# Question-3

import pandas as pd
import time

# Record start time
start_time = time.time()

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Determine the total revenue generated for each make over the years
total_revenue_by_make_year = df.groupby(['make'])['sellingprice'].sum()

# Convert the Series to a DataFrame, reset the index, and rename the columns
total_revenue_by_make = total_revenue_by_make_year.reset_index()
total_revenue_by_make.columns = ['Make', 'TotalRevenue']

# Print the formatted DataFrame
print(total_revenue_by_make)

# Record end time
end_time = time.time()

# Calculate processing time
processing_time = end_time - start_time

print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(start_time)))
print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(end_time)))
print("Processing Time:", processing_time, "seconds")

total_revenue_by_make.to_csv('Q3.csv', index=False)
```

Output

```
            Make  TotalRevenue
0          Acura    82715900.0
1   Aston Martin     1370300.0
2           Audi   116883675.0
3            BMW   444254638.0
4        Bentley     8626650.0
..           ...           ...
91        subaru      222625.0
92        suzuki       24050.0
93        toyota      697215.0
94    volkswagen      147500.0
95            vw      328150.0

```

```
[96 rows x 2 columns]
Start Time: 2024-04-24 14:43:47
End Time: 2024-04-24 14:43:51
Processing Time: 4.287430286407471 seconds
```

Visualization code

```python
# Graph Code for 3rd Question

import pandas as pd
import matplotlib.pyplot as plt

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Determine the total revenue generated for each make over the years
total_revenue_by_make_year = df.groupby(['make'])['sellingprice'].sum()

# Convert the Series to a DataFrame, reset the index, and rename the columns
total_revenue_by_make = total_revenue_by_make_year.reset_index()
total_revenue_by_make.columns = ['Make', 'TotalRevenue']

# Plotting
plt.figure(figsize=(12, 6))
plt.bar(total_revenue_by_make['Make'], total_revenue_by_make['TotalRevenue'], color='skyblue')
plt.xlabel('Make')
plt.ylabel('Total Revenue')
plt.title('Total Revenue Generated for Each Make')
plt.xticks(rotation=90)
plt.tight_layout()
plt.show()
```

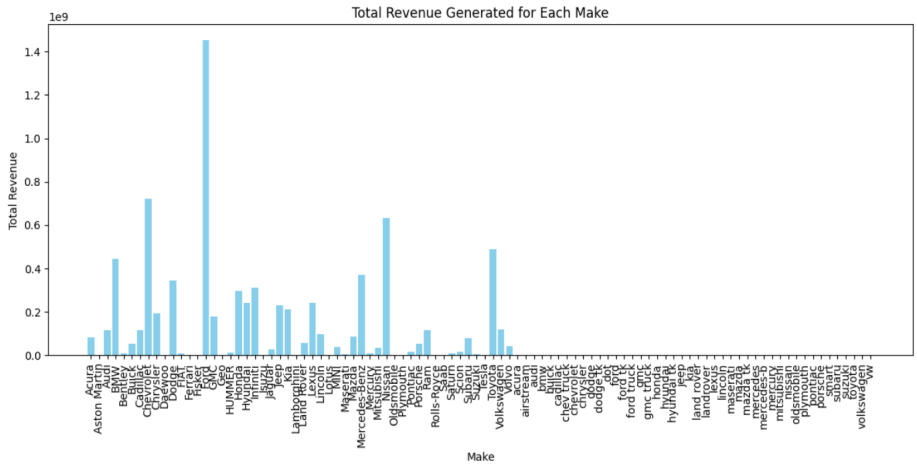

Figure 9: Visualization Image using Mat-plot Library

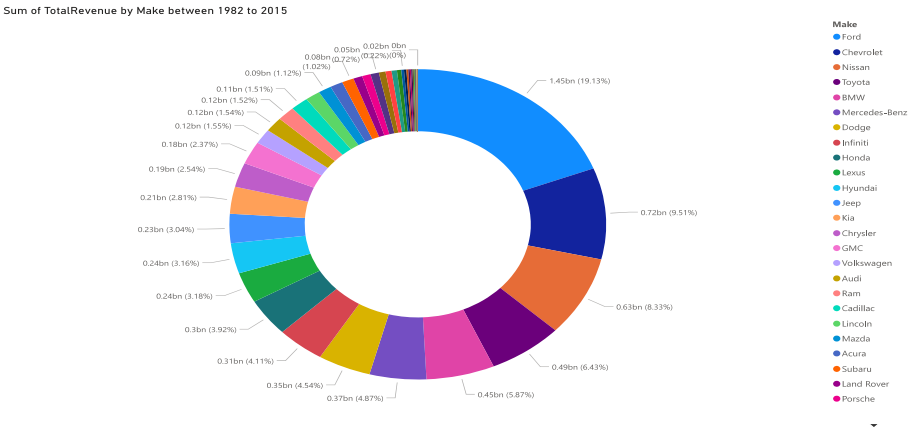

Figure 10: Visualization Image using POWER BI

Question-4

Query

```
import pandas as pd
import time

# Record start time
start_time = time.time()

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by body type and count the number of vehicles sold for each body type
vehicles_sold_by_body_type = df.groupby('body')['model'].count().reset_index(name='TotalVehiclesSold')

print(vehicles_sold_by_body_type)

# Record end time
end_time = time.time()

# Calculate processing time
processing_time = end_time - start_time

print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(start_time)))
print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(end_time)))
print("Processing Time:", processing_time, "seconds")

vehicles_sold_by_body_type.to_csv('Q4.csv', index=False)
```

Output

```
                 body  TotalVehiclesSold
0          Access Cab                232
1   Beetle Convertible                 52
2           CTS Coupe                129
3           CTS Wagon                 13
4          CTS-V Coupe                 28
..                 ...                ...
82         transit van                  7
83    tsx sport wagon                  8
84                van                570
85              wagon               2499
86           xtracab                  4

[87 rows x 2 columns]
Start Time: 2024-04-24 14:45:37
End Time: 2024-04-24 14:45:41
Processing Time: 3.9222331047058105 seconds
```

Visualization code

```
# Graph code for Question-4

```

```
import pandas as pd
import matplotlib.pyplot as plt

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by body type and count the number of vehicles sold for each body type
vehicles_sold_by_body_type = df.groupby('body')['model'].count().reset_index(name='TotalVehiclesSold')

# Plotting
plt.figure(figsize=(10, 20))
plt.scatter(vehicles_sold_by_body_type['TotalVehiclesSold'], vehicles_sold_by_body_type['body'], color='skyblue')  # Reverse x and y
plt.title('Number of Vehicles Sold by Body Type')
plt.xlabel('Number of Vehicles Sold')
plt.ylabel('Body Type')
plt.yticks(rotation=45, ha='right')  # Rotate y-axis labels
plt.grid(True)
plt.tight_layout()
plt.show()
```

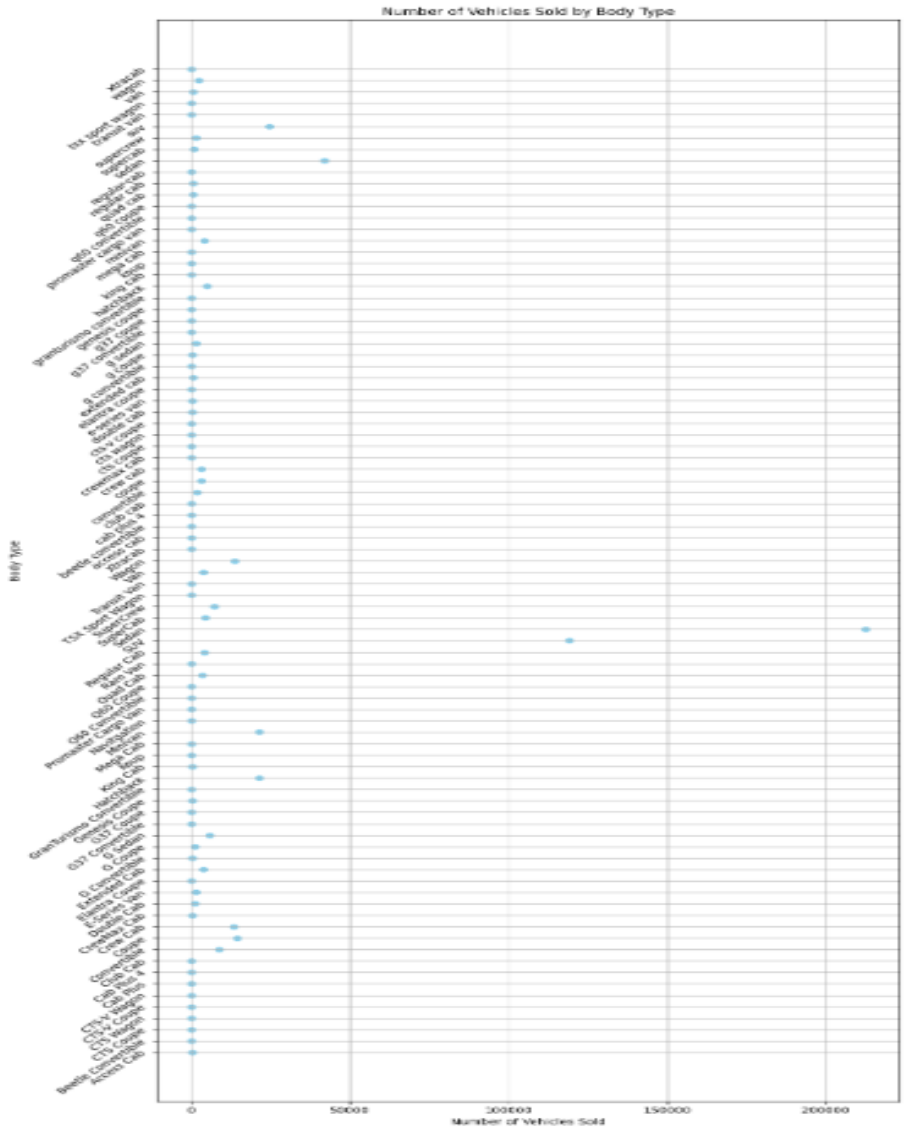

Figure 11: Visualization Image

Question-5

Query

```
import pandas as pd
import time

# Record start time
start_time = time.time()

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by vehicle make and calculate the average odometer reading
average_odometer_by_make = df.groupby('make')['odometer'].mean()

# Convert the Series to a DataFrame, reset the index, and rename the columns
average_odometer_by_make_df = average_odometer_by_make.reset_index()
average_odometer_by_make_df.columns = ['Make', 'AverageOdometer']

# Print the formatted DataFrame
print("Average Odometer Reading by Vehicle Make:")
print(average_odometer_by_make_df)

# Record end time
end_time = time.time()

# Calculate processing time
processing_time = end_time - start_time

print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(start_time)))
print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(end_time)))
print("Processing Time:", processing_time, "seconds")

# Write the output to a CSV file
average_odometer_by_make_df.to_csv('Q5.csv', index=False)
```

Output

```
Average Odometer Reading by Vehicle Make:
           Make  AverageOdometer
0         Acura     85829.219285
1   Aston Martin     26603.640000
2          Audi     66040.140913
3           BMW     64298.103096
4       Bentley     39239.698276
..          ...              ...
91       subaru    134812.600000
92       suzuki     80901.400000
93       toyota    145414.336842
94   volkswagen    103235.083333
95           vw     67813.583333
```

```
[96 rows x 2 columns]
Start Time: 2024-04-24 14:49:01
End Time: 2024-04-24 14:49:05
Processing Time: 3.7955965995788574 seconds
```

Visualization code

```python
# Graph code for Question-5

import pandas as pd
import matplotlib.pyplot as plt

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by vehicle make and year, then calculate the average odometer reading
average_odometer_by_make_year = df.groupby(['make', 'year'])['odometer'].mean()

# Convert the Series to a DataFrame, reset the index
average_odometer_by_make_year_df = average_odometer_by_make_year.reset_index()

# Aggregate the average odometer readings by vehicle make
average_odometer_by_make = average_odometer_by_make_year_df.groupby('make')['odometer'].mean()

# Plotting
plt.figure(figsize=(12, 30))
average_odometer_by_make.plot(kind='barh', color='skyblue')  # Change kind to 'barh' for horizontal bar plot
plt.title('Average Odometer Reading by Vehicle Make')
plt.xlabel('Average Odometer Reading')
plt.ylabel('Vehicle Make')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

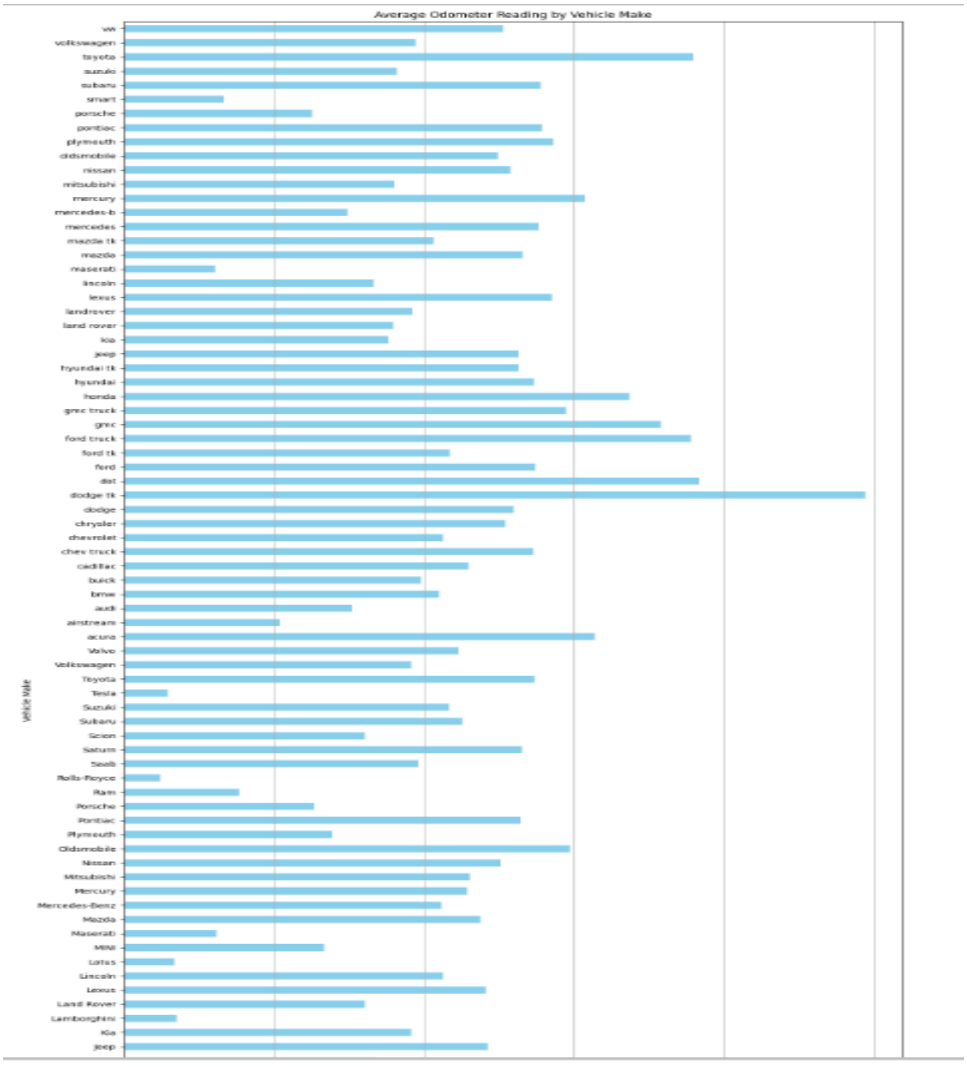

Figure 12: Visualization Image

Question-6

Query

```
# Question-6

import pandas as pd
import time

# Record start time
start_time = time.time()

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by make and model, then count the total sales for each group
top_selling_models = df.groupby(['make', 'model']).size().reset_index(name='TotalSales')

# Sort the data by make and total sales in descending order
top_selling_models = top_selling_models.sort_values(by=['make', 'TotalSales'], ascending=[True, False])

# Print the formatted DataFrame
print('+------------+---------------+----------+')
print('|        Make|          Model|TotalSales|')
print('+------------+---------------+----------+')

for index, row in top_selling_models.iterrows():
    print('|{:12}|{:15}|{:10}|'.format(row['make'], row['model'], row['TotalSales']))

print('+------------+---------------+----------+')

# Record end time
end_time = time.time()

# Calculate processing time
processing_time = end_time - start_time

print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(start_time)))
print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(end_time)))
print("Processing Time:", processing_time, "seconds")
```

Output

```
+-------------+---------------+----------+
|         Make|          Model|TotalSales|
+-------------+---------------+----------+
|Acura        |TL             |      2135|
|Acura        |MDX            |      1581|
|Acura        |TSX            |      1119|
|Acura        |RDX            |       420|
|Acura        |RSX            |       151|
|Acura        |ILX            |       140|
|Acura        |RL             |       108|
|Acura        |CL             |        97|
|Acura        |Integra        |        44|
|Acura        |ZDX            |        39|
|Acura        |TSX Sport Wagon|        36|
|Acura        |RLX            |        17|
|Acura        |Legend         |         9|
|Acura        |mdx            |         4|
|Acura        |TLX            |         1|
|Aston Martin |V8 Vantage     |        17|
|Aston Martin |DB9            |         6|
|Aston Martin |Rapide         |         2|
**** Placed only small piece of output only ****

+-------------+---------------+----------+
Start Time: 2024-04-24 14:50:29
End Time: 2024-04-24 14:50:33
Processing Time: 4.164934873580933 seconds
```

Visualization code

```
# Graph code for question-6

import pandas as pd
import matplotlib.pyplot as plt

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by make and model, then count the total sales for each group
top_selling_models = df.groupby(['make', 'model']).size().reset_index(name='TotalSales')

# Sort the data by make and total sales in descending order
top_selling_models = top_selling_models.sort_values(by=['make', 'TotalSales'], ascending=[True, False])

# Get unique makes
unique_makes = top_selling_models['make'].unique()

# Plotting
for make in unique_makes:
    make_data = top_selling_models[top_selling_models['make'] == make]
    plt.figure(figsize=(12, 6))
```

```
    plt.bar(make_data['model'], make_data['TotalSales'], color='
    skyblue')
    plt.title(f'Top Selling Models for {make}')
    plt.xlabel('Model')
    plt.ylabel('Total Sales')
    plt.xticks(rotation=45, ha='right')
    plt.grid(axis='y')
    plt.tight_layout()
    plt.show()
```

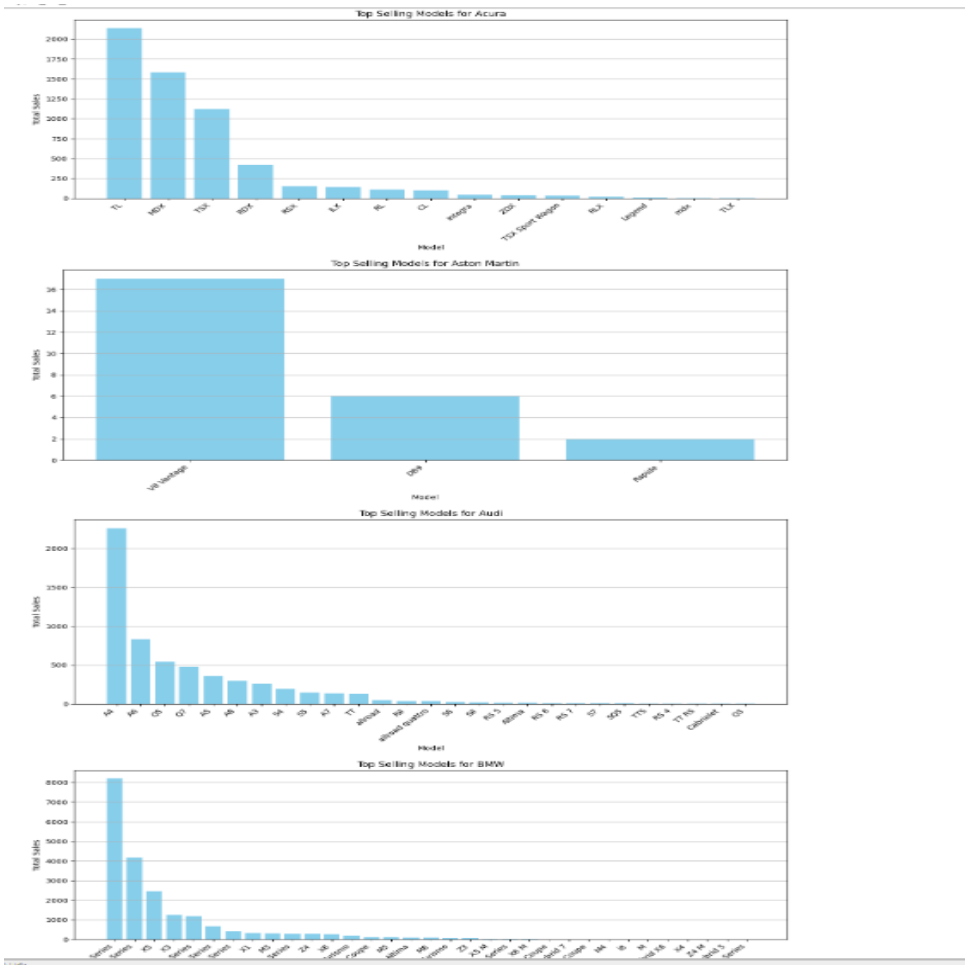

Figure 13: Visualization Image

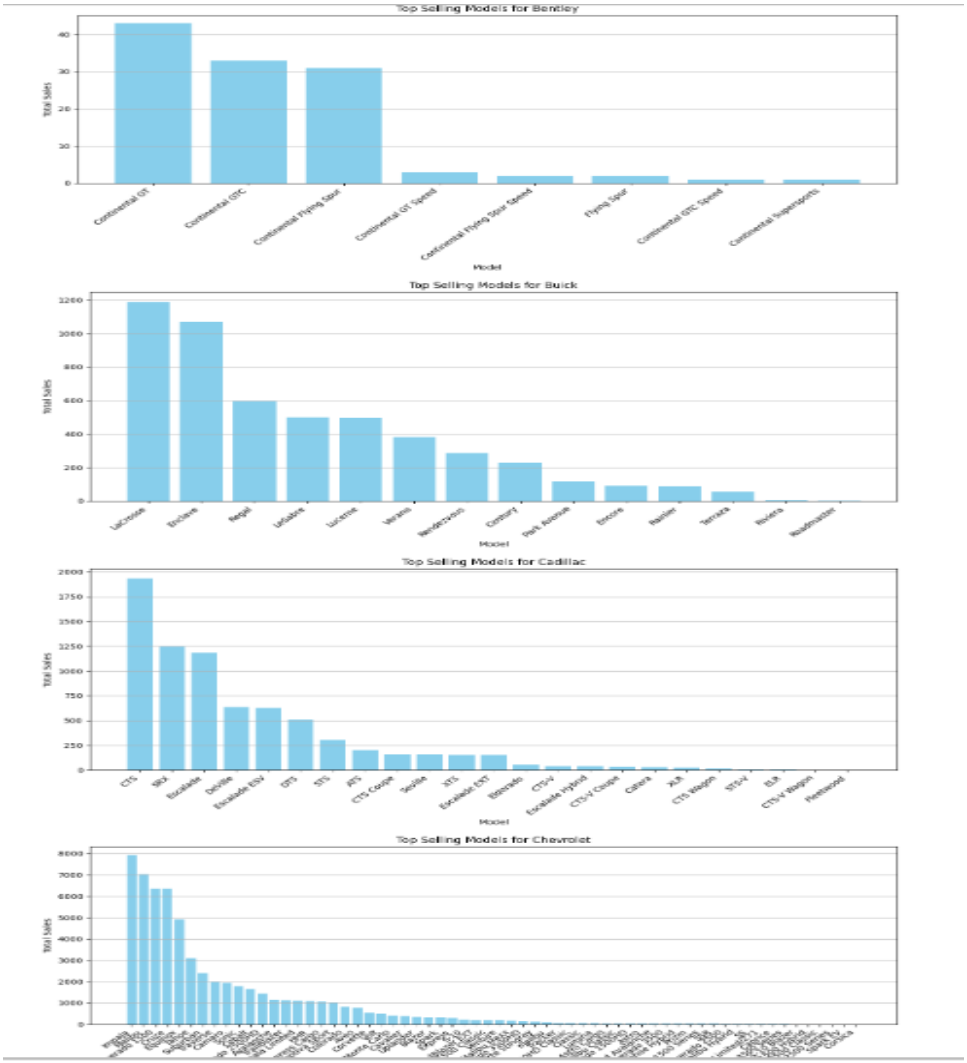

Figure 14: Visualization Image

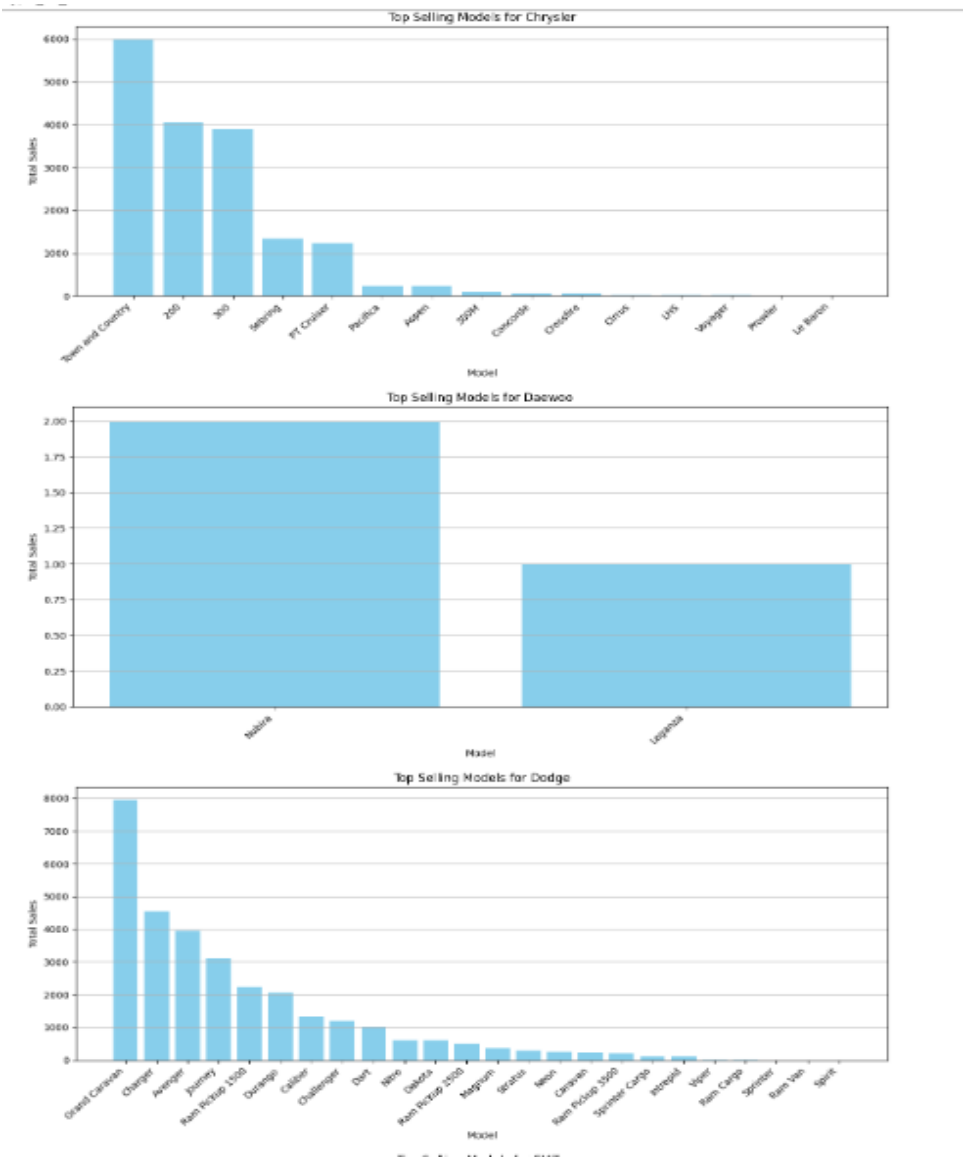

Figure 15: Visualization Image

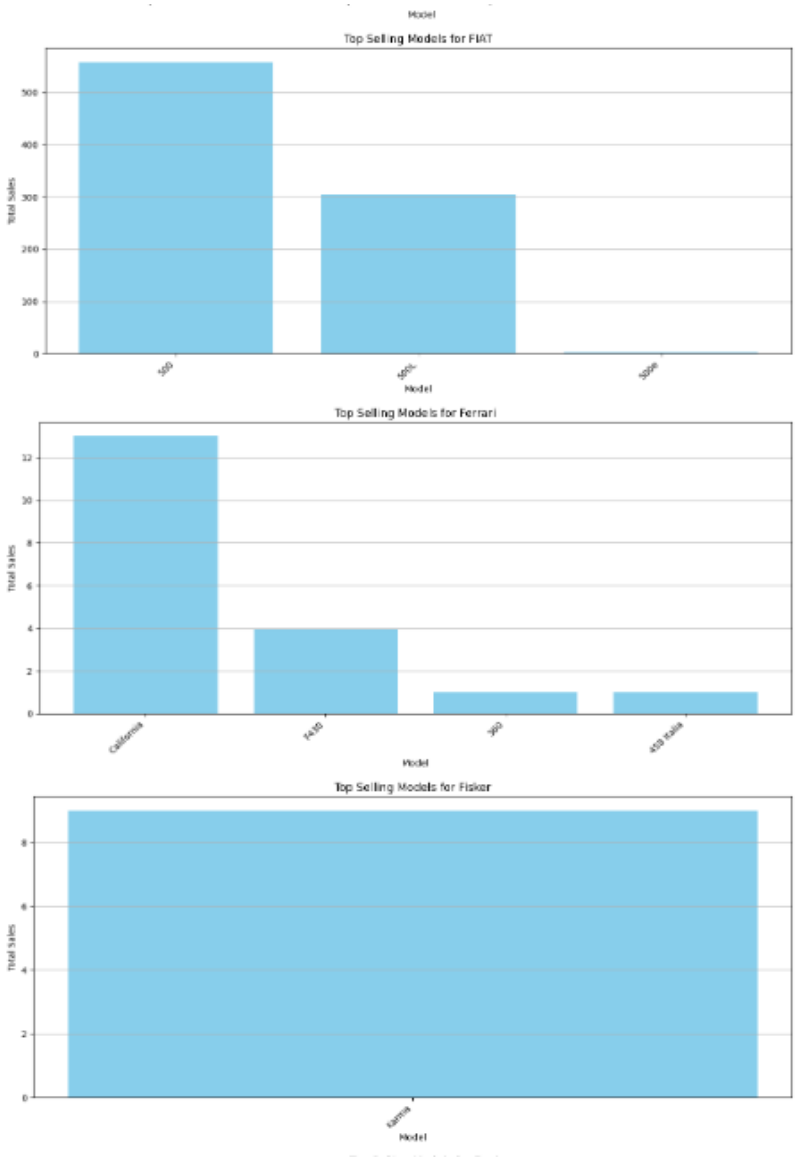

Figure 16: Visualization Image

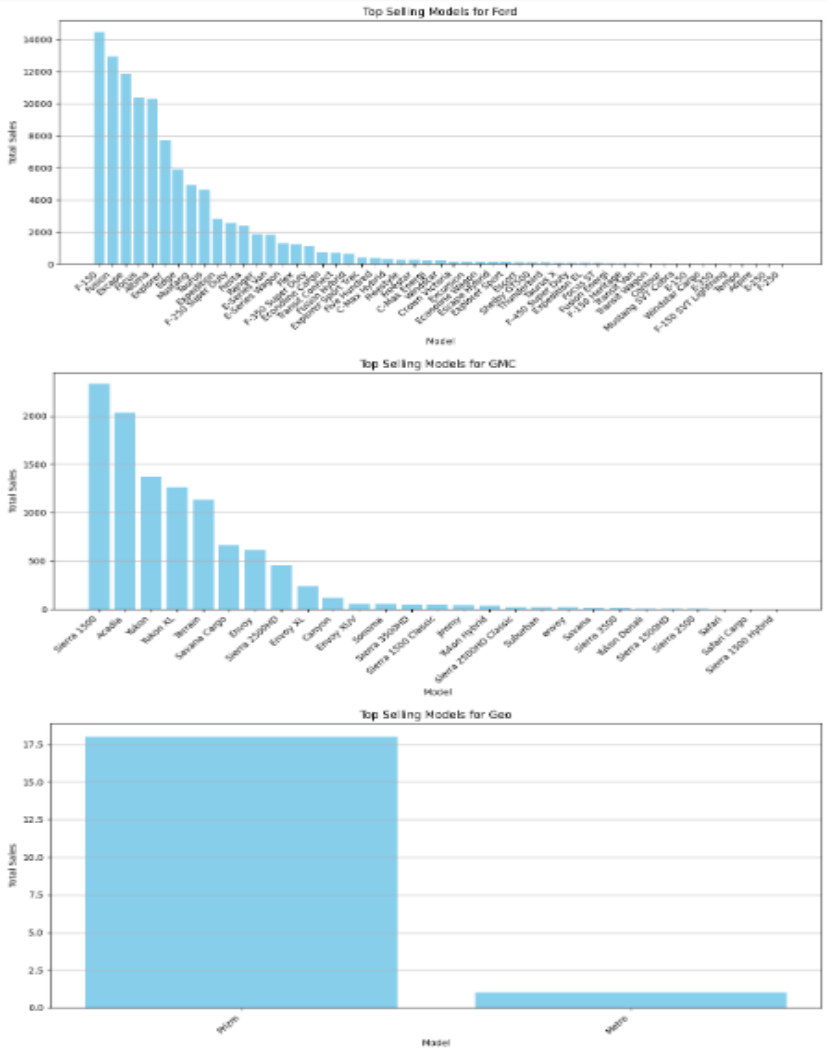

Figure 17: Visualization Image

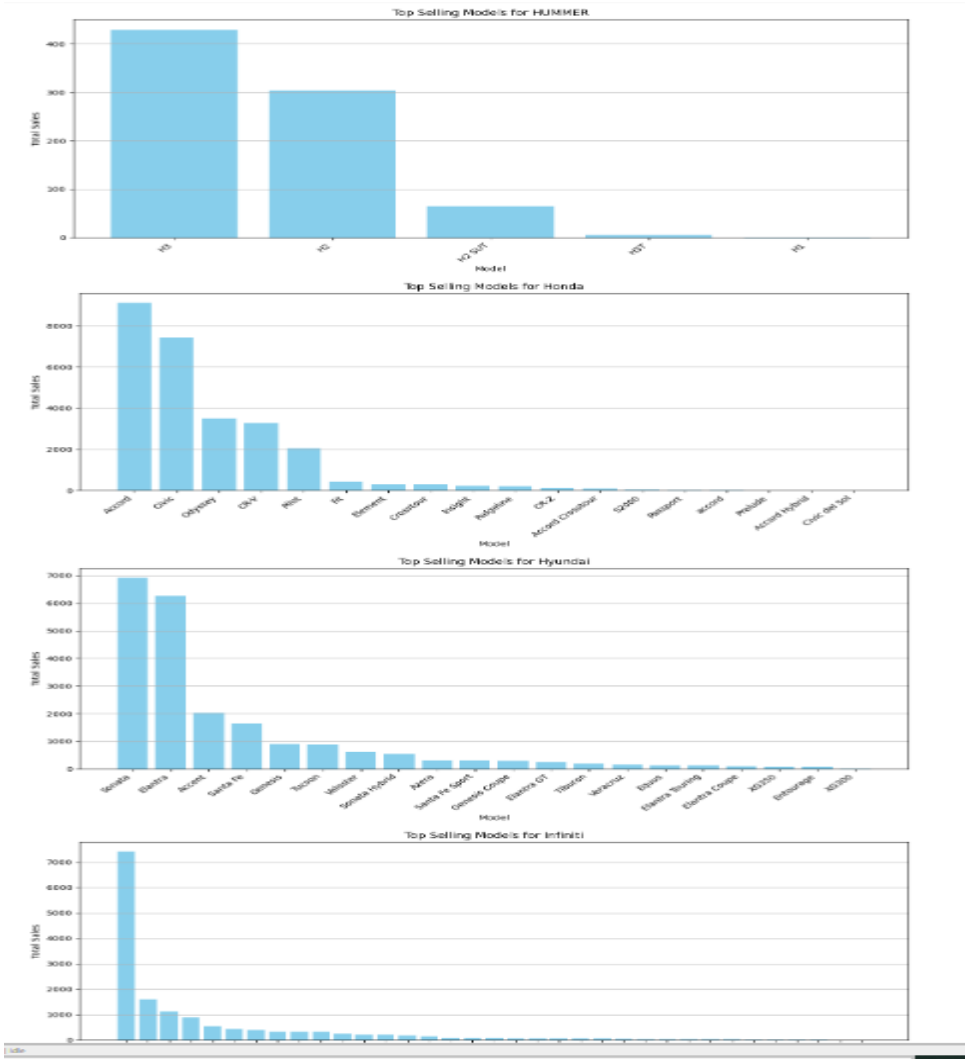

Figure 18: Visualization Image

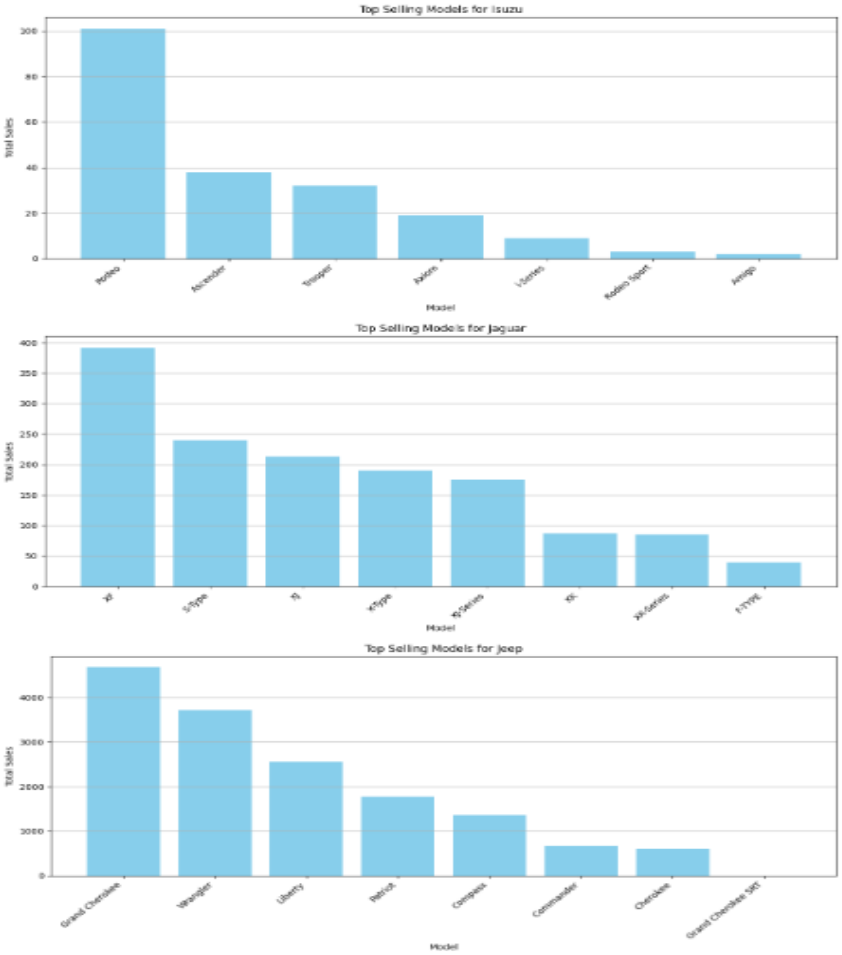

Figure 19: Visualization Image

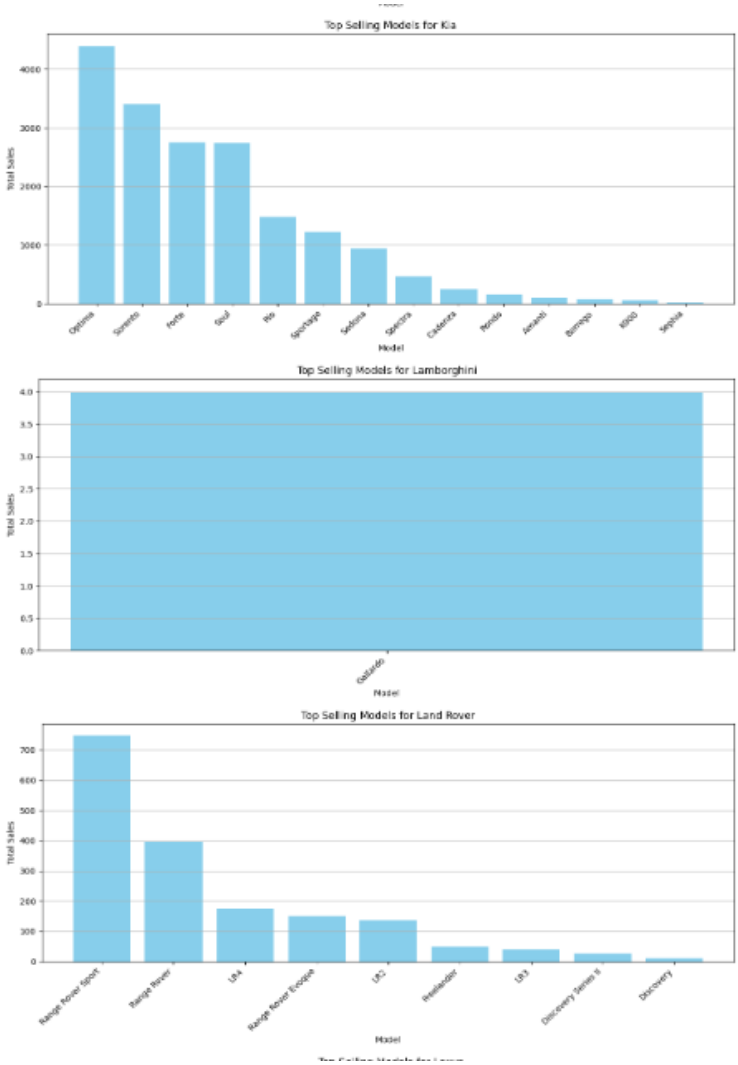

Figure 20: Visualization Image

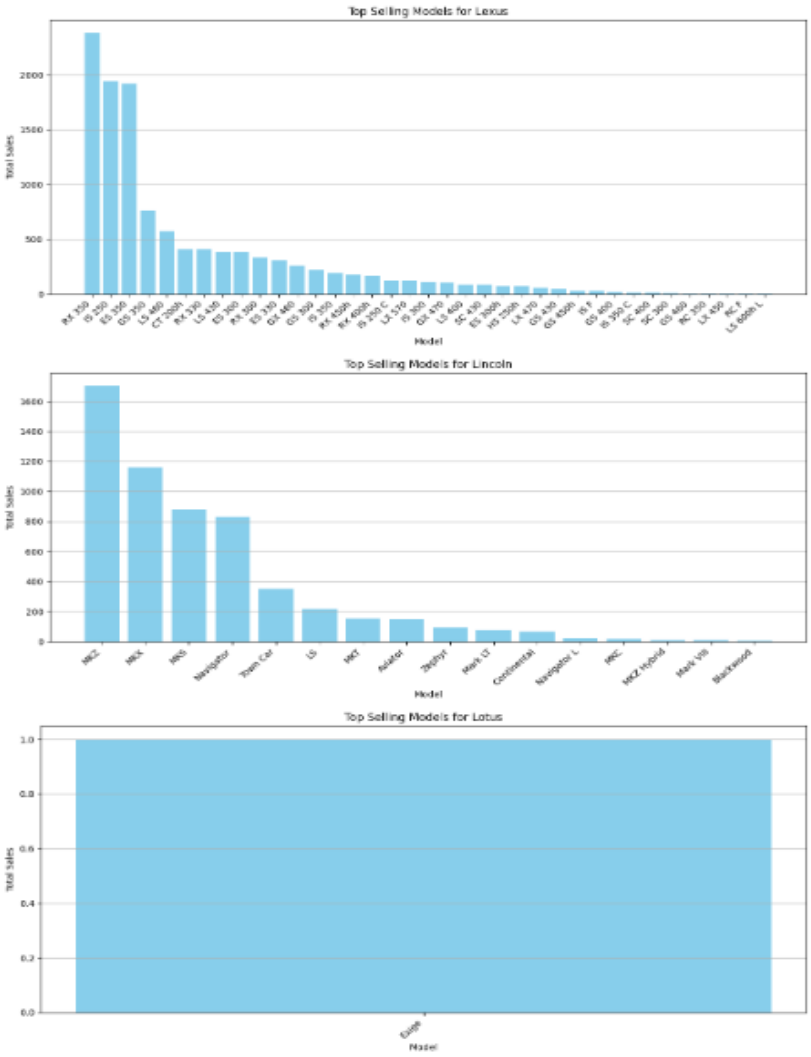

Figure 21: Visualization Image

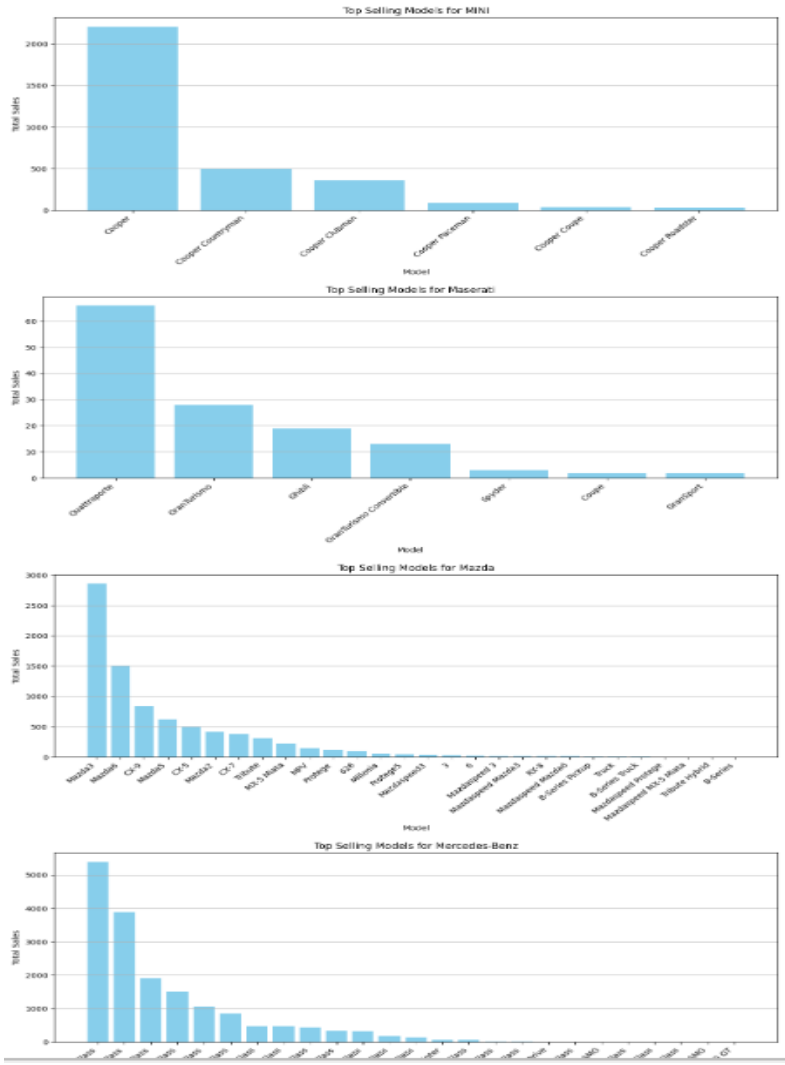

Figure 22: Visualization Image

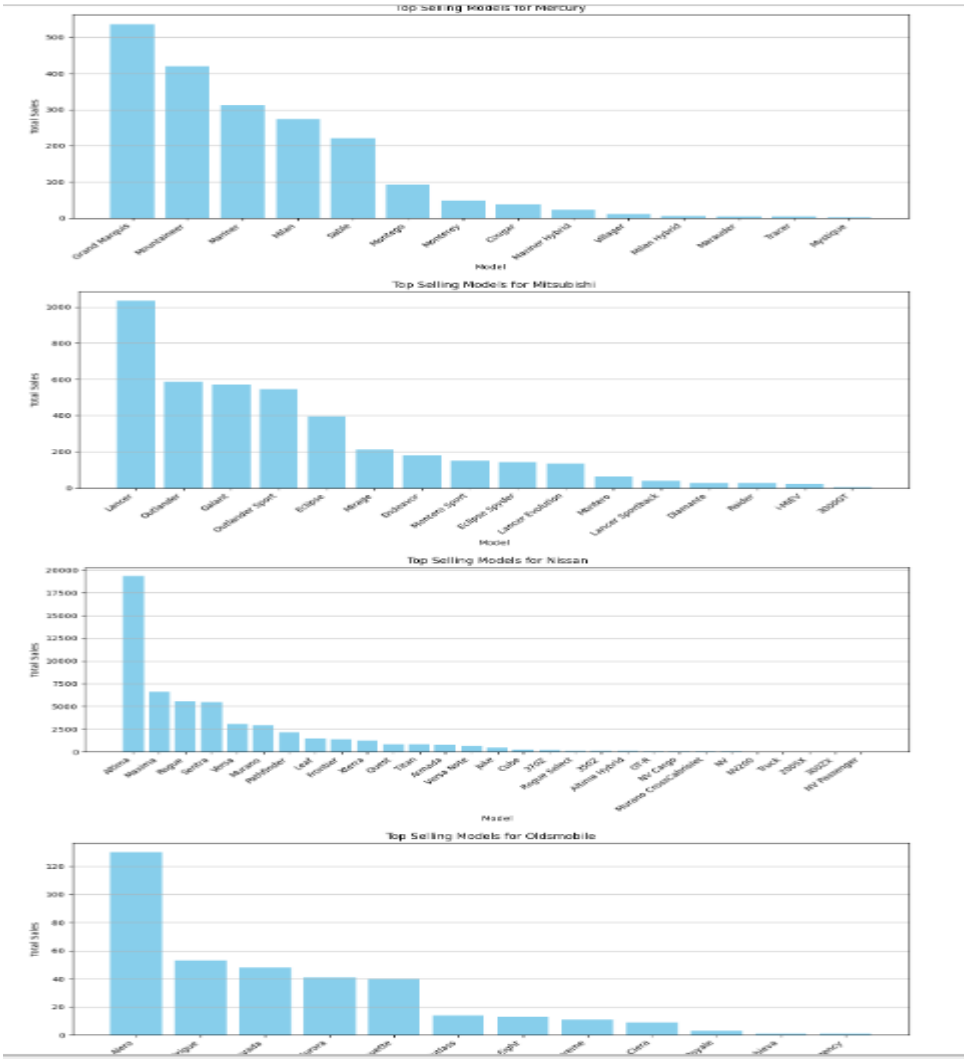

Figure 23: Visualization Image

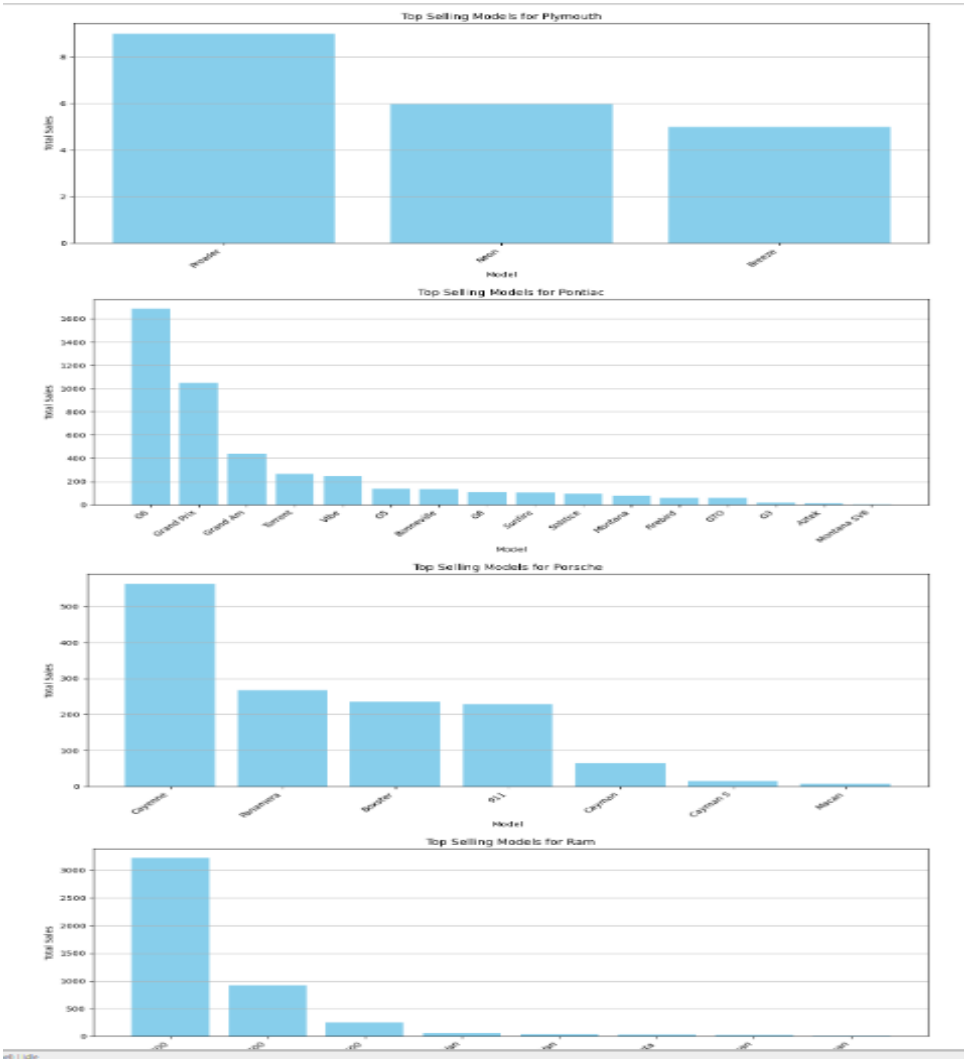

Figure 24: Visualization Image

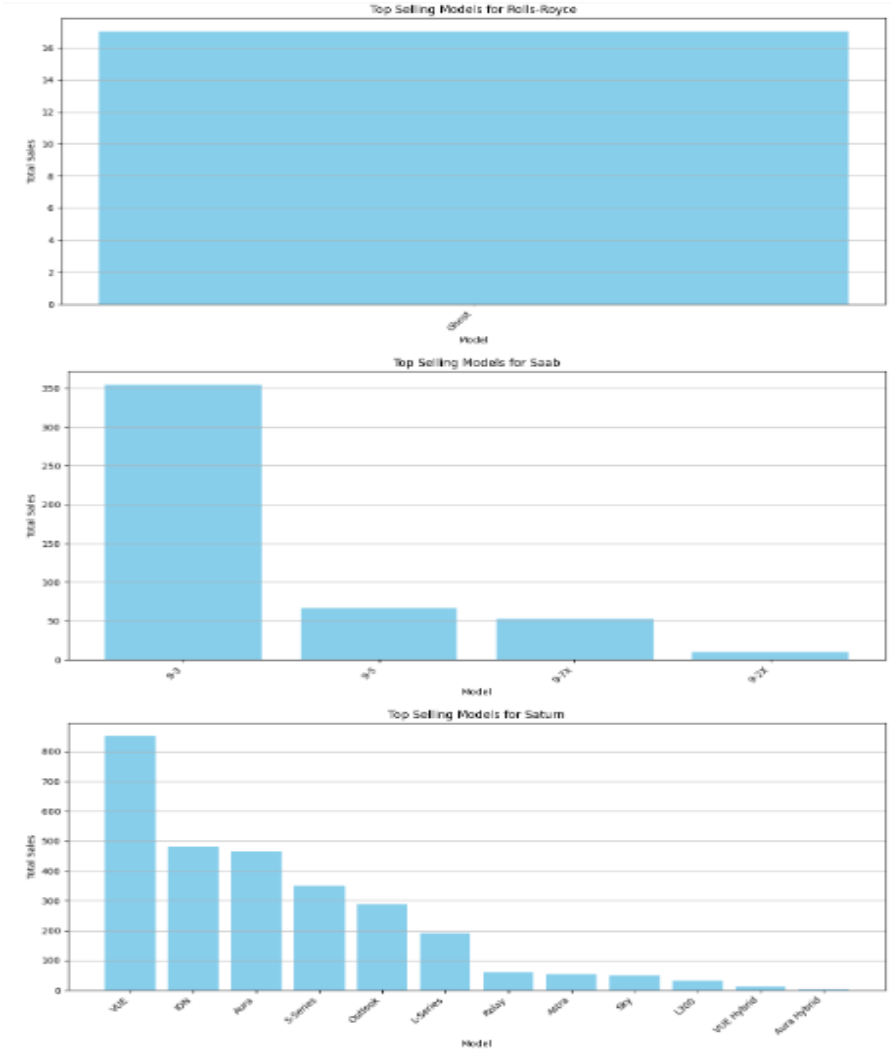

Figure 25: Visualization Image

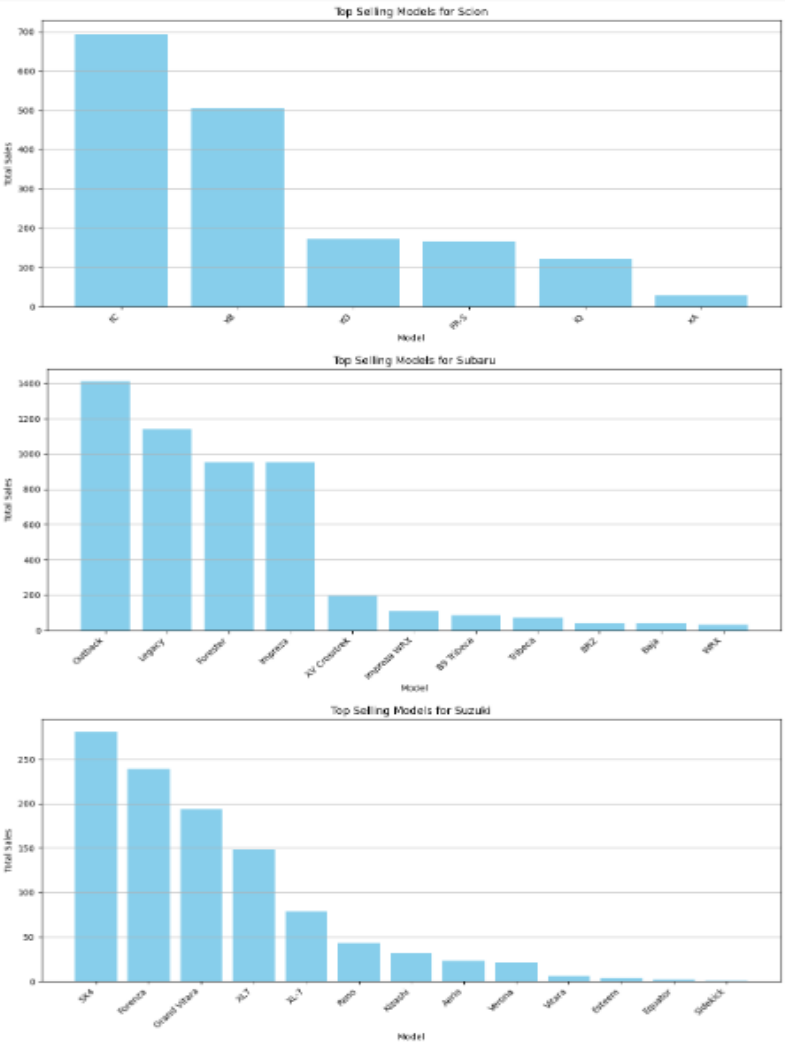

Figure 26: Visualization Image

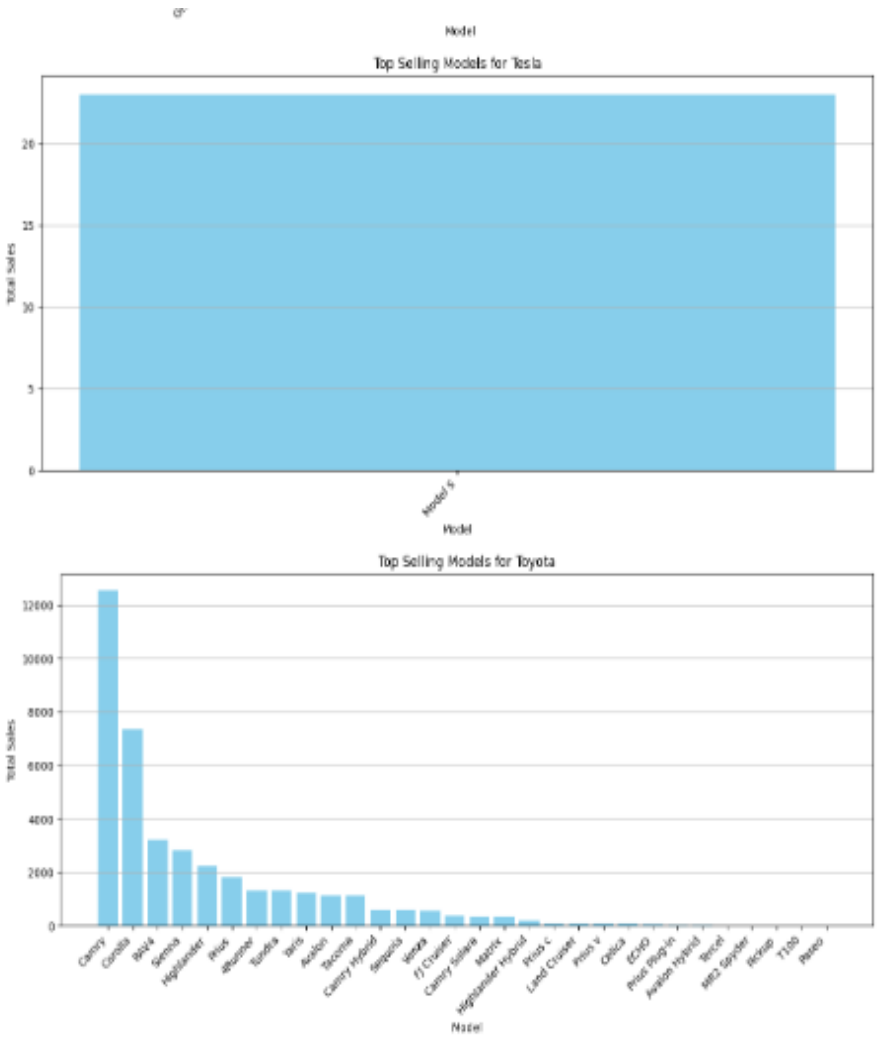

Figure 27: Visualization Image

Question-7

Query

```
import pandas as pd
import time

state_mapping = {
    'al': 'Alabama', 'ak': 'Alaska', 'az': 'Arizona', 'ar': '
    Arkansas', 'ca': 'California',
    'co': 'Colorado', 'ct': 'Connecticut', 'de': 'Delaware', 'fl':
    'Florida', 'ga': 'Georgia',
    'hi': 'Hawaii', 'id': 'Idaho', 'il': 'Illinois', 'in': 'Indiana
    ', 'ia': 'Iowa', 'ks': 'Kansas',
    'ky': 'Kentucky', 'la': 'Louisiana', 'me': 'Maine', 'md': '
    Maryland', 'ma': 'Massachusetts',
    'mi': 'Michigan', 'mn': 'Minnesota', 'ms': 'Mississippi', 'mo':
     'Missouri', 'mt': 'Montana',
    'ne': 'Nebraska', 'nv': 'Nevada', 'nh': 'New Hampshire', 'nj':
    'New Jersey', 'nm': 'New Mexico',
    'ny': 'New York', 'nc': 'North Carolina', 'nd': 'North Dakota',
     'oh': 'Ohio', 'ok': 'Oklahoma',
    'or': 'Oregon', 'pa': 'Pennsylvania', 'ri': 'Rhode Island', 'sc
    ': 'South Carolina',
    'sd': 'South Dakota', 'tn': 'Tennessee', 'tx': 'Texas', 'ut': '
    Utah', 'vt': 'Vermont',
    'va': 'Virginia', 'wa': 'Washington', 'wv': 'West Virginia', '
    wi': 'Wisconsin', 'wy': 'Wyoming'
}

# Record start time
start_time = time.time()

# Read the data into a DataFrame
df = pd.read_csv('output.csv')

# Group the data by state and count the total vehicle sales for
    each state
vehicle_sales_by_state = df.groupby('state').size().reset_index(
    name='TotalSales')

# Convert state column to lowercase for case insensitivity
vehicle_sales_by_state['state'] = vehicle_sales_by_state['state'].
    str.lower()

# Sort the data by total sales in descending order
vehicle_sales_by_state = vehicle_sales_by_state.sort_values(by='
    TotalSales', ascending=False)

# Replace state abbreviations with full names
vehicle_sales_by_state['state'] = vehicle_sales_by_state['state'].
    map(state_mapping)

# Get the top 5 states
top_5_states = vehicle_sales_by_state.head(5)

# Print the top 5 states
print("Top 5 States with the Highest Number of Vehicle Sales:")
```

```
print(top_5_states)

# Record end time
end_time = time.time()

# Calculate processing time
processing_time = end_time - start_time

print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
print("Processing Time:", processing_time, "seconds")

top_5_states.to_csv('Q7.csv', index=False)
```

Output

```
Top 5 States with the Highest Number of Vehicle Sales:
           state  TotalSales
31       Florida       82945
29    California       73148
54  Pennsylvania       53907
59         Texas       45913
32       Georgia       34750
Start Time: 2024-04-24 14:54:49
End Time: 2024-04-24 14:54:53
Processing Time: 3.996584177017212 seconds
```

Visualization code

```
# Graph code for Question-7

import pandas as pd
import matplotlib.pyplot as plt

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by state and count the total vehicle sales for
    each state
vehicle_sales_by_state = df.groupby('state').size().reset_index(
    name='TotalSales')

# Sort the data by total sales in descending order
vehicle_sales_by_state = vehicle_sales_by_state.sort_values(by='
    TotalSales', ascending=False)

# Get the top 5 states
top_5_states = vehicle_sales_by_state.head(5)

# Mapping of state abbreviations to full names
state_mapping = {
    'ca': 'California',
    'tx': 'Texas',
    'fl': 'Florida',
    'ga': 'Georgia',
    'pa': 'Pennsylvania'
```

```
}

# Replace state abbreviations with full names
top_5_states_copy = top_5_states.copy()  # Create a copy of the
    DataFrame slice
top_5_states_copy['state'] = top_5_states_copy['state'].map(
    state_mapping)

# Plotting
plt.figure(figsize=(8, 8))
plt.pie(top_5_states_copy['TotalSales'], labels=top_5_states_copy['
    state'], autopct='%1.1f%%', startangle=140)
plt.title('Top 5 States with the Highest Number of Vehicle Sales')
plt.axis('equal')
plt.tight_layout()
plt.show()
```

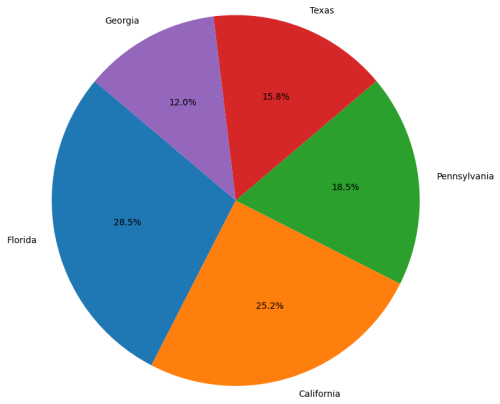

Figure 28: Visualization Image using Mat-plot Library

We also used POWER BI to represent the same information using different chart

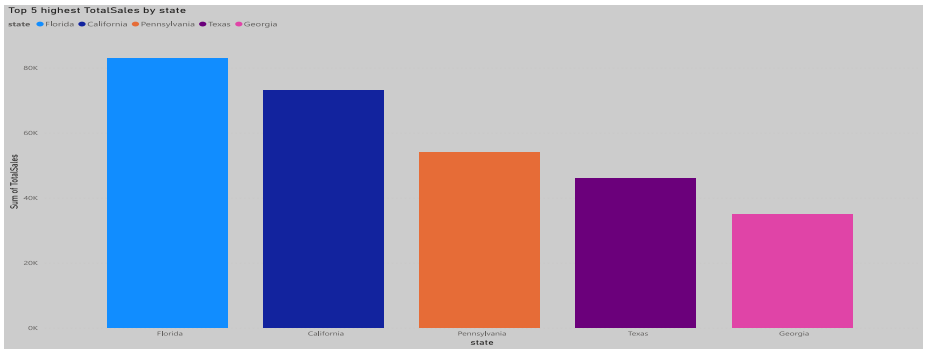

Figure 29: Visualization Image using POWER BI

Question-8

Query

```python
# Question-8

import pandas as pd
import time

# Record start time
start_time = time.time()

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by make and calculate the average MMR value for each make
average_mmr_by_make = df.groupby('make')['mmr'].mean()

# Convert the Series to a DataFrame, reset the index, and rename the columns
average_mmr_by_make_df = average_mmr_by_make.reset_index()
average_mmr_by_make_df.columns = ['Make', 'AvgMMR']
print(average_mmr_by_make_df)

# Record end time
end_time = time.time()

# Calculate processing time
processing_time = end_time - start_time

print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(start_time)))
print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.localtime(end_time)))
print("Processing Time:", processing_time, "seconds")

```

```
# Write the output to a CSV file
average_mmr_by_make_df.to_csv('Q8.csv', index=False)
```

Output

```
             Make         AvgMMR
0           Acura   14076.813252
1    Aston Martin   53560.000000
2            Audi   20080.213835
3             BMW   21575.547806
4         Bentley   75928.448276
..            ...            ...
91        subaru     3911.666667
92        suzuki     4805.000000
93        toyota     7367.631579
94    volkswagen     5984.375000
95            vw   13847.916667

[96 rows x 2 columns]
Start Time: 2024-04-24 14:57:01
End Time: 2024-04-24 14:57:05
Processing Time: 4.0517213344573975 seconds
```

Visualization code

```
# Graph Code for Question-8

import pandas as pd
import matplotlib.pyplot as plt

# Read the data into a DataFrame
# Replace 'output.csv' with the actual path to your data file
df = pd.read_csv('output.csv')

# Group the data by make and calculate the average MMR value for
    each make
average_mmr_by_make = df.groupby('make')['mmr'].mean().sort_values(
    ascending=False)

# Plotting
plt.figure(figsize=(12, 20))
average_mmr_by_make.plot(kind='barh', color='skyblue')  # Change
    kind to 'barh' for horizontal bar plot
plt.title('Average MMR Value by Make')
plt.xlabel('Average MMR Value')
plt.ylabel('Make')
plt.grid(axis='x')
plt.tight_layout()
plt.show()
```

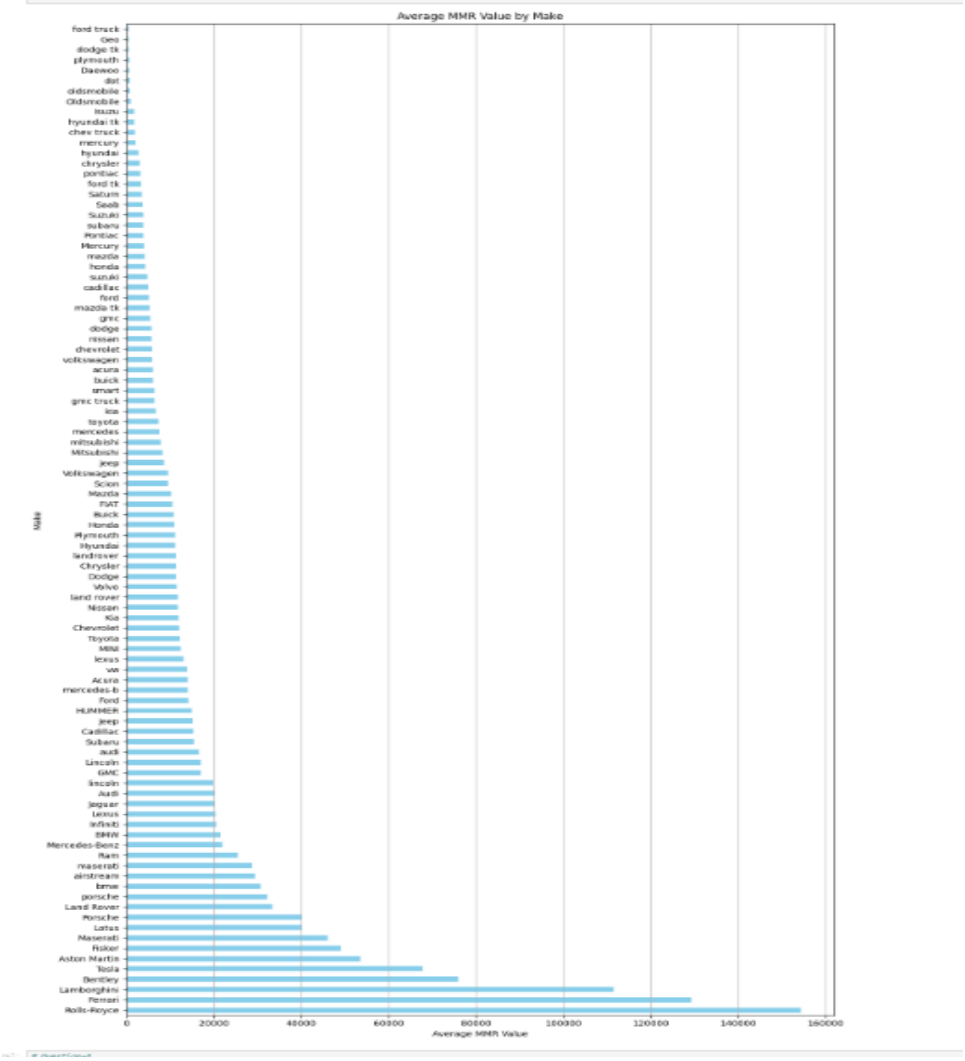

Figure 30: Visualization Image

8 Insights

1. **Year-wise, identify the top-selling vehicle for each manufacturer:**
    - By analyzing the top-selling vehicles for each manufacturer year-wise, we can understand the popularity of specific models over time. This insight can help manufacturers make informed decisions about production and marketing strategies.
2. **Calculate the average selling price for each car type or manufacturer:**
    - Understanding the average selling price for different car types or manufacturers provides valuable insights into market trends and consumer preferences. It helps in pricing strategies, identifying competitive advantages, and evaluating the overall market positioning of different brands.
3. **Determine the total revenue generated for each make over the years:**
    - Tracking the total revenue generated for each make over the years provides insights into the financial performance of different automotive brands. It helps in assessing brand profitability, market share, and overall brand health in the automotive industry.
4. **Count the number of vehicles sold for each body type:**
    - Analyzing the number of vehicles sold for each body type helps in understanding consumer preferences and market demand for different vehicle categories. It provides insights into which body types are more popular among consumers and can inform production and inventory management decisions.
5. **Average Odometer Reading by Vehicle Make:**
    - Examining the average odometer reading by vehicle make offers insights into the usage patterns and driving behaviors of vehicles from different manufacturers. It can help in assessing the reliability and longevity of vehicles from various brands and inform decisions related to vehicle maintenance and resale value.
6. **Top Selling models for each make/manufacturer:**
    - Identifying the top-selling models for each make or manufacturer helps in understanding consumer preferences within specific brand portfolios. It provides insights into which models contribute the most to overall sales and can guide marketing efforts and product development strategies.

7. **Identify the Top 5 States with the Highest Number of Vehicle Sales:**

    - Knowing the top states with the highest number of vehicle sales can provide insights into regional market trends and preferences. It helps in targeting marketing campaigns, allocating resources effectively, and understanding geographic variations in consumer behavior.

8. **Identify the Average MMR Value by Make and Model:**

    - Analyzing the average MMR (Manheim Market Report) value by make and model helps in assessing the resale value and depreciation rates of different vehicles. It provides insights into the perceived value of vehicles from different manufacturers and can inform decisions related to vehicle pricing and inventory management.

9 Metrics: Suitable V's for Analytical Goals

1. **Year-wise, identify the top-selling vehicle for each manufacturer.**

    - **Value:** This analysis provides valuable insights into the performance of each manufacturer's vehicles over time, aiding in strategic decision-making.

2. **Calculate the average selling price for each car type or manufacturer.**

    - **Volume:** The sheer volume of sales data is essential for accurately calculating the average selling price for different car types or manufacturers.

3. **Determine the total revenue generated for each make over the years.**

    - **Volume:** Analyzing the volume of sales data over multiple years is crucial for determining the total revenue generated for each vehicle make.

4. **Count the number of vehicles sold for each body type.**

    - **Volume:** The volume of sales data is essential for accurately counting the number of vehicles sold for each body type.

5. **Average Odometer Reading by Vehicle Make.**

    - **Variety:** Analyzing the variety of odometer readings across different vehicle makes ensures a comprehensive understanding of average odometer readings.

6. **Top Selling models for each make/manufacturer.**

- **Variety:** Identifying the variety of top-selling models for each vehicle make or manufacturer ensures comprehensive coverage of popular models.

7. **Identify the Top 5 States with the Highest Number of Vehicle Sales.**

   - **Volume:** Analyzing the volume of sales data is essential for identifying the states with the highest number of vehicle sales.

8. **Identify the Average MMR Value by Make and Model.**

   - **Variety:** Analyzing the variety of MMR values across different vehicle makes and models ensures a comprehensive understanding of market pricing.

10 Conclusion

In conclusion, our exploration of the "Vehicle Sales and Market Trends Dataset" has yielded profound insights into the multifaceted dynamics of the automotive industry. Through a systematic approach encompassing comprehensive market analysis and advanced predictive modeling techniques, we've uncovered intricate trends, patterns, and underlying factors that shape pricing strategies and consumer preferences.

One of the significant contributions of our project lies in the generation of actionable business insights for a wide spectrum of stakeholders, including automotive industry professionals, dealerships, and financial institutions. By distilling complex data into digestible insights, we've empowered decision-makers to navigate the competitive landscape with confidence, optimizing pricing strategies, refining inventory management practices, and strategically positioning themselves in the market.

Central to our success has been the adept utilization of cutting-edge tools and technologies such as Apache Spark, Jupyter Notebook, and Power BI. These robust platforms have enabled us to process and analyze vast volumes of data with efficiency and precision, while maintaining the integrity and consistency of our findings. The visual representation of our analyses through Power BI has been instrumental in enhancing the clarity and accessibility of our insights, facilitating seamless interpretation and informed decision-making.

Our project objectives, meticulously crafted to address key facets of the automotive market, have been met through rigorous data analysis and visualization. By identifying top-selling vehicles for each manufacturer, calculating average selling prices, and assessing revenue generation trends, we've provided a granular understanding of market dynamics. Moreover, our comparative analysis across different vehicle types, makes, and models, coupled with an exploration of geographical influences on sales trends and pricing, has enriched our comprehension of the market landscape.

The preparatory steps taken, including the thorough cleaning of the dataset and the formulation of tailored queries, have been pivotal in ensuring the reliability and accuracy of our analyses. By addressing outliers and meticulously curating our dataset, we've laid a robust foundation for deriving meaningful insights and driving informed decision-making processes.

In essence, our project underscores the transformative potential of data-driven approaches in shaping the future of the automotive industry. The wealth of insights gleaned from the rich and diverse dataset not only illuminates current market trends but also paves the way for future research endeavors, including predictive modeling initiatives aimed at forecasting vehicle prices and anticipating evolving consumer behaviors. As we continue to harness the power of data analytics, we stand poised to navigate the ever-evolving landscape of the automotive industry with foresight and agility.

11 References

Kaggle Dataset

Vehicle Sales and Market Trends Dataset. Retrieved from Kaggle: `https://www.kaggle.com/datasets/syedanwarafridi/vehicle-sales-data`.

Github

`https://github.com/tallam-git/Big-Data`.

Tools & Technologies

- Apache Spark:
  "Apache Spark. Retrieved from `https://spark.apache.org`."
- Jupyter Notebook:
  "Project Jupyter. Retrieved from `https://jupyter.org`."
- Power BI:
  "Microsoft Power BI. Retrieved from `https://powerbi.microsoft.com`."
- Pandas:
  "Pandas Library. Retrieved from `https://pandas.pydata.org/docs`."
- ChatGPT:
  "ChatGPT. Retrieved from OpenAI. `https://chat.openai.com/`"