Car Price Prediction Using Machine Learning Techniques

Name: Thoufik Umar

Reg. No: 12304034

Faculty Name: Mrinalini Rana

Computer Science Engineering Department

Lovely Professional University Jalandhar, India

Predicting car prices accurately is an important task in the automobile and financial industry. With the growth of machine learning, regression models can be used to analyze historical car data and predict prices based on various features. This project focuses on implementing **Random Forest Regression** to predict car prices. The model was trained on a preprocessed dataset and evaluated using standard regression performance metrics such as **MAE, MSE, RMSE, and R^2 score**. The results demonstrate that Random Forest provides strong predictive performance by capturing complex nonlinear relationships in the data.

I. INTRODUCTION

In today's rapidly growing automobile market, determining the correct price of a vehicle has become a challenging task. Car prices depend on several factors such as brand reputation, model, year of manufacture, engine specifications, mileage, fuel type, and overall condition of the vehicle. Traditional pricing methods rely heavily on manual assessment, market trends, and expert opinion, which may lead to inaccurate or inconsistent pricing. As a result, there is a growing need for intelligent systems that can analyze historical data and provide accurate price predictions.

Machine learning has emerged as a powerful tool for solving real-world prediction problems by learning patterns from large datasets. Regression techniques in machine learning are especially useful when the target variable is continuous, such as predicting house prices, stock values, or car prices. By analyzing relationships between multiple input features and the target variable, regression models can generate reliable predictions with minimal human intervention.

II. RELATED WORK

Predicting vehicle prices using data-driven approaches has attracted significant attention from researchers and industry professionals due to its practical importance in automobile markets, online resale platforms, and financial decision-making systems. Traditional approaches to car price estimation relied on expert knowledge, fixed depreciation rules, and basic statistical techniques. These methods often failed to capture the complex interactions among multiple vehicle attributes and market dynamics, resulting in inaccurate and inconsistent price estimations .Early research in vehicle price prediction primarily utilized linear regression models, where car price was expressed as a linear combination of features such as vehicle age, mileage, engine capacity, and brand. While linear regression models are easy to interpret and computationally efficient, they assume a linear relationship between input features and the target variable. Several studies reported that linear models perform reasonably well for small and clean datasets but struggle to handle nonlinear patterns and multicollinearity present in real-world automobile data.

To overcome the limitations of linear regression, researchers explored polynomial regression and multivariate regression models, which introduced nonlinear terms to improve predictive accuracy. Although these models showed improved performance over simple linear regression, they were prone to overfitting and required careful feature engineering. Moreover, as the number of features increased, polynomial models became computationally expensive and difficult to interpret.

With advancements in machine learning, decision tree-based models gained popularity in car price prediction tasks. Decision Trees can model nonlinear relationships and feature interactions without requiring explicit feature transformations. Several studies demonstrated that decision trees outperform linear models in predicting used car prices by effectively capturing threshold-based decisions, such as sudden depreciation after a certain vehicle age or mileage. However, single decision trees are highly sensitive to noise in the data and often suffer from overfitting, leading to poor generalization on unseen data. To address the instability of individual decision trees, ensemble learning techniques such as Random Forest Regression were introduced. Random Forest, proposed by Leo Breiman, constructs multiple decision trees using random subsets of the training data and features, and aggregates their predictions through averaging. Numerous studies have shown that Random Forest significantly improves prediction accuracy and robustness compared to standalone decision trees. Researchers reported that Random Forest performs particularly well in automobile price prediction due to its ability to ha ndle heterogeneous features, missing values, and nonlinear relationships.

III. METHODOLOGY

The car price prediction system was developed using a structured machine learning approach. The dataset was collected and preprocessed by removing missing values and selecting relevant features. The data was then split into training and testing sets. A Random Forest Regression model was trained on the training data to learn complex relationships between car attributes and price. Finally, the model was evaluated on unseen test data using performance metrics such as MAE, MSE, RMSE, and R^2 score to assess prediction accuracy and reliability.

Data Description

The dataset used in this project consists of automobile-related information collected for the purpose of predicting car prices. Each record in the dataset represents a single vehicle along with its associated attributes. The dataset contains both numerical and categorical features that influence the market value of a car.

Key attributes in the dataset include vehicle specifications such as engine capacity, mileage, power, year of manufacture, fuel type, transmission type, and ownership details. The target variable is the **car price**, which represents the market value of the vehicle. These features collectively capture the technical, performance, and usage-related aspects of automobiles.

Data Preprocessing

Data preprocessing plays a vital role in enhancing the accuracy and reliability of machine learning models, as car price datasets often contain missing values, inconsistencies, and features with varying scales. The following preprocessing steps were carried out in this study:

1. Handling Missing Values:

   Several pollutant-related attributes contained missing observations. Removing all such records would have resulted in a significant loss of data; therefore, missing values in pollutant concentration columns were addressed using **median imputation**. The median was selected as it is less affected by extreme values and helps maintain the natural distribution of the data. However, records with missing **PM2.5** values were excluded, as PM2.5 was required for generating the target air quality categories.

2. Feature Selection and Data Leakage Prevention

   Only relevant features that contribute to car price prediction were selected for model training. The target variable, car price, was strictly excluded from the input feature set to avoid data leakage. This ensured that the model learned patterns only from independent variables such as vehicle age, mileage, engine specifications, fuel type, and transmission. Careful feature selection improved model efficiency and generalization performance.

3. Data Scaling

   The selected numerical features exhibited different units and value ranges. Although Random Forest models are less sensitive to feature scaling, standardization was applied to maintain consistency and support comparative analysis with other regression models. The StandardScaler technique was used to transform the features to have zero mean and unit variance, ensuring fair contribution of all features during model training.

Feature Engineering

Feature engineering was performed to enhance model learning and capture meaningful relationships between vehicle characteristics and car prices. Numerical attributes such as engine capacity, mileage, power, vehicle age, and other specification-related features were used as input variables. The target variable, car price, was excluded from the input feature set to avoid data leakage and ensure realistic model evaluation.

Model Selection and Training

To satisfy academic requirements and enable a fair evaluation, multiple supervised machine learning regression models were implemented in this study. Each model was chosen based on its learning characteristics and suitability for car price prediction:

1. Linear Regression:

   Linear Regression was used as a baseline regression model due to its simplicity and interpretability. As a linear model, it helps assess whether car prices can be approximated as a linear combination of vehicle features such as mileage, engine capacity, and age. The model also provides insight into how individual features influence car price prediction.

2. K-Nearest Neighbors (KNN):

   K-Nearest Neighbors Regression predicts car prices based on the average price of similar vehicles in the feature space. It was included to capture local patterns in vehicle characteristics. KNN does not assume a predefined relationship between features and price, making it effective for modeling nonlinear relationships in car price data.

3. Decision Tree Classifier:

   Decision Tree Regression was applied to model complex and nonlinear relationships between

vehicle features and car prices. Car pricing dynamics often involve threshold-based decisions, such as sudden depreciation after a certain age or mileage. The tree-based structure allows the model to capture such patterns while also providing interpretability.

4. Random Forest Classifier:

Random Forest Regression, an ensemble learning technique based on multiple decision trees, was selected for its robustness and strong generalization performance. By aggregating predictions from several trees trained on random subsets of data and features, Random Forest reduces overfitting and improves prediction accuracy. It effectively handles feature interactions and high-dimensional car datasets, making it well-suited for real-world price prediction tasks.

5. Naive Bayes Classifier:

Naive Bayes was not included in the final model set, as it is primarily designed for classification problems. Since this study focuses on predicting continuous car prices, regression-based models were preferred to ensure meaningful and accurate predictions.

.

IV. TRAINING AND VALIDATING STRATEGY

To ensure reliable and unbiased evaluation of the car price prediction model, a systematic training and validation strategy was adopted. The cleaned and preprocessed dataset was divided into **training and testing sets** using an appropriate data-splitting technique. A majority portion of the data was used for training the model, while the remaining portion was reserved for testing to evaluate model performance on unseen data.
The training dataset was used to fit the regression models, allowing them to learn relationships between vehicle attributes and car prices. During training, the Random Forest Regression model constructed multiple decision trees using random subsets of data and features, improving robustness and reducing overfitting. Other regression models were also trained using the same training data to ensure a fair comparison.

V. PERFORMANCE METRICS

To comprehensively evaluate the predictive performance of the proposed **regression models**, multiple evaluation metrics were employed. Relying on a single metric may provide a misleading assessment of model accuracy, especially when predicting continuous values such as car prices. Therefore, the following standard regression performance metrics were used:

1. Mean Absolute Error:

Mean Absolute Error measures the average absolute difference between the actual car prices and the predicted prices. It provides a clear indication of how close the model's predictions are to the true values. Lower MAE values indicate better prediction accuracy and more reliable model performance.

2. Mean Squared Error:

Mean Squared Error calculates the average of the squared differences between actual and predicted car prices. By squaring the errors, this metric penalizes larger prediction errors more heavily, making it useful for identifying models that produce extreme inaccuracies.

3. Root Mean Squared Error:

Root Mean Squared Error is the square root of the Mean Squared Error. It represents the prediction error in the same unit as the target variable (car price), making it easier to interpret. Lower RMSE values indicate that the model predicts car prices more accurately with fewer large errors.

4. R^2 Score:

The R^2 score measures how well the regression model explains the variance in car prices. An R^2 value closer to 1 indicates that the model successfully captures most of the variability in the data, while a value closer to 0 indicates poor predictive performance.

VI. RESULTS AND DISCUSSION

All implemented machine learning regression models were evaluated on an unseen test dataset using the performance metrics described in the previous section. The experimental results clearly highlight performance differences between baseline regression models and ensemble-based methods.

Model Performance Comparison

Linear Regression, used as the baseline model, achieved reasonable prediction accuracy for car prices. Its linear nature allowed it to capture general trends between

vehicle attributes such as mileage, engine capacity, and age. However, the model struggled to represent complex nonlinear relationships present in real-world car pricing data, resulting in higher prediction errors compared to tree-based models**.**

K-Nearest Neighbours (KNN) Regression showed improved performance over Linear Regression by capturing local similarities between vehicles. It performed well for cars with characteristics similar to those in the training data but was sensitive to noise and feature distribution. The model's performance decreased for vehicles with rare or extreme feature values, affecting overall generalization..

Decision Tree Regression demonstrated better ability to model nonlinear relationships and threshold-based price variations, such as sudden depreciation after certain mileage or age limits. While it achieved lower error values compared to Linear Regression and KNN, the model showed signs of overfitting, leading to reduced stability on unseen test data**.**

Random Forest Regression outperformed all other regression models in terms of accuracy and robustness. By combining predictions from multiple decision trees trained on random subsets of data and features, Random Forest effectively reduced overfitting and captured complex interactions among vehicle attributes. The model achieved the lowest MAE and RMSE values along with the highest R^2 score, indicating strong predictive capability and better generalization.

VII. CONCLUSION

This project successfully demonstrated the application of machine learning techniques for predicting car prices using real-world automobile data. A structured machine learning pipeline was implemented, including data preprocessing, feature engineering, model selection, training, and evaluation. Multiple regression models were developed and compared to analyze their effectiveness in predicting car prices.

Among the evaluated models, **Random Forest Regression** achieved the best performance in terms of prediction accuracy and generalization. The ensemble-based approach effectively captured complex and nonlinear relationships between vehicle attributes and price, resulting in lower prediction errors and a higher R^2 score compared to baseline models such as Linear Regression, K-Nearest Neighbors, and Decision Tree Regression.

The use of multiple performance metrics, including MAE, MSE, RMSE, and R^2 score, provided a comprehensive evaluation of model performance and ensured reliable comparison. The results confirm that Random Forest Regression is a robust and reliable model for car price prediction tasks.

Overall, this study highlights the effectiveness of machine learning in supporting data-driven decision-making in the automobile domain. Future work may focus on incorporating advanced ensemble techniques, performing hyperparameter tuning, and expanding the dataset to further improve prediction accuracy and real-world applicability.

VIII. REFERENCES

[1] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[2] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., Springer, 2009.

[3] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[4] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, 2nd ed., O'Reilly Media, 2019.

[5] Kaggle, "Car Price Prediction Dataset," Available online: https://www.kaggle.com

[6] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques*, Morgan Kaufmann, 2016.

[7] J. Brownlee, "Introduction to Random Forest for Regression," *Machine Learning Mastery*, 2020.

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

from sklearn.model_selection import train_test_spli
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_absolute_error, me


file_path = r"C:\Users\thouf\Downloads\car_price_pr
df = pd.read_csv(file_path)

print(" FIRST 5 ROWS ")
print(df.head())

print("\n DATASET SHAPE ")
print(df.shape)

print("\n COLUMN NAMES ")
print(df.columns)

print("\nDATASET INFO")
print(df.info())
```

```
print("\n STATISTICAL SUMMARY")
print(df.describe())

print("\n MISSING VALUES ")
print(df.isna().sum())


df = df.dropna()

print("\nShape after dropping missing
print(df.shape)


sns.set(style="whitegrid")

# Distribution of Price
plt.figure(figsize=(8,4))
sns.histplot(df["Price"], kde=True)
plt.title("Distribution of Car Prices"
plt.show()

# Price vs Mileage
plt.figure(figsize=(8,4))
sns.scatterplot(x="Mileage", y="Price"
plt.title("Price vs Mileage")
plt.show()
```

```python
# Price vs Year
plt.figure(figsize=(8,4))
sns.scatterplot(x="Year", y="Price", data=df)
plt.title("Price vs Year")
plt.show()
# Price vs Engine Size
plt.figure(figsize=(8,4))
sns.scatterplot(x="Engine Size", y="Price", data=df)
plt.title("Price vs Engine Size")
plt.show()

# Correlation Heatmap
numeric_cols = df.select_dtypes(include="number")
plt.figure(figsize=(8,6))
sns.heatmap(numeric_cols.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()

target_column = "Price"

X = df.drop(columns=[target_column, "Car ID"])
y = df[target_column]

X = pd.get_dummies(X, drop_first=True)

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

```python
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)


lr_model = LinearRegression()
lr_model.fit(X_train_scaled, y_train)

lr_pred = lr_model.predict(X_test_scaled)

print("\nLINEAR REGRESSION")
print("MAE:", mean_absolute_error(y_test, lr_pred))
print("MSE:", mean_squared_error(y_test, lr_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, lr_pred)))
print("R2 Score:", r2_score(y_test, lr_pred))


dt_model = DecisionTreeRegressor(random_state=42)
dt_model.fit(X_train, y_train)

dt_pred = dt_model.predict(X_test)

print("\n DECISION TREE REGRESSION")
print("MAE:", mean_absolute_error(y_test, dt_pred))
print("MSE:", mean_squared_error(y_test, dt_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, dt_pred)))
print("R2 Score:", r2_score(y_test, dt_pred))
```

```python
knn_model = KNeighborsRegressor(n_neighbors=5)
knn_model.fit(X_train_scaled, y_train)

knn_pred = knn_model.predict(X_test_scaled)

print("\nKNN REGRESSION")
print("MAE:", mean_absolute_error(y_test, knn_pred))
print("MSE:", mean_squared_error(y_test, knn_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, knn_pred)))
print("R2 Score:", r2_score(y_test, knn_pred))



rf_model = RandomForestRegressor(
    n_estimators=100,
    random_state=42
)
rf_model.fit(X_train, y_train)

rf_pred = rf_model.predict(X_test)

print("\nRANDOM FOREST REGRESSION")
print("MAE:", mean_absolute_error(y_test, rf_pred))
print("MSE:", mean_squared_error(y_test, rf_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, rf_pred)))
print("R2 Score:", r2_score(y_test, rf_pred))
```

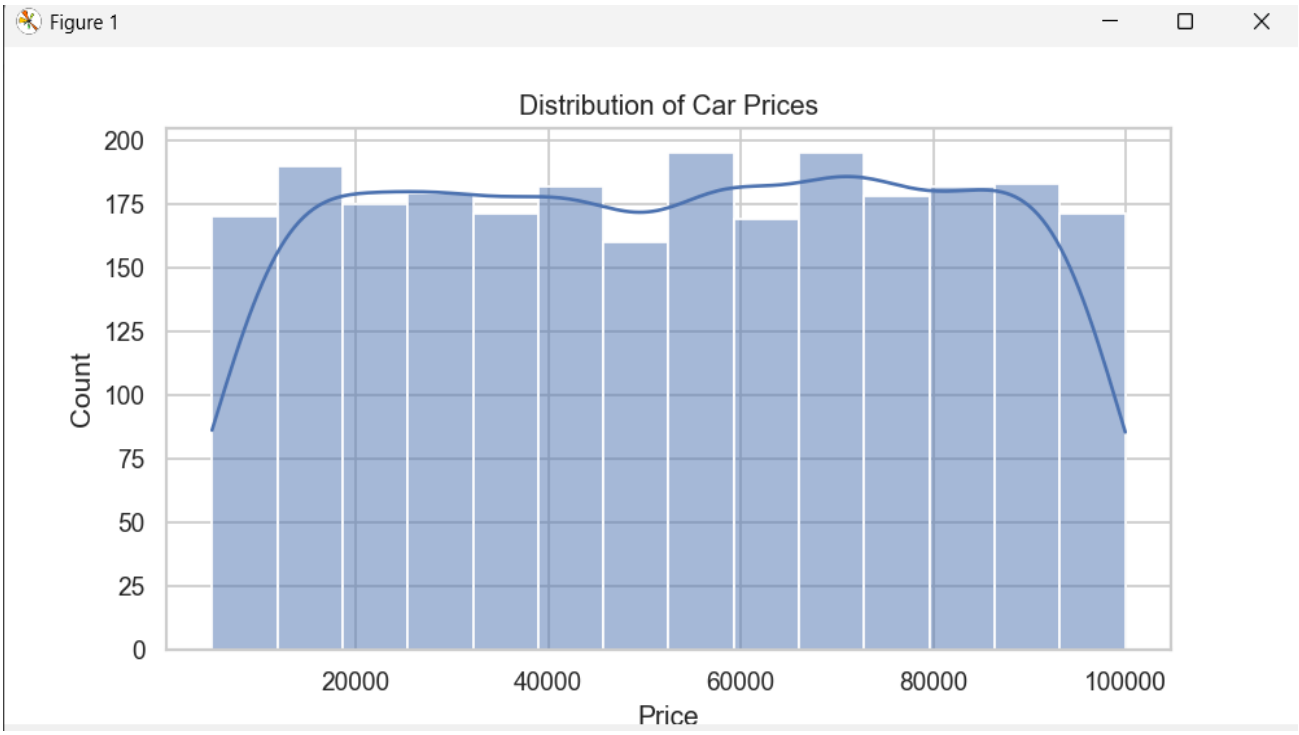


Figure 1
Price vs Mileage
100000
80000
60000
40000
20000
Price
0
50000
100000
150000
200000
250000
300000
Mileage

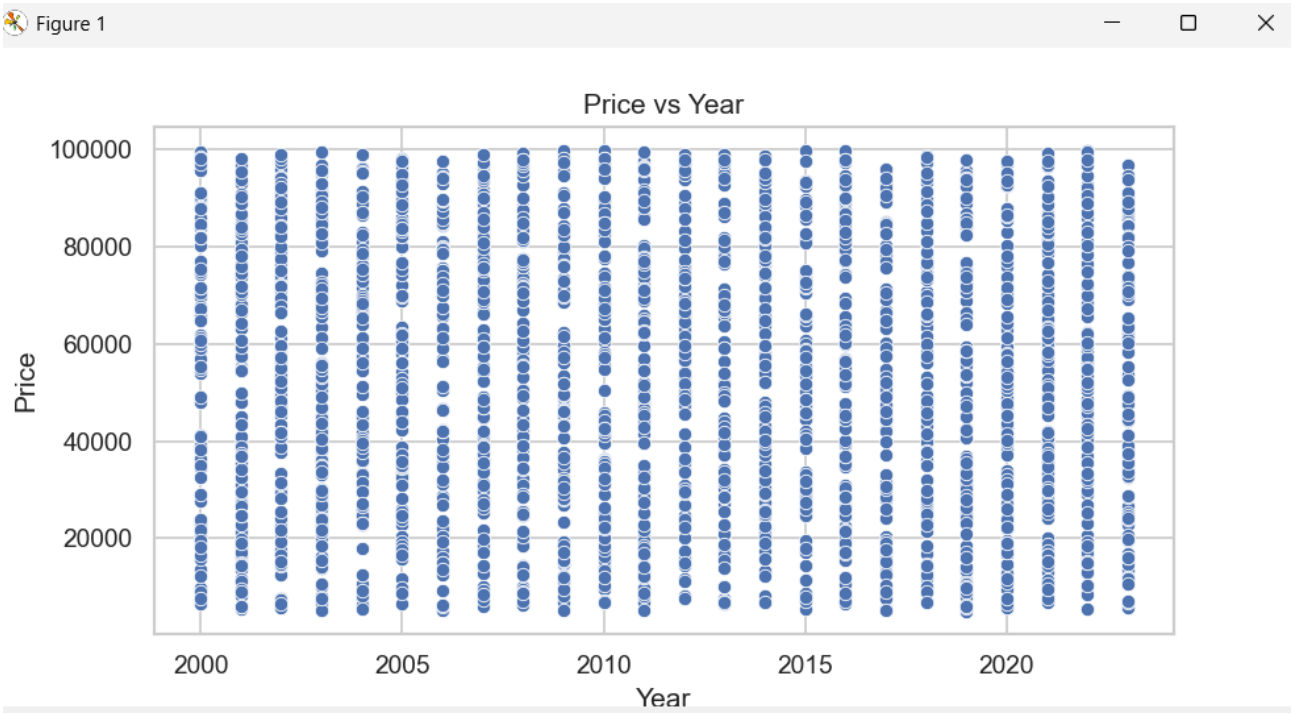
Figure 1
Price vs Year
100000
80000
60000
40000
20000
Price
2000
2005
2010
2015
2020
Year


Figure 1
Price vs Engine Size
Price
100000
80000
60000
40000
20000
1
2
3
4
5
6

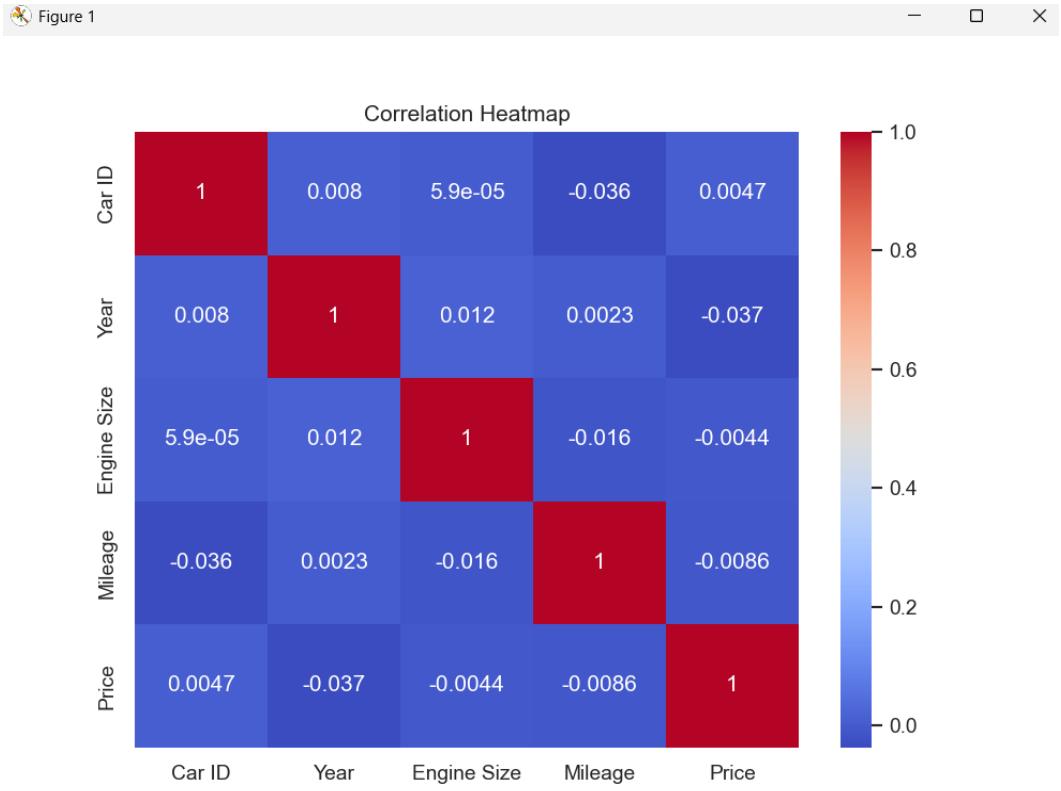
Figure 1
Correlation Heatmap
Car ID
Year
Engine Size
Mileage
Price
1
0.008
5.9e-05
-0.036
0.0047
0.008
1
0.012
0.0023
-0.037
5.9e-05
0.012
1
-0.016
-0.0044
-0.036
0.0023
-0.016
1
-0.0086
0.0047
-0.037
-0.0044
-0.0086
1
1.0
0.8
0.6
0.4
0.2
0.0