Car Price Prediction with Machine Learning

AMAN KHARWAL / AUGUST 4, 2021 / MACHINE LEARNING

The price of a car depends on a lot of factors like the goodwill of the brand of the car, features of the car, horsepower and the mileage it gives and many more. Car price prediction is one of the major research areas in machine learning. So if you want to learn how to train a car price prediction model then this article is for you. In this article, I will take you through how to train a car price prediction model with machine learning using Python.

Car Price Prediction with Machine Learning

One of the main areas of research in machine learning is the prediction of the price of cars. It is based on **finance** and the **marketing** domain. It is a major research topic in machine learning because the price of a car depends on many factors. Some of the factors that contribute a lot to the price of a car are:

1. Brand
2. Model
3. Horsepower
4. Mileage

5. Safety Features
6. GPS and many more.

If one ignores the brand of the car, a car manufacturer primarily fixes the price of a car based on the features it can offer a customer. Later, the brand may raise the price depending on its goodwill, but the most important factors are what features a car gives you to add value to your life. So, in the section below, I will walk you through the task of training a car price prediction model with machine learning using the Python programming language.

Car Price Prediction Model using Python

The dataset I'm using here to train a car price prediction model was downloaded from Kaggle. It contains data about all the main features that contribute to the price of a car. So let's start this task by importing the necessary Python libraries and the dataset:

Dataset

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor

data = pd.read_csv("CarPrice.csv")
data.head()
```

car price.py hosted with ❤ by GitHub view raw

```
   car_ID  symboling                   CarName  ... citympg  
highwaympg    price
0       1          3        alfa-romero giulia  ...      21  
27  13495.0
1       2          3       alfa-romero stelvio  ...      21  
27  16500.0
2       3          1  alfa-romero Quadrifoglio  ...      19  
26  16500.0
3       4          2               audi 100 ls  ...      24  
30  13950.0
4       5          2                audi 100ls  ...      18  
22  17450.0
```

```
[5 rows x 26 columns]
```

There are 26 columns in this dataset, so it is very important to check whether or not this dataset contains null values before going any further:


```
data.isnull().sum()
```

```
car_ID              0
symboling           0
CarName             0
fueltype            0
aspiration          0
doornumber          0
carbody             0
drivewheel          0
enginelocation      0
wheelbase           0
carlength           0
carwidth            0
carheight           0
curbweight          0
enginetype          0
cylindernumber      0
enginesize          0
fuelsystem          0
boreratio           0
stroke              0
compressionratio    0
horsepower          0
peakrpm             0
citympg             0
highwaympg          0
price               0
dtype: int64
```

So this dataset doesn't have any null values, now let's look at some of the other important insights to get an idea of what kind of data we're dealing with:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 26 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   car_ID            205 non-null    int64
 1   symboling         205 non-null    int64
 2   CarName           205 non-null    object
 3   fueltype          205 non-null    object
 4   aspiration        205 non-null    object
 5   doornumber        205 non-null    object
 6   carbody           205 non-null    object
 7   drivewheel        205 non-null    object
 8   enginelocation    205 non-null    object
 9   wheelbase         205 non-null    float64
 10  carlength         205 non-null    float64
 11  carwidth          205 non-null    float64
 12  carheight         205 non-null    float64
 13  curbweight        205 non-null    int64
 14  enginetype        205 non-null    object
 15  cylindernumber    205 non-null    object
 16  enginesize        205 non-null    int64
 17  fuelsystem        205 non-null    object
 18  boreratio         205 non-null    float64
 19  stroke            205 non-null    float64
 20  compressionratio  205 non-null    float64
 21  horsepower        205 non-null    int64
 22  peakrpm           205 non-null    int64
 23  citympg           205 non-null    int64
 24  highwaympg        205 non-null    int64
 25  price             205 non-null    float64
dtypes: float64(8), int64(8), object(10)
memory usage: 41.8+ KB
```

```
print(data.describe())
```

```
           car_ID   symboling   wheelbase  ...     citympg
highwaympg         price
count  205.000000  205.000000  205.000000  ...  205.000000
205.000000    205.000000
mean   103.000000    0.834146   98.756585  ...   25.219512
30.751220  13276.710571
std     59.322565    1.245307    6.021776  ...    6.542142
6.886443   7988.852332
min      1.000000   -2.000000   86.600000  ...   13.000000
16.000000   5118.000000
25%     52.000000    0.000000   94.500000  ...   19.000000
25.000000   7788.000000
50%    103.000000    1.000000   97.000000  ...   24.000000
30.000000  10295.000000
```

```
75%    154.000000    2.000000  102.400000  ...   30.000000
34.000000  16503.000000
max    205.000000    3.000000  120.900000  ...   49.000000
54.000000  45400.000000

[8 rows x 16 columns]
```

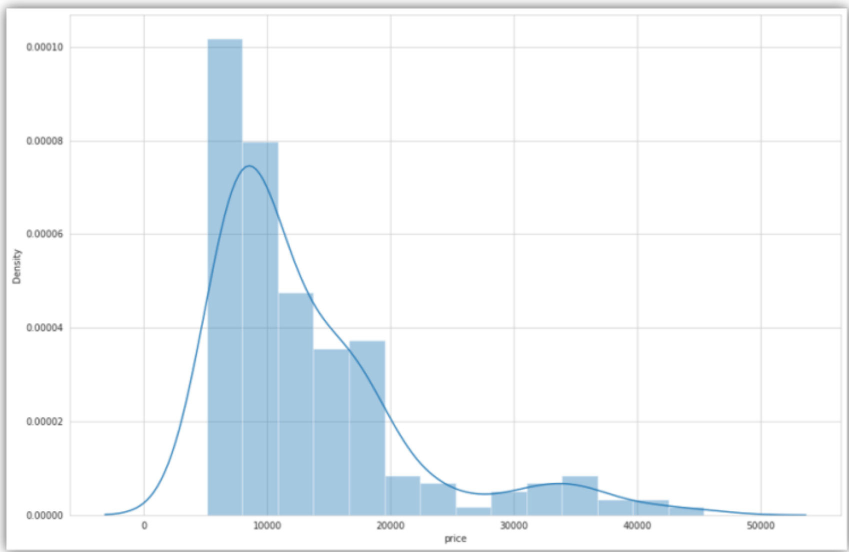
```
1 data.CarName.unique()
```

```
array(['alfa-romero giulia', 'alfa-romero stelvio',
       'alfa-romero Quadrifoglio', 'audi 100 ls', 'audi 100ls',
       'audi fox', 'audi 5000', 'audi 4000', 'audi 5000s (diesel)',
       'bmw 320i', 'bmw x1', 'bmw x3', 'bmw z4', 'bmw x4', 'bmw x5',
       'chevrolet impala', 'chevrolet monte carlo', 'chevrolet vega
2300',
       'dodge rampage', 'dodge challenger se', 'dodge d200',
       'dodge monaco (sw)', 'dodge colt hardtop', 'dodge colt (sw)',
       'dodge coronet custom', 'dodge dart custom',
       'dodge coronet custom (sw)', 'honda civic', 'honda civic
cvcc',
       'honda accord cvcc', 'honda accord lx', 'honda civic 1500 gl',
       'honda accord', 'honda civic 1300', 'honda prelude',
       'honda civic (auto)', 'isuzu MU-X', 'isuzu D-Max ',
       'isuzu D-Max V-Cross', 'jaguar xj', 'jaguar xf', 'jaguar xk',
       'maxda rx3', 'maxda glc deluxe', 'mazda rx2 coupe', 'mazda rx-
4',
       'mazda glc deluxe', 'mazda 626', 'mazda glc', 'mazda rx-7 gs',
       'mazda glc 4', 'mazda glc custom l', 'mazda glc custom',
       'buick electra 225 custom', 'buick century luxus (sw)',
       'buick century', 'buick skyhawk', 'buick opel isuzu deluxe',
       'buick skylark', 'buick century special',
       'buick regal sport coupe (turbo)', 'mercury cougar',
       'mitsubishi mirage', 'mitsubishi lancer', 'mitsubishi
outlander',
       'mitsubishi g4', 'mitsubishi mirage g4', 'mitsubishi montero',
       'mitsubishi pajero', 'Nissan versa', 'nissan gt-r', 'nissan
rogue',
       'nissan latio', 'nissan titan', 'nissan leaf', 'nissan juke',
       'nissan note', 'nissan clipper', 'nissan nv200', 'nissan
dayz',
       'nissan fuga', 'nissan otti', 'nissan teana', 'nissan kicks',
       'peugeot 504', 'peugeot 304', 'peugeot 504 (sw)', 'peugeot
604sl',
       'peugeot 505s turbo diesel', 'plymouth fury iii',
       'plymouth cricket', 'plymouth satellite custom (sw)',
       'plymouth fury gran sedan', 'plymouth valiant', 'plymouth
duster',
       'porsche macan', 'porcshce panamera', 'porsche cayenne',
       'porsche boxter', 'renault 12tl', 'renault 5 gtl', 'saab 99e',
       'saab 99le', 'saab 99gle', 'subaru', 'subaru dl', 'subaru
brz',
       'subaru baja', 'subaru r1', 'subaru r2', 'subaru trezia',
       'subaru tribeca', 'toyota corona mark ii', 'toyota corona',
```

```
       'toyota corolla 1200', 'toyota corona hardtop',
       'toyota corolla 1600 (sw)', 'toyota carina', 'toyota mark ii',
       'toyota corolla', 'toyota corolla liftback',
       'toyota celica gt liftback', 'toyota corolla tercel',
       'toyota corona liftback', 'toyota starlet', 'toyota tercel',
       'toyota cressida', 'toyota celica gt', 'toyouta tercel',
       'vokswagen rabbit', 'volkswagen 1131 deluxe sedan',
       'volkswagen model 111', 'volkswagen type 3', 'volkswagen 411
(sw)',
       'volkswagen super beetle', 'volkswagen dasher', 'vw dasher',
       'vw rabbit', 'volkswagen rabbit', 'volkswagen rabbit custom',
       'volvo 145e (sw)', 'volvo 144ea', 'volvo 244dl', 'volvo 245',
       'volvo 264gl', 'volvo diesel', 'volvo 246'], dtype=object)
```

The price column in this dataset is supposed to be the column whose values we need to predict. So let's see the distribution of the values of the price column:

```
sns.set_style("whitegrid")
plt.figure(figsize=(15, 10))
sns.distplot(data.price)
plt.show()
```

Now let's have a look at the correlation among all the features of this dataset:

```
1 print(data.corr())
```

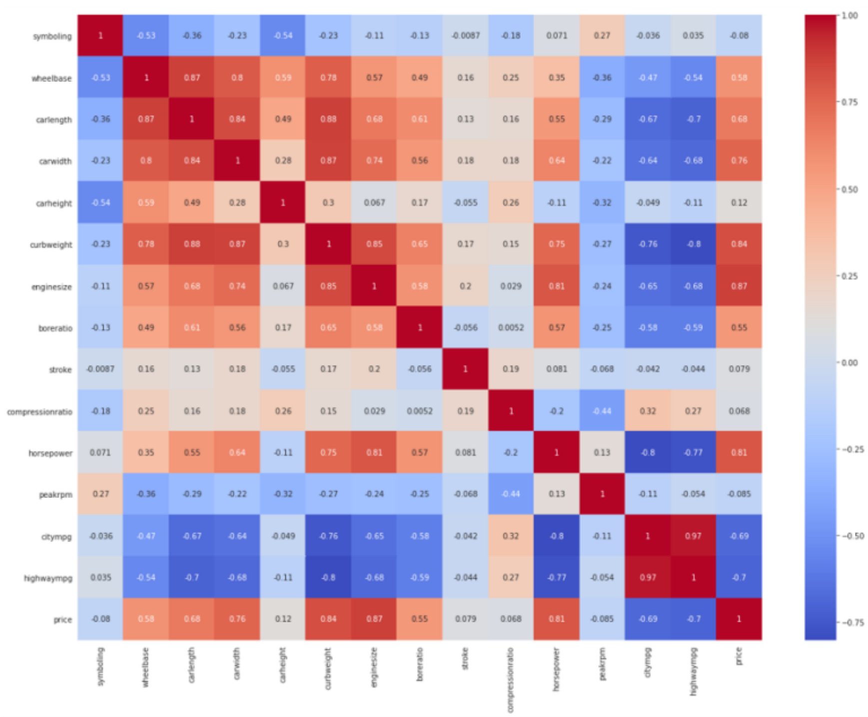
```
                   car_ID  symboling  ...  highwaympg     price
car_ID           1.000000  -0.151621  ...    0.011255 -0.109093
symboling       -0.151621   1.000000  ...    0.034606 -0.079978
wheelbase        0.129729  -0.531954  ...   -0.544082  0.577816
```

```
carlength         0.170636  -0.357612  ...  -0.704662  0.682920
carwidth          0.052387  -0.232919  ...  -0.677218  0.759325
carheight         0.255960  -0.541038  ...  -0.107358  0.119336
curbweight        0.071962  -0.227691  ...  -0.797465  0.835305
enginesize       -0.033930  -0.105790  ...  -0.677470  0.874145
boreratio         0.260064  -0.130051  ...  -0.587012  0.553173
stroke           -0.160824  -0.008735  ...  -0.043931  0.079443
compressionratio  0.150276  -0.178515  ...   0.265201  0.067984
horsepower       -0.015006   0.070873  ...  -0.770544  0.808139
peakrpm          -0.203789   0.273606  ...  -0.054275 -0.085267
citympg           0.015940  -0.035823  ...   0.971337 -0.685751
highwaympg        0.011255   0.034606  ...   1.000000 -0.697599
price            -0.109093  -0.079978  ...  -0.697599  1.000000

[16 rows x 16 columns]
```

```python
plt.figure(figsize=(20, 15))
correlations = data.corr()
sns.heatmap(correlations, cmap="coolwarm", annot=True)
plt.show()
```

car price3.py hosted with ❤ by GitHub view raw



Training a Car Price Prediction Model

I will use the decision tree regression algorithm to train a car price prediction model. So let's split the data into training and test sets and use the decision tree regression algorithm to train the model:

```
predict = "price"
data = data[["symboling", "wheelbase", "carlength",
             "carwidth", "carheight", "curbweight",
             "enginesize", "boreratio", "stroke",
             "compressionratio", "horsepower", "peakrpm",
             "citympg", "highwaympg", "price"]]
x = np.array(data.drop([predict], 1))
y = np.array(data[predict])

from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size=0.2)

from sklearn.tree import DecisionTreeRegressor
model = DecisionTreeRegressor()
model.fit(xtrain, ytrain)
predictions = model.predict(xtest)

from sklearn.metrics import mean_absolute_error
model.score(xtest, predictions)
```

car price4.py hosted with ❤ by GitHub view raw

```
1.0
```

The model gives 100% accuracy on the test set, which is excellent.

Summary

So this is how you can train a machine learning model for the task of predicting car prices by using the Python programming language. It is a major research topic in machine learning because the price of a car depends on many factors. I hope you liked this article on the task of training a model for predicting car prices with machine learning. Feel free to ask your valuable questions in the comments section below.

Aman Kharwal

I'm a writer and data scientist on a mission to educate others about the incredible power of data📈.

ARTICLES: 1353

 PREVIOUS

NEXT 

Recommended For You

Food Delivery Time Prediction using Python

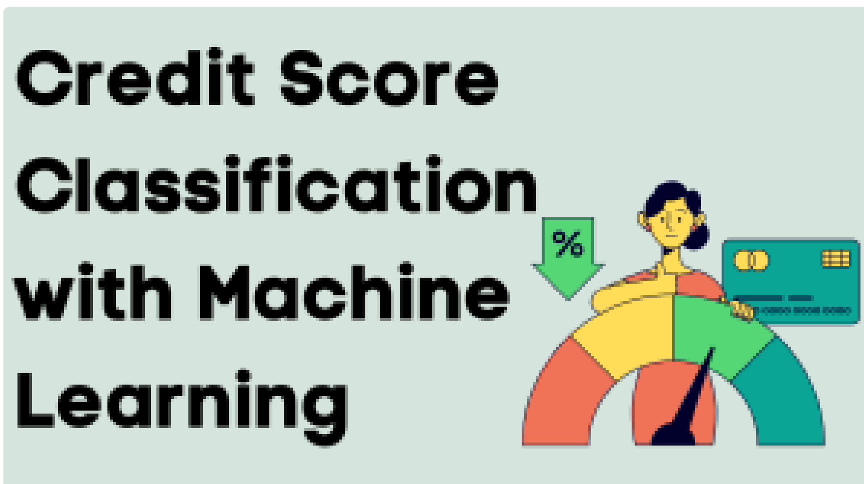
January 2, 2023

Job Recommendation System using Python

December 12, 2022

Machine Learning Roadmap

December 6, 2022 / 2 Comments



Credit Score Classification with Machine Learning

December 5, 2022

Leave a Reply

Enter your comment here...