CAR PRICE PREDICTION

Submitted by:

Kishan Verma

ACKNOWLEDGMENT

I would like to thank Flip Robo Technologies for providing me with the opportunity to work on this project from which I have learned a lot. I am also grateful to Mr. Shubham Yadav for his constant guidance. Some of the reference sources are as follows:

- Stack Overflow
- Medium.com
- scikit-learn.org
- Python official documentation

Contents

INTRODUCTION

BUSINESS PROBLEM FRAMING

A car price prediction has been a high interest research area, as it requires noticeable effort and knowledge of the field expert. Accurate car price prediction involves expert knowledge, because price usually depends on many distinctive features and factors.

Typically, most significant ones are brand and model, age, horsepower and mileage. The fuel type used in the car as well as fuel consumption per mile highly affect price of a car due to a frequent changes in the price of a fuel. 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

Predicting the price of used cars in both an important and interesting problem. According to data obtained from the National Transport Authority In many developed countries, it is common to lease a car rather than buying it outright. A lease is a binding contract between a buyer and a seller (or a third party –usually a bank, insurance firm or other financial institutions) in which the buyer must pay fixed instalments for a pre-defined number of months/years to the seller/financier.

Predicting the resale value of a car is not a simple task. It is trite knowledge that the value of used cars depends on a number of factors. The most important ones are usually the age of the car, its make (and model), the origin of the car (the original country of the manufacturer), its mileage (the number of kilometres it has run) and its horsepower.

REVIEW OF LITERATURE

The number of cars registered between 2003 and 2013 has witnessed a spectacular increase of 234%. To build a model for predicting the price of used cars in India, we applied eight machine learning techniques (Linear Regression, Random Forest Regression, Bagging Regressor, XGB Regressor, ADA Boost Regressor, Regularization (Lasso), Regularization (Ridge), Gradient Boosting Regressor).

The data used for the prediction was collected from the web portal Cars24.com using web scraper that was written in python programming language. Respective performances of different algorithms were then compared to find one that best suits the available data set.

MOTIVATION FOR THE PROBLEM UNDERTAKEN

In this problem, we investigate the application of supervised machine learning techniques to predict the price of used cars in India. The predictions are based on historical data collected from websites. Different techniques like (Linear Regression, Random Forest Regression, Bagging Regressor, XGB Regressor, ADA Boost Regressor, Regularization (Lasso), Regularization (Ridge), Gradient Boosting Regressor) have been used to make the predictions.

The predictions are then evaluated and compared in order to find those which provide the best performances. All the four methods provided comparable performance. In the future, we intend to use more sophisticated algorithms to make the predictions

ANALYTICAL PROBLEM FRAMING

MATHEMATICAL/ ANALYTICAL MODELING OF THE PROBLEM

As an aspiring Data Scientist, the goal is to create a model that will predict the used car prices with the available independent variables. This model will then be used by the management to understand how exactly the prices vary with the variables.

The company can accordingly manipulate the strategy of the firm and concentrate on areas that will yield high returns. Further, the model will be a good way for the management to understand the pricing dynamics of a new market. Based on all the independent variables the model needs to predict dependent variable (Price) A total of 8 regression models were used in order to predict the target variable Price.

DATA SOURCES AND THEIR FORMATS

The sample data was scrapped from Cars24.com by using selenium web-driver in python programming language. A total of three data types in the data set Int and Object. The data collected was saved in .CSV (comma-separated values) format. The data set has a total of 10 columns:

- Name: Name of the car
- Variant: Variant of the car
- Fuel: Fuel type (Petrol or Diesel)
- Kilometers: Kilometers driven per car
- Purchase date: Original purchase year of the car
- Owners: Number of previous owners
- Transmission: Transmission type (Manual or Automatic)
- Accidental: Whether the car is accidental or not
- Price: Car Price in Rs
- Location: Location of car

Name	Variant	Fuel	Kilometers	Purchase Date	Owners	Transmission	Accidential	Price	Location
2014 Maruti Alto 800 VXI MANUAL	VXI MANUAL	-	12,535 km	-	-	-	-	₹2,91,999	Delhi
2014 Hyundai Grand i10 ASTA 1.2 AT VTVT AUTOMATIC	ASTA 1.2 AT VTVT AUTOMATIC	Petrol	26,779 km	-	1st Owner	HR-26-x-xxxx	Non-Accidental	₹4,20,000	Delhi
2020 Maruti Alto LXI MANUAL	LXI MANUAL	-	2,723 km	-	-	-	-	₹3,72,099	Delhi
2011 Maruti Alto K10 VXI MANUAL	VXI MANUAL	Petrol	20,354 km	April 2011	1st Owner	MANUAL	Non-Accidental	₹2,18,499	Delhi
2021 Renault Kwid 1.0 RXT Opt AT AUTOMATIC	1.0 RXT Opt AT AUTOMATIC	-	659 km	-	-	-	-	₹4,76,000	Delhi

DATA PREPROCESSING DONE

- We can drop the Accidental column as it only has one value and it will not have any significant affect on the target variable.
- We have created two new columns Brand and Car_name from Name column.
- We can drop the name column now as we have already created two new columns from it.
- Now we need to extract Transmission type(Manual or Automatic) from Variant column to create a new column name Transmission
- We need to remove the string km from Kilometres column as we only need the numerical data.
- We can remove the month name and only keep the year data.
- Removing the owner text from owner column as we only need the numerical data (1, 2 or more).
- We need to remove the rupee symbol from the Price column.
- We can drop the null values as there are only 14 null values, so it won't have a significant effect on the target variable.
- All the data-types are Objects so we need to convert (Kilometres, Purchase_date, Owners and Price to Float).
- We need to handle outliers and transformations. Fortunately Except Kilometres we don't have to worry about other independent variables as they are categorical in nature. Using box-plot to visually detect outliers.
- Checking the distribution using Q-Q plot, we can clearly see that the distribution is right-skewed so we need to reduce the skewness and try to make the plot normally distributed.
- Checking the Q-Q plot after outlier removal. We can clearly observe that the plots looks more like normally distributed and the skewness has been reduced significantly.
- We need to encode the categorical data. Using Target encoding on Car_name and Variant columns.
- Using pandas get_dummies method on rest of the categorical columns.

- Using Train Test Split on Data-frame.
- Scaling the data using StandardScalar() method.

HARDWARE AND SOFTWARE REQUIREMENTS AND TOOLS USED

HARDWARE:

HP Pavilion X360

SOFTWARE:

Jupyter Notebook (Anaconda 3) – Python 3

Microsoft Office 365 Package

Chrome Web Browser

LIBRARIES USED:

```
#Importing  libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

```
#Importing required libraries
import re
import string
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import SnowballStemmer, WordNetLemmatizer
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
```

MODEL/S DEVELOPMENT AND EVALUATION

TESTING OF IDENTIFIED APPROACHES (ALGORITHMS)

Model Building

```
#Importing train_test_split, Logistic Regression and accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

```
def max_acc_score(reg,x,y):
    max_score=0
    for r_state in range (42,101):
        x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=r_state,test_size=0.20)
        reg.fit(x_train,y_train)
        pred=reg.predict(x_test)
        acc_score=accuracy_score(y_test,pred)
        #print("The accuracy score at r_state", r_state, "is", acc_score)
        if acc_score>max_score:
            max_score=acc_score
            final_r_state=r_state
    print("The maximum accuracy score", max_score, "is achieved at", final_r_state)
    return max_score
```

```
LR=LogisticRegression()
max_acc_score(LR,x,y)
```

```
The maximum accuracy score 0.5425839047275351 is achieved at 52
```

```
50]: 0.5425839047275351
```

RUN AND EVALUATE SELECTED MODELS

```
#Importing required modules and metrices
from sklearn.metrics import confusion_matrix,classification_report
from sklearn.model_selection import cross_val_score
```

```

#Making a for loop and calling the algorithm one by one and save data to respective model using append function
Model=[]
score=[]
cvs=[]
rocscore=[]
for name,model in models:
    print('****************************',name,'*************************')
    print('\n')
    Model.append(name)
    model.fit(x_train,y_train)
    print(model)
    pre=model.predict(x_test)
    print('\n')
    AS=accuracy_score(y_test,pre)
    print('accuracy_score: ',AS)
    score.append(AS*100)
    print('\n')
    sc=cross_val_score(model,x,y,cv=5,scoring='accuracy').mean()
    print('cross_val_score: ',sc)
    cvs.append(sc*100)
    print('\n')
    print('Classification report:\n ')
    print(classification_report(y_test,pre))
    print('\n')
    print('Confusion matrix: \n')
    cm=confusion_matrix(y_test,pre)
    print(cm)
    print('\n\n\n')
```

```

#Finalizing the result
result=pd.DataFrame({'Model':Model, 'Accuracy_score': score,'Cross_val_score':cvs})
result
```

7]:

	Model	Accuracy_score	Cross_val_score
0	Logistic Regression	54.258390	45.324338
1	MultinomialNB	53.464453	45.046441
2	DecisionTreeClassifier	52.580296	44.754201
3	KNeighborsClassifier	37.026344	32.566944
4	RandomForestClassifier	58.444605	49.182286
5	AdaBoostClassifier	46.210754	40.434344
6	GradientBoostingClassifier	50.469145	42.577998

We can see that RandomForest Classifier is giving us the best results, therefore we will be tuning this model for further prediction.

KEY METRICS FOR SUCCESS IN SOLVING PROBLEM UNDER CONSIDERATION

The key metrics used here were accuracy_score, cross_val_score, classification report, and confusion matrix. We tried to find out the best parameters and also to increase our scores by using Hyperparameter Tuning and we will be using GridSearchCV method.

1. Cross Validation:

Cross-validation helps to find out the over fitting and under fitting of the model. In the cross validation the model is made to run on different subsets of the dataset which will get multiple measures of the model. If we take 5 folds, the data will be divided into 5 pieces where each part being 20% of full dataset. While running the Cross-validation the 1st part (20%) of the 5 parts will be kept out as a holdout set for validation and everything else is used for training data. This way we will get the first estimate of the model quality of the dataset. In the similar way further iterations are made for the second 20% of the dataset is held as a holdout set and remaining 4 parts are used for training data during process. This way we will get the second estimate of the model quality of the dataset. These steps are repeated during the cross-validation process to get the remaining estimate of the model quality.

2. Hyperparameter Tuning:

There is a list of different machine learning models. They all are different in some way or the other, but what makes them different is nothing but input parameters for the model. These input parameters are named as Hyperparameters. These hyperparameters will define the architecture of the model, and the best part about these is that you get a choice to select these for your model. You must select from a specific list of hyperparameters for a given model as it varies from model to model. We are not aware of optimal values for hyperparameters which would generate the best model output. So, what we tell the model is to explore and select the optimal model architecture automatically. This selection procedure for hyperparameter is known as Hyperparameter Tuning.

We can do tuning by using GridSearchCV. GridSearchCV is a function that comes in Scikit-learn (or SK-learn) model selection package. An important point here to note is that we need to have Scikit-learn library installed on the computer. This function helps to loop through predefined hyperparameters and fit your estimator (model) on your training set. So, in the end, we can select the best parameters from the listed hyperparameters.

CONCLUSION

In this project, eight different machine learning techniques have been used to forecast the price of used cars in Indian market. The first step I took, was to visualize the distribution of each feature and its effect on the Price (dependent variable). From the analysis, I conclude that some of the most useful features for predictions were "Car Name", "Variant", "Purchase Year", "Kilometers". The Gradient Boosting Regression Algorithm proved to be the best model for regression based on the Cross-Validation scores. An accuracy (r2 score) of 0.96 % was achieved by hyper parametric tuning of the model.

THANK YOU