Car prediction - Case Study

SELVARAJ X(2577127)

04-11-2023

Introduction

- ❖ Car manufacturing companies need to have a good understanding of car prices in the market in order to launch new cars in different categories.
- ❖ This can be done by developing a car price prediction model based on the specifications of cars available in the market.
- ❖ The first step in developing a car price prediction model is to perform EDA on the available data. This involves understanding the data distribution, identifying outliers, and checking for multicollinearity.

Model Selection And Criteria

- ❖ There are a variety of machine learning models that can be used for car price prediction. Some of the most popular models include linear regression, decision trees, and random forests.

  Model Selection Criteria..

- ❖ When selecting a model, it is important to consider the following criteria:
  - ➢ Accuracy: How well does the model predict the car prices on the training data?
  - ➢ Overfitting: Does the model overfit the training data, such that it performs poorly on new data?
  - ➢ Interpretability: How easy is it to understand how the model makes predictions?

Feature Selection

```python
# Split dataset

# Feature Selected which has high correlation
x = train_data[[
    'wheel.base',
    'length',
    'width',
    'height',
    'curb.weight',
    'engine.size',
    'bore',
    'stroke',
    'compression.ratio',
    'horsepower'
]]
y = train_data.iloc[:,-1:]
```

LASSO & Ridge Optimization

- ❖ LASSO and ridge regularization are techniques that can be used to prevent overfitting.
- ❖ Both Lasso and Ridge give same accuracy.

```
<------------ Lasso Regression model ----------->

Lasso Train score        0.721122667416157

Lasso Test score         0.7528484260110737

R-Square        0.7528484260110737

MAE      3087.395817735319
MSE      15241482.150770431
RMSE     3904.0340867838786
```

```
<------------ Ridge Regression model ----------->

Lasso Train score:       0.721122664211696

Lasso Test score:        0.7528516727436303

R-Square:        0.7528516727436303

MAE      3087.392653706613
MSE      15241281.929442637
RMSE     3904.0084438231734
```

Model Creation

Selecting the model which give high accuracy

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score,mean_squared_error
ranc = RandomForestRegressor(n_estimators=10,random_state=1)

ranc.fit(xtest,ytest)
ypred  = ranc.predict(xtest)

print("Accuracy score :\t",r2_score(ytest,ypred))
print()
print('Mean squared Error :\t',mean_squared_error(ytest,ypred))
print()
print('RMSE :\t', np.sqrt(mean_squared_error(ytest,ypred)))
```

```
Accuracy score :         0.9603279459599829

Mean squared Error :     2446518.522121212

RMSE :   1564.1350715718934
```

Testing the model

➔ Testing the model which has unseen data.

```
# Split dataset

X = test_data[[
    'wheel.base',
    'length',
    'width',
    'height',
    'curb.weight',
    'engine.size',
    'bore',
    'stroke',
    'compression.ratio',
    'horsepower',
]]
```

Car price for Unseen data

```
ranc.fit(x,y)
X['Car_Price']  = ranc.predict(X)
```

```
X.head()
```

gth	width	height	curb.weight	engine.size	bore	stroke	compression.ratio	horsepower	Car_Price
58.8	64.1	48.8	2548	130	3.47	2.68	9.0	111	16500.0
76.6	66.4	54.3	2824	136	3.19	3.40	8.0	115	13176.5
77.3	66.3	53.1	2507	136	3.19	3.40	8.5	110	12469.1
92.7	71.4	55.7	2954	136	3.19	3.40	8.5	110	16882.0
92.7	71.4	55.9	3086	131	3.13	3.40	8.3	140	18625.1

Ensemble

```
from sklearn.ensemble import AdaBoostRegressor

ada = AdaBoostRegressor(base_estimator=ranc,n_estimators=15)
ada.fit(xtrain,ytrain)
ada_pred = ada.predict(xtest)

print("Accuracy score :\t",r2_score(ytest,ada_pred))
print()
print('Mean squared Error :\t',mean_squared_error(ytest,ada_pred))
print()
print('RMSE :\t', np.sqrt(mean_squared_error(ytest,ada_pred)))
```

```
Accuracy score :        0.9022018841764445

Mean squared Error :    6031069.163939394

RMSE :   2455.82352052003
```

★ Boosting - Adaboost with base model

```python
from sklearn.ensemble import GradientBoostingRegressor


grad = GradientBoostingRegressor()
grad.fit(xtrain,ytrain)
grad_pred = grad.predict(xtest)

print("Accuracy score :\t",r2_score(ytest,grad_pred))
print()
print('Mean squared Error :\t',mean_squared_error(ytest,grad_pred))
print()
print('RMSE :\t', np.sqrt(mean_squared_error(ytest,grad_pred)))
```

```
Accuracy score :         0.8821436572606656

Mean squared Error :     7268031.173036882

RMSE :   2695.928629069561
```

Boosting - Gradient Boost

Conclusion

The RandomForest Regression model performed the best in terms of accuracy, overfitting, and interpretability. Therefore, this model is recommended for predicting car prices.