

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 24, 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.

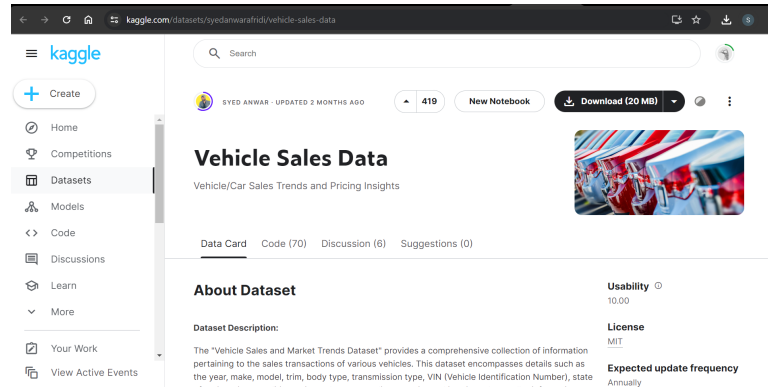


Figure 2: Dataset

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

(a) Before Cleaning

(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

```
1 import pandas as pd
2 import time
3
4 # Record the start time
5 start_time = time.time()
6
7 # Read the data into a DataFrame
8 # Replace 'output.csv' with the actual path to your data file
9 df = pd.read_csv('output.csv')
10
11 # Group the data by year, make, and model, then count the total
    sales for each group
12 sales_counts = df.groupby(['year', 'make', 'model']).size().
    reset_index(name='TotalSales')
13
14 # Record the end time
15 end_time = time.time()
16
17 # Calculate processingtime
18 processingtime = end_time - start_time
19 print("Processing Time: {:.4f} seconds".format(processingtime))
20
21 # Display the top-selling vehicles for each manufacturer year-wise
    in the desired format
22 print(sales_counts)
23
24 # Save the result to a CSV file
25 sales_counts.to_csv('Q1.csv', index=False)
```

Output

```
1 Top-selling vehicles for each manufacturer year-wise:
2   year  make  model  TotalSales
3 0   1982   Ford  Altima         2
4 1   1983   Ford  Altima         1
5 2   1984   Ford  Altima         4
6 3   1984  chevrolet  corvette         1
7 4   1985   Ford  Altima         8
8 ...   ...   ...   ...         ...
9 5608  2015   Volvo    V60         83
10 5609  2015   Volvo   XC60         78
11 5610  2015   Volvo   XC70         25
12 5611  2015  chevrolet   capt         3
13 5612  2015   smart  fortwo         3
14
15 [5613 rows x 4 columns]
16
17 Execution started at: Wed Apr 24 14:29:31 2024
18 Execution ended at: Wed Apr 24 14:29:36 2024
19 Execution time: 4.409718751907349 seconds
```

Visualization code

```
1 # Graph Code for 1st Question
2
```

```

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:
13     x.loc[x['sellingprice'].idxmax()]).reset_index(drop=True)
14
15 # Count the number of top-selling vehicles for each manufacturer
16     year-wise
17 top-selling-counts = top-selling-vehicles.groupby(['year', 'make'])
18     .size().unstack(fill_value=0)
19
20 # Plotting
21 top-selling-counts.plot(kind='bar', stacked=True, figsize=(12, 6))
22 plt.title('Top Selling Vehicles by Manufacturer (Year-wise)')
23 plt.xlabel('Year')
24 plt.ylabel('Number of Top Selling Vehicles')
25 plt.legend(title='Manufacturer', bbox_to_anchor=(1.05, 1), loc='
    upper left')
26 plt.tight_layout()
27 plt.show()

```

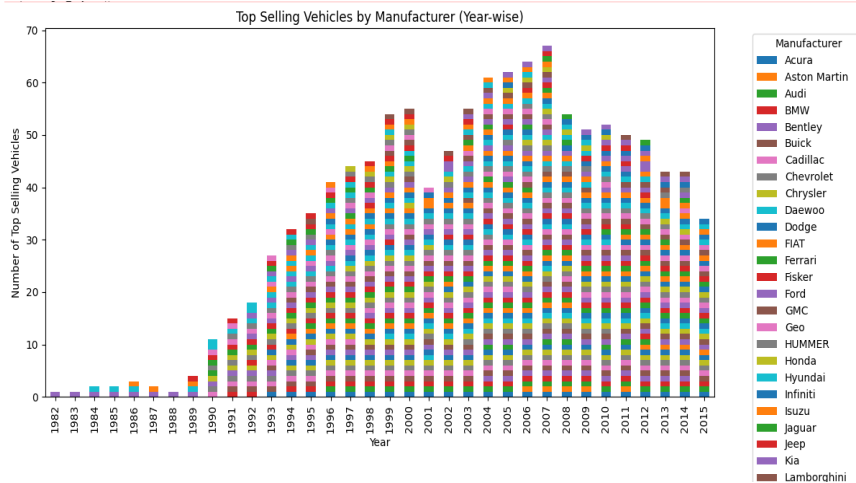


Figure 5: Visualization Image using Mat-Plot Library

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

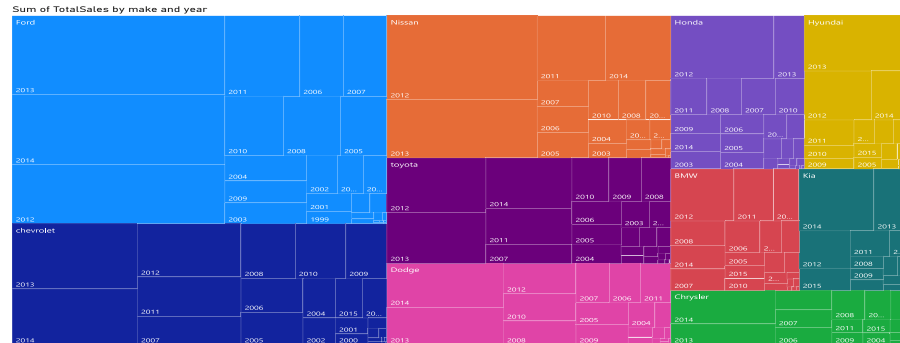


Figure 6: Visualization Image using POWER BI

Question-2

Query

```

1 # Question-2
2
3 import pandas as pd
4 import time
5
6 # Record start time
7 start_time = time.time()
8
9 # Read the data into a DataFrame
10 # Replace 'output.csv' with the actual path to your data file
11 df = pd.read_csv('output.csv')
12
13 # Group the data by car type (model) and calculate the average
    selling price
14 average_price_by_model = df.groupby('model')['sellingprice'].mean()
15
16 # Group the data by manufacturer (make) and calculate the average
    selling price
17 average_price_by_manufacturer = df.groupby('make')['sellingprice'].
    mean()
18
19 # Reset index and rename columns for average selling price by
    manufacturer
20 average_price_by_manufacturer = average_price_by_manufacturer.
    reset_index()
21 average_price_by_manufacturer.columns = ['Make', 'AvgSellingPrice']
22
23 # Record end time
24 end_time = time.time()
25
26 # Calculate processing time
27 processing_time = end_time - start_time
28

```



```

29 print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
30 print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
31 print("Processing Time:", processing_time, "seconds")
32
33 print(average_price_by_manufacturer)
34
35 average_price_by_manufacturer.to_csv('Q2.csv', index=False)

```

Output

```

1 Start Time: 2024-04-24 14:41:27
2 End Time: 2024-04-24 14:41:31
3 Processing Time: 3.8979320526123047 seconds
4      Make  AvgSellingPrice
5 0      Acura    14017.268260
6 1  Aston Martin    54812.000000
7 2      Audi    19915.432782
8 3      BMW    21441.895748
9 4    Bentley    74367.672414
10 ..      ...      ...
11 91    subaru    3710.416667
12 92    suzuki    4810.000000
13 93    toyota    7339.105263
14 94  volkswagen    6145.833333
15 95         vw    13672.916667
16
17 [96 rows x 2 columns]

```

Visualization code

```

1 # Graph Code for 2nd Question
2
3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Read the data into a DataFrame
7 # Replace 'output.csv' with the actual path to your data file
8 df = pd.read_csv('output.csv')
9
10 # Group the data by car type (model) and calculate the average
    selling price
11 average_price_by_model = df.groupby('model')['sellingprice'].mean()
12
13 # Group the data by manufacturer (make) and calculate the average
    selling price
14 average_price_by_manufacturer = df.groupby('make')['sellingprice'].
    mean()
15
16 # Plotting
17 plt.figure(figsize=(10, 25))
18
19 # Plotting average selling price by manufacturer (make)
20 plt.subplot(2, 1, 1)
21 average_price_by_manufacturer.plot(kind='barh', color='lightgreen')
    # Use 'barh' for horizontal bar plot
22 plt.title('Average Selling Price by Manufacturer (Make)')
23 plt.xlabel('Average Selling Price')

```

```

24 plt.ylabel('Manufacturer (Make)')
25 plt.tight_layout()
26
27 plt.show()

```

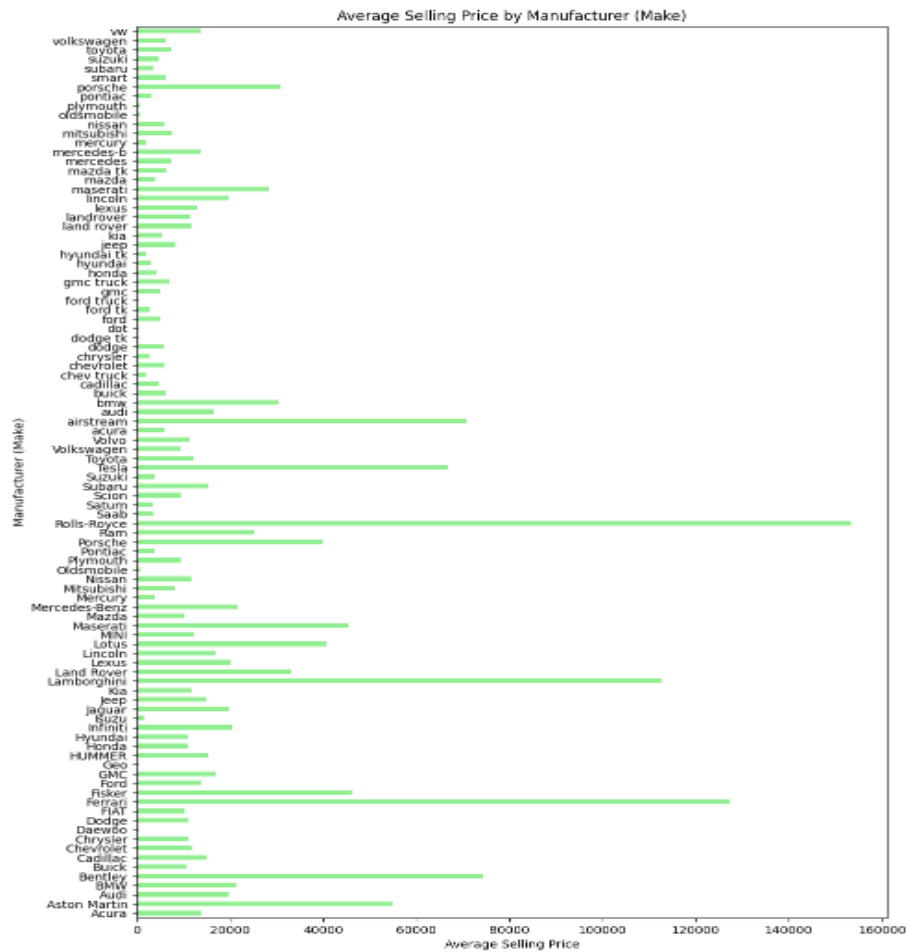


Figure 7: Visualization Image using Mat-plot Library

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

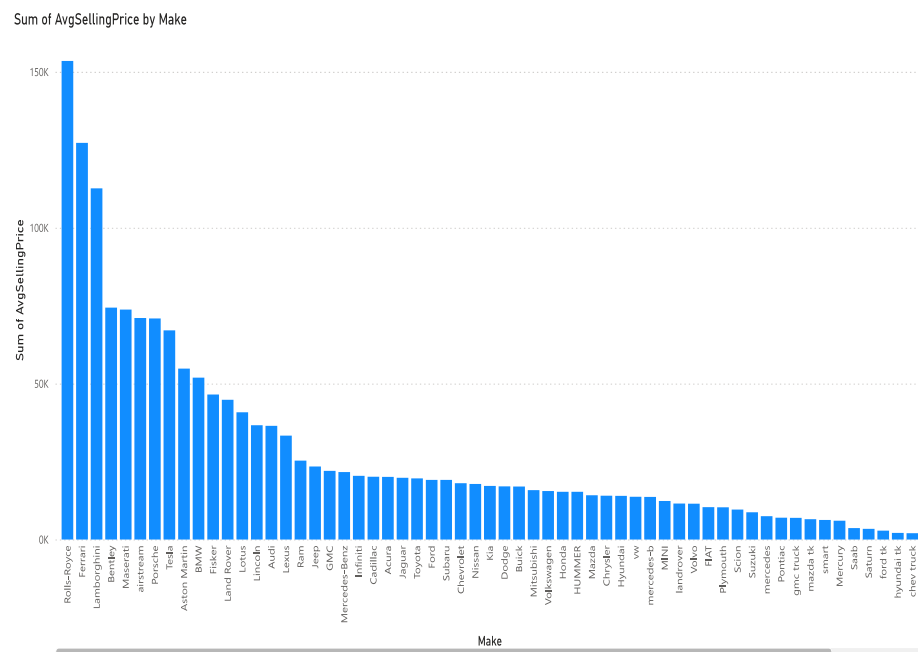


Figure 8: Visualization Image using POWER BI

Question-3

Query

```
1 # Question-3
2
3 import pandas as pd
4 import time
5
6 # Record start time
7 start_time = time.time()
8
9 # Read the data into a DataFrame
10 # Replace 'output.csv' with the actual path to your data file
11 df = pd.read_csv('output.csv')
12
13 # Determine the total revenue generated for each make over the
    years
14 total_revenue_by_make_year = df.groupby(['make'])['sellingprice'].
    sum()
15
16 # Convert the Series to a DataFrame, reset the index, and rename
    the columns
17 total_revenue_by_make = total_revenue_by_make_year.reset_index()
18 total_revenue_by_make.columns = ['Make', 'TotalRevenue']
19
20 # Print the formatted DataFrame
21 print(total_revenue_by_make)
22
23 # Record end time
24 end_time = time.time()
25
26 # Calculate processing time
27 processing_time = end_time - start_time
28
29 print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
30 print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
31 print("Processing Time:", processing_time, "seconds")
32
33 total_revenue_by_make.to_csv('Q3.csv', index=False)
```

Output

```
1      Make  TotalRevenue
2 0      Acura    82715900.0
3 1  Aston Martin    1370300.0
4 2      Audi    116883675.0
5 3      BMW    444254638.0
6 4    Bentley    8626650.0
7 ..      ...
8 91    subaru    222625.0
9 92    suzuki     24050.0
10 93    toyota    697215.0
11 94  volkswagen    147500.0
12 95      vw    328150.0
13
```

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

Visualization code

```
1 # Graph Code for 3rd Question
2
3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Read the data into a DataFrame
7 # Replace 'output.csv' with the actual path to your data file
8 df = pd.read_csv('output.csv')
9
10 # Determine the total revenue generated for each make over the
    years
11 total_revenue_by_make_year = df.groupby(['make'])['sellingprice'].
    sum()
12
13 # Convert the Series to a DataFrame, reset the index, and rename
    the columns
14 total_revenue_by_make = total_revenue_by_make_year.reset_index()
15 total_revenue_by_make.columns = ['Make', 'TotalRevenue']
16
17 # Plotting
18 plt.figure(figsize=(12, 6))
19 plt.bar(total_revenue_by_make['Make'], total_revenue_by_make['
    TotalRevenue'], color='skyblue')
20 plt.xlabel('Make')
21 plt.ylabel('Total Revenue')
22 plt.title('Total Revenue Generated for Each Make')
23 plt.xticks(rotation=90)
24 plt.tight_layout()
25 plt.show()
```

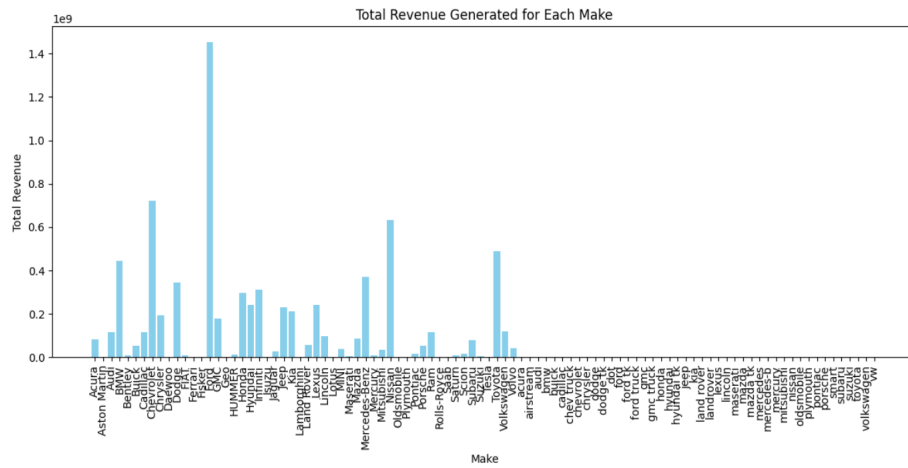


Figure 9: Visualization Image using Mat-plot Library

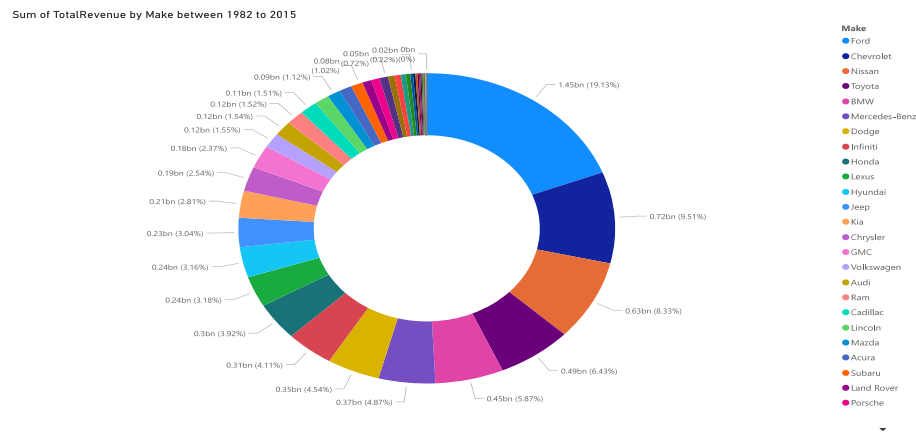


Figure 10: Visualization Image using POWER BI

Question-4

Query

```
1 import pandas as pd
2 import time
3
4 # Record start time
5 start_time = time.time()
6
7 # Read the data into a DataFrame
8 # Replace 'output.csv' with the actual path to your data file
9 df = pd.read_csv('output.csv')
10
11 # Group the data by body type and count the number of vehicles sold
    for each body type
12 vehicles_sold_by_body_type = df.groupby('body')['model'].count().
    reset_index(name='TotalVehiclesSold')
13
14 print(vehicles_sold_by_body_type)
15
16 # Record end time
17 end_time = time.time()
18
19 # Calculate processing time
20 processing_time = end_time - start_time
21
22 print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
23 print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
24 print("Processing Time:", processing_time, "seconds")
25
26 vehicles_sold_by_body_type.to_csv('Q4.csv', index=False)
```

Output

```
1          body  TotalVehiclesSold
2 0      Access Cab             232
3 1  Beetle Convertible             52
4 2          CTS Coupe            129
5 3          CTS Wagon             13
6 4      CTS-V Coupe             28
7 ..          ...             ...
8 82      transit van              7
9 83  tsx sport wagon              8
10 84          van            570
11 85          wagon          2499
12 86      xtracab              4
13
14 [87 rows x 2 columns]
15 Start Time: 2024-04-24 14:45:37
16 End Time: 2024-04-24 14:45:41
17 Processing Time: 3.9222331047058105 seconds
```

Visualization code

```
1 # Graph code for Question-4
2
```

```

3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Read the data into a DataFrame
7 # Replace 'output.csv' with the actual path to your data file
8 df = pd.read_csv('output.csv')
9
10 # Group the data by body type and count the number of vehicles sold
    for each body type
11 vehicles_sold_by_body_type = df.groupby('body')['model'].count().
    reset_index(name='TotalVehiclesSold')
12
13 # Plotting
14 plt.figure(figsize=(10, 20))
15 plt.scatter(vehicles_sold_by_body_type['TotalVehiclesSold'],
    vehicles_sold_by_body_type['body'], color='skyblue') # Reverse
    x and y
16 plt.title('Number of Vehicles Sold by Body Type')
17 plt.xlabel('Number of Vehicles Sold')
18 plt.ylabel('Body Type')
19 plt.yticks(rotation=45, ha='right') # Rotate y-axis labels
20 plt.grid(True)
21 plt.tight_layout()
22 plt.show()

```


Question-5

Query

```
1 import pandas as pd
2 import time
3
4 # Record start time
5 start_time = time.time()
6
7 # Read the data into a DataFrame
8 # Replace 'output.csv' with the actual path to your data file
9 df = pd.read_csv('output.csv')
10
11 # Group the data by vehicle make and calculate the average odometer
    reading
12 average_odometer_by_make = df.groupby('make')['odometer'].mean()
13
14 # Convert the Series to a DataFrame, reset the index, and rename
    the columns
15 average_odometer_by_make_df = average_odometer_by_make.reset_index
    ()
16 average_odometer_by_make_df.columns = ['Make', 'AverageOdometer']
17
18 # Print the formatted DataFrame
19 print("Average Odometer Reading by Vehicle Make:")
20 print(average_odometer_by_make_df)
21
22 # Record end time
23 end_time = time.time()
24
25 # Calculate processing time
26 processing_time = end_time - start_time
27
28 print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
29 print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
30 print("Processing Time:", processing_time, "seconds")
31
32 # Write the output to a CSV file
33 average_odometer_by_make_df.to_csv('Q5.csv', index=False)
```

Output

```
1 Average Odometer Reading by Vehicle Make:
2      Make  AverageOdometer
3 0      Acura      85829.219285
4 1  Aston Martin      26603.640000
5 2      Audi      66040.140913
6 3      BMW      64298.103096
7 4      Bentley      39239.698276
8 ..      ...      ...
9 91     subaru      134812.600000
10 92     suzuki      80901.400000
11 93     toyota      145414.336842
12 94  volkswagen      103235.083333
13 95      vw      67813.583333
```

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

Visualization code

```
1 # Graph code for Question-5
2
3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Read the data into a DataFrame
7 # Replace 'output.csv' with the actual path to your data file
8 df = pd.read_csv('output.csv')
9
10 # Group the data by vehicle make and year, then calculate the
    average odometer reading
11 average_odometer_by_make_year = df.groupby(['make', 'year'])['
    odometer'].mean()
12
13 # Convert the Series to a DataFrame, reset the index
14 average_odometer_by_make_year_df = average_odometer_by_make_year.
    reset_index()
15
16 # Aggregate the average odometer readings by vehicle make
17 average_odometer_by_make = average_odometer_by_make_year_df.groupby
    ('make')['odometer'].mean()
18
19 # Plotting
20 plt.figure(figsize=(12, 30))
21 average_odometer_by_make.plot(kind='barh', color='skyblue') #
    Change kind to 'barh' for horizontal bar plot
22 plt.title('Average Odometer Reading by Vehicle Make')
23 plt.xlabel('Average Odometer Reading')
24 plt.ylabel('Vehicle Make')
25 plt.grid(axis='x')
26 plt.tight_layout()
27 plt.show()
```

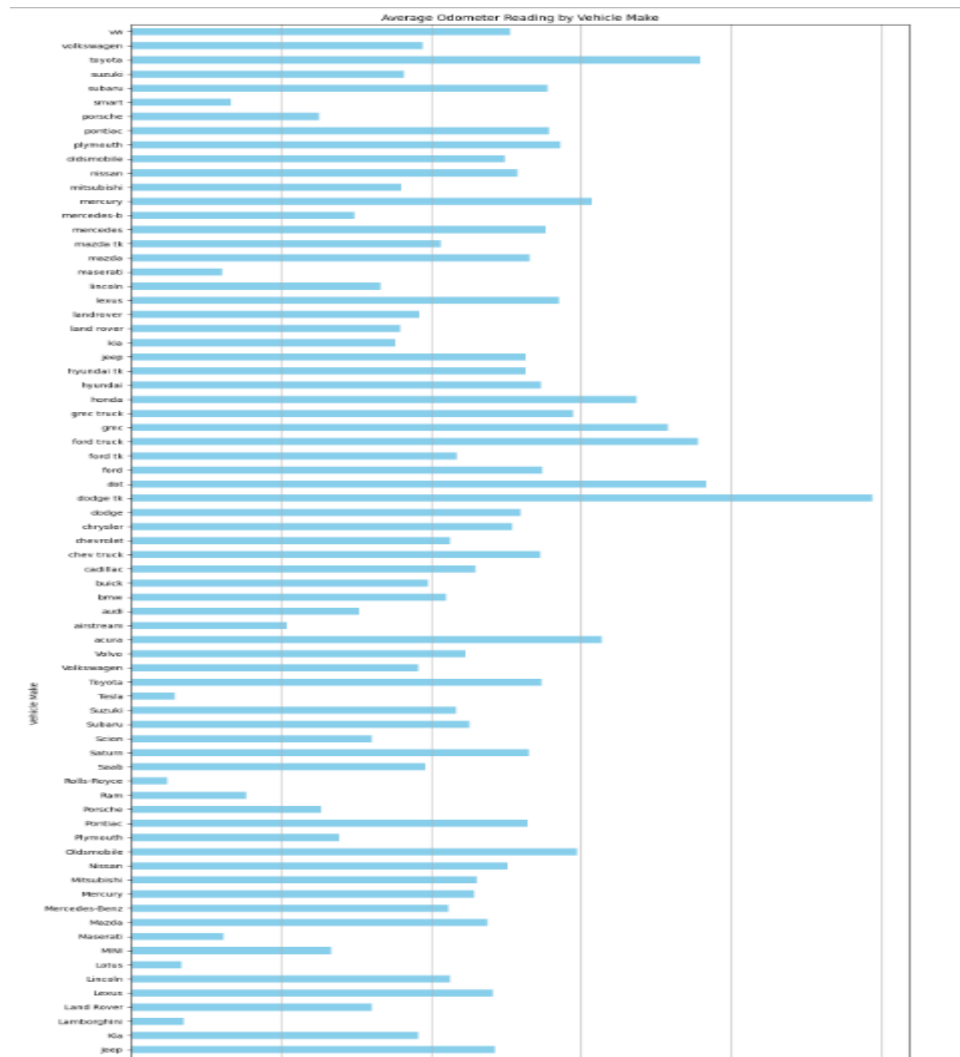


Figure 12: Visualization Image

Question-6

Query

```
1 # Question-6
2
3 import pandas as pd
4 import time
5
6 # Record start time
7 start_time = time.time()
8
9 # Read the data into a DataFrame
10 # Replace 'output.csv' with the actual path to your data file
11 df = pd.read_csv('output.csv')
12
13 # Group the data by make and model, then count the total sales for
    each group
14 top_selling_models = df.groupby(['make', 'model']).size().
    reset_index(name='TotalSales')
15
16 # Sort the data by make and total sales in descending order
17 top_selling_models = top_selling_models.sort_values(by=['make', '
    TotalSales'], ascending=[True, False])
18
19 # Print the formatted DataFrame
20 print('+-----+-----+')
21 print('|           Make|           Model| TotalSales|')
22 print('+-----+-----+')
23
24 for index, row in top_selling_models.iterrows():
25     print('{:12}|{:15}|{:10}|'.format(row['make'], row['model'],
    row['TotalSales']))
26
27 print('+-----+-----+')
28
29 # Record end time
30 end_time = time.time()
31
32 # Calculate processing time
33 processing_time = end_time - start_time
34
35 print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
36 print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
37 print("Processing Time:", processing_time, "seconds")
```

Output

```

1 |-----|
2 |      Make      |      Model      | TotalSales |
3 |-----|
4 | Acura          | TL              | 2135      |
5 | Acura          | MDX             | 1581      |
6 | Acura          | TSX             | 1119      |
7 | Acura          | RDX             | 420       |
8 | Acura          | RSX             | 151       |
9 | Acura          | ILX             | 140       |
10 | Acura          | RL              | 108       |
11 | Acura          | CL              | 97        |
12 | Acura          | Integra         | 44        |
13 | Acura          | ZDX             | 39        |
14 | Acura          | TSX Sport Wagon | 36        |
15 | Acura          | RLX             | 17        |
16 | Acura          | Legend          | 9         |
17 | Acura          | mdx             | 4         |
18 | Acura          | TLX             | 1         |
19 | Aston Martin   | V8 Vantage      | 17        |
20 | Aston Martin   | DB9             | 6         |
21 | Aston Martin   | Rapide          | 2         |
22 |**** Placed only small piece of output only ****|
23 |-----|
24 |-----|
25 |Start Time: 2024-04-24 14:50:29|
26 |End Time: 2024-04-24 14:50:33|
27 |Processing Time: 4.164934873580933 seconds|

```

Visualization code

```

1 # Graph code for question-6
2
3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Read the data into a DataFrame
7 # Replace 'output.csv' with the actual path to your data file
8 df = pd.read_csv('output.csv')
9
10 # Group the data by make and model, then count the total sales for
    each group
11 top_selling_models = df.groupby(['make', 'model']).size().
    reset_index(name='TotalSales')
12
13 # Sort the data by make and total sales in descending order
14 top_selling_models = top_selling_models.sort_values(by=['make', '
    TotalSales'], ascending=[True, False])
15
16 # Get unique makes
17 unique_makes = top_selling_models['make'].unique()
18
19 # Plotting
20 for make in unique_makes:
21     make_data = top_selling_models[top_selling_models['make'] ==
        make]
22     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()
```

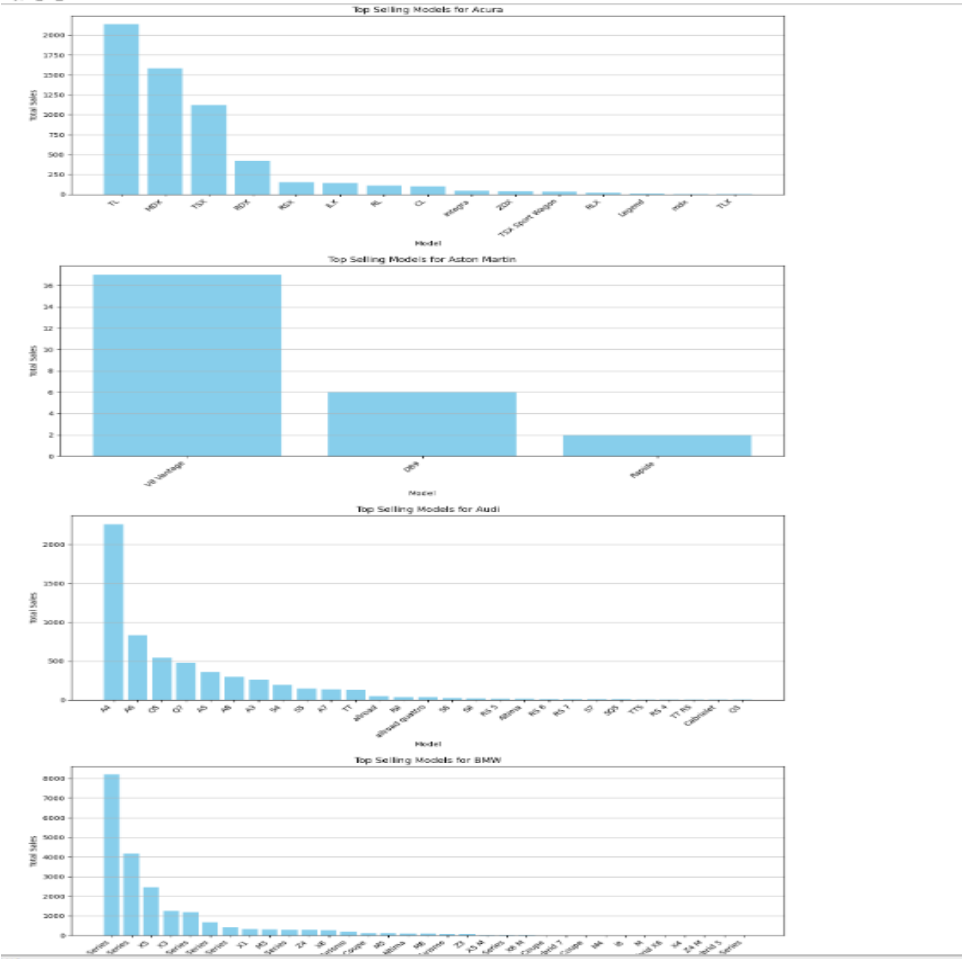


Figure 13: Visualization Image

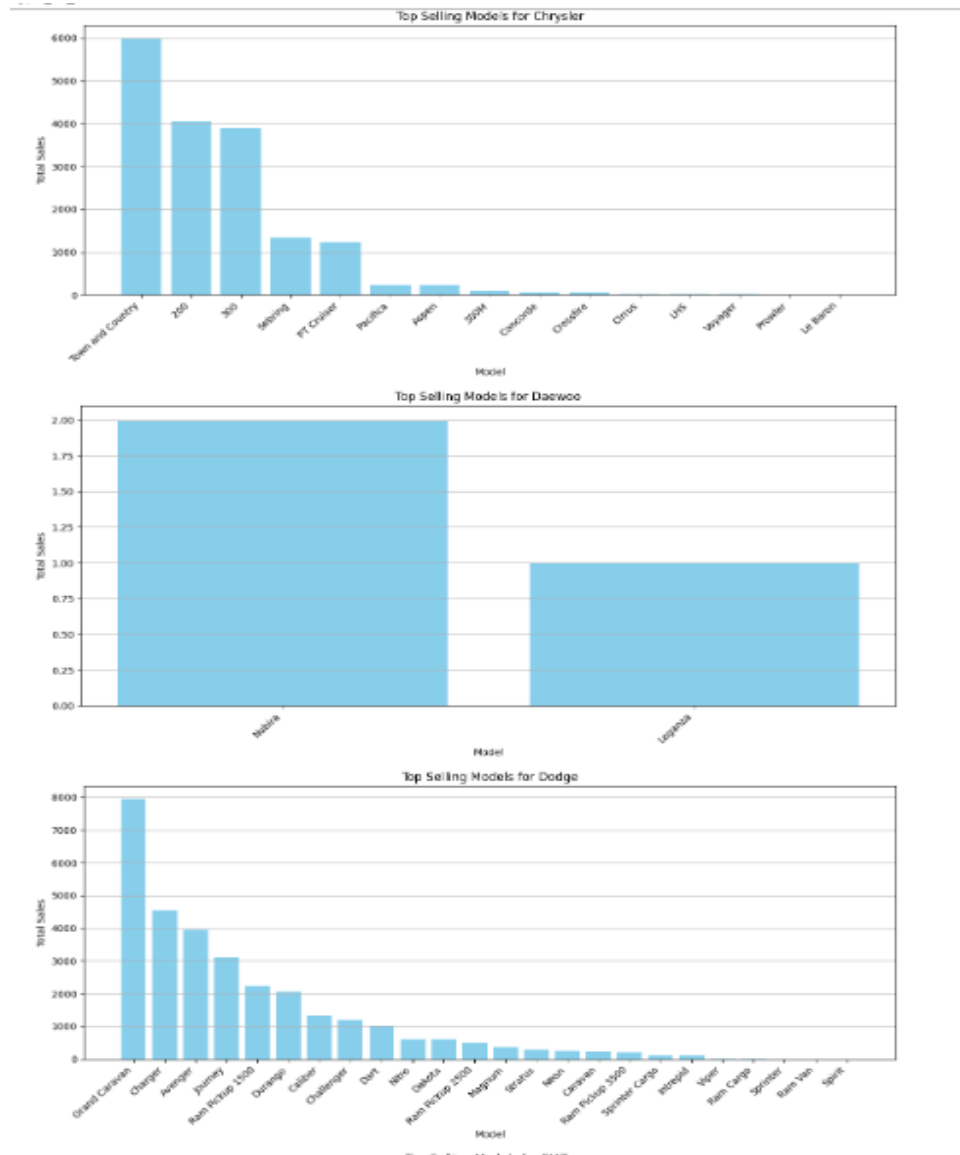


Figure 15: Visualization Image

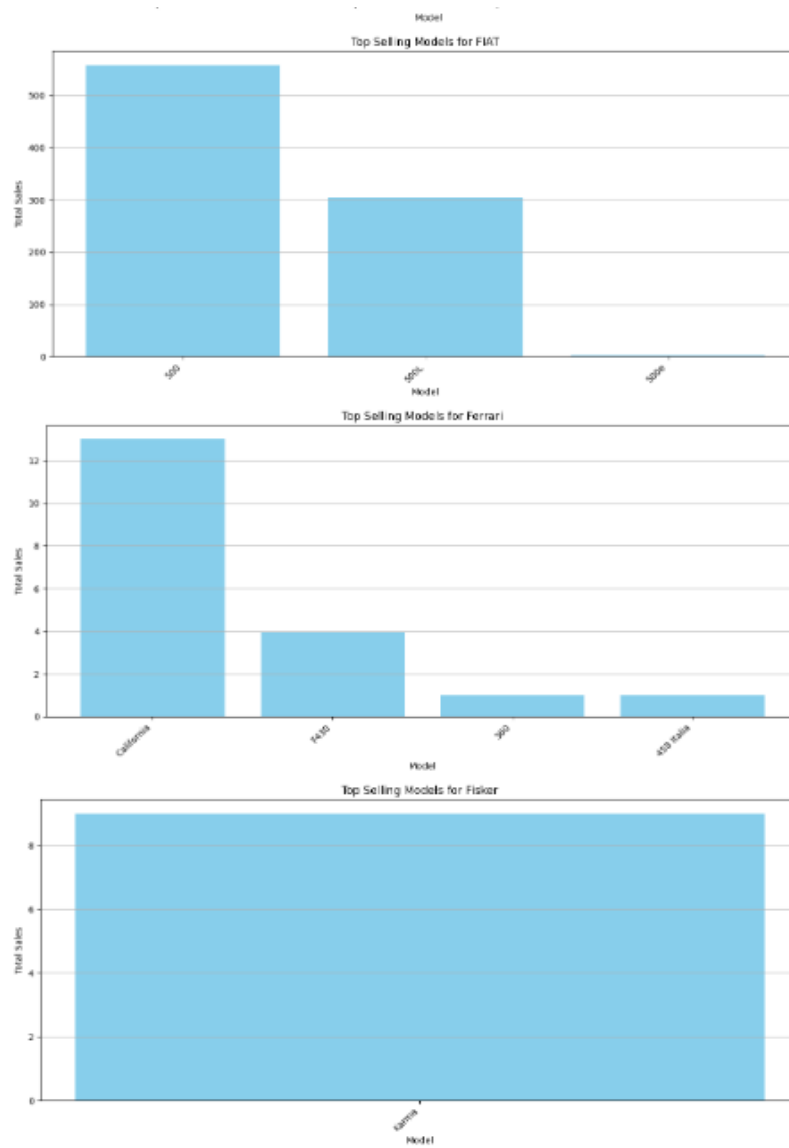


Figure 16: Visualization Image

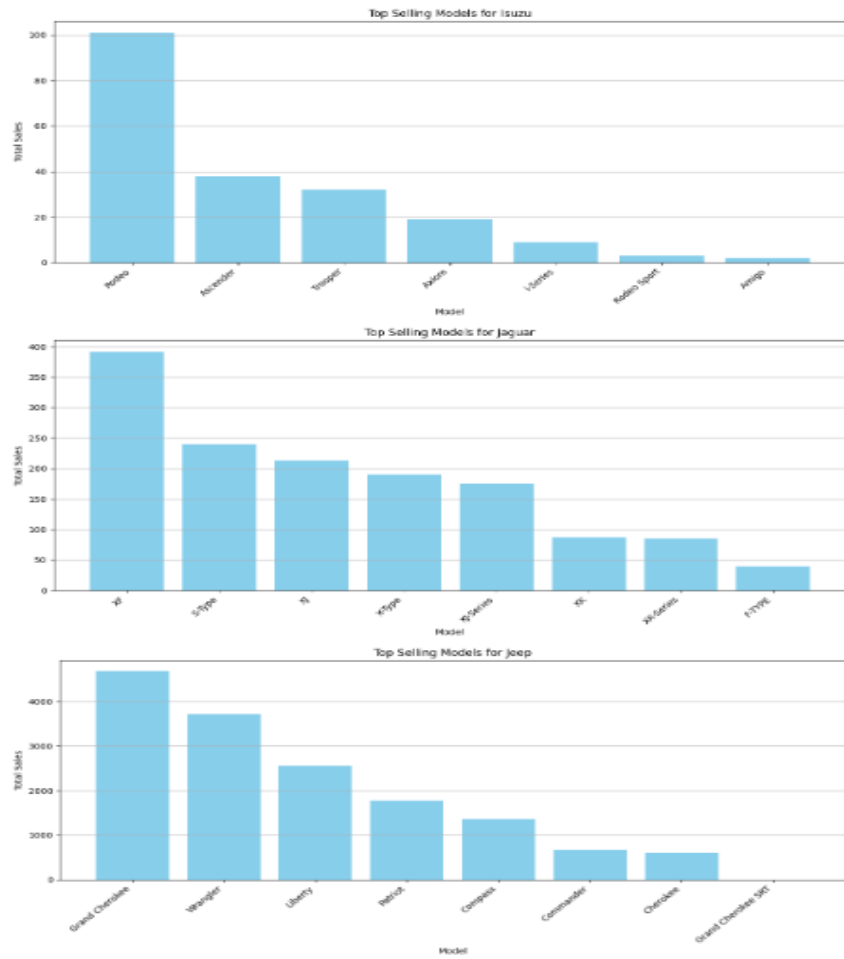


Figure 19: Visualization Image

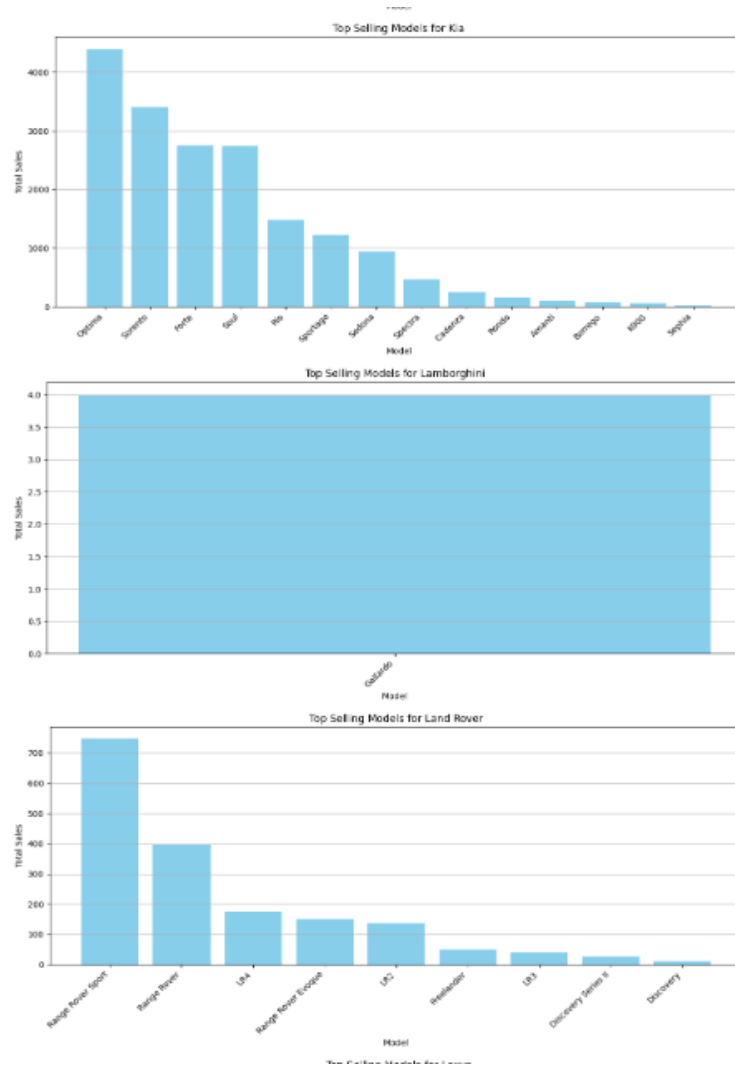


Figure 20: Visualization Image

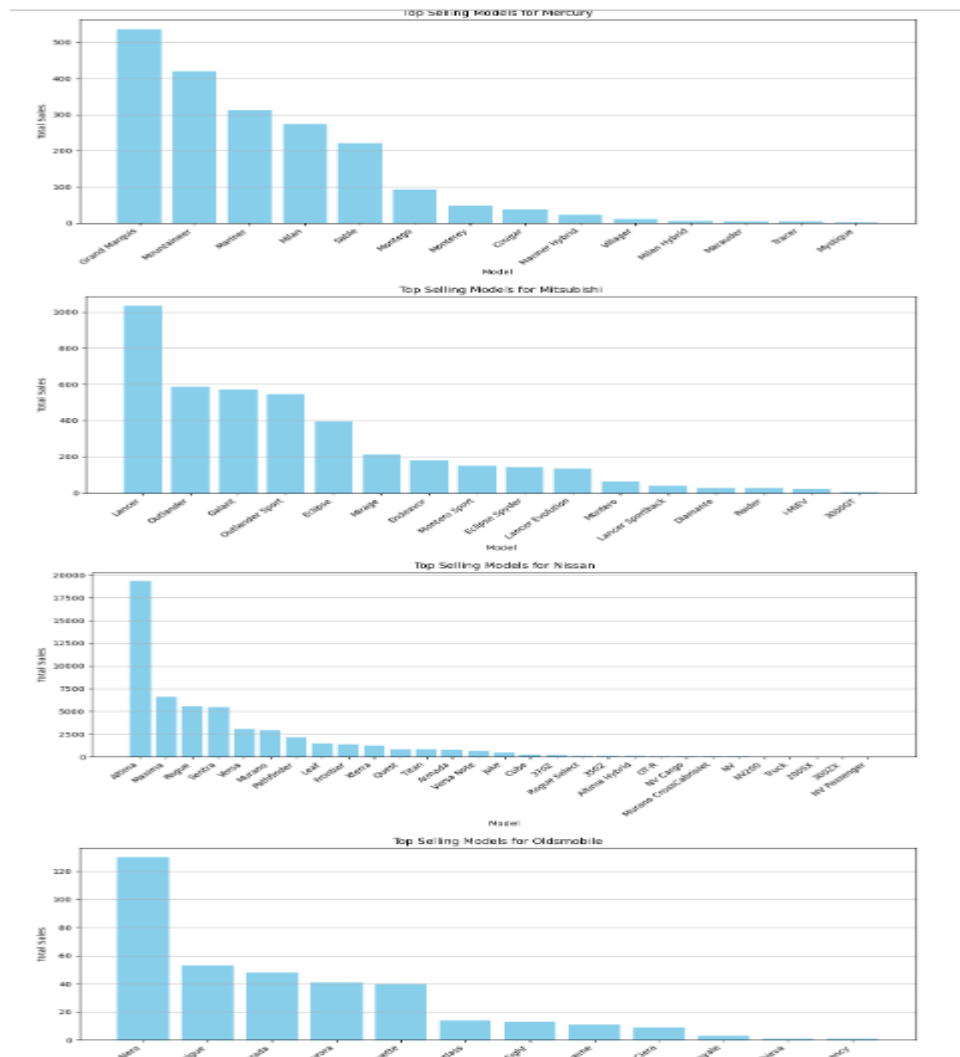


Figure 23: Visualization Image

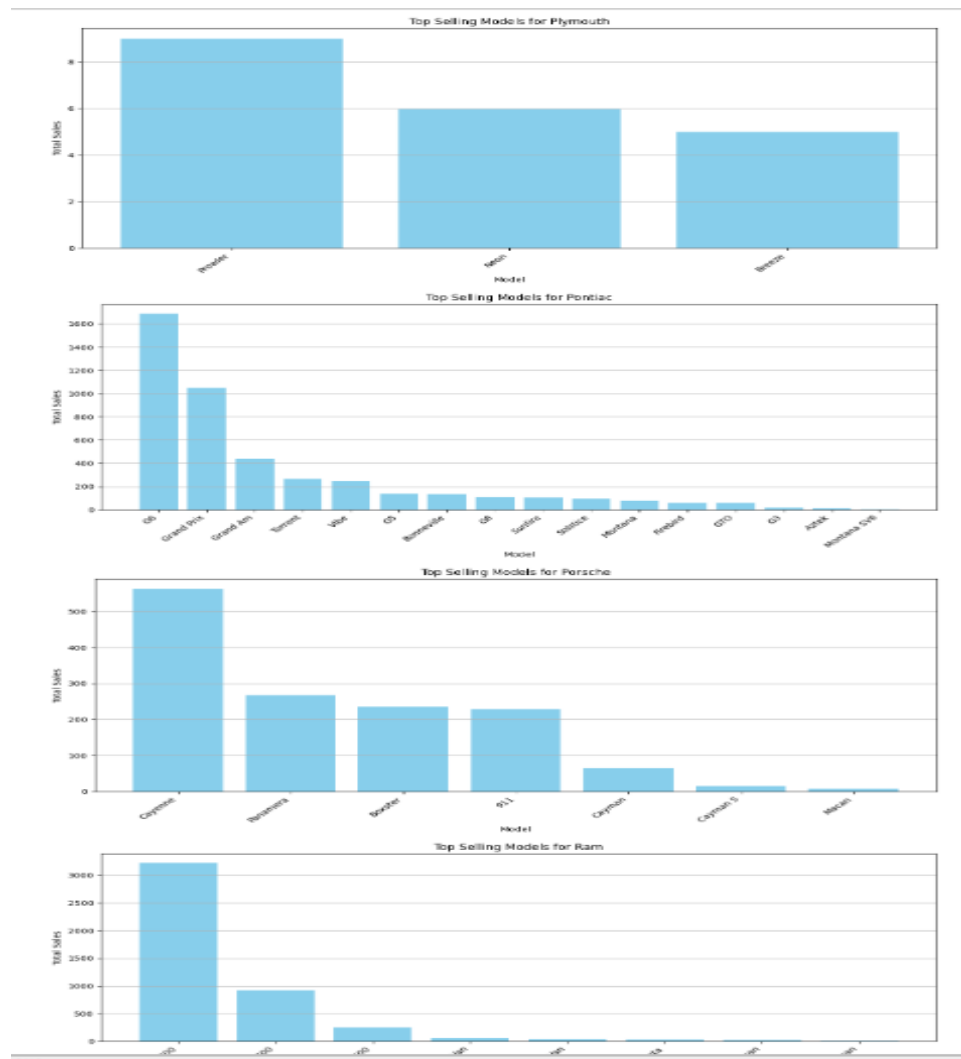


Figure 24: Visualization Image

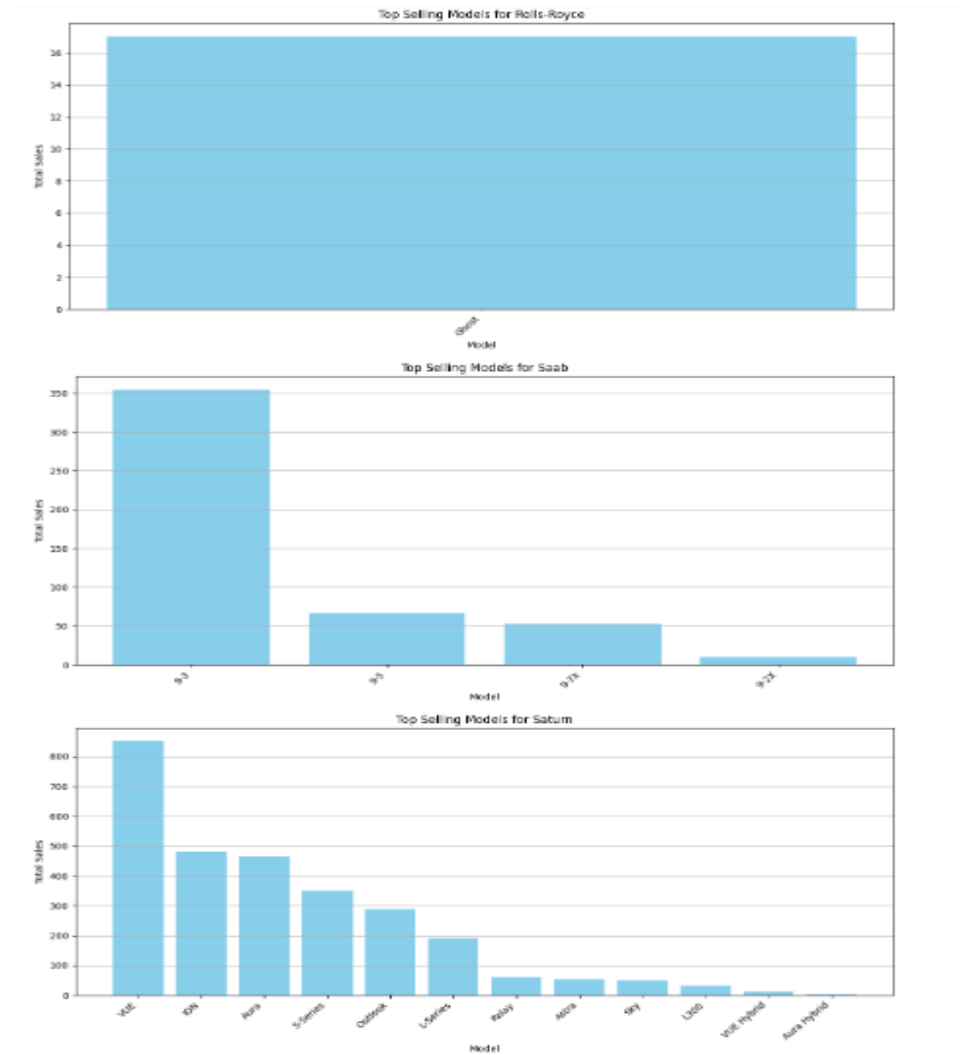


Figure 25: Visualization Image

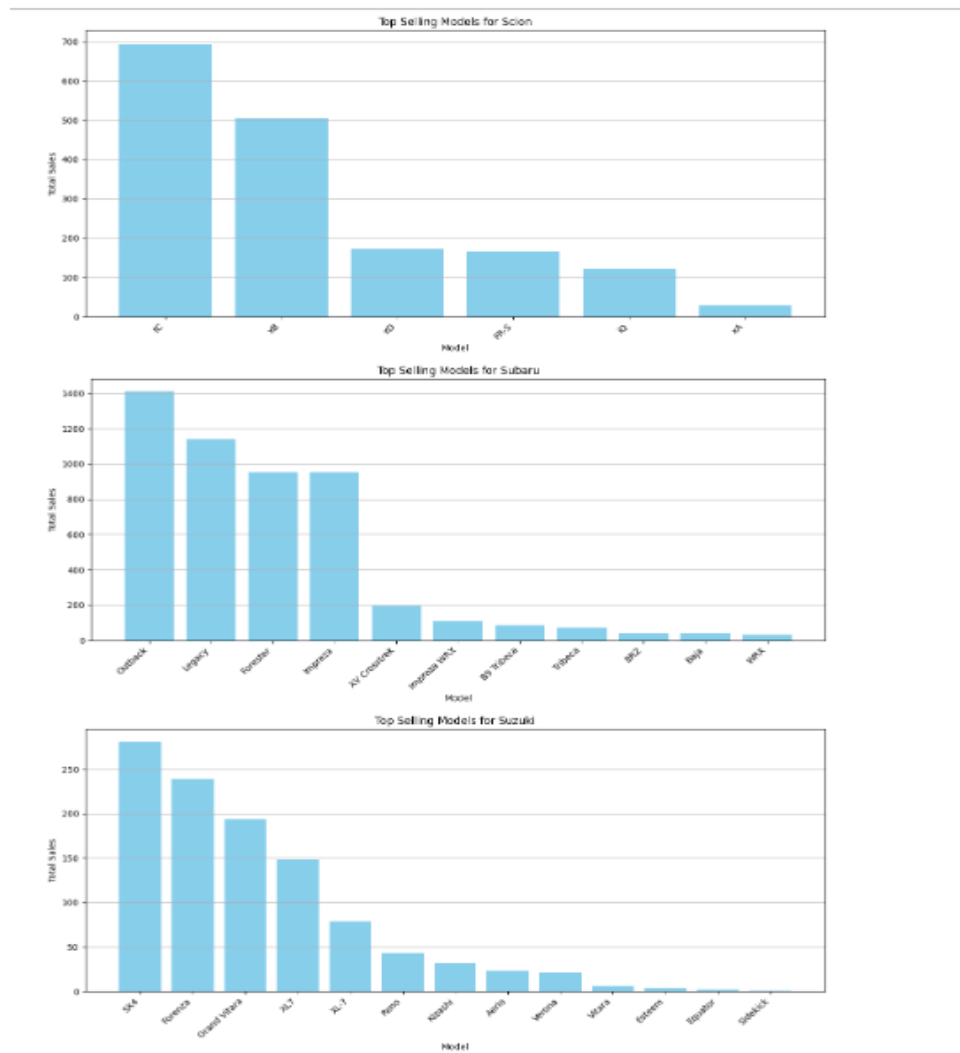


Figure 26: Visualization Image

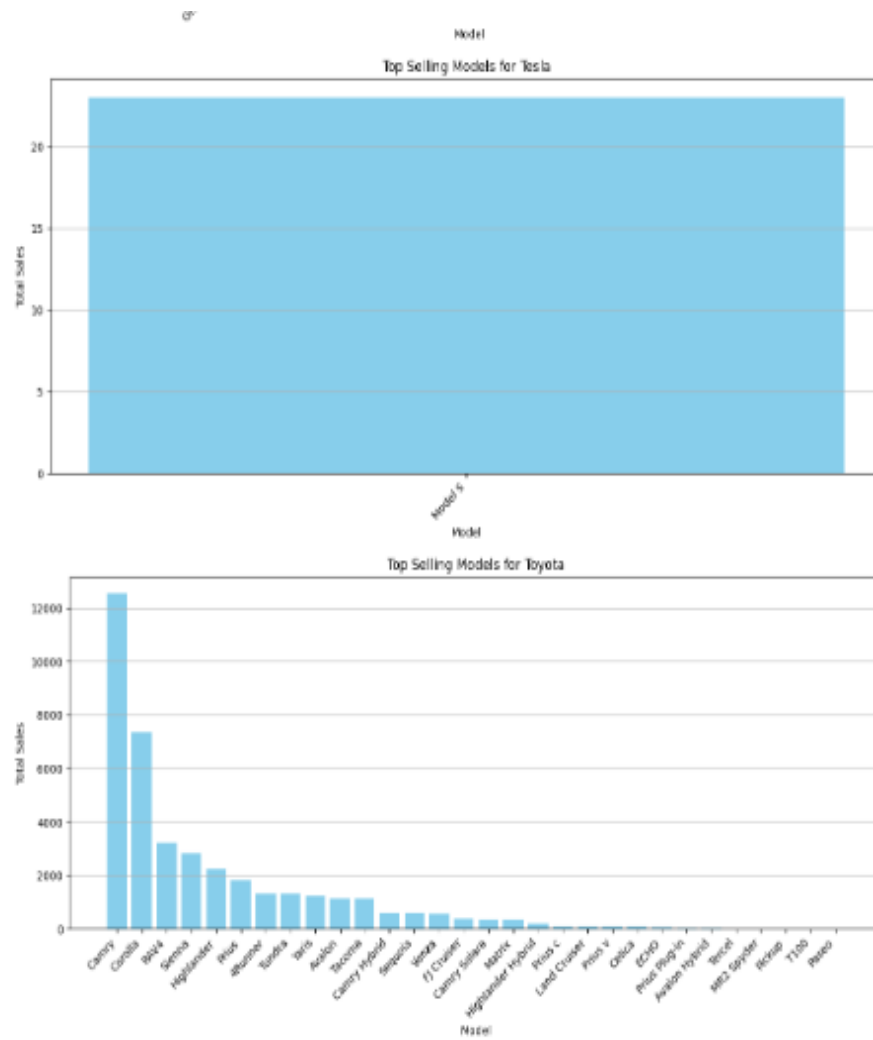


Figure 27: Visualization Image

Question-7

Query

```
1 import pandas as pd
2 import time
3
4 state_mapping = {
5     'al': 'Alabama', 'ak': 'Alaska', 'az': 'Arizona', 'ar': '
6     Arkansas', 'ca': 'California',
7     'co': 'Colorado', 'ct': 'Connecticut', 'de': 'Delaware', 'fl':
8     'Florida', 'ga': 'Georgia',
9     'hi': 'Hawaii', 'id': 'Idaho', 'il': 'Illinois', 'in': 'Indiana',
10    'ia': 'Iowa', 'ks': 'Kansas',
11    'ky': 'Kentucky', 'la': 'Louisiana', 'me': 'Maine', 'md': '
12    Maryland', 'ma': 'Massachusetts',
13    'mi': 'Michigan', 'mn': 'Minnesota', 'ms': 'Mississippi', 'mo':
14    'Missouri', 'mt': 'Montana',
15    'ne': 'Nebraska', 'nv': 'Nevada', 'nh': 'New Hampshire', 'nj':
16    'New Jersey', 'nm': 'New Mexico',
17    'ny': 'New York', 'nc': 'North Carolina', 'nd': 'North Dakota',
18    'oh': 'Ohio', 'ok': 'Oklahoma',
19    'or': 'Oregon', 'pa': 'Pennsylvania', 'ri': 'Rhode Island', 'sc
20    ': 'South Carolina',
21    'sd': 'South Dakota', 'tn': 'Tennessee', 'tx': 'Texas', 'ut': '
22    Utah', 'vt': 'Vermont',
23    'va': 'Virginia', 'wa': 'Washington', 'wv': 'West Virginia', '
24    wi': 'Wisconsin', 'wy': 'Wyoming'
25 }
26
27 # Record start time
28 start_time = time.time()
29
30 # Read the data into a DataFrame
31 df = pd.read_csv('output.csv')
32
33 # Group the data by state and count the total vehicle sales for
34 # each state
35 vehicle_sales_by_state = df.groupby('state').size().reset_index(
36     name='TotalSales')
37
38 # Convert state column to lowercase for case insensitivity
39 vehicle_sales_by_state['state'] = vehicle_sales_by_state['state'].
40     str.lower()
41
42 # Sort the data by total sales in descending order
43 vehicle_sales_by_state = vehicle_sales_by_state.sort_values(by='
44     TotalSales', ascending=False)
45
46 # Replace state abbreviations with full names
47 vehicle_sales_by_state['state'] = vehicle_sales_by_state['state'].
48     map(state_mapping)
49
50 # Get the top 5 states
51 top_5_states = vehicle_sales_by_state.head(5)
52
53 # Print the top 5 states
54 print("Top 5 States with the Highest Number of Vehicle Sales:")
```

```

40 print(top_5_states)
41
42 # Record end time
43 end_time = time.time()
44
45 # Calculate processing time
46 processing_time = end_time - start_time
47
48 print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
49 print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
50 print("Processing Time:", processing_time, "seconds")
51
52 top_5_states.to_csv('Q7.csv', index=False)

```

Output

```

1 Top 5 States with the Highest Number of Vehicle Sales:
2      state  TotalSales
3 31  Florida      82945
4 29  California    73148
5 54  Pennsylvania    53907
6 59   Texas      45913
7 32   Georgia      34750
8 Start Time: 2024-04-24 14:54:49
9 End Time: 2024-04-24 14:54:53
10 Processing Time: 3.996584177017212 seconds

```

Visualization code

```

1 # Graph code for Question-7
2
3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Read the data into a DataFrame
7 # Replace 'output.csv' with the actual path to your data file
8 df = pd.read_csv('output.csv')
9
10 # Group the data by state and count the total vehicle sales for
    each state
11 vehicle_sales_by_state = df.groupby('state').size().reset_index(
    name='TotalSales')
12
13 # Sort the data by total sales in descending order
14 vehicle_sales_by_state = vehicle_sales_by_state.sort_values(by='
    TotalSales', ascending=False)
15
16 # Get the top 5 states
17 top_5_states = vehicle_sales_by_state.head(5)
18
19 # Mapping of state abbreviations to full names
20 state_mapping = {
21     'ca': 'California',
22     'tx': 'Texas',
23     'fl': 'Florida',
24     'ga': 'Georgia',
25     'pa': 'Pennsylvania'

```

```

26 }
27
28 # Replace state abbreviations with full names
29 top_5_states_copy = top_5_states.copy() # Create a copy of the
    DataFrame slice
30 top_5_states_copy['state'] = top_5_states_copy['state'].map(
    state_mapping)
31
32 # Plotting
33 plt.figure(figsize=(8, 8))
34 plt.pie(top_5_states_copy['TotalSales'], labels=top_5_states_copy['
    state'], autopct='%1.1f%%', startangle=140)
35 plt.title('Top 5 States with the Highest Number of Vehicle Sales')
36 plt.axis('equal')
37 plt.tight_layout()
38 plt.show()

```

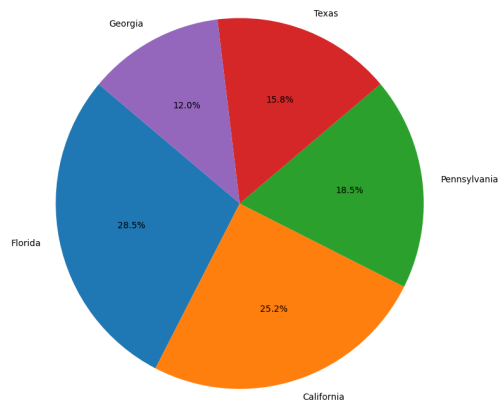


Figure 28: Visualization Image using Mat-plot Library

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

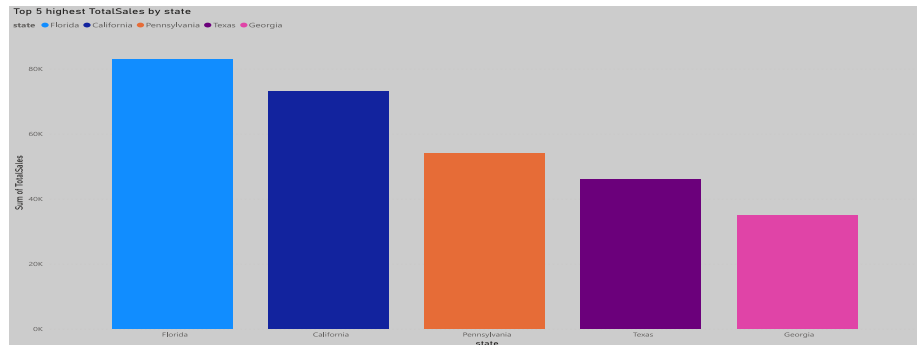


Figure 29: Visualization Image using POWER BI

Question-8

Query

```

1 # Question-8
2
3 import pandas as pd
4 import time
5
6 # Record start time
7 start_time = time.time()
8
9 # Read the data into a DataFrame
10 # Replace 'output.csv' with the actual path to your data file
11 df = pd.read_csv('output.csv')
12
13 # Group the data by make and calculate the average MMR value for
    each make
14 average_mmr_by_make = df.groupby('make')['mmr'].mean()
15
16 # Convert the Series to a DataFrame, reset the index, and rename
    the columns
17 average_mmr_by_make_df = average_mmr_by_make.reset_index()
18 average_mmr_by_make_df.columns = ['Make', 'AvgMMR']
19 print(average_mmr_by_make_df)
20
21 # Record end time
22 end_time = time.time()
23
24 # Calculate processing time
25 processing_time = end_time - start_time
26
27 print("Start Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(start_time)))
28 print("End Time:", time.strftime('%Y-%m-%d %H:%M:%S', time.
    localtime(end_time)))
29 print("Processing Time:", processing_time, "seconds")
30

```

```

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

```

Output

```

1      Make      AvgMMR
2 0      Acura  14076.813252
3 1  Aston Martin  53560.000000
4 2      Audi   20080.213835
5 3      BMW    21575.547806
6 4    Bentley   75928.448276
7 ..      ...      ...
8 91    subaru   3911.666667
9 92    suzuki   4805.000000
10 93    toyota   7367.631579
11 94  volkswagen   5984.375000
12 95      vw     13847.916667
13
14 [96 rows x 2 columns]
15 Start Time: 2024-04-24 14:57:01
16 End Time: 2024-04-24 14:57:05
17 Processing Time: 4.0517213344573975 seconds

```

Visualization code

```

1 # Graph Code for Question-8
2
3 import pandas as pd
4 import matplotlib.pyplot as plt
5
6 # Read the data into a DataFrame
7 # Replace 'output.csv' with the actual path to your data file
8 df = pd.read_csv('output.csv')
9
10 # Group the data by make and calculate the average MMR value for
    each make
11 average_mmr_by_make = df.groupby('make')['mmr'].mean().sort_values(
    ascending=False)
12
13 # Plotting
14 plt.figure(figsize=(12, 20))
15 average_mmr_by_make.plot(kind='barh', color='skyblue') # Change
    kind to 'barh' for horizontal bar plot
16 plt.title('Average MMR Value by Make')
17 plt.xlabel('Average MMR Value')
18 plt.ylabel('Make')
19 plt.grid(axis='x')
20 plt.tight_layout()
21 plt.show()

```


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/syednwarafriidi/vehicle-sales-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/>"