FLIP ROBO

Car Price Prediction

Submitted by:

C S Manjunath Reddy

ACKNOWLEDGMENT

I would like to thank FlipRobo for giving me this opportunity. The DataTrained institute classes helped me to solve this problem.

The language used for web scrapping is Selenium and Pandas. For the project, the language used Python with Pandas, NumPy, Sklearn.

The data was collected from Olx and Cardekho websites for training the ML model for price prediction.

INTRODUCTION

- Business Problem Framing

  Client works with small traders, who sell used cars. With the change in market due to covid 19 impact, client is facing problems with their previous car price valuation machine learning models. So, they are looking for new machine learning models from new data. We collected data from car dekho and OLx and we are using that data here to train the model

- Conceptual Background of the Domain Problem

  The domain-related concepts are web scrapping, Data science, Machine Learning.

- Review of Literature

  I researched different locations from olx and cardekho and collected the car data as data sources to train the model. Once I scrapped the data I used it to train the model with the help of pandas and Sickit-learn.

- Motivation for the Problem Undertaken

  Client wants to predict the cars price. So, I collected the data from different websites to train Ml model and to predict prices for future use for clients.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

  The data we needed should contain Brand, model, variant, manufacturing year, driven kilometres, fuel, number of owners, location and at last target variable.The data is collected from olx and cardekho.

- Data Sources and their formats

  The data is collected from Olx and Cardekho websites. The data is collected with Selenium webscrapping.

- Data Preprocessing Done

  The data contains much wrong information, we replaced such data with nan values and removed all the null values to get better data to train the model.

- Hardware and Software Requirements and Tools Used

  Selenium : used to web scrapping to collect data multiple websites.

  Pandas : used to create data frames,clean the data.

  Scikit-Learn : Used Ml algorithms to train the data and evaluate best model.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

  Collected the data by using selenium webdriver.
  Created the data frame for the scraped data.
  Used the data to train the modem and to predict the car price.
  Saved the best model for future usage

- Testing of Identified Approaches (Algorithms)

  Linear Regression
  DecisionTreeRegressor
  RandomforestRegressor
  KNeighborsRegressor

- Run and Evaluate selected models
  Describe all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.
- Key Metrics for success in solving problem under consideration

Required libraries to train the model.

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import r2_score, mean_squared_error,mean_absolute_error
```

```
lr=LinearRegression()
dr=DecisionTreeRegressor()
rf=RandomForestRegressor()
kn=KNeighborsRegressor()
```

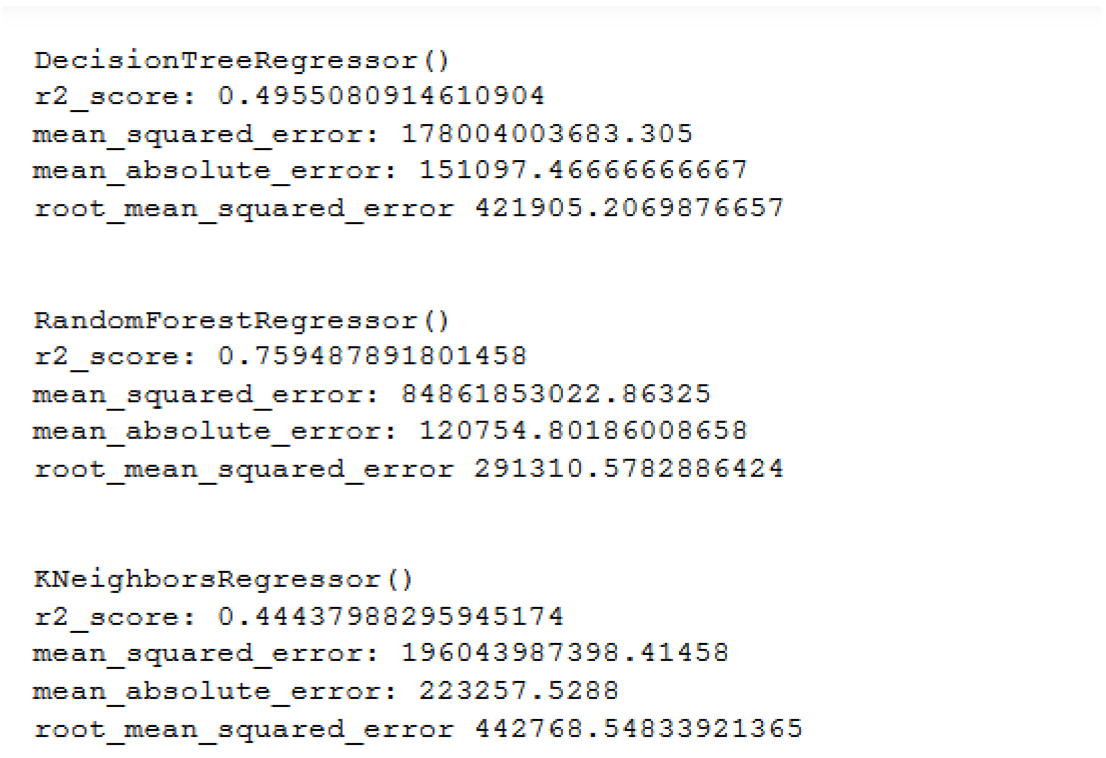
```
model=[lr,dr,rf,kn]
```

```
Splitting data for training and testing
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=40)
```

```
for m in model:
    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=34)
    m.fit(x_train,y_train)
    predv=m.predict(x_test)
    print(m)
    print('r2_score:',r2_score(y_test,predv))
    print('mean_squared_error:',mean_squared_error(y_test,predv))
    print('mean_absolute_error:',mean_absolute_error(y_test,predv))
    print('root_mean_squared_error',np.sqrt(mean_squared_error(y_test,predv)))
    print('\n')
```

```
LinearRegression()
r2_score: 0.3646650679868234
mean_squared_error: 224170417134.6158
mean_absolute_error: 275994.81065743545
root_mean_squared_error 473466.38437656354
```

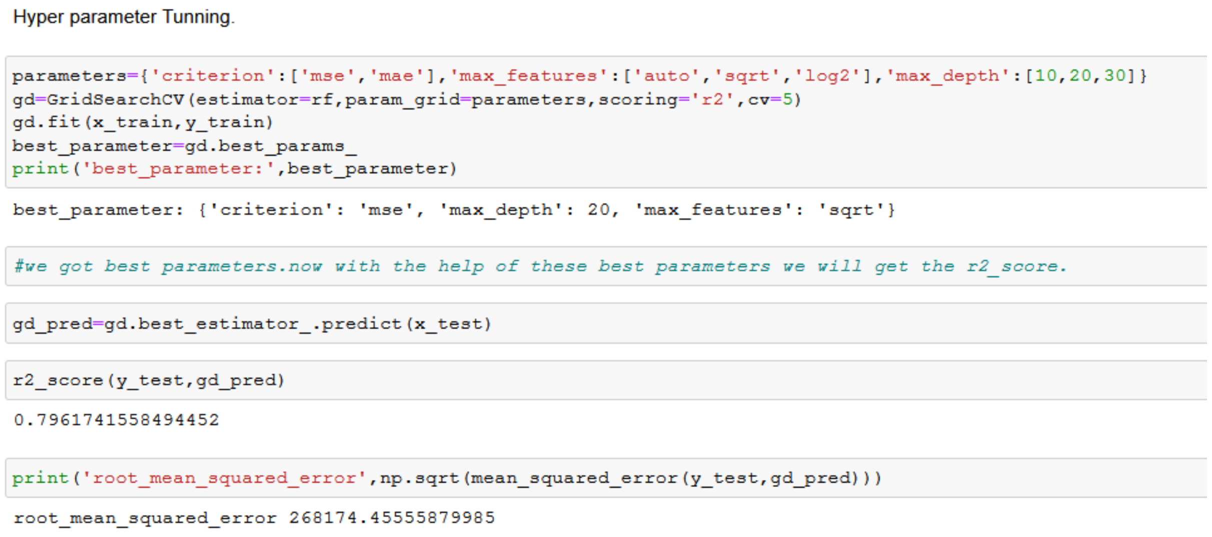


```
DecisionTreeRegressor()
r2_score:  0.4955080914610904
mean_squared_error:  178004003683.305
mean_absolute_error:  151097.46666666667
root_mean_squared_error 421905.2069876657


RandomForestRegressor()
r2_score:  0.759487891801458
mean_squared_error:  84861853022.86325
mean_absolute_error:  120754.80186008658
root_mean_squared_error 291310.5782886424


KNeighborsRegressor()
r2_score:  0.44437988295945174
mean_squared_error:  196043987398.41458
mean_absolute_error:  223257.5288
root_mean_squared_error 442768.54833921365
```

Hyper parameter tuning for Randomforestregressor



Hyper parameter Tunning.

```
parameters={'criterion':['mse','mae'],'max_features':['auto','sqrt','log2'],'max_depth':[10,20,30]}
gd=GridSearchCV(estimator=rf,param_grid=parameters,scoring='r2',cv=5)
gd.fit(x_train,y_train)
best_parameter=gd.best_params_
print('best_parameter:',best_parameter)
```

```
best_parameter: {'criterion': 'mse', 'max_depth': 20, 'max_features': 'sqrt'}
```

```
#we got best parameters.now with the help of these best parameters we will get the r2_score.
```

```
gd_pred=gd.best_estimator_.predict(x_test)
```

```
r2_score(y_test,gd_pred)
```

```
0.7961741558494452
```

```
print('root_mean_squared_error',np.sqrt(mean_squared_error(y_test,gd_pred)))
```

```
root_mean_squared_error 268174.45555879985
```

- ## Interpretation of the Results

Key Metrics are RMSE (which shows how good the model is working) and r2_score shows the accuracy of the model.

CONCLUSION

- Key Findings and Conclusions of the Study

  Trained the data with multiple algorithms and checked RandomForestRegressor is the better algorithm with better r2_score and RMSE. Did Hyper Parameter tuning for this algorithm to get better model and saved that model for future use.