Oasis Infobyte Data Science Internship

Task3

Car Price Prediction With Machine Learning

JONNADA HANIRUDH REDDY

In [1]:
```
#importing basic libraries

import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
#reading the dataset

df=pd.read_csv('car.csv')
```

In [3]:
```
df.sample(5)
```

Out[3]:

	name	company	year	Price	kms_driven	fuel_type
40	Renault Duster 85 PS RxE Diesel	Renault	2013	4,89,999	27,000 kms	Diesel
592	Tata Indigo Marina LS	Tata	2004	1,80,000	70,000 kms	Diesel
684	Maruti Suzuki Swift Dzire VXi 1.2 BS IV	Maruti	2009	2,10,000	59,000 kms	Petrol
258	Chevrolet Enjoy 1.4 LS 8 STR	Chevrolet	2013	2,45,000	55,000 kms	Diesel
846	Maruti Suzuki Alto 800	Maruti	2016	2,50,000	2,450 kms	Petrol

In [4]:
```
backup=df.copy()
```

In [5]:
```
df.describe()
```

Out[5]:

	name	company	year	Price	kms_driven	fuel_type
count	892	892	892	892	840	837
unique	525	48	61	274	258	3
top	Honda City	Maruti	2015	Ask For Price	45,000 kms	Petrol
freq	13	235	117	35	30	440

In [6]:
```
#cheking for Null Values

df.isna().sum()
```

Out[6]:
```
name           0
company        0
year           0
Price          0
kms_driven    52
fuel_type     55
dtype: int64
```

In [7]:
```
#removing the Null data

df=df[~df['fuel_type'].isna()]
```

In [8]:
```
#reseting the index after droping

df.reset_index(drop=True)
```

Out[8]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing XO eRLX Euro III	Hyundai	2007	80,000	45,000 kms	Petrol
1	Mahindra Jeep CL550 MDI	Mahindra	2006	4,25,000	40 kms	Diesel
2	Maruti Suzuki Alto 800 Vxi	Maruti	2018	Ask For Price	22,000 kms	Petrol
3	Hyundai Grand i10 Magna 1.2 Kappa VTVT	Hyundai	2014	3,25,000	28,000 kms	Petrol
4	Ford EcoSport Titanium 1.5L TDCi	Ford	2014	5,75,000	36,000 kms	Diesel
...	...	...	...	...	...	...
832	Maruti Suzuki Ritz VXI ABS	Maruti	2011	2,70,000	50,000 kms	Petrol
833	Tata Indica V2 DLE BS III	Tata	2009	1,10,000	30,000 kms	Diesel
834	Toyota Corolla Altis	Toyota	2009	3,00,000	1,32,000 kms	Petrol
835	Tata Zest XM Diesel	Tata	2018	2,60,000	27,000 kms	Diesel
836	Mahindra Quanto C8	Mahindra	2013	3,90,000	40,000 kms	Diesel

837 rows × 6 columns

In [9]:
```
#Removing non year from Year Column

df=df[df['year'].str.isnumeric()]
```

In [10]:
```
# making year as numeric

df['year']=df['year'].astype(int)
```

In [11]:
```
# changing "Ask for Price" rather removing them from price column

df=df[df['Price']!="Ask For Price"]
```

In [12]:
```
# making price as int type

df['Price']=df['Price'].str.replace(',','').astype(int)
```

In [13]:
```
# removing "," & "kms" from kms-driven column

df['kms_driven']=df['kms_driven'].str.split(' ').str.get(0).str.replace(',','')
```

In [14]:
```
# making kms_driven column as int type

df=df[df['kms_driven'].str.isnumeric()]
df['kms_driven']=df['kms_driven'].astype(int)
```

In [15]:
```
# slicing the car name to 3 words only

df['name']=df['name'].str.split(' ').str.slice(0,3).str.join(' ')
```

In [16]:
```
df.head()
```

Out[16]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
3	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
4	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
6	Ford Figo	Ford	2012	175000	41000	Diesel

In [17]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 816 entries, 0 to 889
Data columns (total 6 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   name        816 non-null    object
 1   company     816 non-null    object
 2   year        816 non-null    int32 
 3   Price       816 non-null    int32 
 4   kms_driven  816 non-null    int32 
 5   fuel_type   816 non-null    object
dtypes: int32(3), object(3)
memory usage: 35.1+ KB
```

In [18]:
```
#restetting the indexes again

df.reset_index(drop=True)
```

Out[18]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
811	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
812	Tata Indica V2	Tata	2009	110000	30000	Diesel
813	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
814	Tata Zest XM	Tata	2018	260000	27000	Diesel
815	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

816 rows × 6 columns

In [19]:
```
#statistical analysis

df.describe()
```

Out[19]:

	year	Price	kms_driven
count	816.000000	8.160000e+02	816.000000
mean	2012.444853	4.117176e+05	46275.531863
std	4.002992	4.751844e+05	34297.428044
min	1995.000000	3.000000e+04	0.000000
25%	2010.000000	1.750000e+05	27000.000000
50%	2013.000000	2.999990e+05	41000.000000
75%	2015.000000	4.912500e+05	56818.500000
max	2019.000000	8.500003e+06	400000.000000

In [20]:
```
#outlier removal

df=df[df['Price']<6e6].reset_index(drop=True)
df
```

Out[20]:

	name	company	year	Price	kms_driven	fuel_type
0	Hyundai Santro Xing	Hyundai	2007	80000	45000	Petrol
1	Mahindra Jeep CL550	Mahindra	2006	425000	40	Diesel
2	Hyundai Grand i10	Hyundai	2014	325000	28000	Petrol
3	Ford EcoSport Titanium	Ford	2014	575000	36000	Diesel
4	Ford Figo	Ford	2012	175000	41000	Diesel
...	...	...	...	...	...	...
810	Maruti Suzuki Ritz	Maruti	2011	270000	50000	Petrol
811	Tata Indica V2	Tata	2009	110000	30000	Diesel
812	Toyota Corolla Altis	Toyota	2009	300000	132000	Petrol
813	Tata Zest XM	Tata	2018	260000	27000	Diesel
814	Mahindra Quanto C8	Mahindra	2013	390000	40000	Diesel

815 rows × 6 columns

In [21]:
```
#make separate input and output data colums

x=df.drop(columns='Price')
y=df['Price']
```

In [22]:
```
# differentiate training and testing data

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

In [23]:
```
y_train.head()
```

Out[23]:
```
732     265000
426     900000
425     220000
82     1200000
310     649999
Name: Price, dtype: int32
```

In [24]:
```
x_train.head()
```

Out[24]:

	name	company	year	kms_driven	fuel_type
732	Maruti Suzuki Alto	Maruti	2015	14000	Petrol
426	Mahindra Scorpio S10	Mahindra	2015	97200	Diesel
425	Mahindra Bolero DI	Mahindra	2012	59466	Diesel
82	Volkswagen Vento Highline	Volkswagen	2019	3600	Diesel
310	Mahindra XUV500	Mahindra	2014	47000	Diesel

In [25]:
```
#'name','company','fuel_type' columns are categorical so we need to make them neumeric

from sklearn.preprocessing import OneHotEncoder, StandardScaler
df1=OneHotEncoder()
df1.fit(df[['name','company','fuel_type']])
```

Out[25]:
```
▾ OneHotEncoder

OneHotEncoder()
```

In [26]:
```
from sklearn.compose import make_column_transformer
column_trans= make_column_transformer((OneHotEncoder(categories=df1.categories_),['name','company','fuel_type']),
                                      remainder='passthrough')
```

In [27]:
```
# applying necessities for models
```

In [30]:
```
!pip install xgboost
from xgboost import XGBRegressor
from sklearn.metrics import mean_squared_error,r2_score

import math
from sklearn.pipeline import make_pipeline
```

```
Collecting xgboost
  Downloading xgboost-1.7.6-py3-none-win_amd64.whl (70.9 MB)
     ---------------------------------------- 70.9/70.9 MB 5.0 MB/s eta 0:00:00
Requirement already satisfied: scipy in c:\users\hanirudh reddy\anaconda3\lib\site-packages (from xgboost) (1.10.0)
Requirement already satisfied: numpy in c:\users\hanirudh reddy\anaconda3\lib\site-packages (from xgboost) (1.23.5)
Installing collected packages: xgboost
Successfully installed xgboost-1.7.6
```

In [32]:
```
#XGB regressor

xgbr=XGBRegressor()
pipe=make_pipeline(column_trans,xgbr)
pipe.fit(x_train,y_train)
y_pred_xgbr=pipe.predict(x_test)
r2=r2_score(y_test,y_pred_xgbr)
r2
```

Out[32]:
```
0.6032891885643739
```

In [33]:
```
#let's run XGBRegression for 50 times and store the highest scores

score=[]
for i in range(50):
    x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=i)
    xgbr=XGBRegressor()
    pipe=make_pipeline(column_trans,xgbr)
    pipe.fit(x_train,y_train)
    y_pred=pipe.predict(x_test)
    score.append(r2_score(y_test,y_pred))
```

In [34]:
```
score[np.argmax(score)]
```

Out[34]:
```
0.887748087736036
```

In [35]:
```
#saving according to the best item

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=np.argmax(score))
xgbr=XGBRegressor()
pipe=make_pipeline(column_trans,xgbr)
pipe.fit(x_train,y_train)
y_pred=pipe.predict(x_test)
r2_score(y_test,y_pred)
```

Out[35]:
```
0.887748087736036
```

In [36]:
```
# testing our model prediction

pipe.predict(pd.DataFrame([['Maruti Suzuki Swift','Maruti',2019,100,'Petrol']],
            columns=['name','company','year','kms_driven','fuel_type']))

# original ans is 530000
```

Out[36]:
```
array([538705.3], dtype=float32)
```

In [37]:
```
pipe.predict(pd.DataFrame([['Ford Figo','Ford',2014,175000,'Diesel']],
            columns=['name','company','year','kms_driven','fuel_type']))

# original ans is 400000
```

Out[37]:
```
array([372975.78], dtype=float32)
```

In [38]:
```
# Saving the model

import pickle
```

In [39]:
```
pickle.dump(pipe,open('car_price.pkl','wb'))
```

In [ ]:
```

```