DEPI Graduation Project

Vehicle Price Prediction

Explore the fascinating world of car pricing in Australia with our comprehensive data science project

Meet our team

[Menna Hesham]

[Arwa Algazzar]

[Karima Mahmoud]

[Toka Mahmoud]

[Mariam Fathi]

[Hager Sherif]

AGENDA

- Business problem
- EDA
- Feature Selection
- Data Visualization
- Machine Learning
- Deployment

BUSINESS PROBLEM

- Analyze and predict car prices in the Australian market for 2023 to understand pricing trends and offer competitive pricing strategies. The dataset contains detailed information on car brands, models, types, and features sold in Australia. Understanding the key factors influencing car prices is crucial for optimizing pricing strategies, enhancing market competitiveness, and providing valuable insights to both buyers and sellers. By leveraging this data, businesses can identify market patterns, improve inventory pricing, and predict future price trends.

Dataset Overview

Car Features

Brand, model, year, and type (car/SUV) are key identifiers. Transmission, engine, and drive type provide technical specs.

Market Factors

New/used status, kilometers driven, and location influence pricing. Fuel type and consumption affect running costs.

Aesthetic Elements

Exterior/interior colors, body type, doors, and seats contribute to a car's appeal and value.

Exploratory Data Analysis (EDA)

1

Data Cleaning

We meticulously cleaned the dataset, handling missing values and outliers. This ensured high-quality insights.

2

Data transformation

converting data from one format or structure into another to make it suitable for analysis or processing.

3

Data exploration

We explored relationships between variables. This helped identify key factors influencing car prices in Australia.

Data cleaning

```
df.replace(['POA', '-', '- / -'], np.nan, inplace=True)
```

- In this step, we are handling missing or problematic data entries by replacing specific values with NaN (Not a Number). The values we are targeting include:
- 'POA' (Price on Application),
- '-' (dash symbols),
- and '- / -' (often used in place of missing data
- By converting these values to NaN, we can easily manage missing data in our analysis, ensuring that the dataset is clean and ready for further processing or modeling.

In this section, we are cleaning and standardizing specific columns in the dataset that contain numerical values stored as text. We use regular expressions to extract the numeric information and convert the columns into the correct data types (float or integer).

FuelConsumption:

- We extract the numeric value from the FuelConsumption column (which may contain text along with numbers) using a regular expression.
- Then, we convert the extracted values into a float type, ensuring the column contains valid numerical data for analysis.

Doors, Seats, CylindersinEngine , Engine:

- For each of these columns, we extract the numerical values (ignoring any text), and convert them into int types.
- If a value is missing, we fill it with 0 to avoid any errors during computations.
- The process ensures that these columns are now properly formatted for analysis, enabling us to perform numerical operations without any data type issues.

```
df['FuelConsumption'] = df['FuelConsumption'].str.extract(r'(\d+\.\d+)').astype(float)
df['Doors'] = df['Doors'].str.extract(r'(\d+)').fillna(0).astype(int)
df['Seats'] = df['Seats'].str.extract(r'(\d+)').fillna(0).astype(int)
df['CylindersinEngine'] = df['CylindersinEngine'].str.extract(r'(\d+)').fillna(0).astype(int)
df['Engine'] = df['Engine'].str.extract(r'(\d+)').fillna(0).astype(int)
```

DATA TRANSFORMATION

- In this step, we separate the dataset columns into two types: numeric and categorical.

Numeric columns:

- We identify columns that contain numerical data types (float32, float64, int64) using select_dtypes() and store their names in a list.

Categorical columns:

- Similarly, we identify columns with categorical data (usually of type object for text) and store them in another list.
- This separation allows us to apply appropriate preprocessing steps based on the data type, ensuring better handling during analysis and modeling.

```
numeric_cols = df.select_dtypes(include=[ 'float32','float64','int64']).columns.tolist()
categorical_cols = df.select_dtypes(include=['object']).columns.tolist()

# Print the identified columns
print("Numeric columns:", numeric_cols)
print("Categorical columns:", categorical_cols)
```

DATA TRANSFORMATION

We applied Label Encoding to convert categorical features into numeric values, which allows us to calculate the correlation between these features and car prices. This process helps machine learning models understand and use categorical data effectively.

Here are the categorical features we transformed:

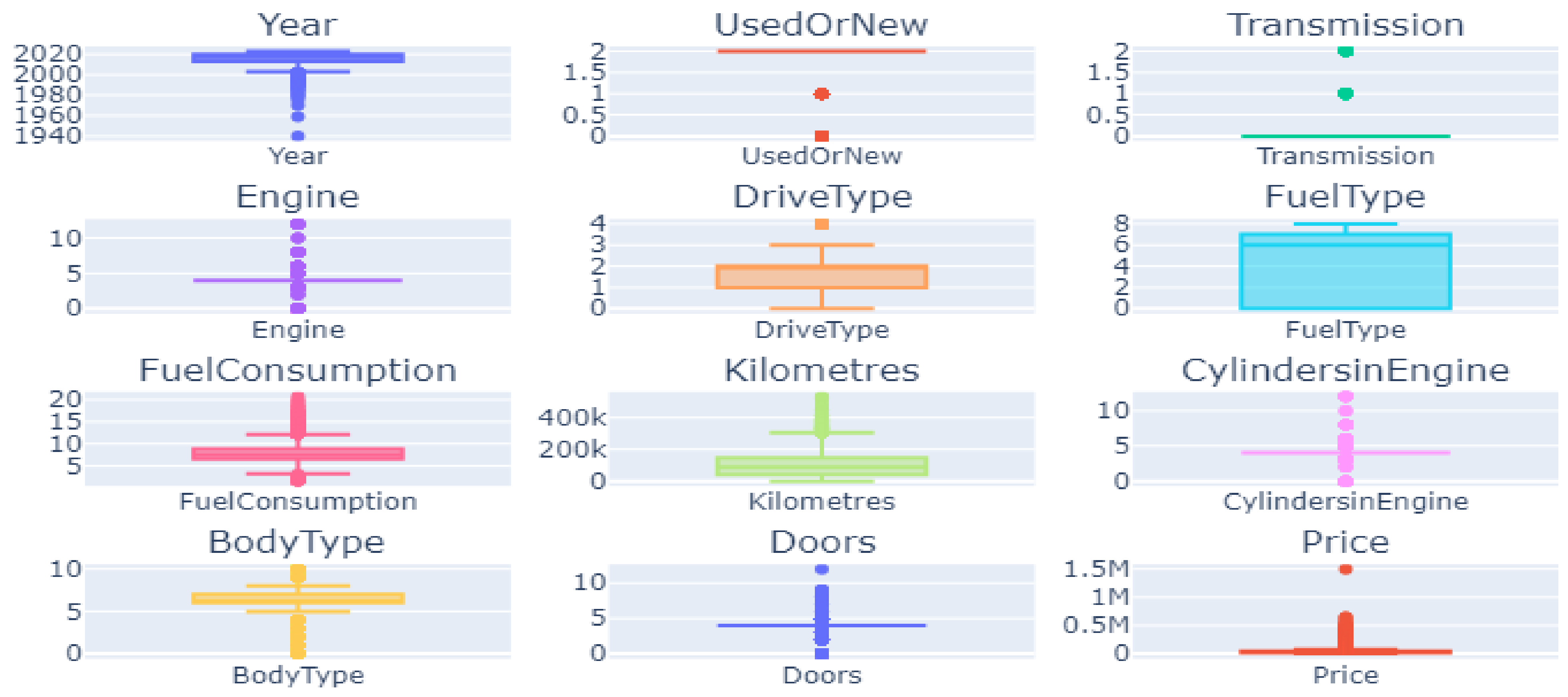
- Brand
- Model
- Transmission
- Fuel Type
- Body Type

Correlation Analysis:
After encoding the categorical data, we calculated the correlation between these features and the Price column. This analysis helps us identify which features have the strongest relationship with car prices, aiding in feature selection and further modeling.

This transformation is crucial for ensuring that categorical data is properly processed and ready for machine learning algorithms.

```python
# Initialize the LabelEncoder
label_encoder = LabelEncoder()

# Iterate through object columns and apply label encoding
for col in df.select_dtypes(include=['object']).columns:
    df[col] = label_encoder.fit_transform(df[col])

# Calculate the correlation matrix
correlation = df.corr()

# Display the correlation between encoded categorical columns and the price
correlation_with_price = correlation['Price']

print(correlation_with_price)
```

DATA EXPLORATION



Made with Gamma

```python
[34] df.drop(columns=['Brand','Model','Car/Suv','Title','Location','ColourExtInt','Seats'],inplace=True)
```

```python
[37] df.dropna(subset=['Year','Price'], inplace=True)
```

```python
[48] df[['Kilometres', 'FuelConsumption']] = df[['Kilometres', 'FuelConsumption']].fillna(df[['Kilometres', 'FuelConsumption']].median())
```

- Handling Outliers
- Removing Irrelevant Columns

Outliers:

Detected outliers using statistical methods (e.g., Z-scores, IQR).

Managed outliers by either capping, transforming, or removing extreme values to ensure robust analysis.

Data Cleaning

Missing Values:

Identified missing data in key columns (e.g., Price, Kilometres).

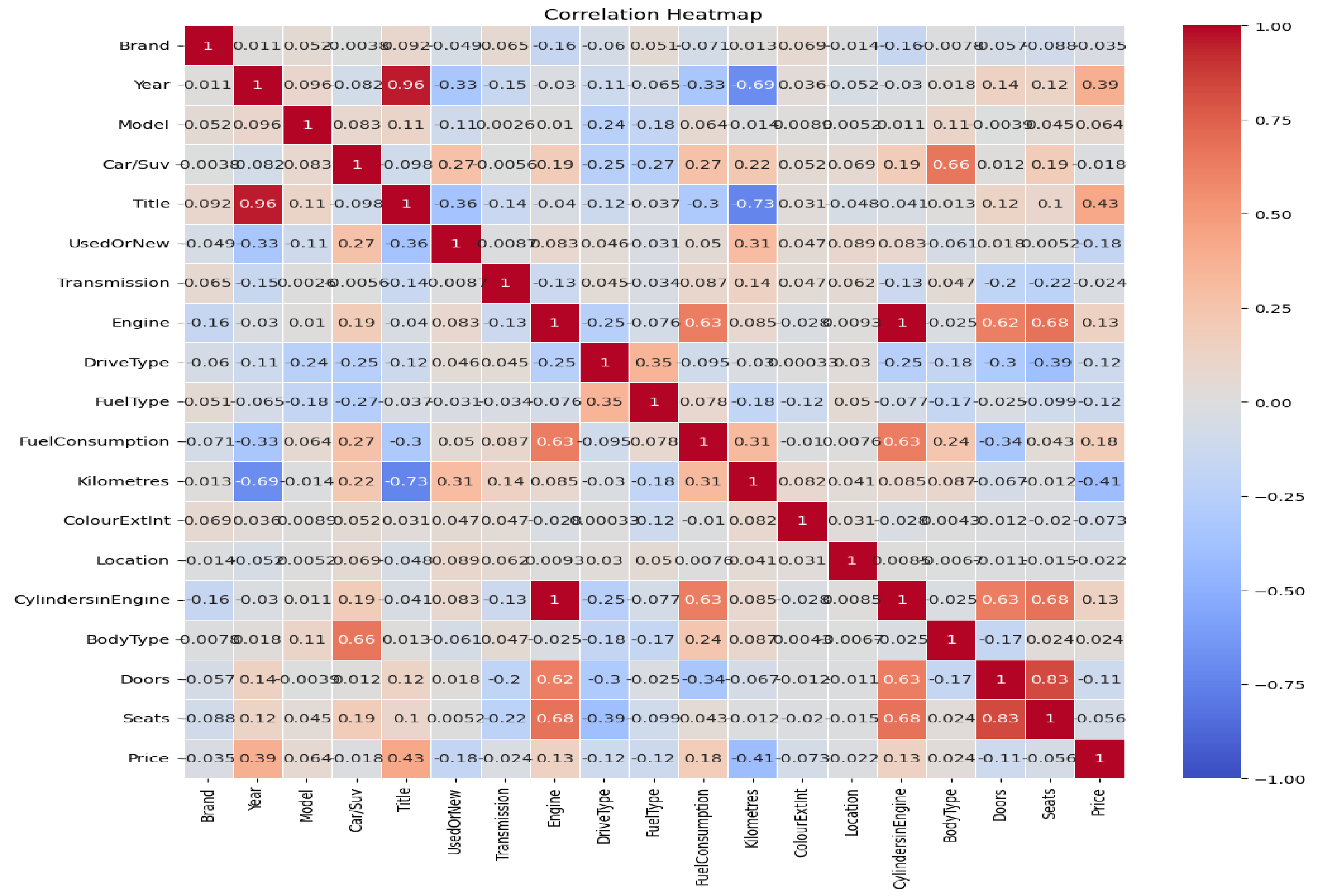
Handled missing values by replacing placeholders like "POA" and "-" with NaN.

Handling Outliers

Made with Gamma

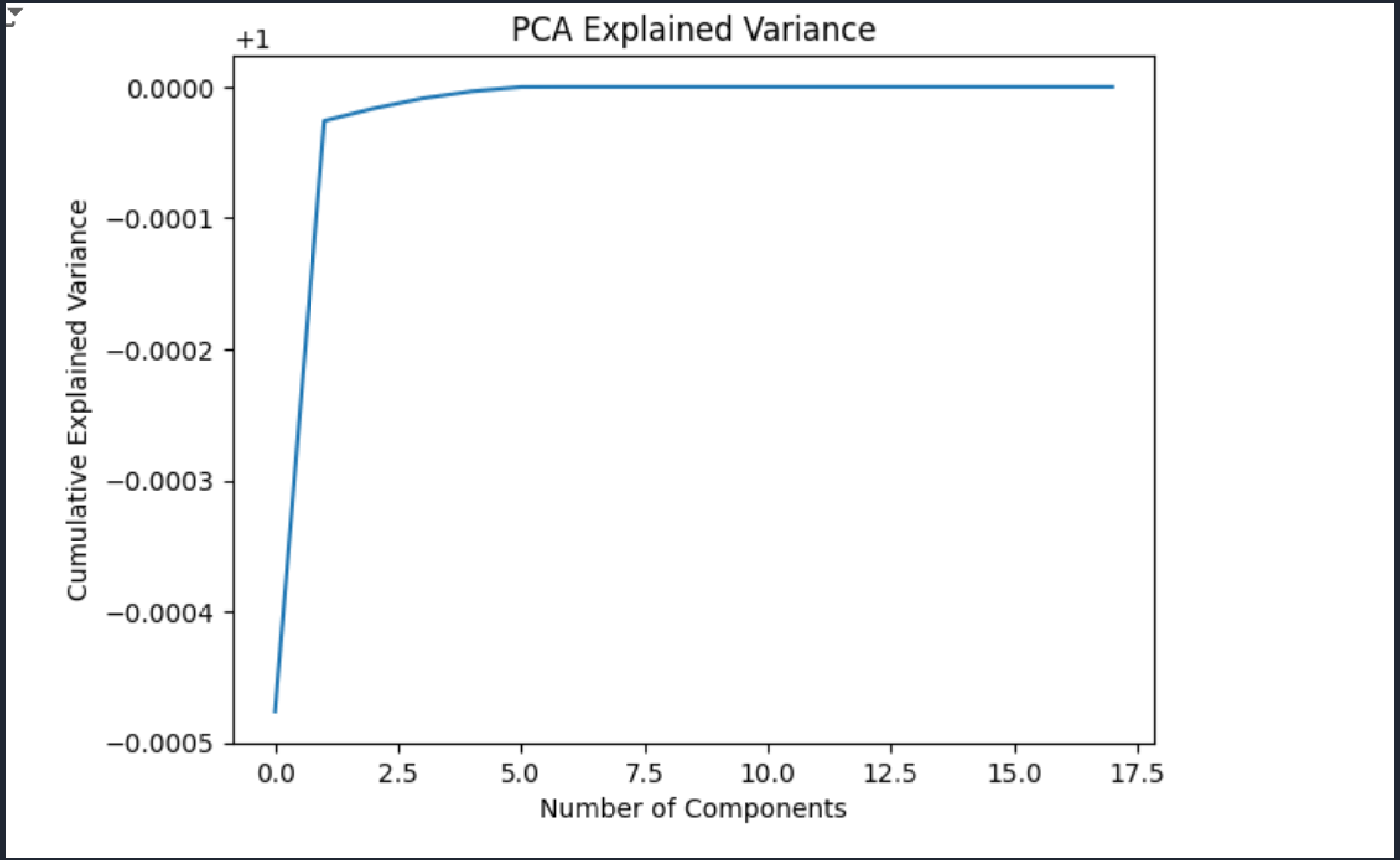
Feature Importance :

First method :
Correlation matrix and heatmap



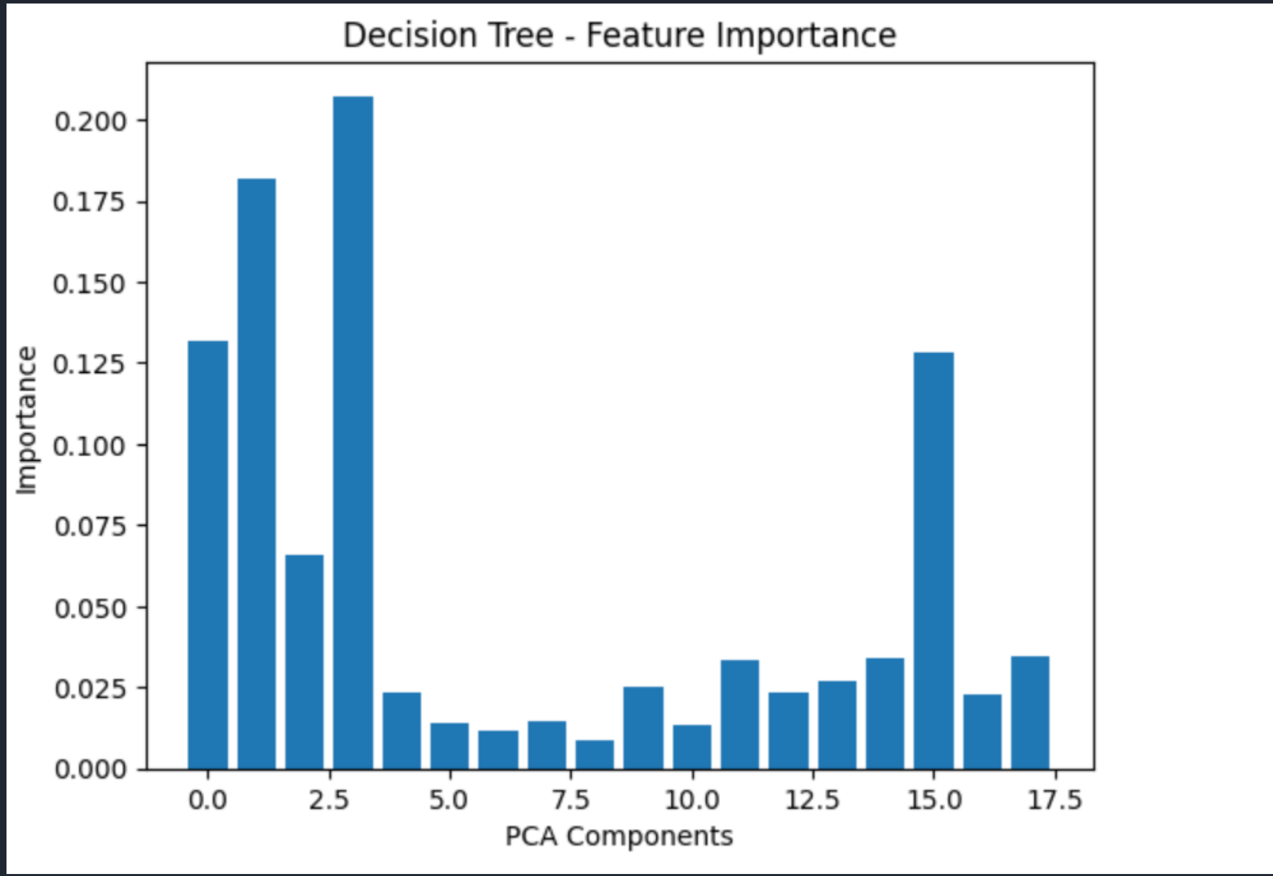
Feature Importance :

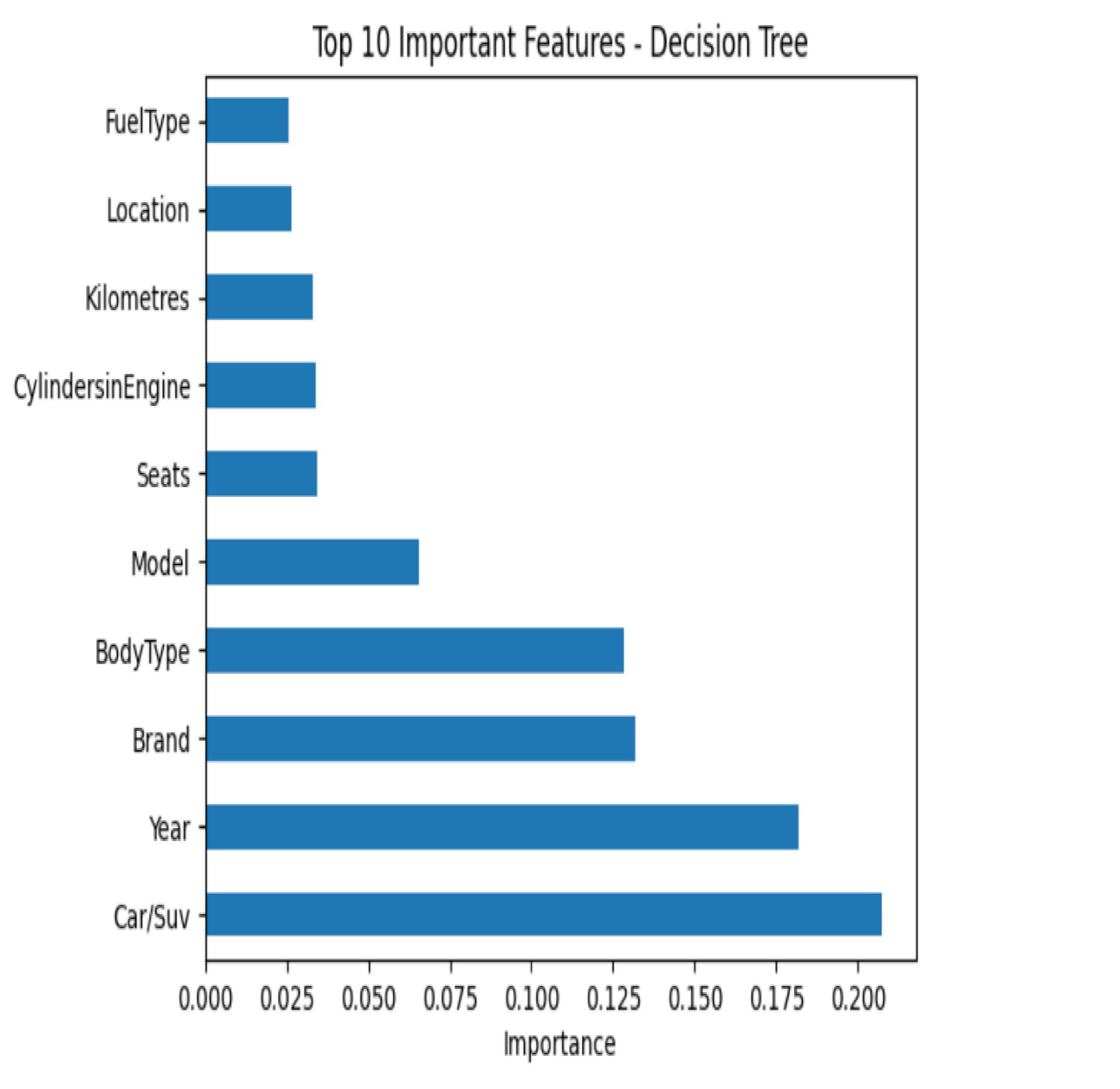
Second method :
PCA (Principal Component Analysis)



Feature Importance:

Third method :
DecisionTreeRegressor for feature Importance





Data Visualization Techniques

Scatter Plots

We used scatter plots to visualize relationships between continuous variables like price and kilometers driven.

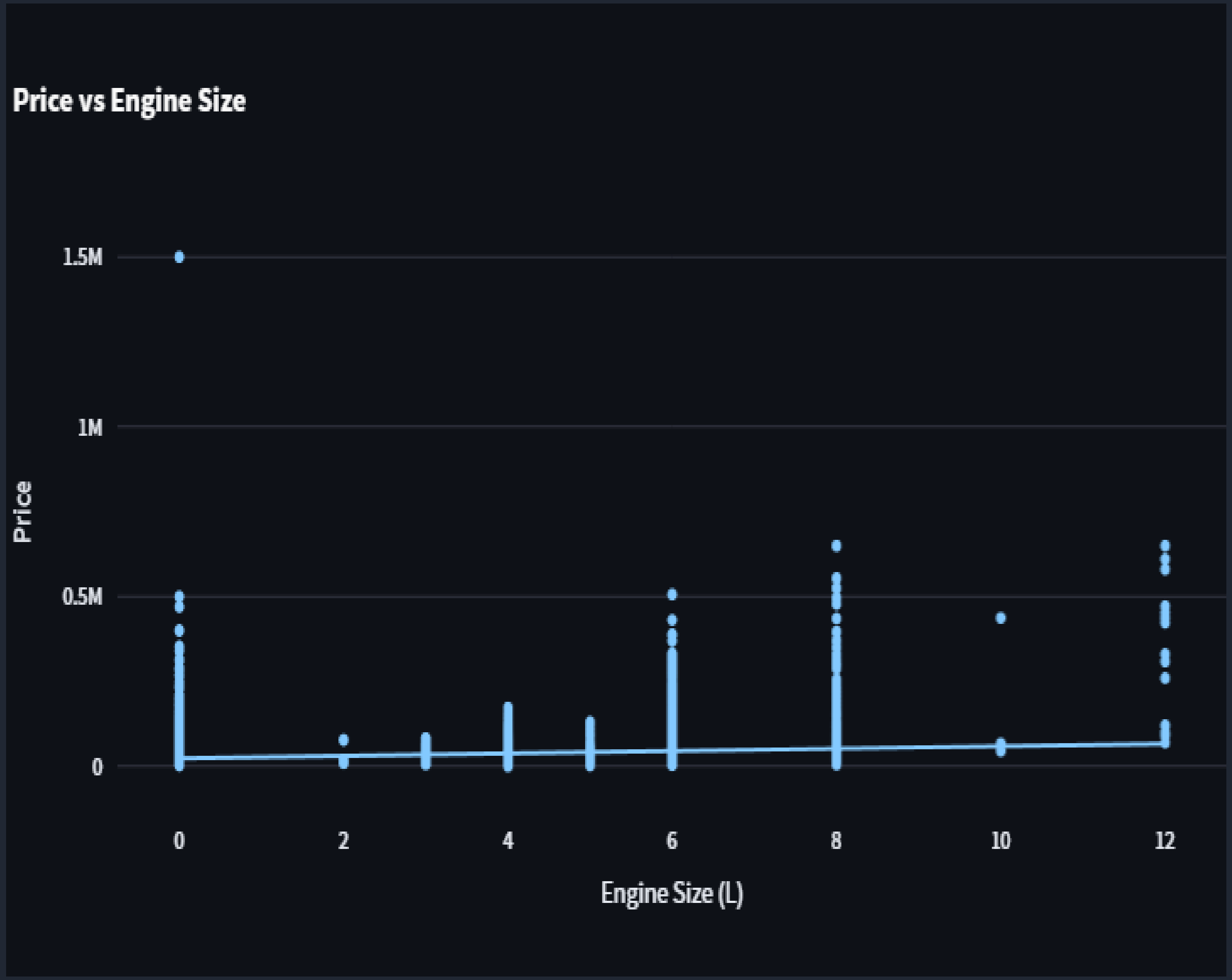
Box Plots

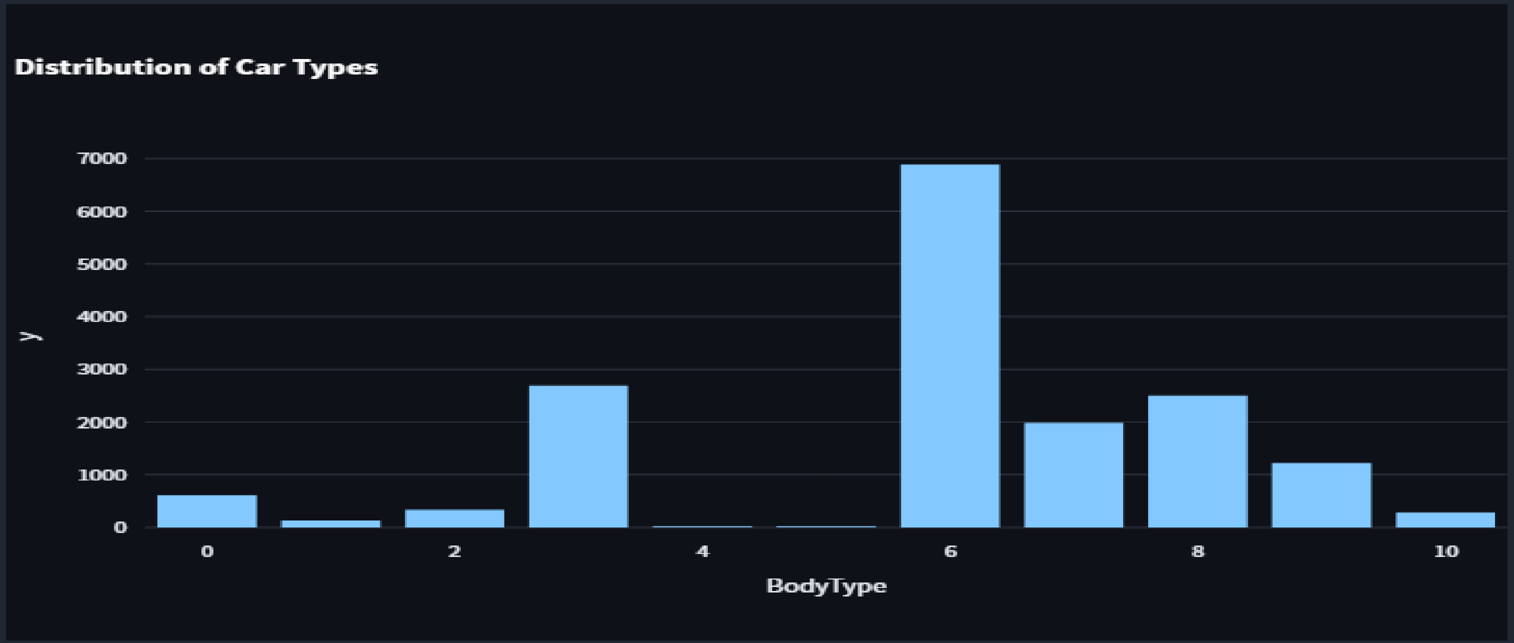
Box plots helped us understand price distributions across categorical variables such as brand and body type.

Heat Maps

We created heat maps to visualize correlations between multiple variables, revealing interesting patterns in the data.

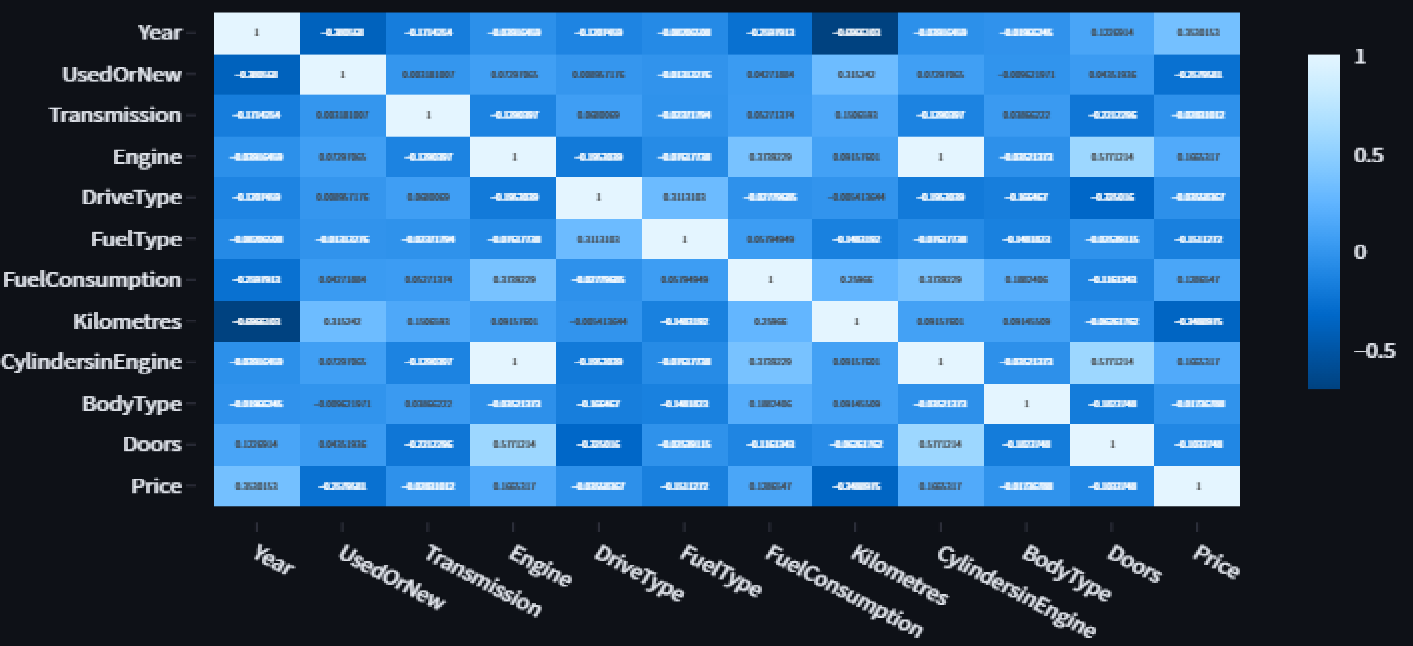
Data Visualization



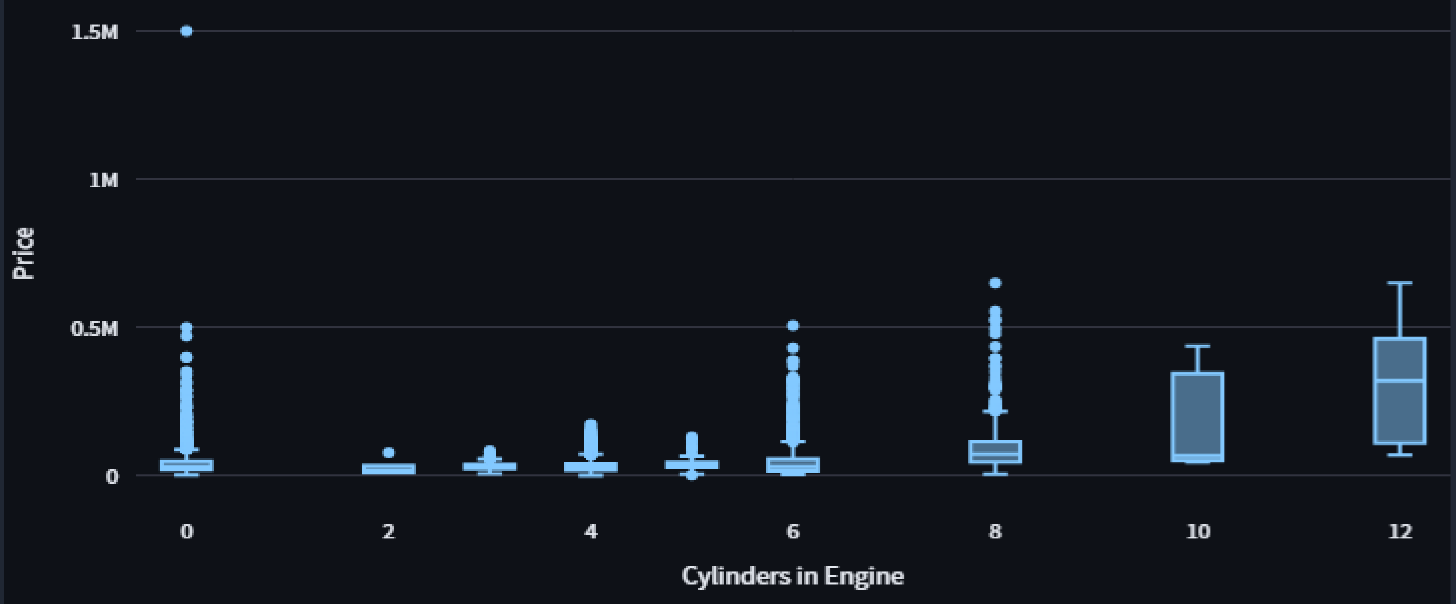


Made with Gamma

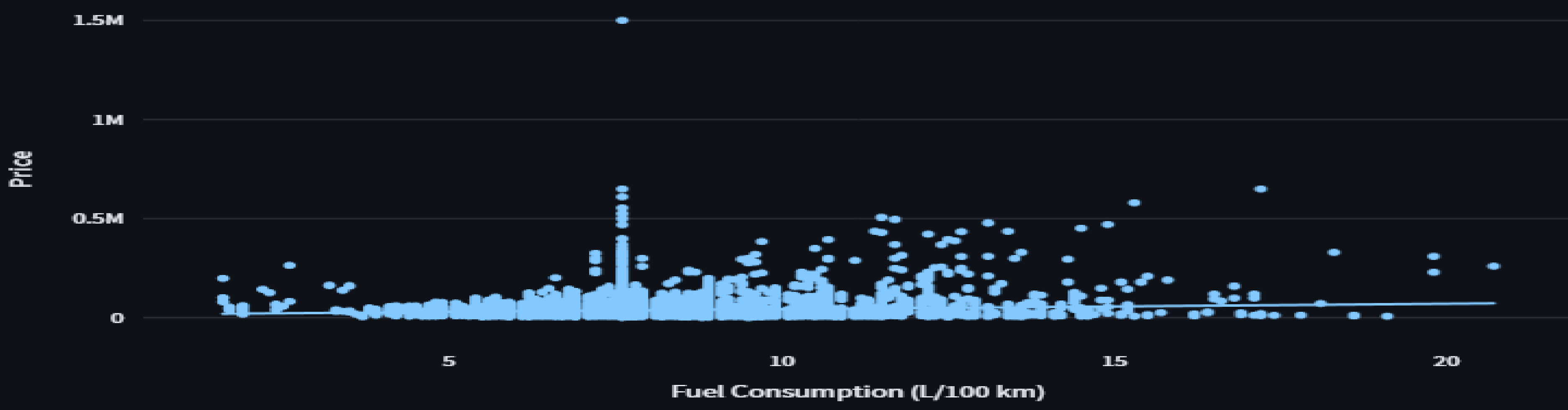
Correlation Heatmap



Price Distribution by Number of Cylinders



Price vs Fuel Consumption



Machine Learning Models

1

Data Preparation

We preprocessed the data, encoding categorical variables and scaling numerical features for optimal model performance.

2

Model Selection

We experimented with various algorithms, including linear regression, random forests, and gradient boosting machines.

3

Model Evaluation

We used cross-validation and metrics like RMSE and R-squared to assess and compare model performance.

4

Hyperparameter Tuning

We fine-tuned our models using grid search and random search techniques for optimal predictive power.

Performance of Regression Models using CV :

KNN:

```
y_pred_train = modelKNN.predict(X_train)
y_pred_val = modelKNN.predict(X_val)
print("Training Result (MSE): ", mean_squared_error(y_train, y_pred_train))
print("Training Result (R-squared): ", r2_score(y_train, y_pred_train))
print("Validation Result (MSE): ", mean_squared_error(y_val, y_pred_val))
print("Validation Result (R-squared): ", r2_score(y_val, y_pred_val))

print("Testing Result (MSE): ", mean_squared_error(y_test, modelKNN.predict(X_test)))
print("Testing Result (R-squared): ", r2_score(y_test, modelKNN.predict(X_test)))
```

```
Training Result (MSE):  146251765.68925118
Training Result (R-squared):  0.8912569294759123
Validation Result (MSE):  458139533.01734227
Validation Result (R-squared):  0.5976833948736279
Testing Result (MSE):  2455295978.461788
Testing Result (R-squared):  -0.5226559262238759
```

Logistic Regression:

```
[573] from sklearn.linear_model import LinearRegression

      linear_reg = LinearRegression()
      eval_model(linear_reg, X_train, y_train, X_val, y_val)
```

```
(0.336788507788831015, 0.339244004442375 46)
```

```
eval_model(linear_reg, X_train, y_train, X_test, y_test)
```

```
(0.33678850778831015, -616062588.1949852)
```

```
[575] import math
      math.sqrt(0.37154725901417607)
```

```
0.6095467652396952
```

Performance of Regression Models using CV :

Highest Accuracy:

- ExtraTreesRegressor: 0.7455340201432604
- RandomForestRegressor: 0.7306825125391688

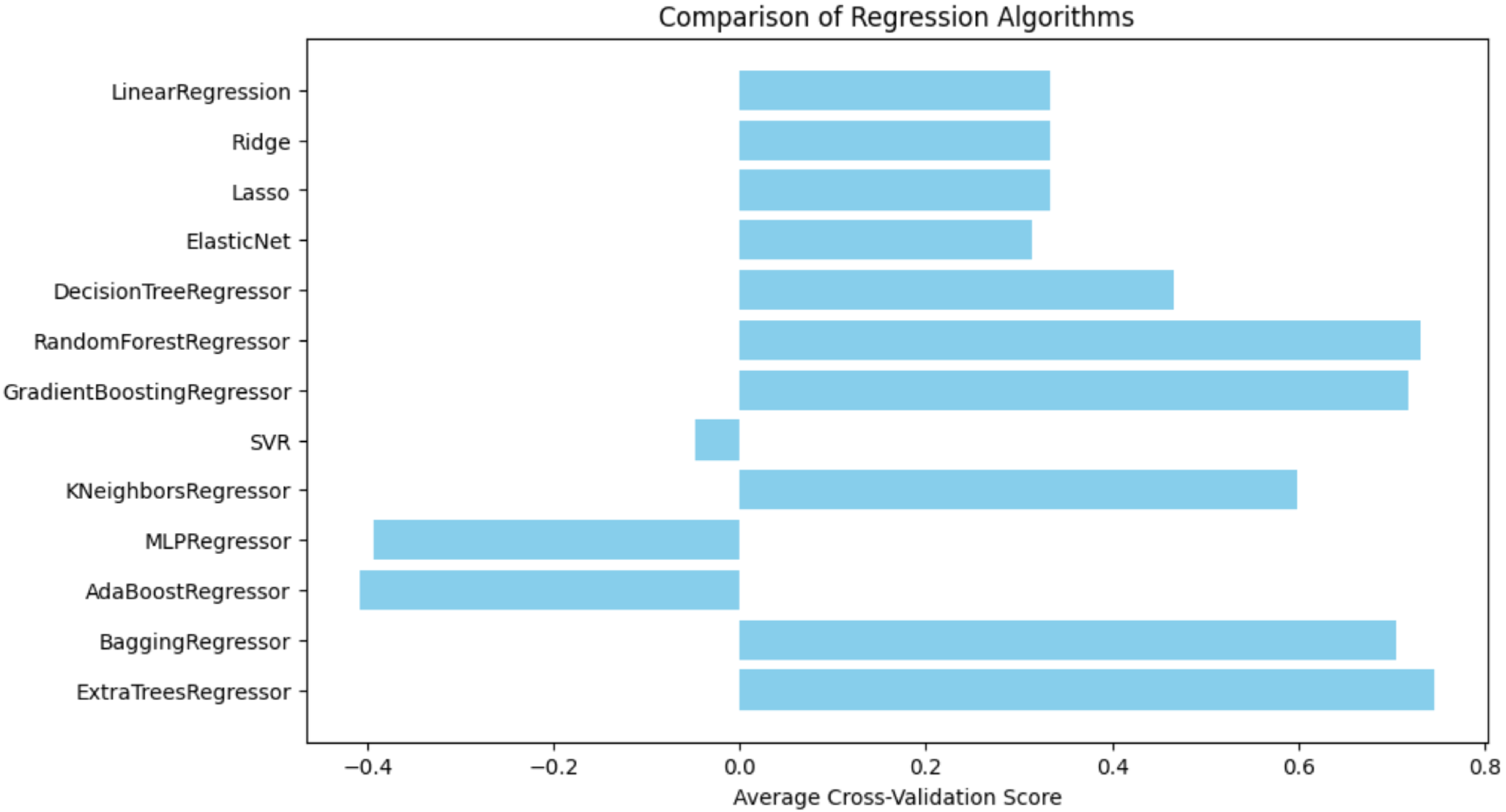
Moderate Accuracy:

- GradientBoostingRegressor : 0.718175473510159
- BaggingRegressor: 0.7052372089848618

Lowest Accuracy:

- MLPRegressor: - -0.393347030243334 23
- AdaBoostRegressor: -0.40771437504202274

```
LinearRegression ------------------------> [0.34805666 0.32438692 0.33019908] 0.3342142222626915
Ridge ------------------------> [0.34805598 0.32438604 0.33020514] 0.3342157200413485
Lasso ------------------------> [0.34805851 0.32438495 0.33020578] 0.334216409025752 9
ElasticNet ------------------------> [0.32399361 0.30144335 0.31529388] 0.31357694672630526
DecisionTreeRegressor ------------------------> [0.51715761 0.58422582 0.29859003] 0.4666578185408528
RandomForestRegressor ------------------------> [0.76146517 0.73660375 0.69397861] 0.7306825125391688
GradientBoostingRegressor ------------------------> [0.73001649 0.74862167 0.67588826] 0.718175473510159
SVR ------------------------> [-0.03428159 -0.05685935 -0.05358424] -0.0482417263291888 5
KNeighborsRegressor ------------------------> [0.60980794 0.61117948 0.57709809] 0.5993618348454192
MLPRegressor ------------------------> [-0.35448261 -0.40032447 -0.42731537] -0.394040817972081 2
AdaBoostRegressor ------------------------> [-0.89343489 -0.08859831 -0.24110993] -0.40771437504202274
BaggingRegressor ------------------------> [0.72161384 0.73487672 0.65922107] 0.7052372089848618
ExtraTreesRegressor ------------------------> [0.7691269  0.75007213 0.71740302] 0.7455340201432604
```


Comparison of Regression Algorithms
LinearRegression
Ridge
Lasso
ElasticNet
DecisionTreeRegressor
RandomForestRegressor
GradientBoostingRegressor
SVR
KNeighborsRegressor
MLPRegressor
AdaBoostRegressor
BaggingRegressor
ExtraTreesRegressor
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
Average Cross-Validation Score

Interactive Streamlit Dashboard

1. ## Home Page

   •We created an engaging landing page that introduces users to the project's scope and objectives.

2. ## Visualizations Page

   Users can explore interactive charts and graphs, uncovering insights about Australian car pricing trends.

3. ## Model Page

   We implemented a real-time prediction feature, allowing users to estimate car prices based on input parameters.

4. ## Prediction History

   Users can save and review their past predictions, enhancing the app's utility and user experience.

Deployment on Streamlit Cloud

Cloud Hosting

We leveraged Streamlit Cloud for seamless deployment, ensuring high availability and scalability for our app.

Security Measures

We implemented robust security protocols to protect user data and ensure safe interactions with our app.

Performance Optimization

We fine-tuned our app for optimal performance, minimizing load times and maximizing user engagement.

Fork

Navigation

Go to

- Home
- Visualizations
- Model

🚗 Australian Vehicle Prices

📊 Overview

A comprehensive dataset for exploring the car market in Australia.

ℹ About Dataset

Description: This dataset contains the latest information on car prices in Australia for the year 2023. It covers various brands, models, types, and features of cars sold in the Australian market. It provides useful insights into the

Navigation

Go to

- Home
- Visualizations
- Model

Model

Year: 2020

Fuel Consumption (L/100km): 8.00

Used or New: Used

Kilometres: 50000

Transmission: Manual

Cylinders in Engine: 4

Engine Size (L): 2.00

Body Type: Sedan

Drive Type: FWD

Number of Doors: 2

Fuel Type: Petrol

Predicted Price: $211,646.16

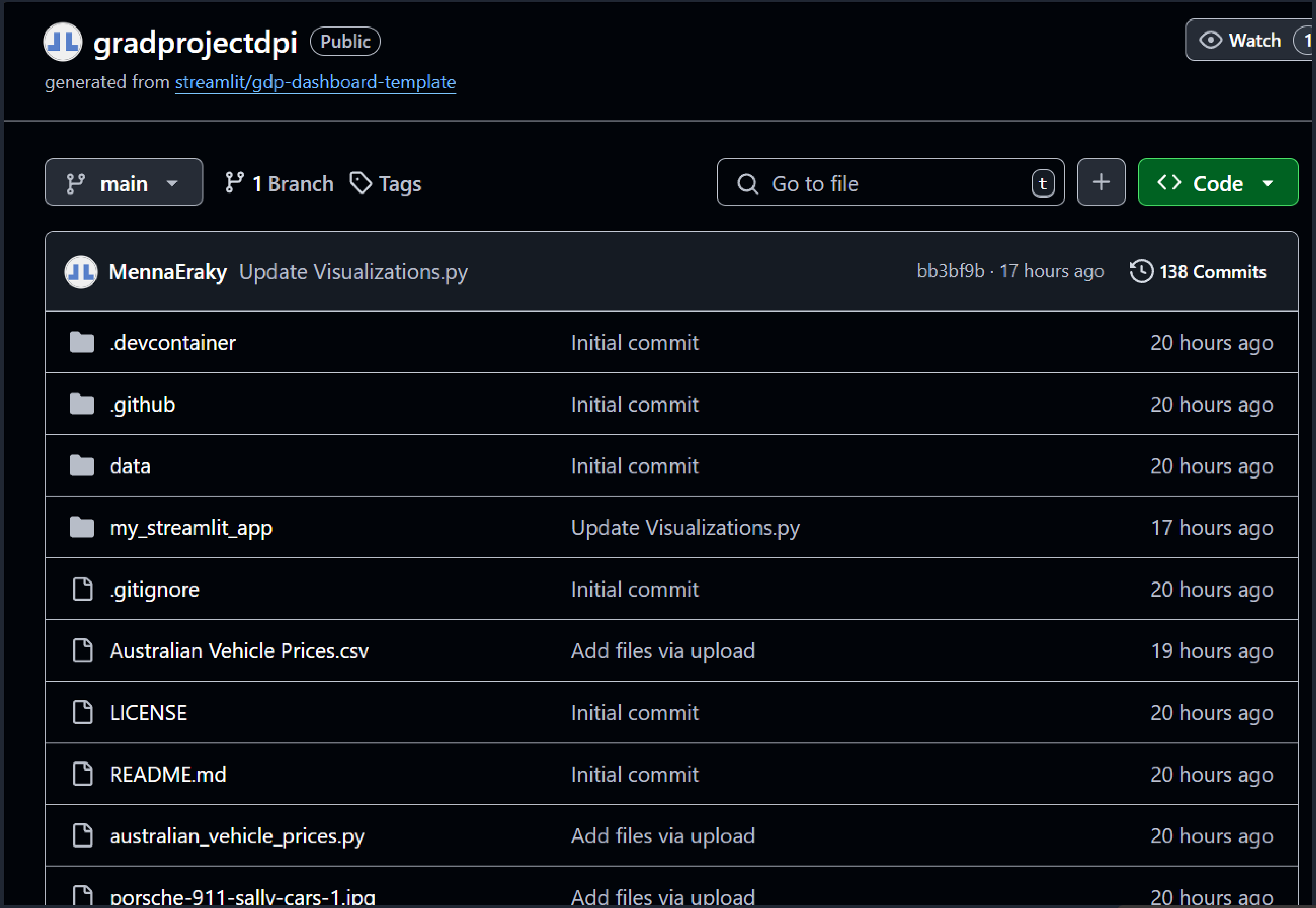
Input Data and Prediction

	Year	UsedOrNew	Transmission	Engine	DriveType	FuelType	FuelConsumption	Kilometres	CylindersinEngine	BodyType	Doors	Predicted Price
0	2,020	Used	Manual	2	FWD	Petrol	8	50,000	4	Sedan	2	$211,646.16

Model Performance Comparison

Best Model: RandomForestRegressor with Mean CrossVal Score: 0.72

GITHUB REPO


gradprojectdpi
Public
Watch
generated from streamlit/gdp-dashboard-template
main
1 Branch
Tags
Go to file
t
Code
MennaEraky Update Visualizations.py
bb3bf9b · 17 hours ago
138 Commits
.devcontainer
Initial commit
20 hours ago
.github
Initial commit
20 hours ago
data
Initial commit
20 hours ago
my_streamlit_app
Update Visualizations.py
17 hours ago
.gitignore
Initial commit
20 hours ago
Australian Vehicle Prices.csv
Add files via upload
19 hours ago
LICENSE
Initial commit
20 hours ago
README.md
Initial commit
20 hours ago
australian_vehicle_prices.py
Add files via upload
20 hours ago
porsche-911-sally-cars-1.jpg
Add files via upload
20 hours ago

Project Impact and Future Directions

Current Impact	Future Possibilities
Empowering consumers with data-driven car pricing insights	Integrating real-time market data for up-to-the-minute predictions
Assisting dealerships in competitive pricing strategies	Expanding the model to predict future market trends
Providing valuable market intelligence for manufacturers	Incorporating AI-driven personalized recommendations for car buyers