CAR PRICE PREDICTION

An Analysis of Used Car Data to Predict Market Prices

PREPARED BY:

TENZIN TSOMU

DATE:

JULY,2024

Table of Contents

1. Introduction

This report presents the process of cleaning and analysing a dataset of used cars to build a predictive model for car prices. The dataset, sourced from `quikr_car.csv`, includes various features such as car name, company, year of manufacture, price, kilometers driven, and fuel type. The objective is to clean the data, perform exploratory analysis, and develop a linear regression model to predict car prices.

2. Data Overview

The dataset comprises 892 entries with the following columns:

- `name`: Car name
- `company`: Car manufacturer
- `year`: Year of manufacture
- `Price`: Price of the car
- `kms_driven`: Kilometers driven
- `fuel_type`: Type of fuel

3. Data Cleaning and Preprocessing

3.1 Initial Data Inspection

```
car.head()
```

Copy

The initial inspection revealed inconsistent data and non-numeric values in several columns.

3.2 Data Cleaning Steps

- **Year**: Filtered out non-numeric values and converted to integer.
- **Price**: Removed rows with 'Ask For Price', handled commas, and converted to integer.
- **Kilometers Driven**: Extracted numeric values from strings and converted to integer.
- **Fuel Type**: Removed rows with missing fuel types.

3.3 Additional Cleaning

- **Name Column**: Simplified by retaining only the first three words.
- **Price Outliers**: Filtered out prices greater than 6 million.
- **Company Column**: Verified and retained relevant company names.

4. Data Transformation

Categorical features (`'name'`, `'company'`, `'fuel_type'`) were one-hot encoded. The transformed dataset was prepared for modeling.

5. Model Building

5.1 Feature and Target Variables

Features (`'X'`) include:

- `'name'`
- `'Company'`
- `'Year'`
- `'Kms_driven'`
- `'fuel_type'`

Target variable (`'y'`):

- `'Price'`

5.2 Train-Test Split

The dataset was split into training and testing sets using a 80-20 split.

5.3 Model Pipeline

A pipeline was created that included:

- `OneHotEncoder` for categorical variables
- `LinearRegression` model

```
from sklearn.pipeline import make_pipeline
from sklearn.compose import make_column_transformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score

# OneHotEncoder setup
ohe = OneHotEncoder()
column_trans = make_column_transformer(
    (OneHotEncoder(categories=ohe.categories_), ['name', 'company', 'fuel_type']
    remainder='passthrough'
)

# Linear Regression model
lr = LinearRegression()
pipe = make_pipeline(column_trans, lr)

# Training the model
pipe.fit(X_train, y_train)
```

5.4 Model Evaluation

The R-squared score for the initial test set was approximately 0.605, indicating moderate explanatory power. After multiple iterations with different random states, the best R-squared score achieved was approximately 0.899.

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
import numpy as np

# Finding the best random state
scores = []
for i in range(1000):
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, ran
    pipe.fit(X_train, y_train)
    y_pred = pipe.predict(X_test)
    scores.append(r2_score(y_test, y_pred))

best_random_state = np.argmax(scores)
best_score = scores[best_random_state]
```

6. Model Prediction

The model was used to predict the price of a sample car:

```
sample_prediction = pipe.predict(pd.DataFrame(columns=X_test.columns, data=np.ar
```

7. Conclusions

The data cleaning and preprocessing steps ensured that the dataset was suitable for modeling. The linear regression model, when evaluated, showed a significant improvement in predictive performance after optimization. The model can be used to estimate car prices based on various features, but further improvements could be made by exploring more advanced models and additional feature engineering.

8. Recommendations

1. **Explore Advanced Models**: Consider using more complex models such as Random Forests or Gradient Boosting Machines.
2. **Feature Engineering**: Introduce new features like the age of the car and interaction terms.
3. **Cross-Validation**: Implement cross-validation to obtain a more reliable measure of model performance.
4. **Visualization**: Conduct exploratory data analysis (EDA) to visualize data distributions and residuals.

9. Next Steps

1. **Outlier Analysis**: Investigate outliers in ‘`Price`’ and ‘`kms_driven`’.
2. **Enhanced Feature Engineering**: Develop additional features and interactions to improve model performance.
3. **Visualization**: Generate plots to better understand data distribution and model performance.