Final Project Report

AIPP Team- A12 12/8/23 2023S-T4 AIP Group A

BUNSEKI

Project Name	Data-driven Analysis of American car market for Kuruma Auto
Project Advisor	Dr. Molham Chikhalsouk
Project Customer	Kuruma Auto
Team Name	A12

Name	Student Id	Task	Total Hours
Dipesh Sharma	500203399	Project Management, researched about USA Car market, Worked on Feature Engineering, managing Trello Board, reviewing slides for project showcase, building status report/graphs, and building a final comprehensive project report and reviewing slides.	420 (30 hours/Week)
Mohammedtareeq Sajjadahmed Shaikh	500204202	Developed a code using python, building project deliverable report, managed github profile and Presentation slides for project showcase	420 (30 hours/Week)
Sohini Roy Chowdhury	500196496	Researched about Success of Japanese Car Market, Developing Power BI Dashboards, worked on website development, built mini presentation and Presentation slides for project showcase	420 (30 hours/Week)

BUNSEKI

1. Introduction

We know that the global car market has very tough competition in terms of pricing, style and changing customers' preferences. Therefore, the prediction of a car price will play a key role for any auto industry who is aiming to establish a strong presence in the highly competitive US automobile market. In this project, we will help Kuruma Auto to gain a successful entry into the US car market by providing useful data-driven insights to purposely position its car models and pricing.

2. Problem Statement

Kuruma Auto- A Japanese automobile company is aiming to enter the US Car market with the goal of selling vehicles to local customers, while competing with already established American and European car manufacturers. The main challenge is to research and identify the hidden factors that determine car pricing in the American market, and later on plan a pricing strategy that aligns with consumer expectations, market conditions, and Kuruma Auto's business goals.

The objective of this report is to present a comprehensive analysis of the Car Market in US to the stakeholders in the industry. The status and some useful insights about already established industries in the US with the predicted average price and trends are presented in the report with the analysis of complex data in user friendly dashboard.

3. Business Value to Our Client

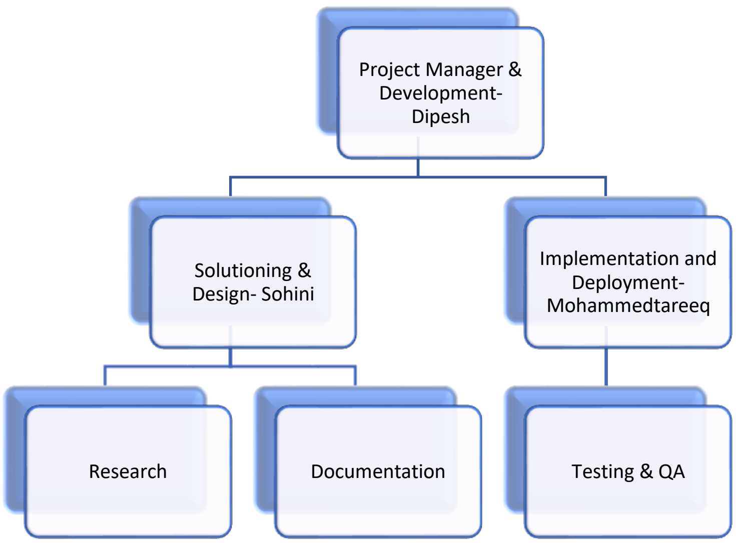
We will help Kuruma Auto to better understand the US market's unique characteristics and enable them to enter and compete effectively in US market.

Providing data insights and reports

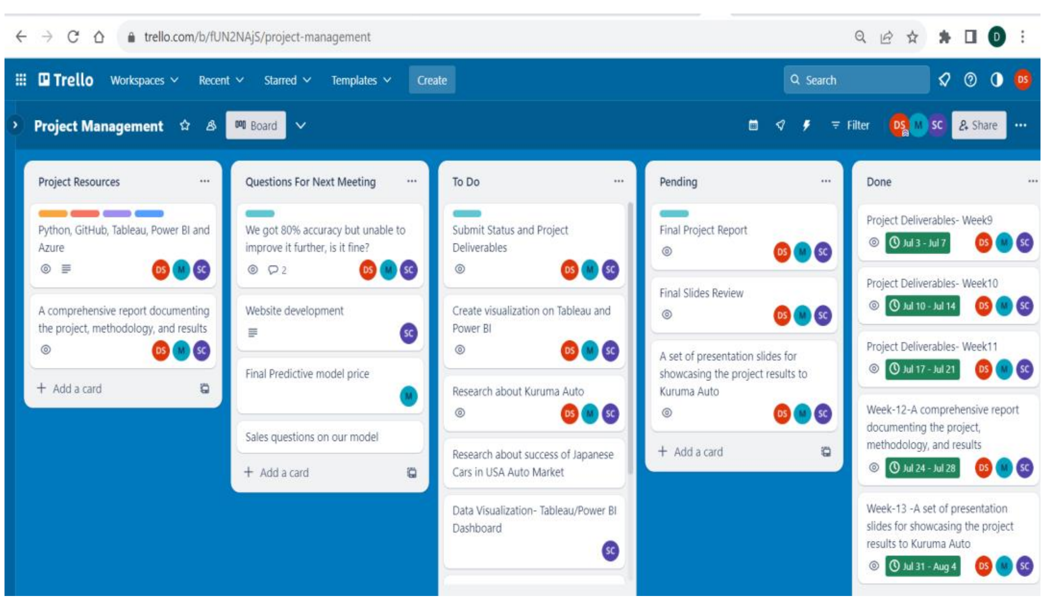
Interactive Dashboard Building

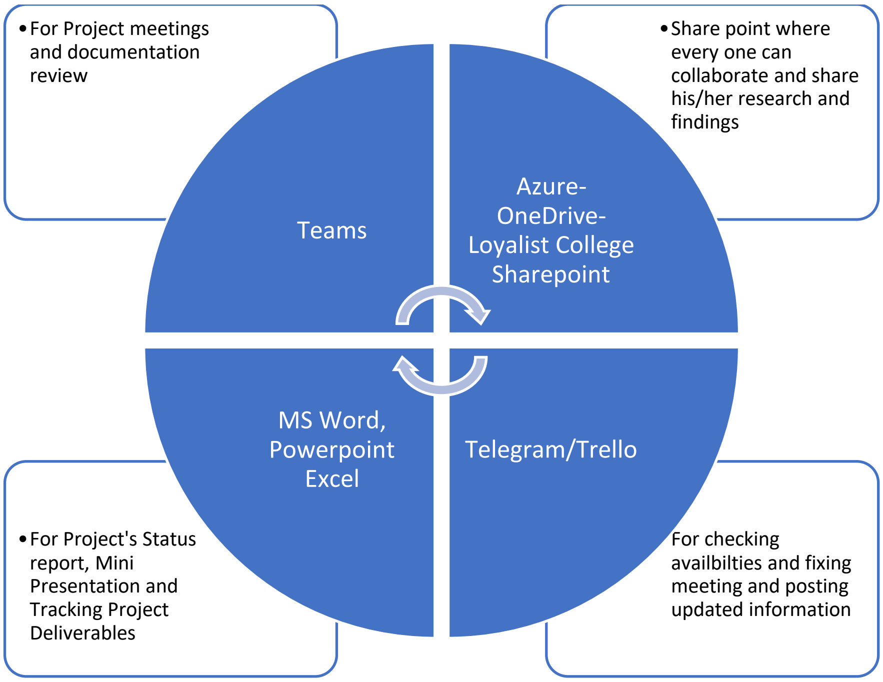
4. Project Management

Organizational Chart- Roles and Responsibilities of Team Members:

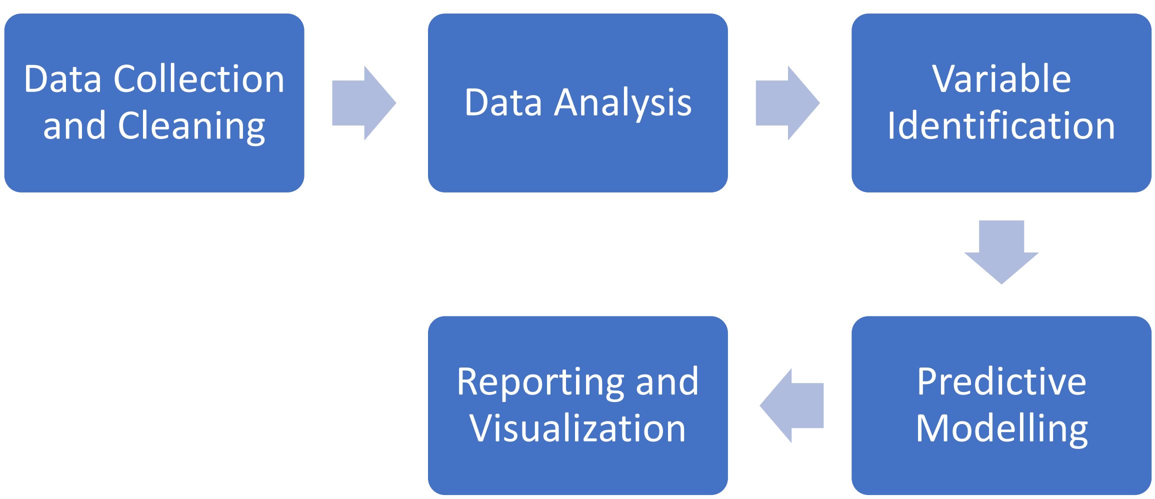


5. Platform and Tools



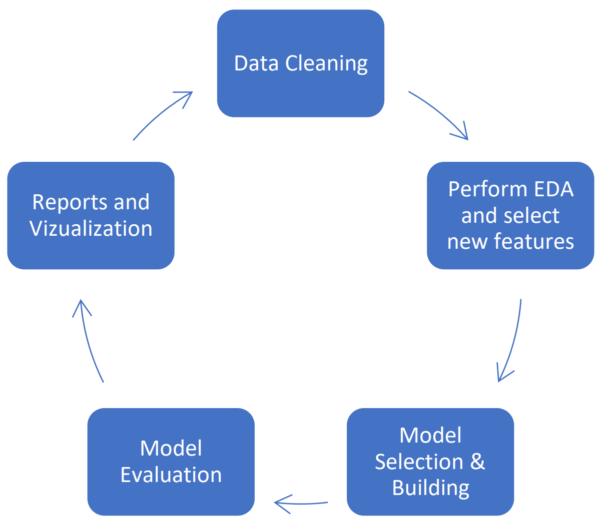


6. Project Scope and Solution Flow



The project's scope includes:

- Collection and Analysis: Gathering and analyzing comprehensive data on car pricing within the US market.
- Variable Identification: Identifying the variables that play a key role in determining car prices.
- Predictive Modeling: Developing predictive models that accurately forecast car prices based on the selected variables.
- Reporting and Visualization: Creating informative reporting and interactive dashboards to convey insights to Kuruma Auto.



The above figure is planned under continuous improvement solution flow.

7. Technical Description

In this project we have explored various technical techniques like statistical analysis, machine learning techniques, and data visualization to extract actionable insights from complex dataset. These insights will play an important role in identifying the unique set of essential variables that have the most significant influence on car pricing in the US market.

BUNSEKI

8. Methodology and Problem Solving

8.1 Data Understanding and Exploration

So, we have this dataset under .csv file - Spring 2023 - AISC - Kuruma_CarPrice_Ass

We looked at this dataset to understand the size and attribute names etc.

Figure 1 shows the names of the columns, count of rows and columns and some more statistics.

In [4]:
```
df_cars.columns
```

Out[4]:
```
Index(['CarCompany', 'fueltype', 'aspiration', 'doornumber', 'carbody',
       'drivewheel', 'enginelocation', 'wheelbase', 'carlength', 'carwidth',
       'carheight', 'curbweight', 'enginetype', 'cylindernumber', 'enginesize',
       'fuelsystem', 'boreratio', 'stroke', 'compressionratio', 'horsepower',
       'peakrpm', 'citympg', 'highwaympg', 'price'],
      dtype='object')
```

In [5]:
```
df_cars.shape
```

Out[5]:
```
(205, 24)
```

Data has 205 rows and 26 columns

In [6]:
```
df_cars.describe()
```

Out[6]:

	wheelbase	carlength	carwidth	carheight	curbweight	enginesize	boreratio	stroke	compressionratio	horsepow
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.00000
mean	98.756585	174.049268	65.907805	53.724878	2555.565854	126.907317	3.329756	3.255415	10.142537	104.1170
std	6.021776	12.337289	2.145204	2.443522	520.680204	41.642693	0.270844	0.313597	3.972040	39.5441
min	86.600000	141.100000	60.300000	47.800000	1488.000000	61.000000	2.540000	2.070000	7.000000	48.00000
25%	94.500000	166.300000	64.100000	52.000000	2145.000000	97.000000	3.150000	3.110000	8.600000	70.00000
50%	97.000000	173.200000	65.500000	54.100000	2414.000000	120.000000	3.310000	3.290000	9.000000	95.00000
75%	102.400000	183.100000	66.900000	55.500000	2935.000000	141.000000	3.580000	3.410000	9.400000	116.00000
max	120.900000	208.100000	72.300000	59.800000	4066.000000	326.000000	3.940000	4.170000	23.000000	288.00000

Figure 1- Understanding the dataset.

Figure 2 shows the data type and non-null count against each Column

```
In [7]:  #Check the null value count and data type
         df_cars.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 24 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   CarCompany        205 non-null    object
 1   fueltype          205 non-null    object
 2   aspiration        205 non-null    object
 3   doornumber        205 non-null    object
 4   carbody           205 non-null    object
 5   drivewheel        205 non-null    object
 6   enginelocation    205 non-null    object
 7   wheelbase         205 non-null    float64
 8   carlength         205 non-null    float64
 9   carwidth          205 non-null    float64
 10  carheight         205 non-null    float64
 11  curbweight        205 non-null    int64
 12  enginetype        205 non-null    object
 13  cylindernumber    205 non-null    object
 14  enginesize        205 non-null    int64
 15  fuelsystem        205 non-null    object
 16  boreratio         205 non-null    float64
 17  stroke            205 non-null    float64
 18  compressionratio  205 non-null    float64
 19  horsepower        205 non-null    int64
 20  peakrpm           205 non-null    int64
 21  citympg           205 non-null    int64
 22  highwaympg        205 non-null    int64
 23  price             205 non-null    float64
dtypes: float64(8), int64(6), object(10)
memory usage: 38.6+ KB
```

Figure 2- Data Types

Figure 3 shows that there are no duplicate values in the given dataset.

```
In [9]:   df_cars['CarCompany'].unique()

Out[9]:  array(['alfa-romero', 'audi', 'bmw', 'chevrolet', 'dodge', 'honda',
                'isuzu', 'jaguar', 'maxda', 'mazda', 'buick', 'mercury',
                'mitsubishi', 'Nissan', 'nissan', 'peugeot', 'plymouth', 'porsche',
                'porcshce', 'renault', 'saab', 'subaru', 'toyota', 'toyouta',
                'vokswagen', 'volkswagen', 'vw', 'volvo'], dtype=object)

In [10]:  df_cars.duplicated().sum()

Out[10]: 0
```

Figure 3- Data Integrity

As a conclusion:

There are 205 rows and 25 columns.
No null values in dataset.
out of 24 features 16 are numeric and rest are categorical.
there are no duplicate values.

8.2 Data Cleaning

Removing inconsistent dataentries in the column of CompanyName
The following companies name have been entered incorrect:-

mazda as maxda
porsche as porcshce
toyota as toyouta
volkswagon as vokswagen, vw

Figure 4 shows the code of performing some cleaning steps which is necessary for data visualization and during model building.

```
In [11]: df_cars['CarCompany'] = df_cars['CarCompany'].str.capitalize()
         df_cars['CarCompany'].unique()

Out[11]: array(['Alfa-romero', 'Audi', 'Bmw', 'Chevrolet', 'Dodge', 'Honda',
                'Isuzu', 'Jaguar', 'Maxda', 'Mazda', 'Buick', 'Mercury',
                'Mitsubishi', 'Nissan', 'Peugeot', 'Plymouth', 'Porsche',
                'Porcshce', 'Renault', 'Saab', 'Subaru', 'Toyota', 'Toyouta',
                'Vokswagen', 'Volkswagen', 'Vw', 'Volvo'], dtype=object)

In [12]: df_cars.CarCompany.replace('Maxda','Mazda',inplace=True)
         df_cars.CarCompany.replace('Porcshce','Porsche',inplace=True)
         df_cars.CarCompany.replace('Toyouta','Toyota',inplace=True)
         df_cars.CarCompany.replace('Vokswagen','Volkswagen',inplace=True)
         df_cars.CarCompany.replace('Vw','Volkswagen',inplace=True)

         #df_cars.CarCompany.unique()

In [13]: df_cars['CarCompany'] = df_cars['CarCompany'].str.capitalize()
         df_cars['CarCompany'].unique()

Out[13]: array(['Alfa-romero', 'Audi', 'Bmw', 'Chevrolet', 'Dodge', 'Honda',
                'Isuzu', 'Jaguar', 'Mazda', 'Buick', 'Mercury', 'Mitsubishi',
                'Nissan', 'Peugeot', 'Plymouth', 'Porsche', 'Renault', 'Saab',
                'Subaru', 'Toyota', 'Volkswagen', 'Volvo'], dtype=object)

In [14]: df_cars.to_csv('Kuruma_Auto_Clean_Data_Spring_2023.csv')
```

Figure 4- Data Cleaning

8.3 Data Visualization

In [16]:

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

# Create a new figure with a specific size
plt.figure(figsize=(20, 8))

# First subplot: Car Price Distribution Plot
plt.subplot(1, 2, 1)
plt.title('Car Price Distribution Plot')
sns.histplot(df_cars.price, kde=True)

# Second subplot: Car Price Spread
plt.subplot(1, 2, 2)
plt.title('Car Price Spread')
sns.boxplot(y=df_cars.price)

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

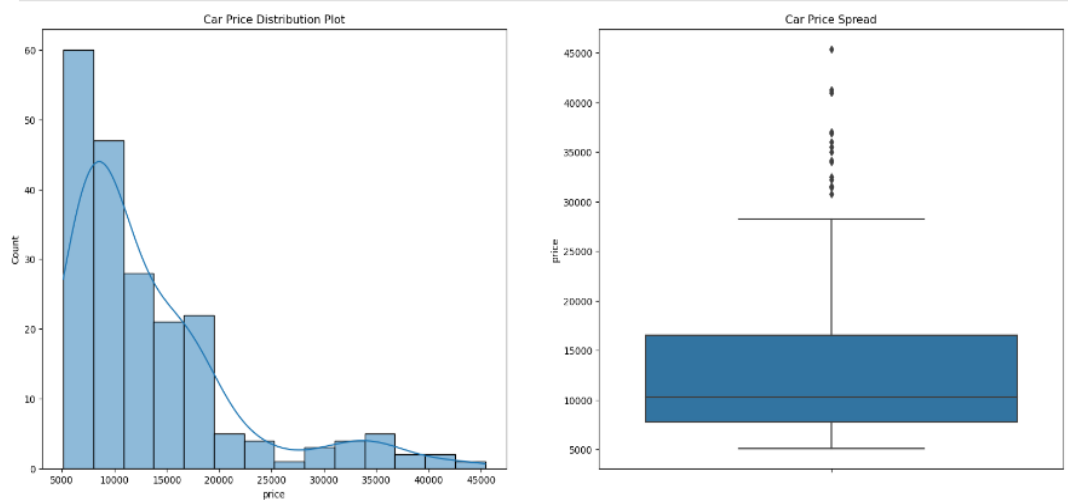


Figure 5- Histogram and Box Plot for Car Price

In [17]:

```python
print(df_cars.price.describe(percentiles = [0.25,0.50,0.75,0.85,0.90,1]))
```

```
count      205.000000
mean     13276.710571
std       7988.852332
min       5118.000000
25%       7788.000000
50%      10295.000000
75%      16503.000000
85%      18500.000000
90%      22563.000000
100%     45400.000000
max      45400.000000
Name: price, dtype: float64
```

Figure 6- Understanding the car price.

Inference:

- The plot seemed to be right-skewed, meaning that the most prices in the dataset are low (Below 15,000).
- There is a significant difference between the mean and the median of the price distribution.
- The data points are far spread out from the mean, which indicates a high variance in the car prices. (85% of the prices are below 18,500, whereas the remaining 15% are between 18,500 and 45,400.)

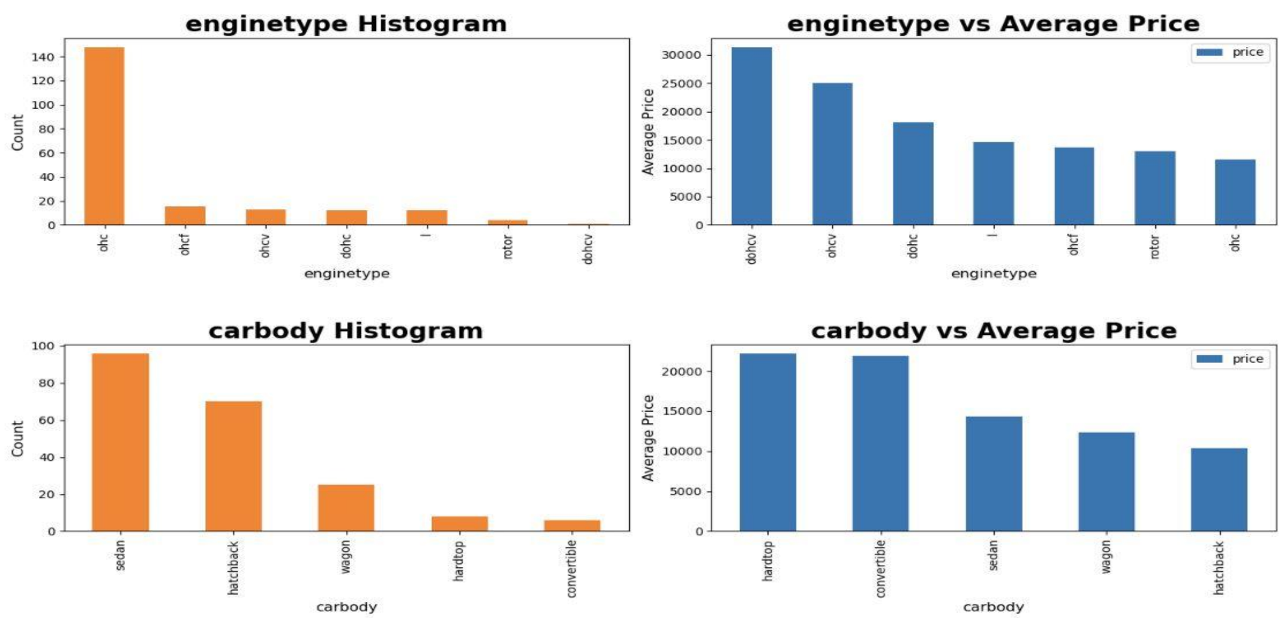


Figure 7- Histogram of Various features.

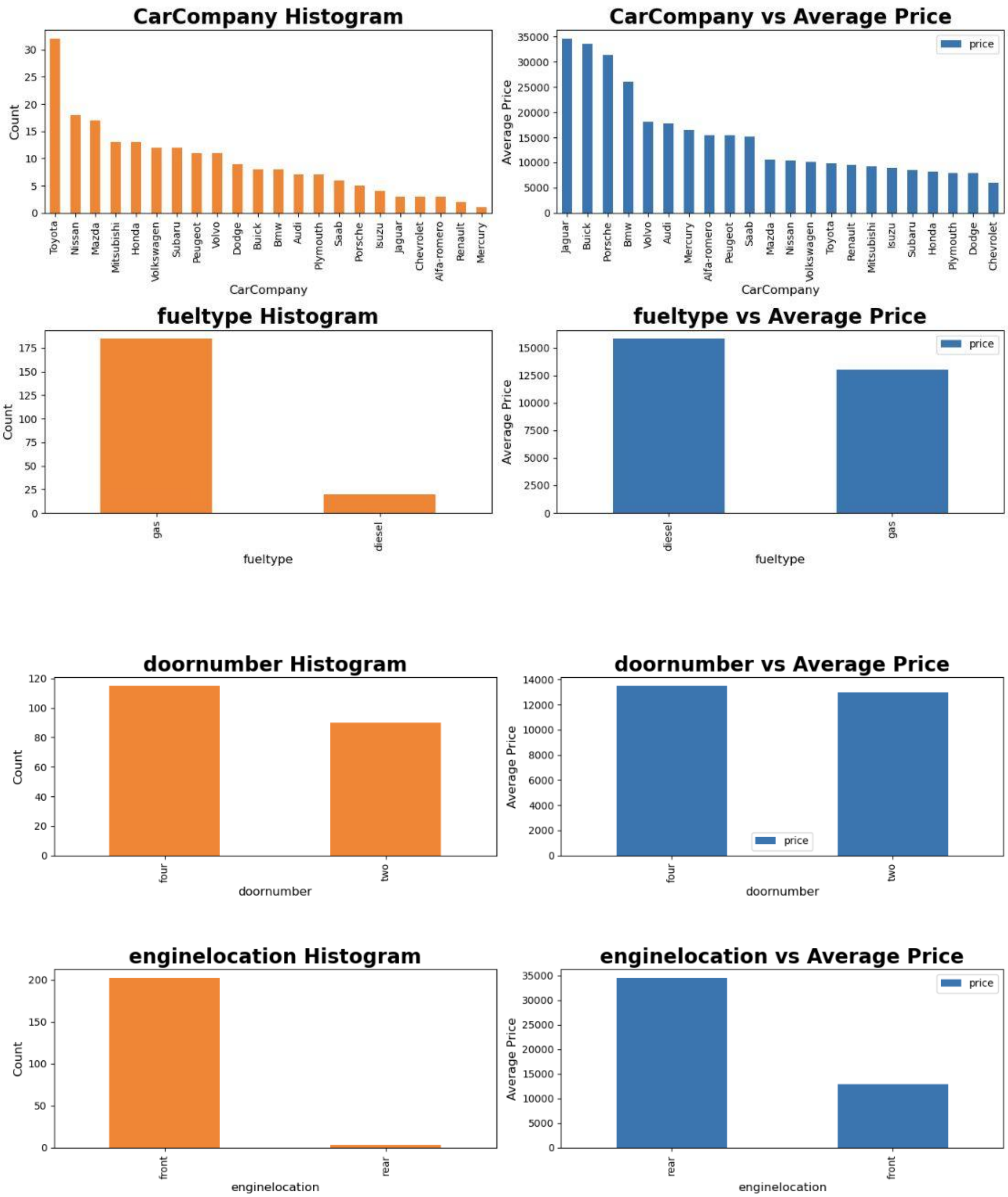


Figure 8-Histogram of various features.

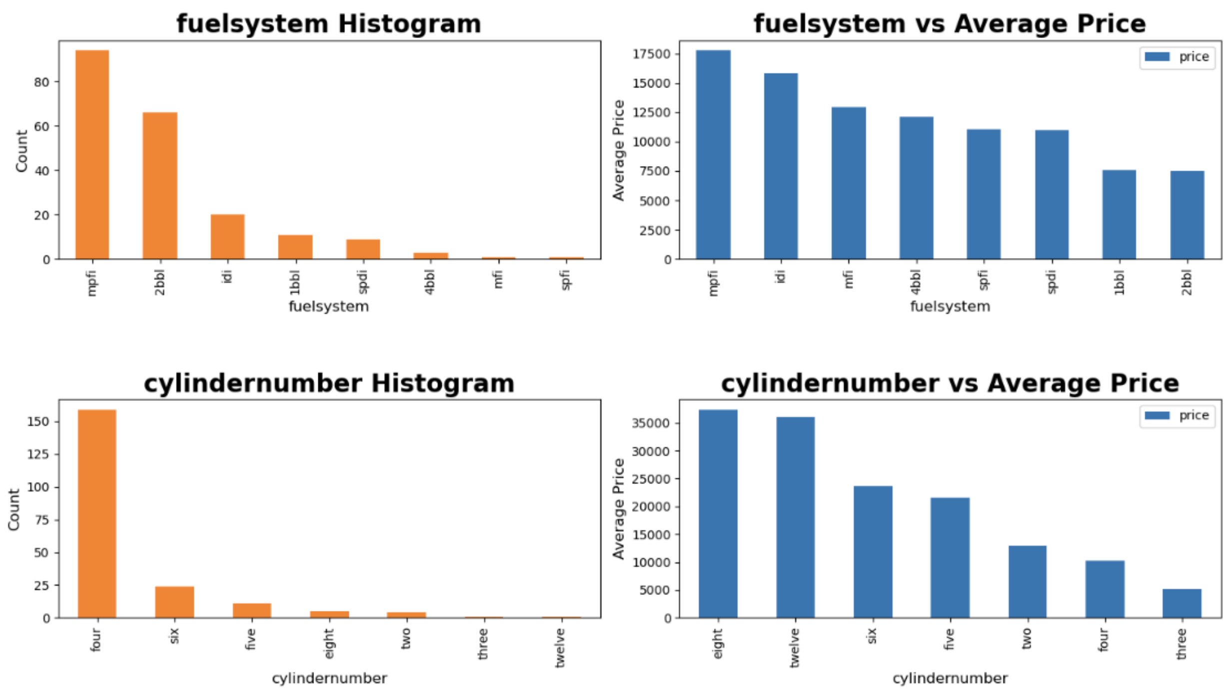


Figure 9- Histogram of various features

Insights:

- Toyota is the most preferred car company.
- Gas is the most preferred fuel type, and the average price of gas type vehicle is also less.
- Ohc is preferred engine type and average price of ohc vehicle is the least among all.
- Hardtop and Convertible vehicles are costlier than others.
- Cars with engines in rear are more than double the average cost of cars with engines in front.
- A four-cylinder car is preferred most, eight- and twelve-cylinder cars are the costliest.

Visualizing the numerical features

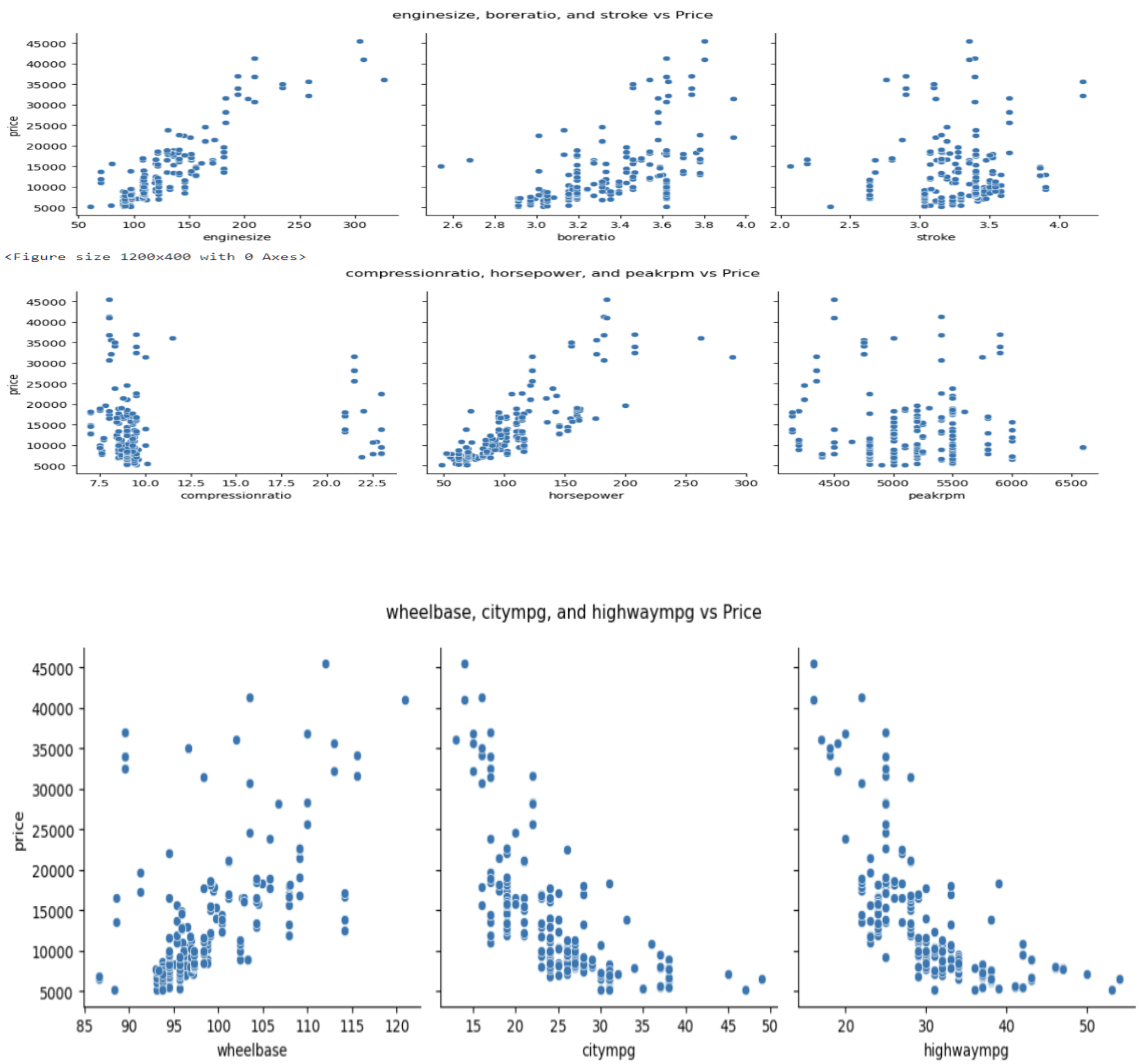


Figure 10- Scatter plot of various features vs price

Inference:

enginesize, boreratio, horsepower, wheelbase - seem to have a significant positive correlation with price.

citympg, highwaympg - seem to have a significant negative correlation with price.

8.4 Model Building and Evaluation

```
# Create and evaluate Decision Tree Regressor model
dt_model = DecisionTreeRegressor(max_depth=3)
dt_model.fit(X_train, y_train)
dt_pred = dt_model.predict(X_test)
print(dt_pred)
dt_mse = mean_squared_error(y_test, dt_pred)
dt_cv_scores = cross_val_score(dt_model, X, y, cv=5, scoring='neg_mean_squared_error')
dt_mse_scores = -dt_cv_scores
dt_scores = cross_val_score(dt_model, X, y, cv=4)
dt_accuracy = dt_scores.mean()

# Print evaluation results
print('Linear Regression:')
print('MSE:', linear_mse)
print('Cross-Validation MSE Scores:', linear_mse_scores)
print('Average Cross-Validation MSE:', linear_mse_scores.mean())
print('Linear Regression Accuracy:', linear_accuracy)
print()


print('Decision Tree Regressor:')
print('MSE:', dt_mse)
print('Cross-Validation MSE Scores:', dt_mse_scores)
print('Average Cross-Validation MSE:', dt_mse_scores.mean())
print('dt Accuracy:', dt_accuracy)
print()

# Plot the regression results
plt.figure(figsize=(12, 8))
```

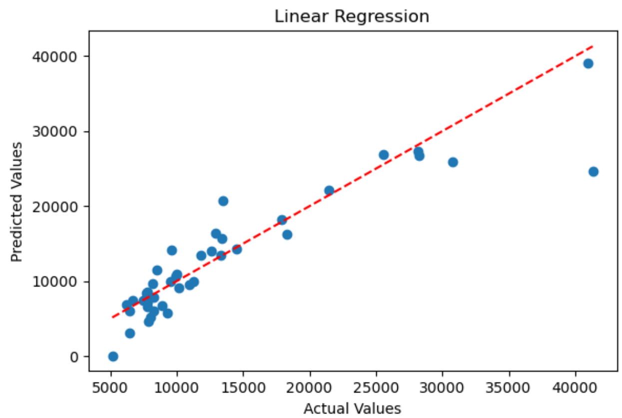
```
# Linear Regression plot
plt.subplot(221)
plt.scatter(y_test, linear_pred)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--')
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Linear Regression')


# Decision Tree Regressor plot
plt.subplot(224)
plt.scatter(y_test, dt_pred)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red', linestyle='--')
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Decision Tree Regressor')

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

```
[25961.14687736 18206.48519092  9903.2219374  13390.77568701
 26774.26934086  6561.60234729  8555.14961019  5823.5145299
  9185.5434292   7141.10195045 13424.87140895  6015.74460999
 16219.62103551 10874.0101599  39143.30948182  6103.55379982
    62.93576758 14051.31725151  9663.30050927 10614.59244262
 11485.52782296 20788.54999049  7914.5095034   3122.55486958
  7457.95088465 24595.2902726  14171.10623282 15766.32251681
  5216.17008215 16372.76429424 26870.89074497  6829.25520531
  4596.91125347 22159.31879722  8363.84415013 27363.81856337
  9930.74470854  9596.48548127  6782.31149865 14312.29823764
  7504.75526471]
[34915.         14816.8358209   8576.68292683 14816.8358209
 34915.          6614.45945946  8576.68292683  8576.68292683
 14816.8358209   8576.68292683 14816.8358209   8576.68292683
 14816.8358209   8576.68292683 45400.          6614.45945946
  6614.45945946 14816.8358209   8576.68292683  8576.68292683
  8576.68292683 14816.8358209   6614.45945946  6614.45945946
  6614.45945946 45400.          8576.68292683 14816.8358209
  6614.45945946 14816.8358209  34915.          6614.45945946
  8576.68292683 22518.28571429  8576.68292683 34915.
 14816.8358209  14816.8358209   6614.45945946 14816.8358209
  8576.68292683]
Linear Regression:
MSE: 12415511.42325991
Cross-Validation MSE Scores: [22351083.89939087 13727576.45056673
21025243.02291986 39554473.74518143
  8168827.02212458]
Average Cross-Validation MSE: 20965440.828036692
Linear Regression Accuracy: 0.548926691770146

Decision Tree Regressor:
MSE: 8532131.517107163
Cross-Validation MSE Scores: [ 9510829.33422777 13350381.5871635
25026994.72214809  5986912.12980365
  4554686.75680382]
Average Cross-Validation MSE: 11685960.906029368
dt Accuracy: 0.8446025597058141
```



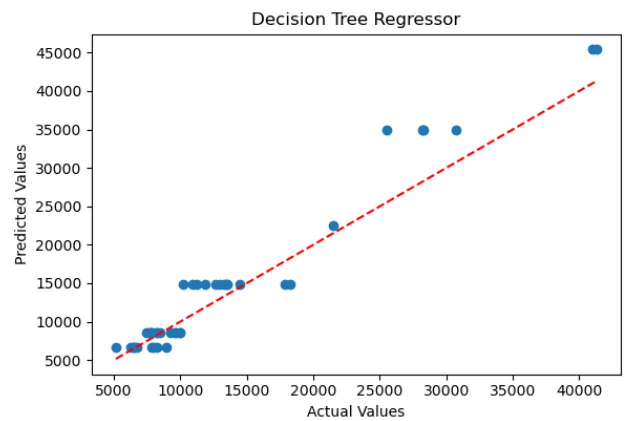


Figure 11- Predicted Price values with LR and DT

In [29]:
```
# Calculate the accuracies
linear_accuracy = linear_scores.mean()
dt_accuracy = dt_scores.mean()

# Print the accuracies
print('Linear Regression Accuracy:', linear_accuracy)
print('Decision Tree Regressor Accuracy:', dt_accuracy)

# Create a bar chart for model accuracies
models = ['Linear Regression', 'Decision Tree Regressor']
accuracies = [linear_accuracy, dt_accuracy]

plt.bar(models, accuracies)
plt.xlabel('Models')
plt.ylabel('Accuracy')
plt.title('Model Accuracies')
plt.show()
```

```
Linear Regression Accuracy: 0.548926691770146
Decision Tree Regressor Accuracy: 0.8446025597058141
```

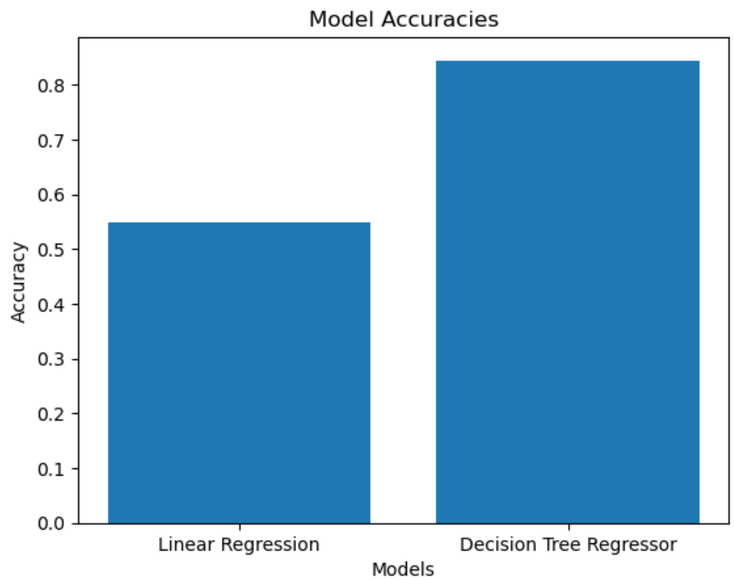


Figure 12- Accuracy -LR vs DT

Conclusion- Figure 12 shows that Decision Tree model performs better on the given data set as it has accuracy of ~85 whereas LR has accuracy of ~55.

We can see that the decision tree regressor has a higher accuracy of 0.8446025597058141. This shows that the decision tree model can explain approximately 84.46% of the variance in the data, which means it has better predictive power compared to the linear regression model.

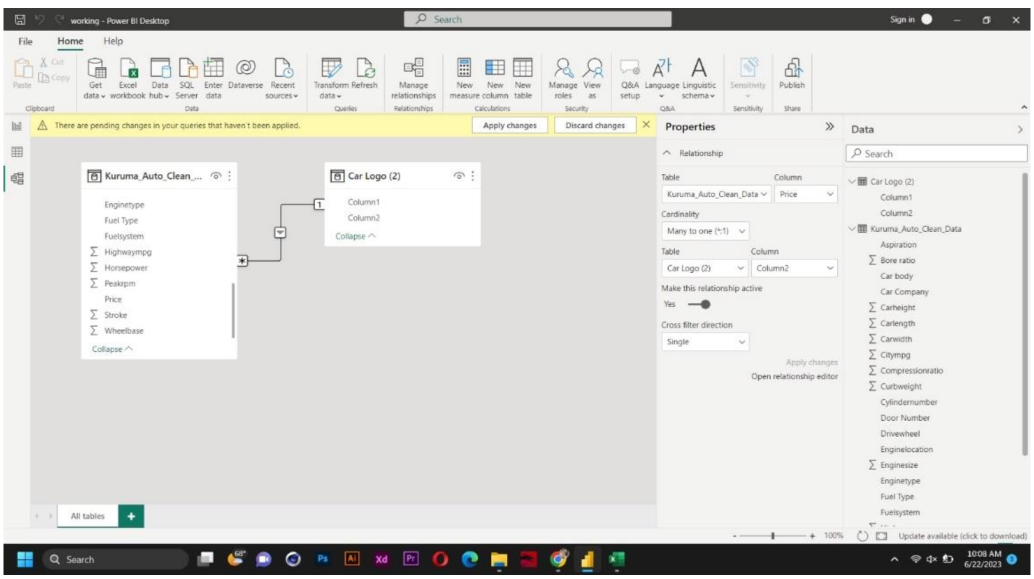
Linear Regressor vs Decision Tree Regressor

We know that both are two different types of algorithms which are used for regression tasks. But their performance results can have huge differences based on the nature of the dataset, relationship between features and target variables.

Decision Trees can model non-linear relations between features and target variables, whereas LR models typically assume a linear relationship between features. Therefore, If the data contains complex non-linear relationships, a Decision Tree will be able to model them better.

Decision Trees are good at dealing with feature interactions, where one feature's impact is dependent on another feature's value. Linear Regression may not be able to capture these interactions. So, that's why DT performs better than LR sometimes

8.5 Building a pipeline in Power BI desktop:



Building an interactive Dashboard using Power BI:

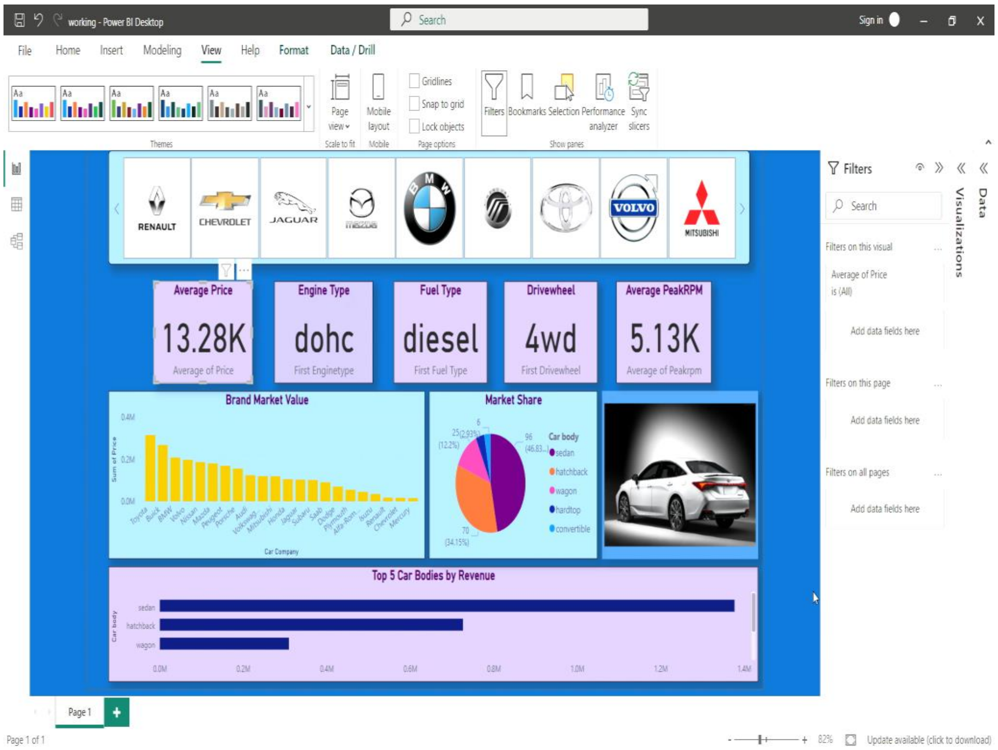


Figure 13- A view of an interactive dashboard

We developed an interactive Dashboard using Power BI tool. We selected this tool as it provides interactive visualizations and BI features which are user friendly and anyone from either technical or non-technical field can take advantage of this.

Generally, the term interactive here means that if we click on any car logo at the top of the dashboard (in figure 13), it will show all the information and insights about that car company in US market and most importantly the present market share, average price, and the most popular car's model.

9. Project Resources

The following tools and techniques we used in our Project:

Python, GitHub, OneDrive, Trello, Wix and Power BI.

https://trello.com/b/fUN2NAjS/project-management
https://github.com/mohammedtareeq786/Kuruma_Auto

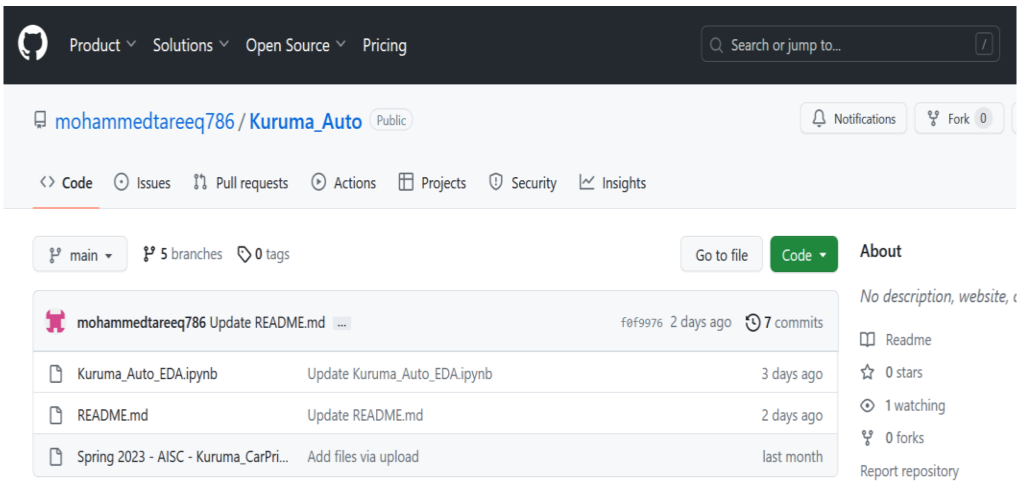


Figure 14- GitHub Profile

Website: https://sohinirchowdhury98.wixsite.com/bunseki

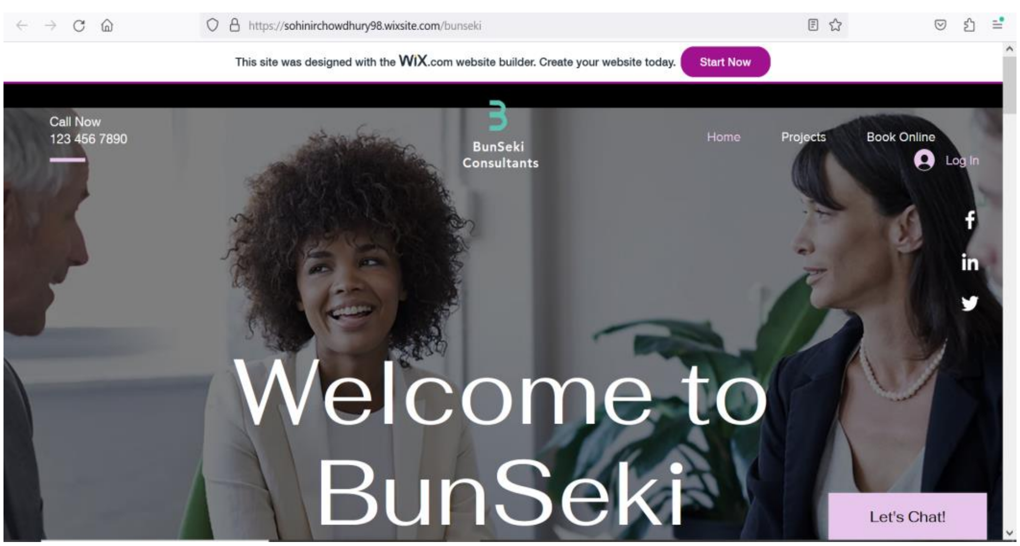


Figure 15- The Project Portfolio website

10. Findings and Analysis

- Toyota is a most preffered car.

Toyota

- Gas is a most preffered fuel type.

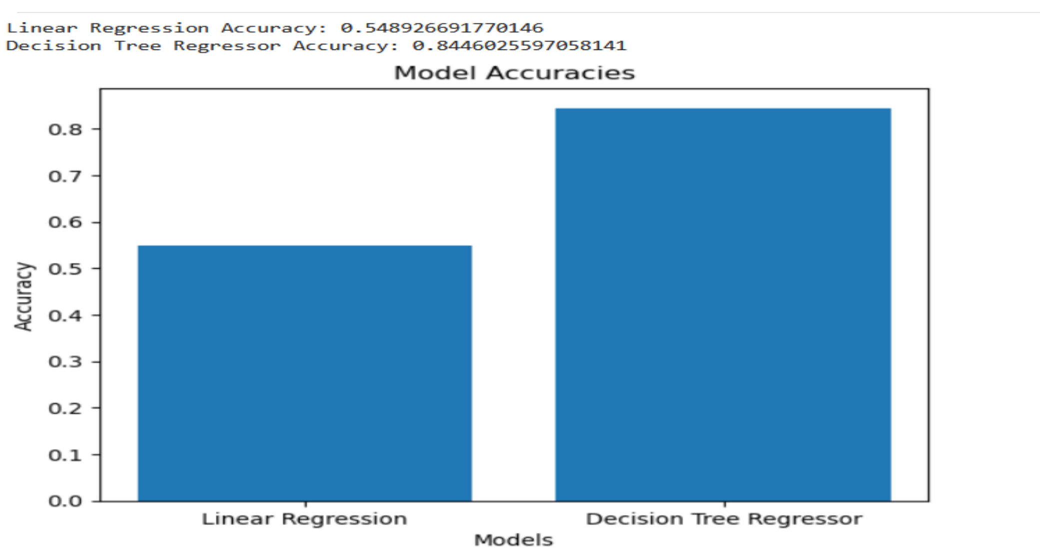
Gas

- A car which is having atleast 4 clyinders is preffered.

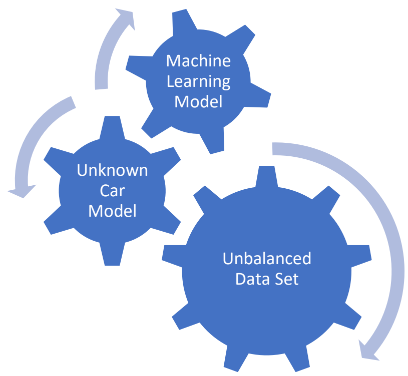
Cylinder

11. Drawbacks and Failings

Accuracy of our model with Linear regression was 55% and we worked on this and tried different model and at present we have reached up to ~85% with Decision Tree.



We are unable to improve it further, maybe because of the nature of the data set we have got after cleaning and picking features for our analysis and model prediction.



12. Project Deliverables Timelines

Schedule	Deliverable	Completion Date
Week-3	Collecting Data	26-06-23
Week-4	Data Cleaning and EDA	01-06-23
Week-5	Feature Engineering	08-06-23
Week-6	Selecting and Building a ML Model	15-06-23
Week-7	Building an initial model	22-06-23
Week-8	Refining a model	29-06-23
Week-9	Final Predictive model with improved accuracy	07-07-23
Week-10	Building Reports and Visualization	14-07-23
Week-11	Starting to build a comprehensive project report	21-07-23
Week-12	Report and analysis	28-07-23
Week-13	Showcasing the progress of project	03-08-23
Week-14	Final Project Report review and submission	10-08-23

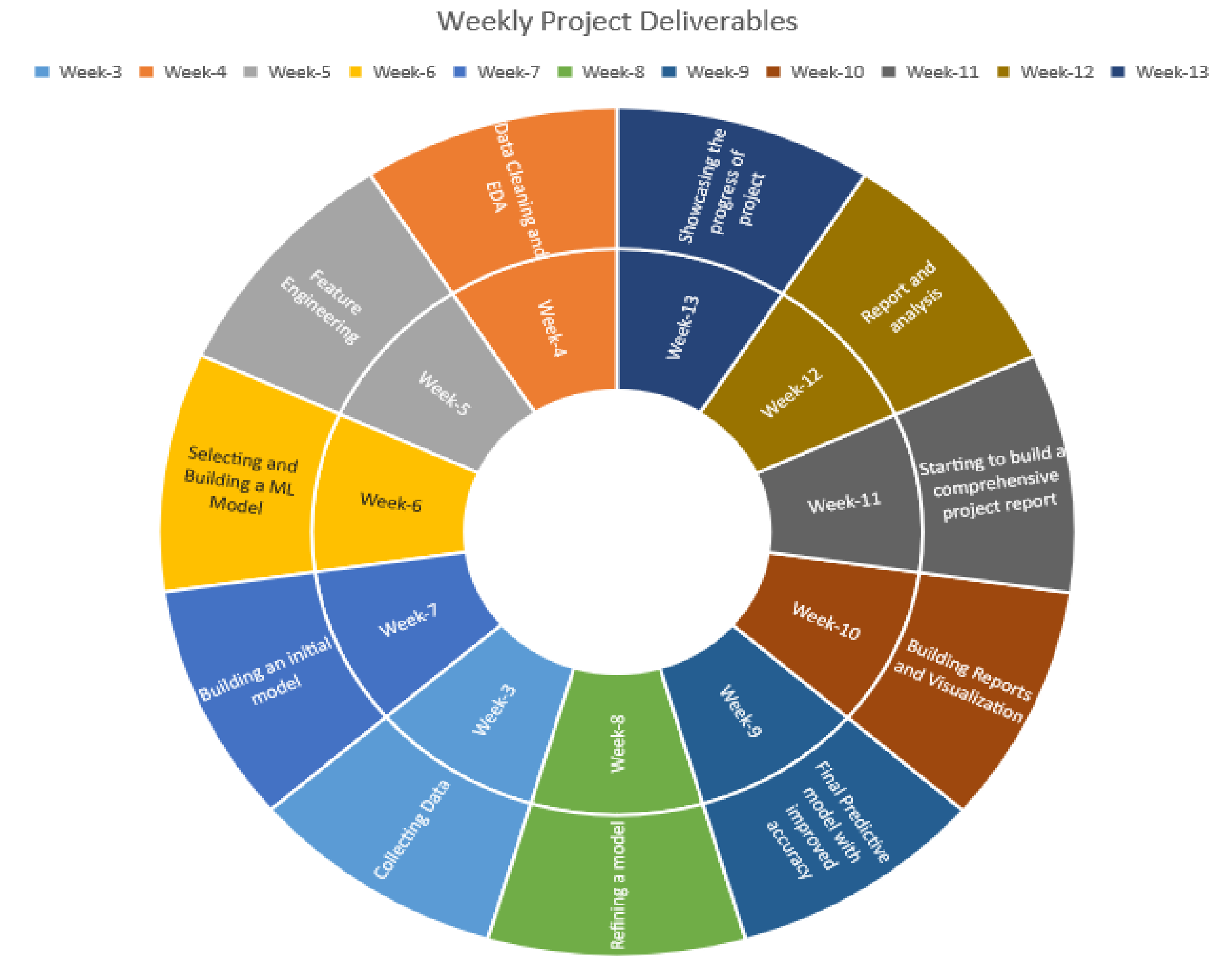


Figure 16- Weekly Project Timelines

13. References

- US Car Price Exploratory Data Analysis | Kaggle
- Predicting Car Prices Using Machine Learning and Data Science | by Suhas Maddali | ODSCJournal | Medium
- Howard Wilner on Why the U.S. Car Market is Dominated by Japanese Cars | by Howard Wilner | Medium
- The rise of Japan: How the car industry was won - The Globe and Mail

14. Members Sign Off

X
Sohini

X
Md Tareeq

X
Dipesh