CAR PRICE PREDICTION

Submitted by:

PRITI CHAUHAN

ACKNOWLEDGMENT

I would like to acknowledge Ms. Khshboo Garg for giving this assignment.

I would want to convey my sincere thanks Datatrained Academy and their guidance without them, the task would not have been accomplished.

The website that I referred are:

https://learning.datatrained.com
https://github.com
https://www.geeksforgeeks.org
https://www.carwale.com
https://www.cardekho.com
https://www.cartrade.com
https://stackoverflow.com
https://www.kaggle.com

INTRODUCTION

- Business Problem Framing

With the covid 19 impact in the market, we have seen lot of changes in the car market. Now some cars are in demand hence making them costly and some are not in demand hence cheaper. One of our clients works with small traders, who sell used cars. With the change in market due to covid 19 impact, our 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 have to make car price valuation model.

- Conceptual Background of the Domain Problem

  This project is about predicting the price of used cars in India, using the data of some websites. There are two phases in this project:
  - Data Collection Phase.
  - Model Building Phase.

- Review of Literature

  1. First phase is Data Scraping using Selenium.
     Scraped data from:
     − www.cardekho.com
     − www.carwale.com
     − www.cartrade.com

Features:

− Brand (Brand of the car)
− Name (Name of the car)
− Year (Year of manufactured)
− Fuel (Fuel used in the car)
− Driven(km) (How much the car is driven in km)
− City (In which city is car available to sell)

Target:

− Price (Price of the used car)

- Motivation for the Problem Undertaken

  This project is on the data scraping, data science and machine learning model, build the model to predict the used car price based on some features.

Analytical Problem Framing

- Mathematical/ Analytical Modelling of the Problem
  - **Information of the dataset:**

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3942 entries, 0 to 3941
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Unnamed: 0   3942 non-null   int64 
 1   Brand        3942 non-null   object
 2   Name         3942 non-null   object
 3   Year         3942 non-null   int64 
 4   Fuel         3942 non-null   object
 5   Driven(km)   3942 non-null   object
 6   City         3942 non-null   object
 7   Price        3942 non-null   object
dtypes: int64(2), object(6)
memory usage: 246.5+ KB
```

  - **Description of the dataset:**

	Unnamed: 0	Year
count	3942.000000	3942.000000
mean	1970.500000	2015.840690
std	1138.101709	3.341433
min	0.000000	1996.000000
25%	985.250000	2014.000000
50%	1970.500000	2016.000000
75%	2955.750000	2018.000000
max	3941.000000	2022.000000

- Data Sources and their formats
- Data Collection Phase.
    I. Collected the data from different websites such as carwale.com, cardekho.com & cartrade.com.
    II. Collected data like name, year, km, fuel, city etc.
    III. Saved the dataset as a csv file.
    IV. Data cleaning from excel and through python.

- Model Building Phase.
    I. Data Cleaning.
    II. EDA
    III. Data Pre-processing
    IV. Model Building
    V. Model Evaluation
    VI. Selecting the best model
    VII. Hyperparameter tuning

- Data Preprocessing Done

  EDA

Description

No null present

Data cleaning

Visualization

Encoding

- Hardware and Software Requirements and Tools Used

  Anaconda-navigator
  jupyter notebook
  matplotlib-inline==0.1.6
  numpy==1.23.2
  packaging==21.3
  pickleshare==0.7.5
  platformdirs==2.5.2
  prompt-toolkit==3.0.30
  pyparsing==3.0.9
  python-dateutil==2.8.2
  scikit-learn==1.1.2
  scipy==1.9.0
  sklearn==0.05

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)
  - EDA
  - Description
  - No null present
  - Data cleaning
  - Visualization
  - Encoding
  - Model Building
  - Select the best model
  - Hyper parameter tuning

- Testing of Identified Approaches (Algorithms)

  Algorithms used for the training and testing:
  - RandomForest Regressor.
  - AdaBoost Regressor.
  - GradientBoosting Regressor.
  - Super Vector Regressor.
  - Kneighbors Regressor.

- Run and Evaluate selected models

- RandomForest Regressor.

```
rf.fit(x_train,y_train)
score(rf, x_train,x_test,y_train,y_test,train = True)
score(rf, x_train,x_test,y_train,y_test,train = False)
```

```
 ----- Train Result -----

R2 Score: 0.9793886664446027

 ----- Test Result -----

R2 Score: 0.8023402936439155
Mean Absolute Error: 228386.78317523902
```

- AdaBoost Regressor.

```
ada.fit(x_train,y_train)
score(ada, x_train,x_test,y_train,y_test,train = True)
score(ada, x_train,x_test,y_train,y_test,train = False)
```

```
 ----- Train Result -----

R2 Score: -0.5244165804043754

 ----- Test Result -----

R2 Score: -1.055646062073242
Mean Absolute Error: 1552650.8041604715
```

- GradientBoosting Regressor.

```
gb.fit(x_train,y_train)
score(gb, x_train,x_test,y_train,y_test,train = True)
score(gb, x_train,x_test,y_train,y_test,train = False)
```

```
 ----- Train Result -----

R2 Score: 0.8653486749511684

 ----- Test Result -----

R2 Score: 0.760550538301414
Mean Absolute Error: 295744.3045491251
```

- SupperVector Regressor.

```
svr.fit(x_train,y_train)
score(svr, x_train,x_test,y_train,y_test,train = True)
score(svr, x_train,x_test,y_train,y_test,train = False)
```

```
utput; double click to hide sult -----

R2 Score: -0.08581709576846541

 ----- Test Result -----

R2 Score: -0.09063739749513333
Mean Absolute Error: 568163.0950604408
```

- KNeighbors Regressor.

```
knn.fit(x_train,y_train)
score(knn, x_train,x_test,y_train,y_test,train = True)
score(knn, x_train,x_test,y_train,y_test,train = False)
```

```
 ----- Train Result -----

R2 Score: 0.6191224987301477

 ----- Test Result -----

R2 Score: 0.3040107458056932
Mean Absolute Error: 442420.1900608519
```

- Interpretation of the Results

RandomForest Regressor is giving the best score.

CONCLUSION

- Key Findings and Conclusions of the Study

```
grid = GridSearchCV(rf, param_grid = param)
grid.fit(x_train,y_train)
print('Best Params = ',grid.best_params_)
```

```
Best Params =  {'max_depth': None, 'max_features': 0.5, 'n_estimators': 200}
```

```
params = {"min_samples_split":[ 2, 5, 10],
          "min_samples_leaf":[1, 2, 3, 5]}
```

```
grid = GridSearchCV(rf, param_grid = params)
grid.fit(x_train,y_train)
print('Best Params = ',grid.best_params_)
```

```
Best Params =  {'min_samples_leaf': 1, 'min_samples_split': 2}
```

```
f_hyp = RandomForestRegressor(max_depth = None, max_features = 0.5, min_samples_leaf = 1, min_samples_split = 2, n_e
```

```
rf_hyp.fit(x_train,y_train)
score(rf_hyp, x_train,x_test,y_train,y_test,train = True)
score(rf_hyp, x_train,x_test,y_train,y_test,train = False)
```

```
 ----- Train Result -----

R2 Score: 0.9783061196967381

 ----- Test Result -----

R2 Score: 0.8283762349117869
Mean Absolute Error: 226478.25855438437
```

```
   Post tuning score, is better than the default parameter, so saving the post tuning parameters for the model.
```

After done hyper parameter tuning is the results are better and the best parameter for RandomForest Regressor.