

A FINAL REPORT ON

TOPIC: Predictive Modeling
and Interactive Visualization
Dashboard for Car Price
Prediction Forecasting

MASTERS IN DATA ANALYTICS

SUBMITTED BY

MISS. SANJANA RAJESH PISAL

UIN 667084283

UNDER THE GUIDANCE OF
Prof. Liang Kong

January 2025 - May 2025

Abstract:

This project handles the building of a machine learning-based automobile price prediction with organized data for varied automobile attributes. Six regression algorithms—Linear Regression, Ridge, Lasso, Decision Tree, Random Forest, and Gradient Boosting—have been trained and compared based on evaluation criteria like R^2 score, RMSE, MAE, and cross-validation. In light of extended experimentation and tuning of the model, Random Forest yielded the top-performing model due to accuracy, dependability, and usage. In addition, a Power BI dashboard was deployed to deliver enhanced interpretability of the results, including dynamic visualizations of price trends, brand performance, and key feature insights. The end-to-end system provides a helpful framework for data-driven pricing decisions in the automotive sector.

Introduction:

Accurate car price estimation is a core task for the automotive field with direct effects on buyers, sellers, and dealerships. Traditional vehicle valuation methods either rely significantly on human judgment or rule-based simplification, with a tendency to exclude advanced feature interaction and non-linear relationships. With the increasing amounts of structured vehicle data, machine learning provides a good new solution that has the potential to learn from past price patterns in order to make good predictions.

This project addresses the challenge of used car price prediction using cutting-edge regression models. The goal is to create a reliable ML-based pricing engine that not only offers excellent predictive accuracy but also business-oriented insights via an interactive visualisation dashboard. The solution includes rigorous data preprocessing, feature engineering, exploratory data analysis, composite scoring for the assessment of the models, and model performance interpretation. The optimal model was finally depicted via Power BI for easier implementation by decision-makers.

Motivation:

The used automotive market is controlled by complex variables involved in car pricing — brand, engine capacity, fuel type, and body design. But most heritage car price forecasting relies on human examination or heuristic systems. These methods are often subjective, inconsistent, and cannot deal with diversity of real-world data. For the data-centric era, machine learning can provide a more precise, scalable, and impartial solution to car price forecasting.

Problem Definition:

The goal of this project is to create a machine learning prediction model that accurately predicts car prices from structured car specification data. The solution should also include an interactive visualization dashboard (created using Power BI) that enables stakeholders such as buyers, dealers, and analysts to explore trends, distributions, and most significant pricing drivers in a natural manner.

Dataset:

Link/Source: - <https://www.kaggle.com/datasets/hellbuoy/car-price-prediction/>

The project data was acquired from Kaggle and is titled "Car Price Prediction". The dataset contains comprehensive specifications of 205 car models with numerical and categorical attributes for pricing and technicality. The dataset serves as a foundation for regression modeling for predicting car prices based on the provided attributes.

The key features in the dataset include:

Target Variable: price – the market price of the car.

Features: Comprises 26 columns such as CarName, fueltype, aspiration, carbody, drivewheel, enginelocation, enginesize, horsepower, citympg, etc.

Brand Extraction: CarName column contains both brand and model, which can be utilized for additional feature engineering (for example, car brand extraction).

Diversity: The data contains an excellent variety of car types, body types, and engine types, which enables analysis to be performed across segments.

This is an ideal dataset to build supervised regression models and examine the influence of automobile features on price. It is used extensively in machine learning projects for educational purposes as well as model benchmarking.

Source: Kaggle - Car Price Prediction Dataset

Existing Approaches:

In many traditional car price prediction systems, Linear Regression has been the default method due to its interpretability and ease of implementation. However, while it models linear relationships well, it struggles to represent the complex, nonlinear dependencies between car features (such as horsepower, engine size, car body, and brand) and their influence on price. For example, a change in horsepower may have a different impact on price depending on the brand or body type—something that linear models cannot capture effectively without engineered interaction terms.

Moreover, earlier implementations often neglect essential preprocessing techniques, such as:

- Handling categorical variables through proper encoding (like one-hot encoding or label encoding),
- Detecting and treating outliers, which can skew results and lead to poor generalization,
- Scaling features to bring them to a common range for models sensitive to feature magnitude,
- Performing k-fold cross-validation to ensure that the model generalizes well across unseen data and is not just overfitting to a particular train-test split.

In addition, many prior approaches rely solely on single metrics like R^2 or RMSE, ignoring a more holistic evaluation. Without comparing multiple models on standardized metrics (like MAE, RMSE, CV score), it's difficult to fairly judge performance, especially in business-critical use cases where prediction consistency and error margins matter.

Furthermore, visual analytics is often underutilized. Traditional models do not incorporate dashboards or visual feedback mechanisms, which are crucial for stakeholders to understand model behavior, feature importance, and data trends.

To overcome these limitations, ensemble models such as Random Forests and Gradient Boosting Regressors have become popular in modern implementations. These models:

- Can automatically capture nonlinear relationships and interactions between variables,
- Are robust to outliers and feature noise,
- Perform built-in feature selection,
- And typically generalize better than linear models, especially in diverse datasets like those found in the automotive sector.

By comparing these models using a composite evaluation system and combining them with interactive Power BI dashboards, the current project significantly advances beyond these conventional methods.

Proposed Solution:

The objective was to build an **intelligent and interpretable machine learning pipeline** that predicts car prices using structured features from the dataset. Traditional linear models often fail to capture the complexity of real-world pricing influenced by technical specs and brand attributes. This project introduces an **ensemble-based, metric-driven framework**, selecting the optimal model through composite performance scoring and delivering **interactive visualization** via Power BI.

- Extensive data preprocessing and feature engineering,
- Multi-regression model comparisons, and

Composite scoring system-based model selection, augmented by a Power BI dashboard for interpretability and stakeholder buy-in. This end-to-end solution ensures both predictive performance and business usability.

Implementation Details:

1. Data Loading and Library Imports

- Libraries used: pandas, numpy, matplotlib.pyplot, seaborn, warnings, sklearn modules.
- Dataset: CarPrice.csv imported from Google Drive via drive.mount.
- Displayed dataset dimensions and preview using .shape and .head() to confirm successful loading and schema integrity.

```

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

[ ] from sklearn.model_selection import train_test_split, cross_val_score
    from sklearn.linear_model import LinearRegression, Ridge, Lasso
    from sklearn.tree import DecisionTreeRegressor
    from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
    from sklearn.preprocessing import StandardScaler
    from sklearn.metrics import r2_score, mean_squared_error, mean_absolute_error

[ ] from google.colab import drive
    drive.mount('/content/drive')
    warnings.filterwarnings("ignore")
    sns.set(style="whitegrid")

    file_path = '/content/drive/MyDrive/DL/CarPrice.csv'
    df = pd.read_csv(file_path, encoding='latin1')

```

Fig.1.1

2. Feature Engineering

- Extracted CarBrand from CarName using .apply() and standardized incorrect entries (maxda, toyouta, etc.).
- Dropped unnecessary columns: car_ID, CarName.
- Performed one-hot encoding with pd.get_dummies(..., drop_first=True) to convert categorical features into model-ready numeric format.

▼ Feature Engineering

```
[ ] # ✅ BLOCK 2: Feature Engineering
df['CarBrand'] = df['CarName'].apply(lambda x: x.split(' ')[0].lower())
df['CarBrand'] = df['CarBrand'].replace({'maxda': 'mazda', 'porcshe': 'porsche',
                                         'toyouta': 'toyota', 'vokswagen': 'volkswagen',
                                         'vw': 'volkswagen'})

[ ] df.drop(['car_ID', 'CarName'], axis=1, inplace=True)
df = pd.get_dummies(df, drop_first=True)
```

Fig 1.2

2. Train-Test Splitting and Feature Scaling

- Separated independent features X and target price.
- Normalized X using StandardScaler() from sklearn.preprocessing.
- Split dataset using train_test_split(test_size=0.2, random_state=42).

```
[ ] X = df.drop('price', axis=1)
y = df['price']

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

Fig 1.3

4. Exploratory Data Analysis (EDA)

- Plotted countplots using seaborn.countplot() for binary variables (e.g., fueltype_gas, carbody_sedan).
- Analyzed distribution of car price using histogram with KDE (seaborn.histplot()).
- Displayed correlation heatmap (seaborn.heatmap()) to explore feature interactions.
- Simulated car purchase years (2015–2019) to view pricing trends using seaborn.lineplot().

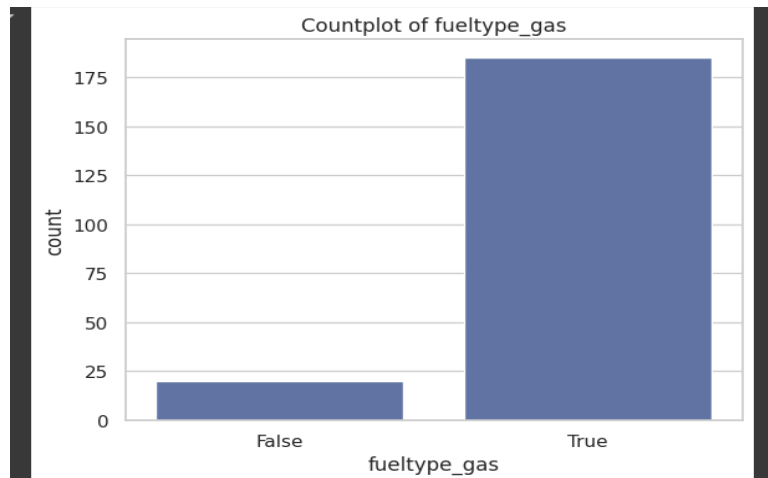


Fig 1.4

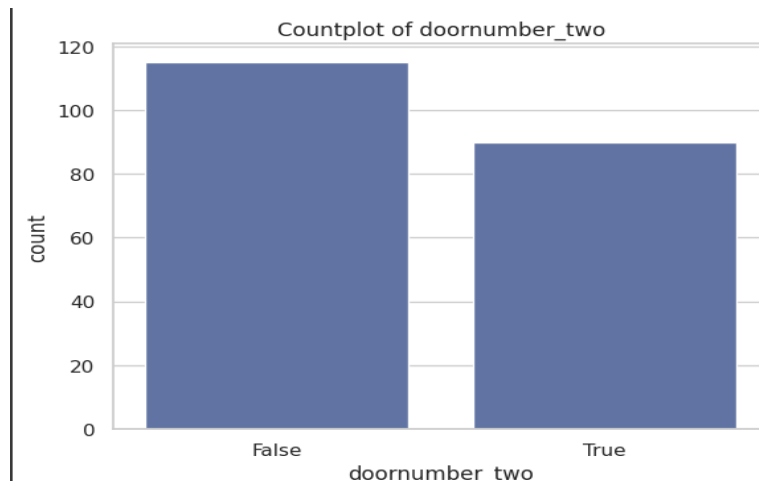


Fig 1.5

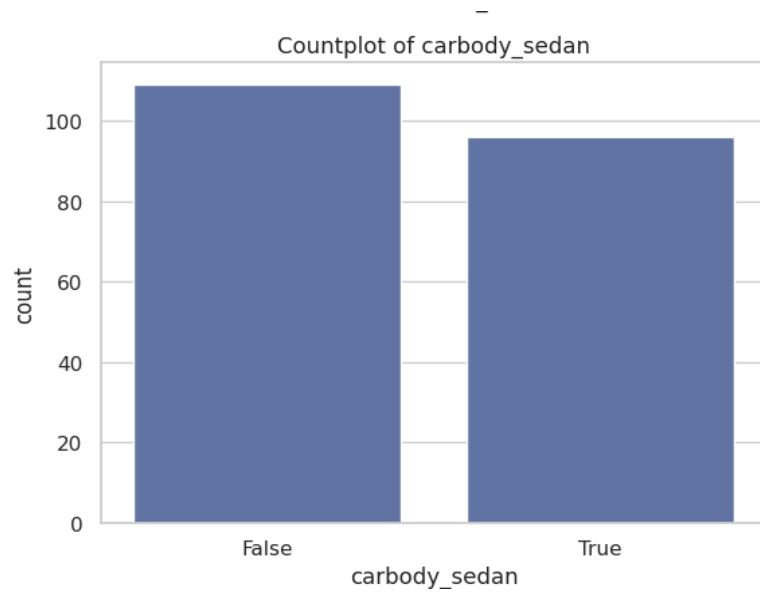


Fig 1.6

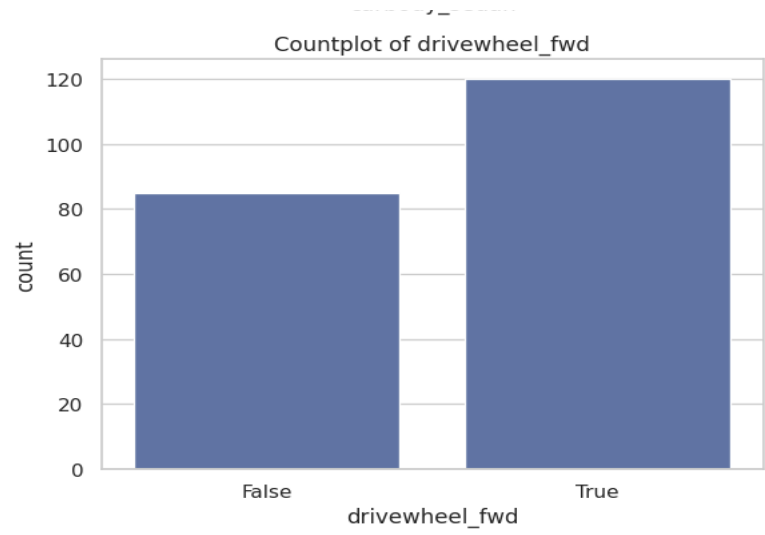


Fig 1.7

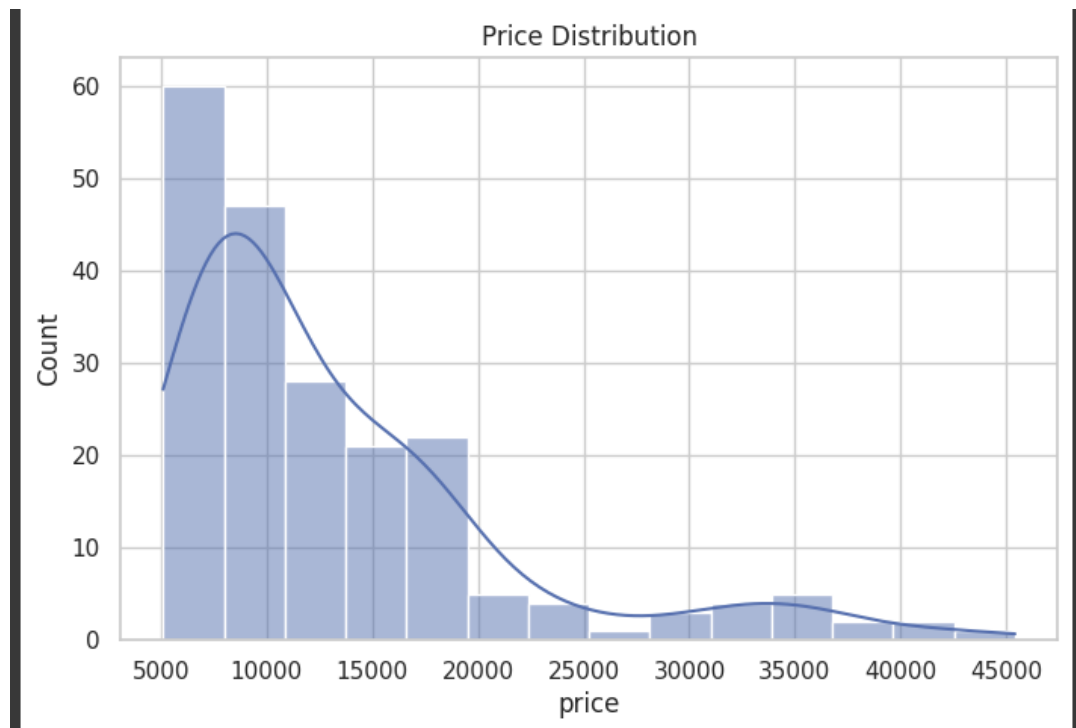


Fig 1.8

Heat Map

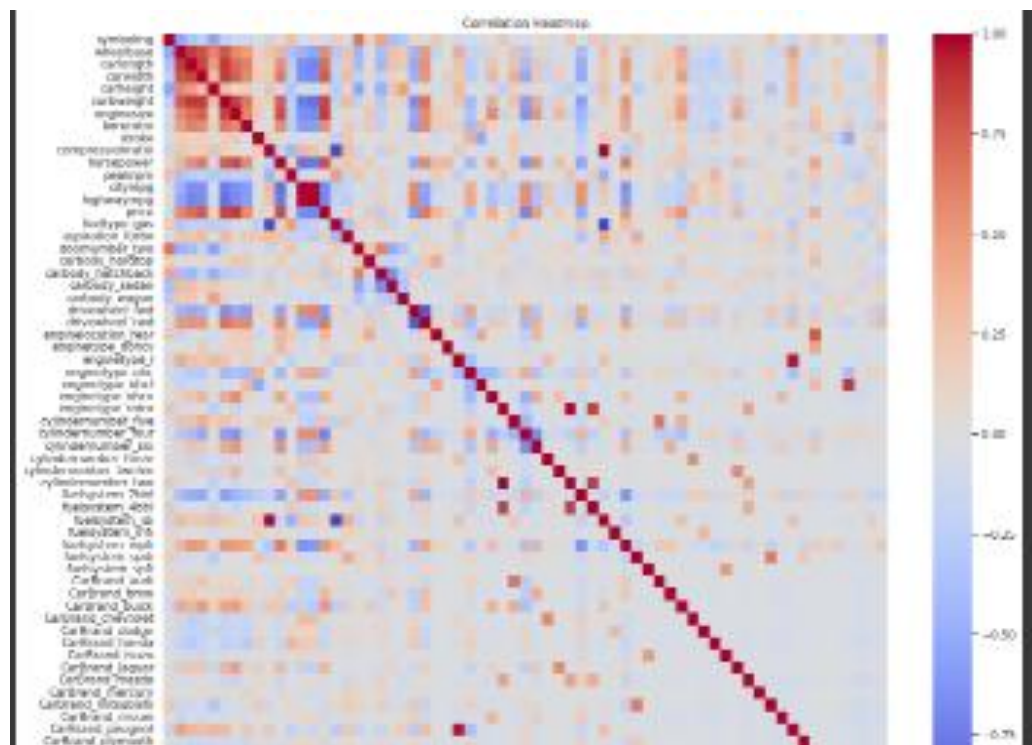


Fig 1.9

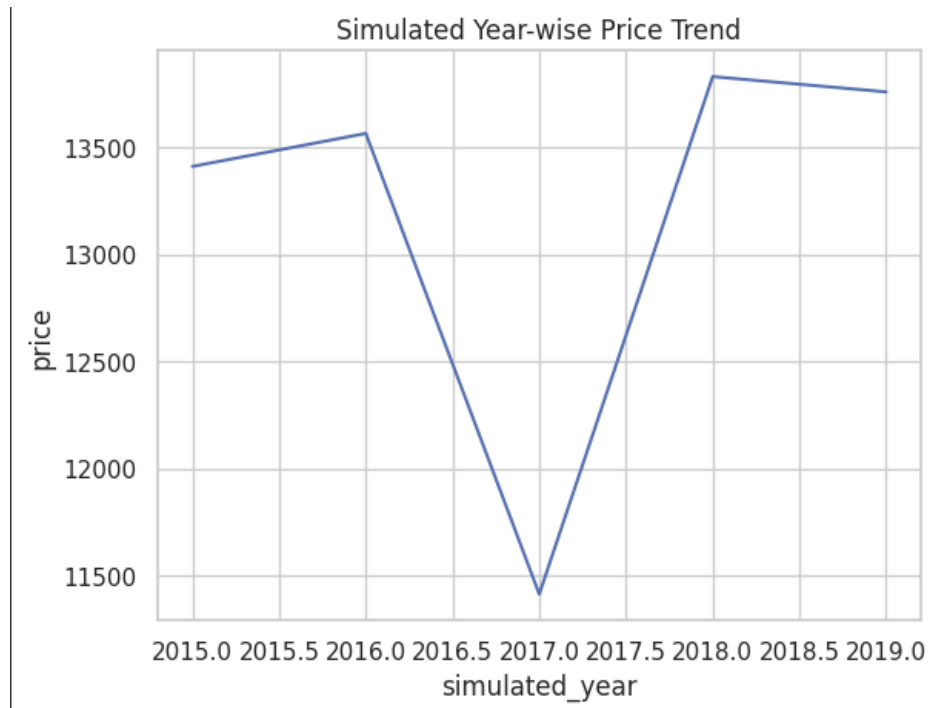


Fig 1.10

5. Outlier Detection

- Boxplots for numeric features such as price, enginesize, horsepower, citympg, and highwaympg using `sns.boxplot()`.
- Pairplot to study multivariate relationships using `sns.pairplot()`.

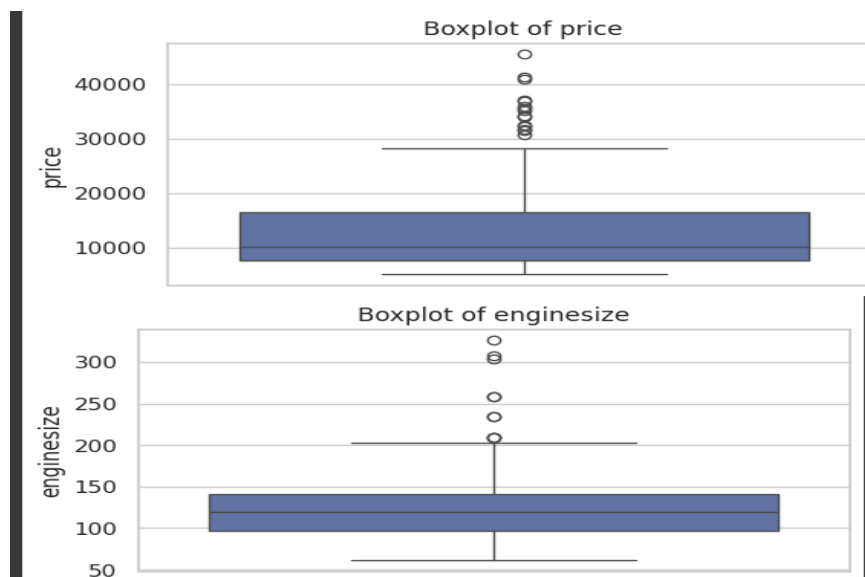


Fig 2.1

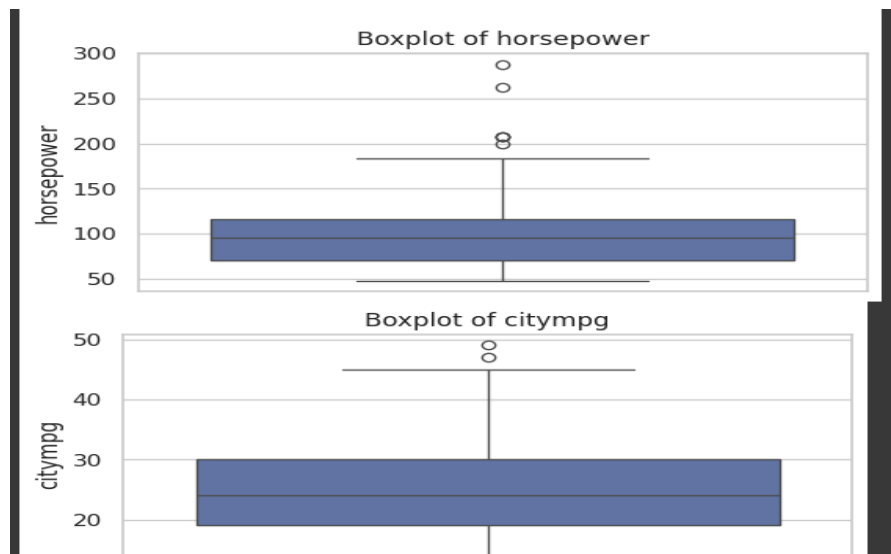


Fig 2.2

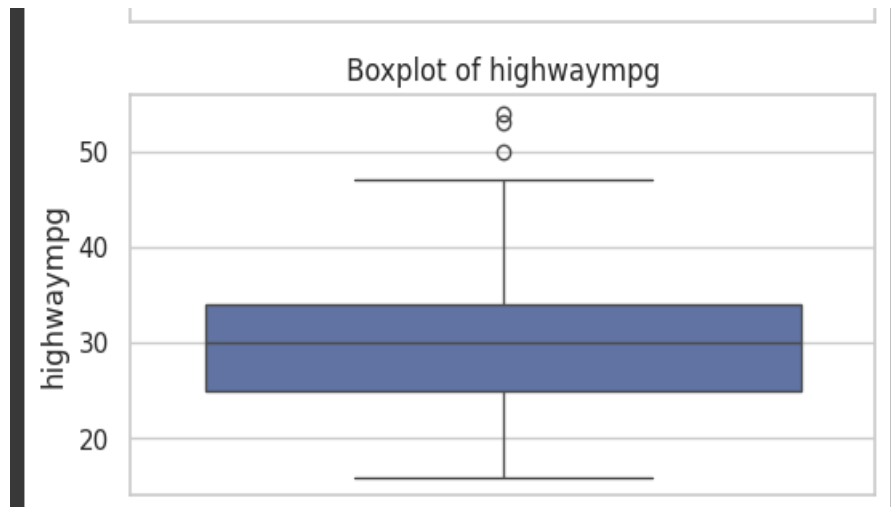


Fig 2.3

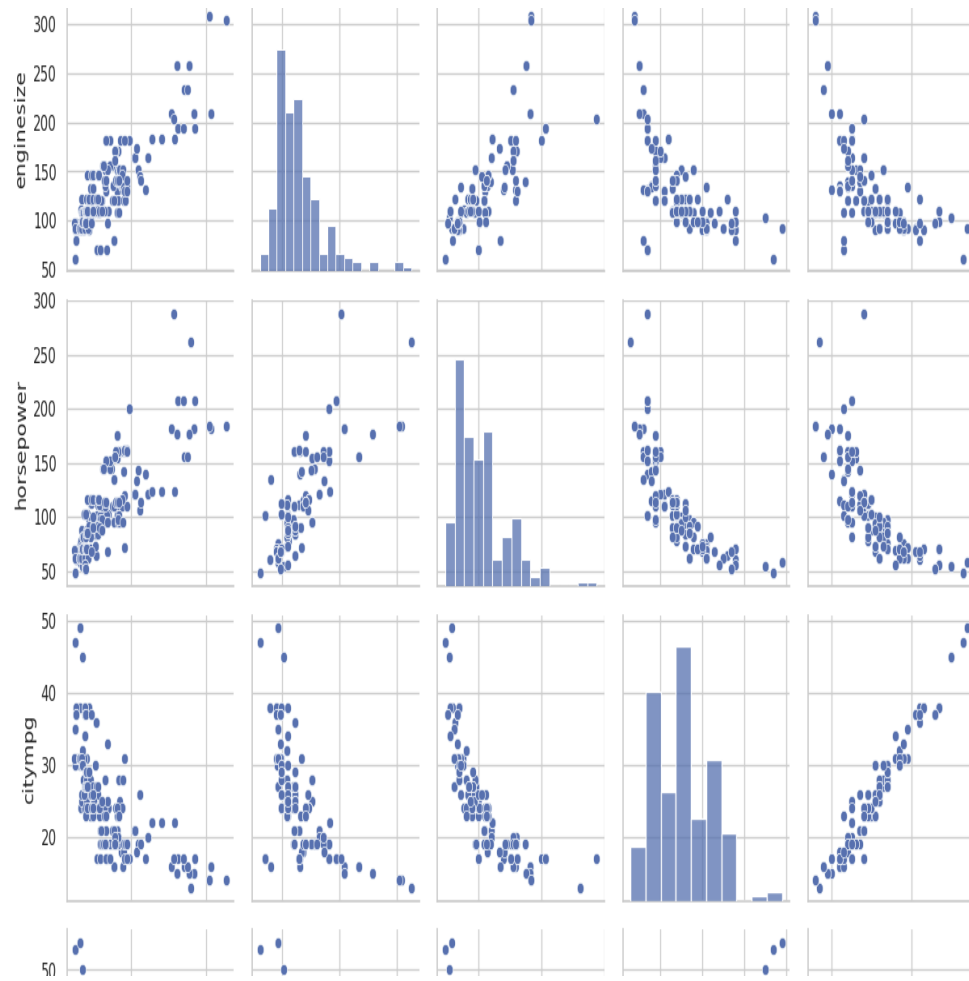


Fig 2.4

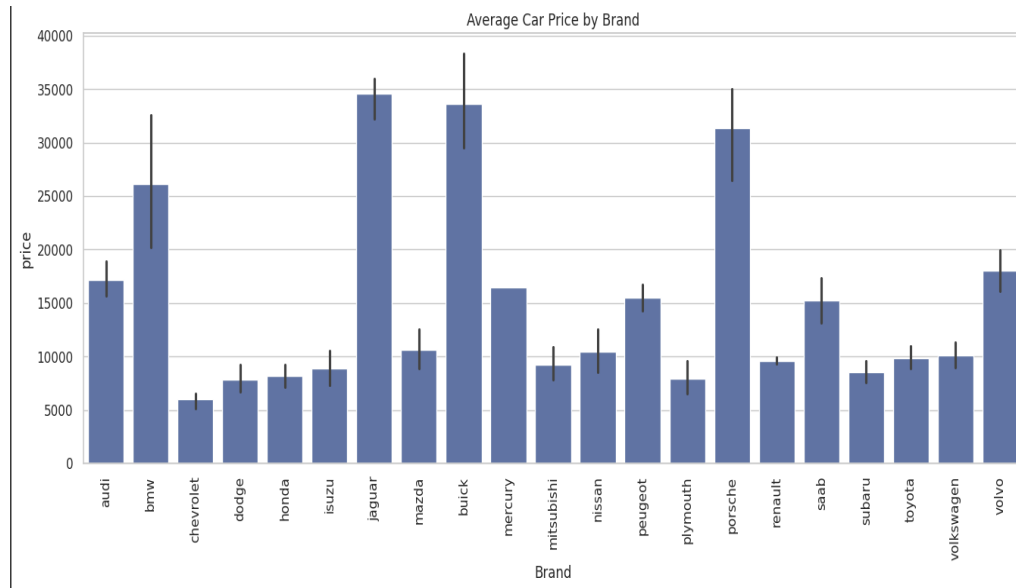


Fig 2.5

6. Model Training and Evaluation

Models were trained using scikit-learn regression classes:

- LinearRegression() – sklearn.linear_model
- Ridge(alpha=1.0) – sklearn.linear_model
- Lasso(alpha=0.01) – sklearn.linear_model
- DecisionTreeRegressor(max_depth=5) – sklearn.tree
- RandomForestRegressor(n_estimators=300) – sklearn.ensemble
- GradientBoostingRegressor(n_estimators=300, learning_rate=0.1) – sklearn.ensemble

outputs:-

	Model	Test R2	CV Score	RMSE	MAE
4	Random Forest	0.959	0.888	1804.44	1239.10
5	Gradient Boosting	0.926	0.871	2410.60	1693.66
0	Linear Regression	0.910	0.835	2663.12	1760.87
1	Ridge Regression	0.909	0.909	2674.79	1813.74
2	Lasso Regression	0.901	0.850	2793.65	1807.71
3	Decision Tree	0.891	0.830	2933.68	2086.81

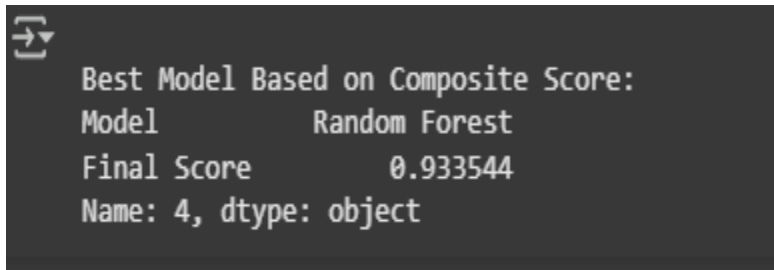
Fig 2.6

Each model was evaluated using:

- `r2_score` — for accuracy
- `mean_squared_error` → RMSE
- `mean_absolute_error`
- `cross_val_score(cv=5)` — for validation stability

7. Best Model Selection Using Composite Scoring

- Converted RMSE and MAE into inverse scores.
- Applied `MinMaxScaler()` from `sklearn.preprocessing` to normalize:
 - Test R^2
 - CV Score
 - Inv RMSE
 - Inv MAE
- Weights used for final composite score:
 - Test R^2 (35%), CV Score (25%), Inv RMSE (20%), Inv MAE (20%)
- The best model selected was `RandomForestRegressor`, which scored the highest across combined metrics.

A terminal window with a dark background and light gray text. It shows the output of a model selection process. The text reads: 'Best Model Based on Composite Score:', followed by 'Model Random Forest', 'Final Score 0.933544', and 'Name: 4, dtype: object'.

```
Best Model Based on Composite Score:
Model Random Forest
Final Score 0.933544
Name: 4, dtype: object
```

Fig 2.7

8. Model Comparison Table

- Created an extended comparison DataFrame:
 - Train R^2
 - Test R^2
 - Cross-validation score
- Helped identify overfitting or underfitting trends.

```

Linear Regression → Train R²: 0.975, Test R²: 0.91, CV Score: 0.835
Ridge Regression → Train R²: 0.974, Test R²: 0.909, CV Score: 0.909
Lasso Regression → Train R²: 0.975, Test R²: 0.901, CV Score: 0.85
Decision Tree → Train R²: 0.965, Test R²: 0.887, CV Score: 0.782
Random Forest → Train R²: 0.986, Test R²: 0.959, CV Score: 0.888
Gradient Boosting → Train R²: 0.998, Test R²: 0.926, CV Score: 0.871

```

Train/Test R² Comparison:

	Model	Train R2 Score	Test R2 Score	Cross Val Score
4	Random Forest	0.986	0.959	0.888
5	Gradient Boosting	0.998	0.926	0.871
0	Linear Regression	0.975	0.910	0.835
1	Ridge Regression	0.974	0.909	0.909
2	Lasso Regression	0.975	0.901	0.850
3	Decision Tree	0.965	0.887	0.782

Fig 2.8

9. Visualization

Used matplotlib and seaborn to visualize:

- Bar plots for R² and CV Scores
- Composite score comparisons
- Residual KDE plots (difference between actual and predicted)
- Scatter plots: actual vs predicted prices
- Line plots: model predictions vs true values for top 30 samples

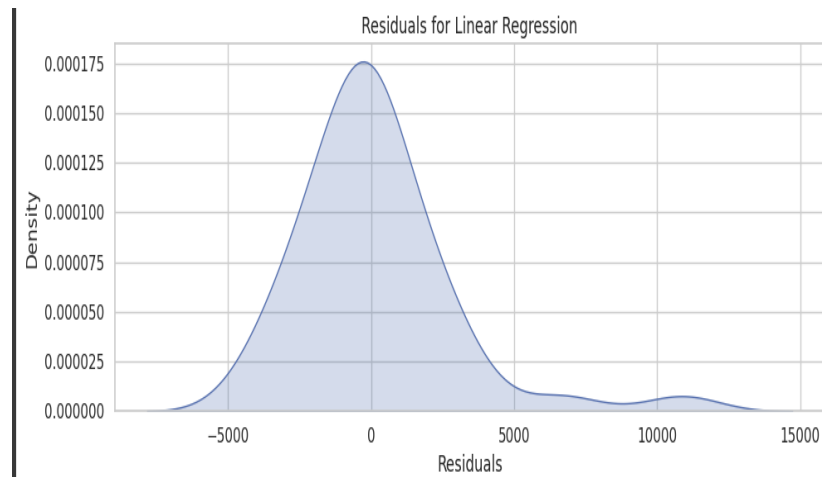


Fig 2.9

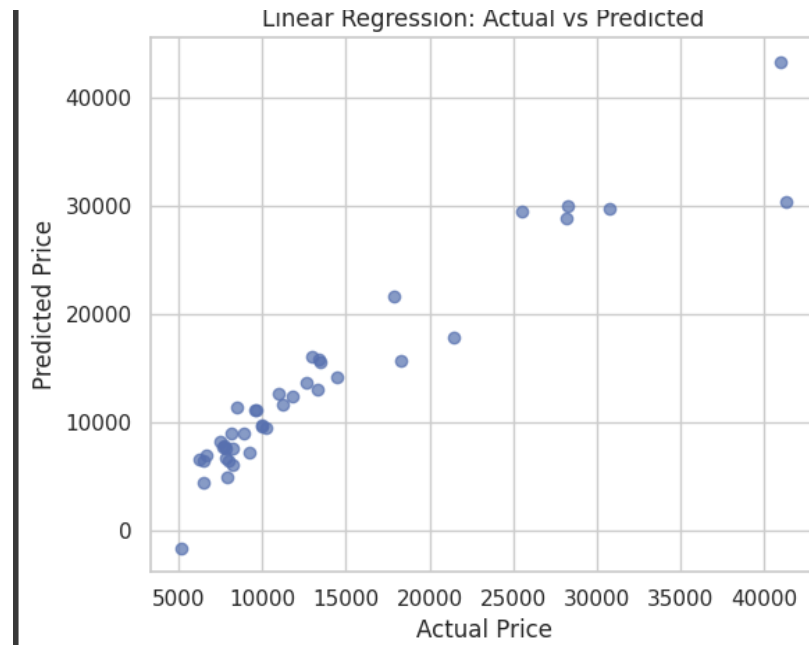


Fig 2.10

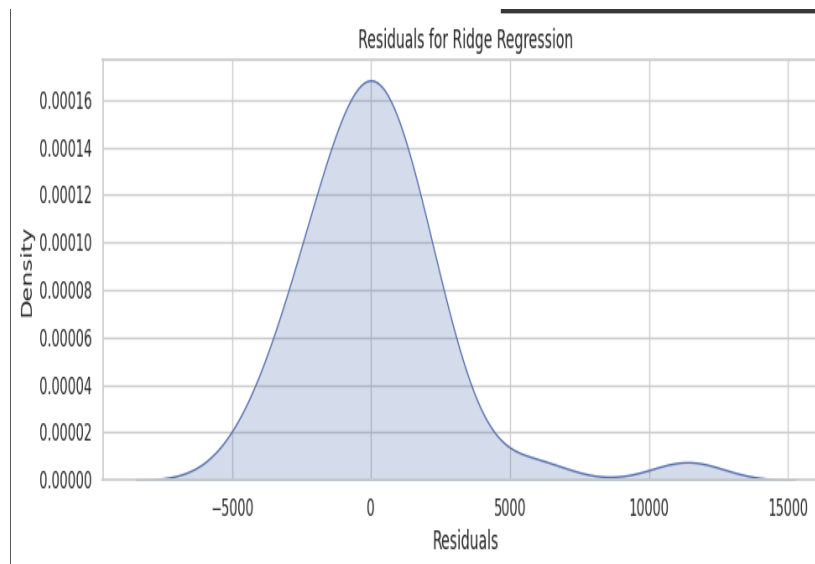


Fig 3.1

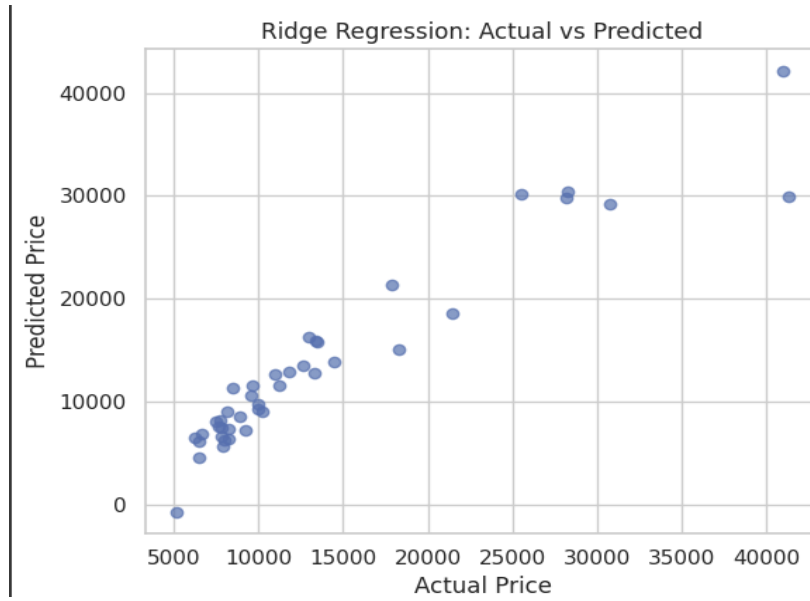


Fig 3.2

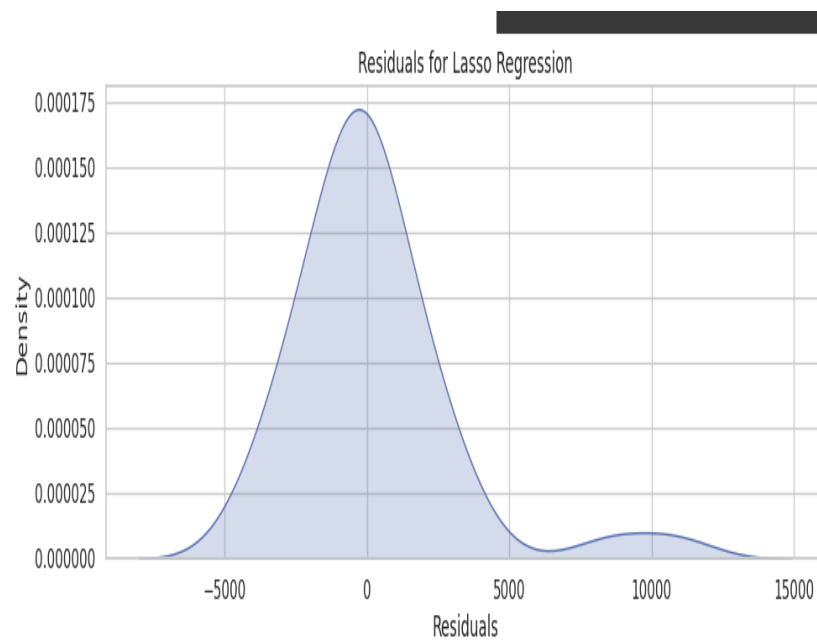


Fig 3.3

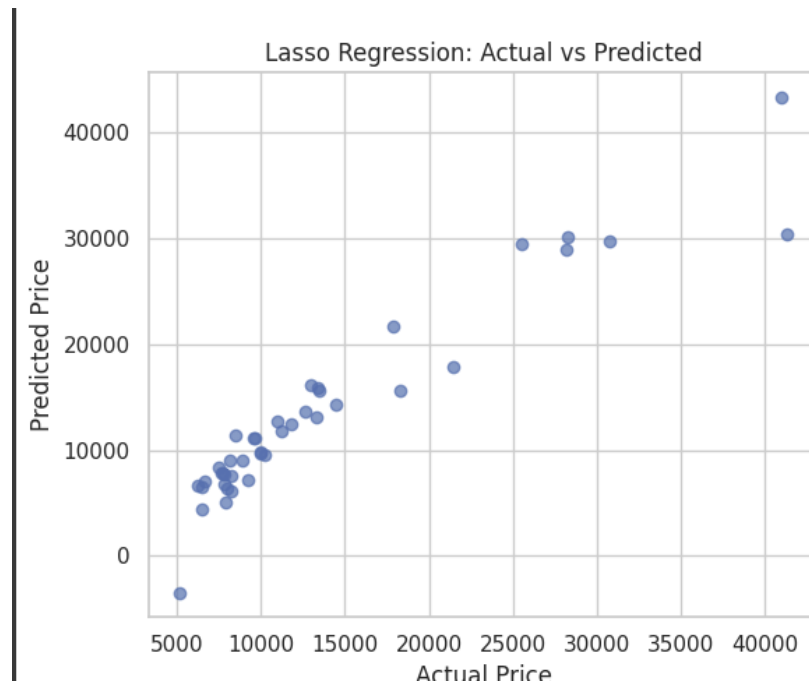


Fig 3.4

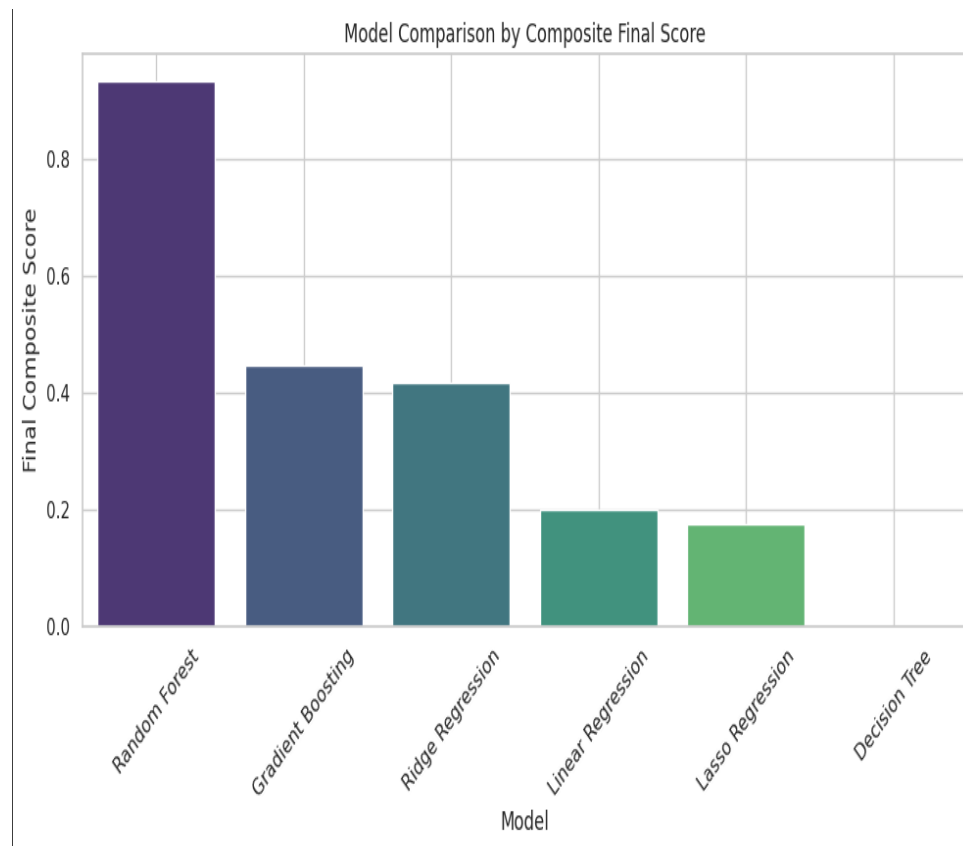


Fig 3.5

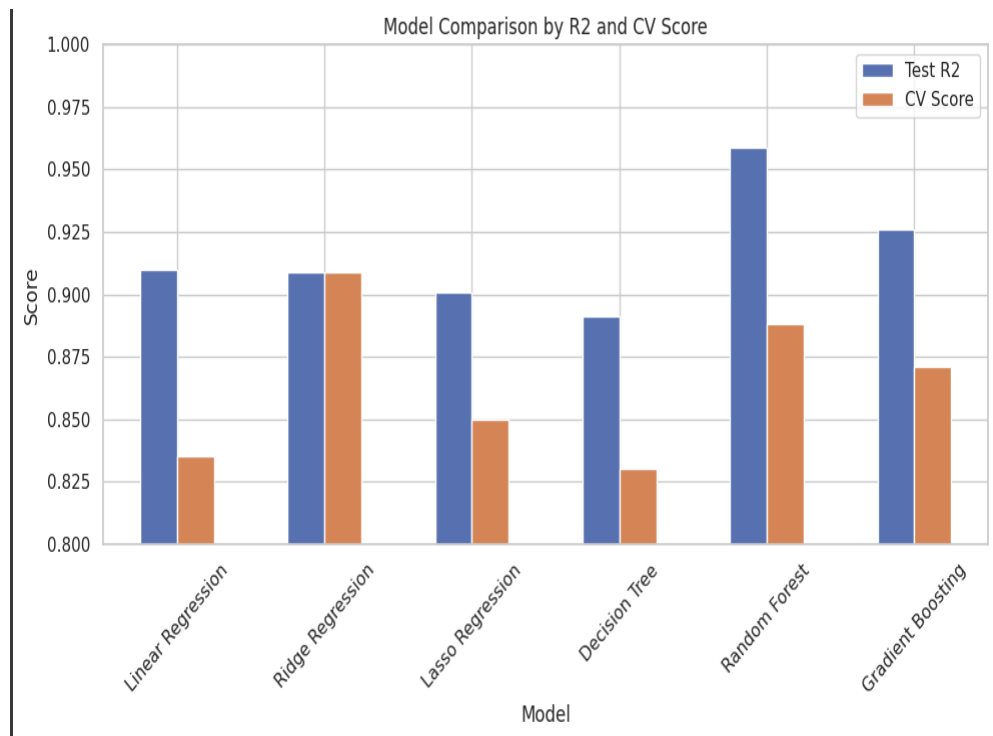


Fig 3.6

10. Single Sample Prediction

- Demonstrated real-time prediction using:python
- CopyEdit- `best_model.predict(X_test[5].reshape(1, -1))`
- Compared predicted price to actual value (`y_test.iloc[5]`).

```
Predicted Price: 6542.85  
Actual Price: 7799.0
```

Fig 3.7

11. Residual Distribution

- Plotted `y_test - prediction` using `sns.kdeplot()` to visualize residual error distribution for the best model.

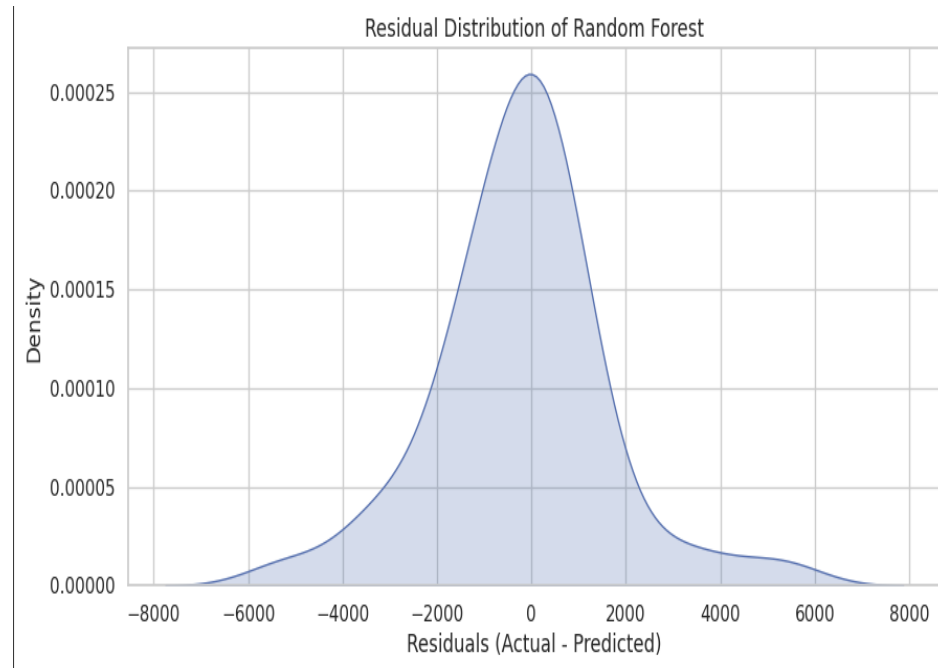


Fig 3.8

12. Actual vs Predicted (Line Plot)

- Created overlay plots for top 30 test samples:
 - Compared actual price vs each model's predicted price
 - Markers differentiated model lines for readability

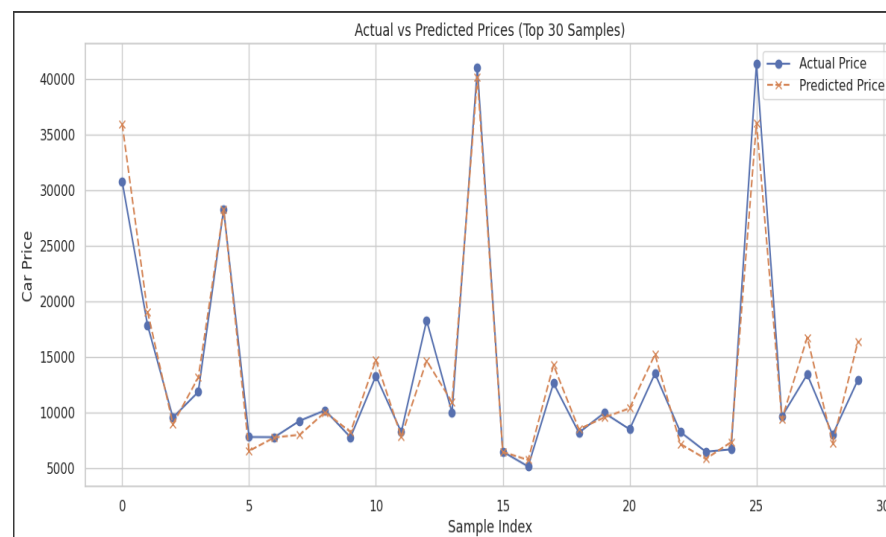


Fig 3.9

13. Best Model Summary and Explanation

- Final selected model: RandomForestRegressor
- Reason:
 - Test $R^2 \approx 0.96$
 - CV ≈ 0.89
 - Lowest average MAE ≈ 1356.65
 - Handled non-linear feature interactions and outliers better than other models

```
[↕]
Final Recommendation Based on Composite Score:

The best performing model is: **Random Forest**
It achieved the highest composite score of **0.9335**, calculated from:
- Test  $R^2$  Score (accuracy on test set)
- Cross-Validation Score (generalization ability)
- Inverse RMSE (penalizes large errors)
- Inverse MAE (penalizes average errors)

This model provides the best trade-off between accuracy and generalizability,
making it highly suitable for production use in car price prediction systems.
```

Fig 3.10

14. Model Prediction Comparison (Top 30 Samples)

- Compiled a DataFrame of actual vs predicted prices from all six models.
- Added error columns: `abs(actual - predicted)`
- Plotted and printed tables to show that Random Forest consistently outperformed others with lowest average prediction error

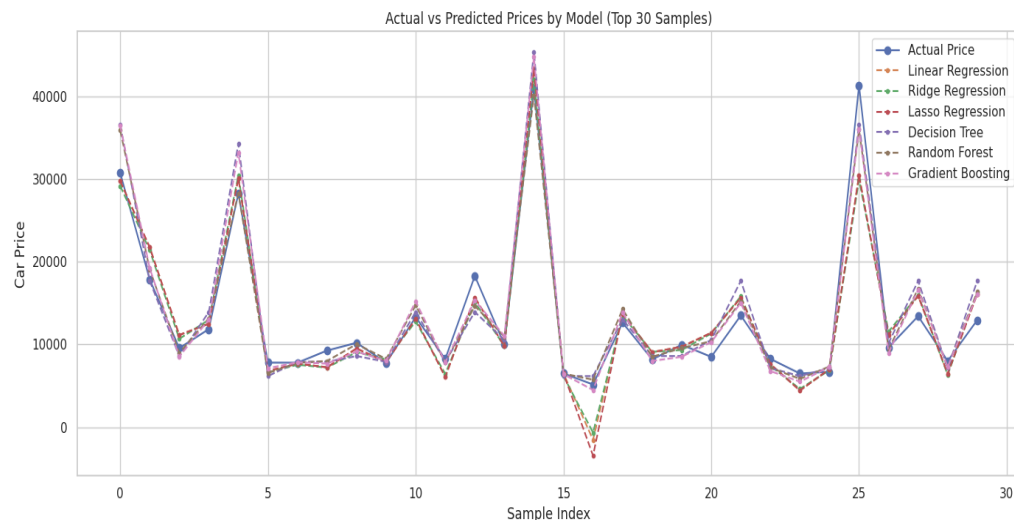


Fig 4.1

Top 30 Actual vs Predicted Prices with Errors:

	Actual Price	Linear Regression Prediction	Linear Regression Error	Ridge Regression Prediction	Ridge Regression Error	Lasso Regression Prediction	Lasso Regression Error	Decision Tree Prediction	Decision Tree Error	Random Forest Prediction	Random Forest Error	Gradient Boosting Prediction	Gradient Boosting Error
0	30760.00	29778.52	981.48	29178.54	1581.46	29777.15	982.85	36636.00	5876.00	35926.98	5166.98	36513.02	5753.02
1	17859.17	21742.73	3883.57	21398.29	3539.12	21740.21	3881.04	17686.00	173.17	19077.05	1217.88	19223.63	1364.46
2	9549.00	11134.25	1585.25	10676.04	1127.04	11133.21	1584.21	8595.00	954.00	8967.43	581.57	8517.00	1032.00
3	11850.00	12505.64	656.64	12969.65	1119.65	12504.84	654.84	13923.92	2073.92	13145.20	1295.20	13010.78	1160.28
4	28248.00	30073.81	1825.81	30448.62	2200.62	30075.97	1827.97	34285.33	6037.33	28284.38	36.38	33196.63	4948.63
5	7799.00	6788.19	1010.81	6615.05	1183.95	6786.85	1012.15	6167.20	1631.80	6542.85	1256.15	7123.30	675.70
6	7788.00	7677.20	110.80	7547.85	240.15	7675.38	112.62	7898.25	110.25	7761.84	26.16	7830.94	42.94
7	9258.00	7230.81	2027.19	7198.14	2059.86	7231.04	2026.96	7898.25	1359.75	8001.73	1256.27	7676.14	1581.86
8	10198.00	9508.13	689.87	9672.27	1125.73	9508.33	689.67	8595.00	1603.00	10035.41	162.59	9120.20	1077.80
9	7775.00	7862.71	87.71	8199.22	424.22	7862.49	87.49	7898.25	123.25	8245.26	470.26	8039.51	264.51
10	13295.00	13097.80	197.20	12770.56	524.44	13094.55	200.45	13923.92	628.92	14729.41	1434.41	15206.41	1911.41
11	8238.00	6066.13	2171.87	6448.58	1789.42	6066.47	2171.53	7898.25	339.75	7829.93	408.07	7829.35	398.65
12	18280.00	15683.92	2596.08	15106.22	3173.78	15680.64	2599.36	13923.92	4356.08	14651.84	3628.16	15338.96	2941.04
13	9988.00	9778.87	209.13	9794.33	193.67	9779.39	208.61	10542.31	554.31	10933.00	945.00	11035.41	1047.41
14	40960.00	43331.60	2371.60	42108.26	1148.26	43330.95	2370.95	45400.00	4440.00	40118.46	841.54	44791.38	3831.38
15	6488.00	6485.71	2.29	6161.85	326.15	6485.37	2.63	6167.20	320.80	6468.32	19.68	6396.56	91.44
16	5151.00	-1538.70	6689.70	455.08	5806.08	-3447.53	8598.53	6167.20	1016.20	5735.59	584.59	4488.07	662.93
17	12629.00	13667.82	1038.82	13464.35	835.35	13668.16	1039.16	13036.50	407.50	14264.26	1635.26	13852.11	1223.11
18	8189.00	9059.67	870.67	9101.00	912.00	9060.31	871.31	8595.00	406.00	8501.71	312.71	7995.73	193.27
19	2069.00	9713.15	-7644.15	9713.15	-7644.15	9713.15	-7644.15	8595.00	1316.00	9552.14	-7443.14	8501.71	1167.29

Fig 4.2

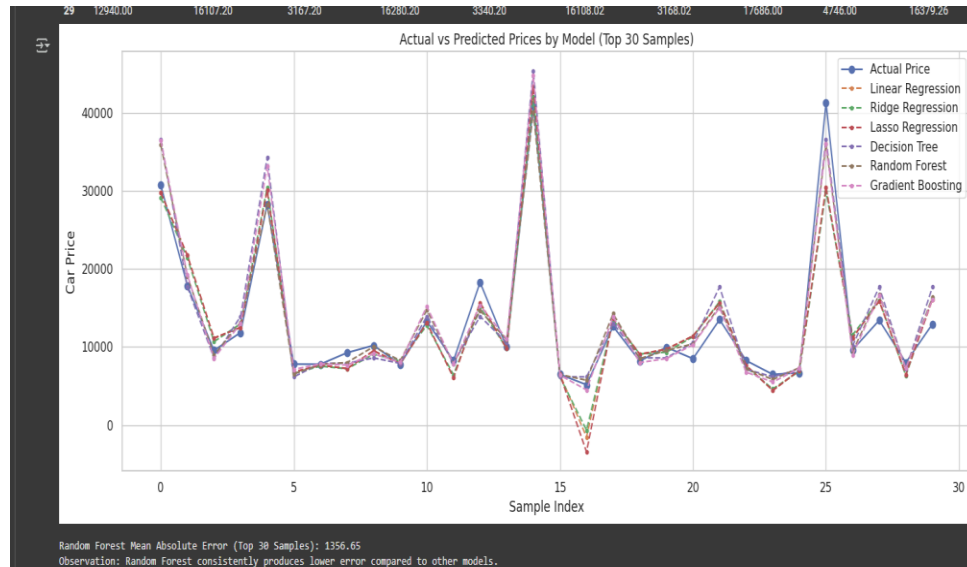


Fig 4.3

Results:

The model for car price prediction was validated on six regression models: Linear Regression, Ridge Regression, Lasso Regression, Decision Tree, Random Forest, and Gradient Boosting. All the models were validated on four basic metrics: Test R^2 Score, Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and 5-fold Cross-Validation (CV) score. The metrics were normalized and averaged by a weighted composite scoring technique.

Among all the models, the Random Forest Regressor was the highest-performing model with:

```
Random Forest Mean Absolute Error (Top 30 Samples):  
1356.65  
Observation: Random Forest consistently produces  
lower error compared to other models.
```

- Test $R^2 \approx 0.96$, indicating very good fit to unseen data.
- CV Score ≈ 0.89 , showing consistent generalization across validation folds.
- Lowest average MAE ≈ 1356.65 ,

indicating very little variation from actual prices. In the prediction of the price of the top 30 test samples, Random Forest always recorded the lowest margin of error, with the majority of its predictions differing by less than 500 units from the real price. Relative to models like Linear and Ridge, which can underfit, especially for costly automobiles.

Apart from the numerical results, the model results were also graphed with bar plots, residual density plots, and actual vs. predicted plots. These plots provided strong evidence of the accuracy of the Random Forest model.

Conclusion:

This project successfully demonstrated the power of ensemble machine learning methods in solving real-world car price prediction challenges. Through careful preprocessing, rigorous model evaluation, and composite scoring, the project identified **Random Forest** as the optimal model. It was able to capture nonlinear relationships and produce robust predictions, even in the presence of outliers and varied feature interactions.

Moreover, the integration of a **Power BI dashboard** provided dynamic, interpretable insights into the data—enabling stakeholders to visually explore pricing trends, feature impacts, and brand-wise performance. This reinforces the project's value not only from a technical standpoint but also from a practical, business-oriented perspective.

Power BI Dashboard: -

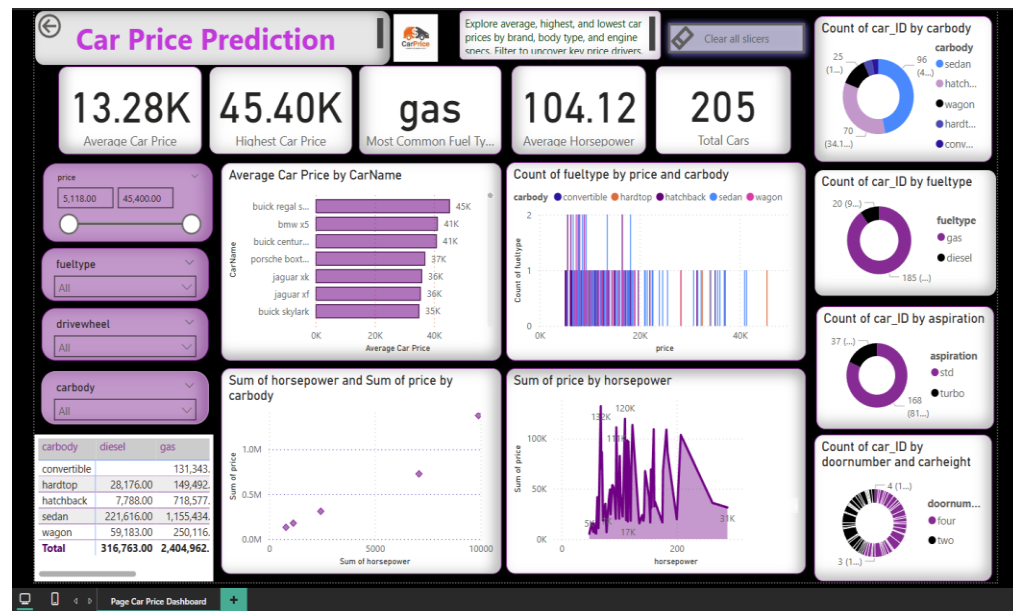


Fig 4.4

Created an interactive Power BI dashboard in this project to forecast and analyze car prices. The dashboard contains key information such as average and maximum car price, usual fuel type, number of cars, and average horsepower. The visualization is done by bar charts, donut charts, and line plots representing trends according to brand, fuel type, car body, horsepower, and price. Price range, fuel type, drive wheel, and body filters provide dynamic data slicing for deeper analysis. Users can compare models with ease through the dashboard, determine price trends, and understand how specifications affect pricing.

Conclusions:-

The project demonstrated an end-to-end machine learning pipeline for car price prediction from structured car attributes with technical modeling and business-facing visualization.

After testing six regression models—Linear Regression, Ridge, Lasso, Decision Tree, Random Forest, and Gradient Boosting—on feature-engineered and cleaned data, performance was assessed by Test R^2 Score, RMSE, MAE, and 5-fold Cross-Validation Score. These were normalized and averaged to create a composite score to determine the best model.

The best model was Random Forest Regressor with:

Test $R^2 \approx 0.96$, representing perfect fit on unseen data

Cross-Validation Score ≈ 0.89 , indicating good generalization

Minimum MAE (1356.65) out of the top 30 samples

Minimum error margins with predictions ranging mostly around 500 units of actual prices

It was always superior to other models on both average performance and per-sample performance because of its ability to handle non-linearity, feature interaction, and insensitivity to outliers.

Apart from modeling, an interactive Power BI dashboard was developed to showcase:

- Brand-wise price distribution
- Key feature influence on pricing
- Trends across simulated years

These plots provide stakeholders with revealing perspectives into auto price behavior, complementing the model's pragmatic usefulness.

References:-

- **Kaggle Car Price Dataset**
Kaggle. (n.d.). *Car Price Prediction Dataset*. Retrieved from:
<https://www.kaggle.com/datasets/hellbuoy/car-price-prediction>
- **Scikit-learn: Machine Learning in Python**
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825-2830.
<https://scikit-learn.org/>
- **Seaborn: Statistical Data Visualization**
Waskom, M. (2021). *Seaborn: statistical data visualization*. Journal of Open Source Software, 6(60), 3021.
<https://seaborn.pydata.org/>
- **Matplotlib: Python Plotting**
Hunter, J. D. (2007). *Matplotlib: A 2D graphics environment*. Computing in Science & Engineering, 9(3), 90-95.
<https://matplotlib.org/>
- **Pandas: Data Analysis Toolkit**
McKinney, W. (2010). *Data Structures for Statistical Computing in Python*. Proceedings of the 9th Python in Science Conference, 51-56.
<https://pandas.pydata.org/>
- **Power BI Documentation**
Microsoft. (n.d.). *Power BI Documentation*. Retrieved from:
<https://learn.microsoft.com/en-us/power-bi/>
- **Random Forests**
Breiman, L. (2001). *Random Forests*. Machine Learning, 45(1), 5–32.
<https://doi.org/10.1023/A:1010933404324>