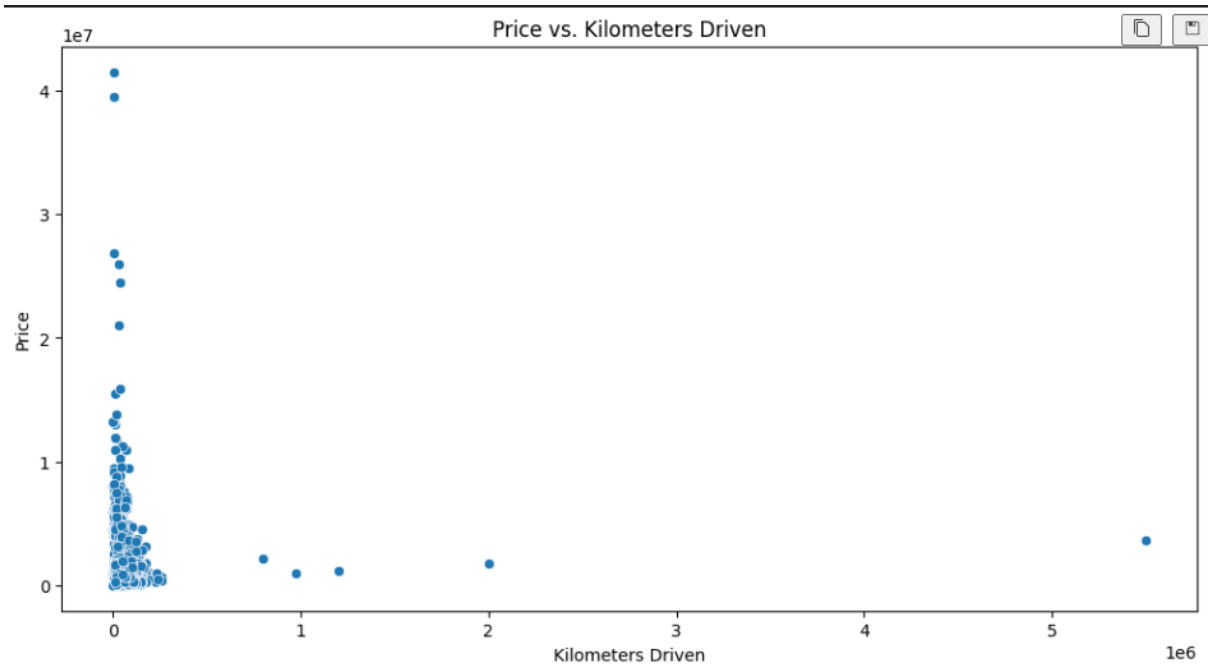
Documentation Report: Used Car Price Prediction Application

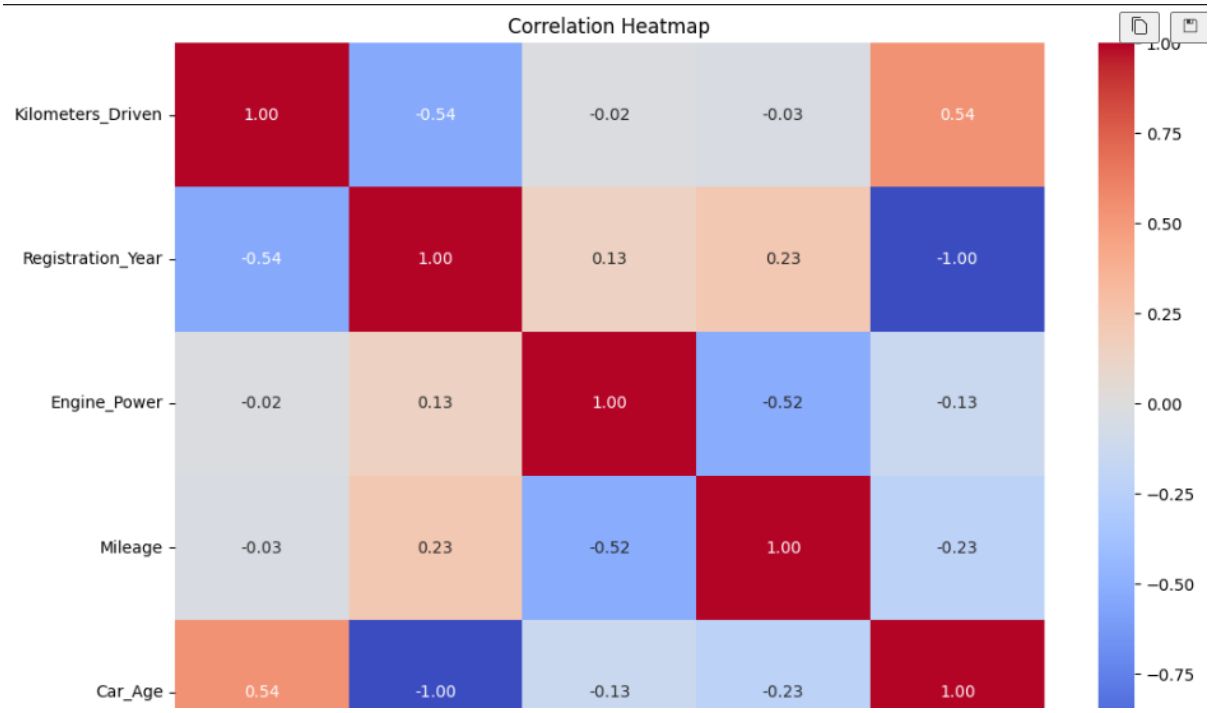
Introduction

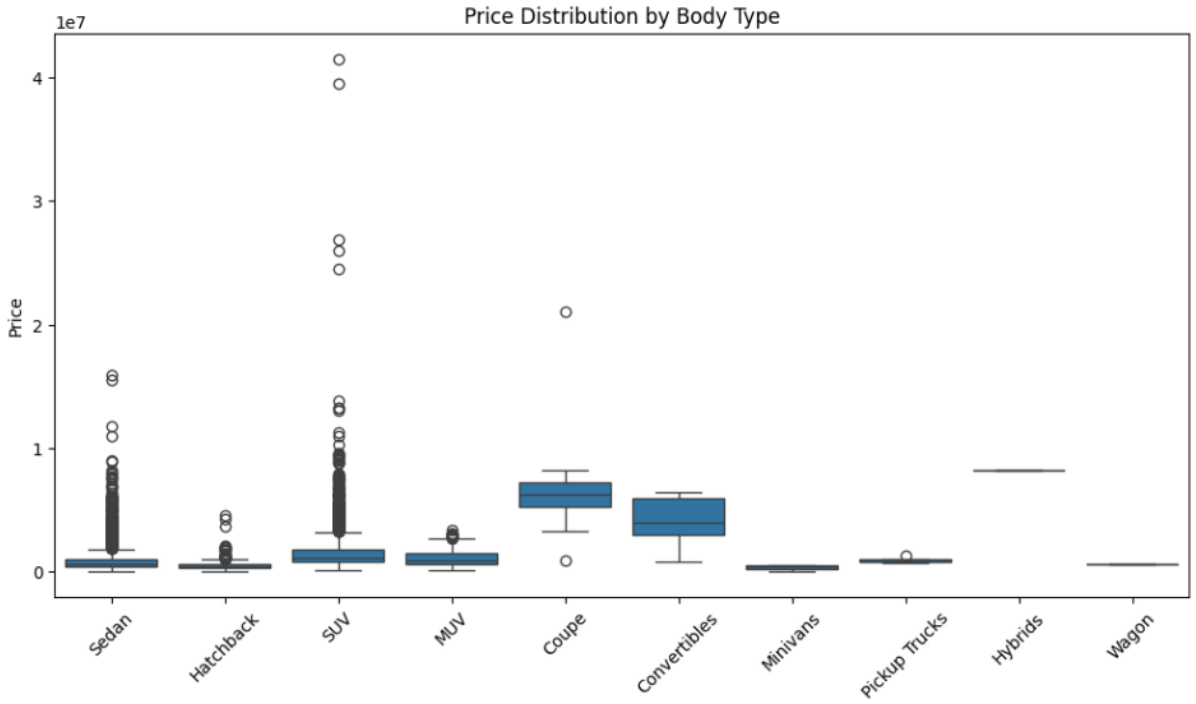
This application is designed to predict the price of a used car based on specific input features provided by the user. The prediction model is built using a linear regression algorithm, and the application interface is developed with Streamlit. The primary goal of this application is to offer a simple, user-friendly experience for estimating the price of a car using essential attributes.

Key Features

1. **User-Centered Input Interface:**
    - All inputs are positioned in the center of the page for a focused and streamlined user experience.
    - The inputs include car make, fuel type, registration year, engine power, kilometers driven, and city.
2. **Prediction Model:**
    - The model uses Linear Regression to predict the price of a used car.
    - Essential features for prediction include Kilometers_Driven, Registration_Year, Mileage, and Engine_Power.







3. **Model Training and Evaluation:**
    - The model is trained on a dataset of used cars and evaluated using metrics like Mean Squared Error (MSE) and R-squared (R2).
    - Missing values in the input data are handled using median imputation.

```
Linear Regression:
  CV Mean Score: 584763541101.57
  CV Std Score: 408779842408.85
  Test MSE: 170811645236.44

Decision Tree:
  CV Mean Score: 576332537945.29
  CV Std Score: 382973616156.42
  Test MSE: 118513947448.69

Random Forest:
  CV Mean Score: 297835904826.83
  CV Std Score: 314691065646.74
  Test MSE: 88753674073.50

Gradient Boosting:
  CV Mean Score: 354158011390.86
  CV Std Score: 341763417743.10
  Test MSE: 101519622914.55

Best parameters for Random Forest: {'max_depth': 30, 'min_samples_split': 2, 'n_estimators': 50}
Best score for Random Forest: 297584318179.78
Best parameters for Gradient Boosting: {'n_estimators': 100, 'max_depth': 5, 'learning_rate': 0.1}
Best score for Gradient Boosting: 307092973031.83
```

4. **Real-Time Price Prediction:**
    - Upon providing the input details, the application predicts the car price and displays it prominently on the page.
    - The prediction is accompanied by model performance metrics to give users an understanding of the model's accuracy.

Data Handling and Model Training

1. Data Loading

The dataset is loaded from a CSV file ('cleaned_combined_cars.csv'). The data is cached to enhance performance, reducing the need to reload the dataset with every interaction.

```python
def load_data():
    combined_df = pd.read_csv('cleaned_combined_cars.csv')
    return combined_df
```

✓ 0.0s

2. Model Training

The Linear Regression model is trained using the following steps:

- **Feature Selection:** The model uses four primary features: Kilometers_Driven, Registration_Year, Mileage, and Engine_Power.
- **Imputation:** Missing values in these features are filled using the median value of the respective feature.

- **Data Splitting:** The data is split into training and testing sets (80/20 split).
- **Model Fitting:** The Linear Regression model is fitted on the training data.

The model is cached to avoid retraining with every interaction, thus improving performance.

Exploratory analysis using numerical statistics and categorical statistics

```
Numerical Statistics:
       Kilometers_Driven  Registration_Year  Engine_Power      Mileage  \
count       7.249000e+03        2020.000000   7198.000000  6168.000000
mean        5.665036e+04        2015.575743    106.193109    19.270323
std         7.777081e+04           4.660192     47.087657     3.643097
min         0.000000e+00        2002.000000     25.000000     7.080000
25%         3.000000e+04        2012.000000     80.000000    17.100000
50%         5.100000e+04        2016.000000     88.000000    18.900000
75%         7.600000e+04        2019.000000    120.000000    21.437500
max         5.500000e+06        2023.000000    576.000000    35.600000

              Price      Car_Age
count  7.249000e+03  2020.000000
mean   9.970425e+05     8.424257
std    1.476826e+06     4.660192
min    2.016100e+04     1.000000
25%    4.100000e+05     5.000000
50%    6.220000e+05     8.000000
75%    9.800000e+05    12.000000
max    4.150000e+07    22.000000

Categorical Statistics:
       Fuel_Type  Body_Type   Car_Model    Ownership  \
count       7249       7245        7249         7218
unique         5         10         310            5
...
Price                                                650000.0
Car_Age                                                   2.0
City                                                    Delhi
```

3. Prediction Process

The user provides the necessary input details through the main interface. The application calculates the mileage based on the car make, city, and registration year. If mileage data is unavailable, the median mileage is used.

The input data is processed, ensuring it matches the format of the training data, and the model predicts the car price based on these inputs.

```
input_data_imputed = imputer.transform(input_data)
predicted_price = model.predict(input_data_imputed)[0]
```

4. Model Performance

The application reports the following performance metrics:

- **Mean Squared Error (MSE):** Indicates the average of the squares of the errors, showing how close the predictions are to the actual values.
- **R-squared (R2):** Represents the proportion of the variance in the dependent variable that is predictable from the independent variables.

These metrics are displayed on the interface to give users insight into the model's reliability.

```
Linear Regression:
  Mean Absolute Error (MAE): 305120.57
  Mean Squared Error (MSE): 495949299211.85
  R-squared: 0.67

Decision Tree:
  Mean Absolute Error (MAE): 221819.31
  Mean Squared Error (MSE): 406887357048.28
  R-squared: 0.73

Random Forest:
  Mean Absolute Error (MAE): 172115.40
  Mean Squared Error (MSE): 271679968135.37
  R-squared: 0.82

Gradient Boosting:
  Mean Absolute Error (MAE): 242754.25
  Mean Squared Error (MSE): 323450834539.71
  R-squared: 0.78
```

User Interface

1. Input Fields

The application provides the following input fields:

- **Car Make:** Drop-down selection of available car models.
- **Fuel Type:** Drop-down selection of fuel types (e.g., Petrol, Diesel).
- **Registration Year:** Drop-down selection of the year the car was registered.
- **Engine Power:** Numeric input for the car’s engine power in bhp.
- **Kilometers Driven:** Numeric input for the total kilometers the car has been driven.
- **City:** Drop-down selection of cities where the car is located.

2. Output

The predicted car price is prominently displayed in the center of the page, ensuring visibility and clarity. The output is formatted to appear professional and easily readable.

3. Model Performance Metrics

At the bottom of the page, the application provides the MSE and R2 scores, giving users a quick overview of the model's accuracy.

Conclusion

This documentation outlines the functionality, data handling, and user interface of the Used Car Price Prediction Application. The application is designed for simplicity, accuracy, and user-friendliness, making it a valuable tool for estimating the price of used cars based on essential attributes. The model's performance metrics are included to assure users of the prediction's reliability.