CAR PRICE PREDICTION WITH 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

Import required Modules

In [1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

%matplotlib inline
```

Read CSV file

In [2]:
```
car = pd.read_csv('CarPrice_Assignment.csv')
car
```

Out[2]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsystem	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg	highwaympg	price
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	13495
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	16500
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47	9.0	154	5000	19	26	16500
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40	10.0	102	5500	24	30	13950
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40	8.0	115	5500	18	22	17450
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
200	201	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114	5400	23	28	16845
201	202	-1	volvo 144ea	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	8.7	160	5300	19	25	19045
202	203	-1	volvo 244dl	gas	std	four	sedan	rwd	front	109.1	...	173	mpfi	3.58	2.87	8.8	134	5500	18	23	21485
203	204	-1	volvo 246	diesel	turbo	four	sedan	rwd	front	109.1	...	145	idi	3.01	3.40	23.0	106	4800	26	27	22470
204	205	-1	volvo 264gl	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114	5400	19	25	22625

205 rows × 26 columns

In [3]:
```
car.head()
```

Out[3]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsystem	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg	highwaympg	price
0	1	3	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	13495
1	2	3	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	...	130	mpfi	3.47	2.68	9.0	111	5000	21	27	16500
2	3	1	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	...	152	mpfi	2.68	3.47	9.0	154	5000	19	26	16500
3	4	2	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	...	109	mpfi	3.19	3.40	10.0	102	5500	24	30	13950
4	5	2	audi 100ls	gas	std	four	sedan	4wd	front	99.4	...	136	mpfi	3.19	3.40	8.0	115	5500	18	22	17450

5 rows × 26 columns

In [4]:
```
car.tail()
```

Out[4]:

	car_ID	symboling	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	...	enginesize	fuelsystem	boreratio	stroke	compressionratio	horsepower	peakrpm	citympg	highwaympg	price
200	201	-1	volvo 145e (sw)	gas	std	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114	5400	23	28	16845
201	202	-1	volvo 144ea	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	8.7	160	5300	19	25	19045
202	203	-1	volvo 244dl	gas	std	four	sedan	rwd	front	109.1	...	173	mpfi	3.58	2.87	8.8	134	5500	18	23	21485
203	204	-1	volvo 246	diesel	turbo	four	sedan	rwd	front	109.1	...	145	idi	3.01	3.40	23.0	106	4800	26	27	22470
204	205	-1	volvo 264gl	gas	turbo	four	sedan	rwd	front	109.1	...	141	mpfi	3.78	3.15	9.5	114	5400	19	25	22625

5 rows × 26 columns

In [5]:
```
car.shape
```

Out[5]:
```
(205, 26)
```

In [11]:
```
car.size
```

Out[11]:
```
5330
```

Data Cleaning

In [13]:
```
car.drop(columns=["car_ID", "symboling", "carheight", "stroke", "compressionratio", "peakrpm"], inplace=True)
car.head()
```

Out[13]:

	CarName	fueltype	aspiration	doornumber	carbody	drivewheel	enginelocation	wheelbase	carlength	carwidth	curbweight	enginetype	cylindernumber	enginesize	fuelsystem	boreratio	horsepower	citympg	highwaympg	price
0	alfa-romero giulia	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	2548	dohc	four	130	mpfi	3.47	111	21	27	13495
1	alfa-romero stelvio	gas	std	two	convertible	rwd	front	88.6	168.8	64.1	2548	dohc	four	130	mpfi	3.47	111	21	27	16500
2	alfa-romero Quadrifoglio	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	2823	ohcv	six	152	mpfi	2.68	154	19	26	16500
3	audi 100 ls	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	2337	ohc	four	109	mpfi	3.19	102	24	30	13950
4	audi 100ls	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	2824	ohc	five	136	mpfi	3.19	115	18	22	17450

In [15]:
```
car.shape
```

Out[15]:
```
(205, 20)
```

In [16]:
```
car.isnull().sum() #glad, no null values
```

Out[16]:
```
CarName           0
fueltype          0
aspiration        0
doornumber        0
carbody           0
drivewheel        0
enginelocation    0
wheelbase         0
carlength         0
carwidth          0
curbweight        0
enginetype        0
cylindernumber    0
enginesize        0
fuelsystem        0
boreratio         0
horsepower        0
citympg           0
highwaympg        0
price             0
dtype: int64
```

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

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 205 entries, 0 to 204
Data columns (total 20 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   CarName         205 non-null    object
 1   fueltype        205 non-null    object
 2   aspiration      205 non-null    object
 3   doornumber      205 non-null    object
 4   carbody         205 non-null    object
 5   drivewheel      205 non-null    object
 6   enginelocation  205 non-null    object
 7   wheelbase       205 non-null    float64
 8   carlength       205 non-null    float64
 9   carwidth        205 non-null    float64
 10  curbweight      205 non-null    int64
 11  enginetype      205 non-null    object
 12  cylindernumber  205 non-null    object
 13  enginesize      205 non-null    int64
 14  fuelsystem      205 non-null    object
 15  boreratio       205 non-null    float64
 16  horsepower      205 non-null    int64
 17  citympg         205 non-null    int64
 18  highwaympg      205 non-null    int64
 19  price           205 non-null    float64
dtypes: float64(5), int64(5), object(10)
memory usage: 32.2+ KB
```

In [18]:
```
car.describe()
```

Out[18]:

	wheelbase	carlength	carwidth	curbweight	enginesize	boreratio	horsepower	citympg	highwaympg	price
count	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000	205.000000
mean	98.756585	174.049268	65.907805	2555.565854	126.907317	3.329756	104.117073	25.219512	30.751220	13276.710571
std	6.021776	12.337289	2.145204	520.680204	41.642693	0.270844	39.544167	6.542142	6.886443	7999.852232
min	86.600000	141.100000	60.300000	1488.000000	61.000000	2.540000	48.000000	13.000000	16.000000	5118.000000
25%	94.500000	166.300000	64.100000	2145.000000	97.000000	3.150000	70.000000	19.000000	25.000000	7788.000000
50%	97.000000	173.200000	65.500000	2414.000000	120.000000	3.310000	95.000000	24.000000	30.000000	10295.000000
75%	102.400000	183.100000	66.900000	2935.000000	141.000000	3.588000	116.000000	30.000000	34.000000	16503.000000
max	120.900000	208.100000	72.300000	4066.000000	326.000000	3.940000	288.000000	49.000000	54.000000	45400.000000

In [19]:
```
# car_features = []
for i in [1, 2, 3, 4, 5, 6, 11, 12, 14]:
    print(f"{car.columns[i]} \t {car[car.columns[i]].unique()}")
    cat_features.append(car.columns[i])

cat_features.append("CarName")
```

```
fueltype 	 ['gas' 'diesel']
aspiration 	 ['std' 'turbo']
doornumber 	 ['two' 'four']
carbody 	 ['convertible' 'hatchback' 'sedan' 'wagon' 'hardtop']
drivewheel 	 ['rwd' 'fwd' '4wd']
enginelocation 	 ['front' 'rear']
enginetype 	 ['dohc' 'ohcv' 'ohc' 'l' 'rotor' 'ohcf' 'dohcv']
cylindernumber 	 ['four' 'six' 'five' 'three' 'twelve' 'two' 'eight']
fuelsystem 	 ['mpfi' '2bbl' 'mfi' '1bbl' 'spfi' '4bbl' 'idi' 'spdi']
```

In [20]:
```
car["drivewheel"].value_counts()
```

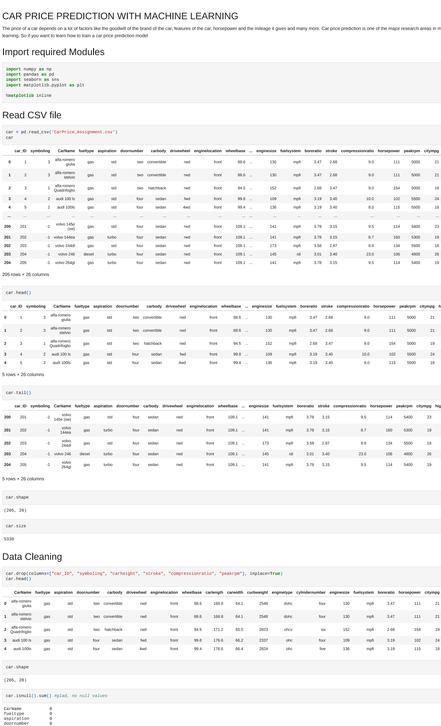
Out[20]:
```
fwd    120
rwd     76
4wd      9
Name: drivewheel, dtype: int64
```

In [21]:
```
car["drivewheel"] = car["drivewheel"].replace('4wd', 'fwd')
car["drivewheel"].value_counts()
```

Out[21]:
```
fwd    129
rwd     76
Name: drivewheel, dtype: int64
```

Heatmap

In [23]:
```
plt.figure(figsize=(12, 5))
corr = car.corr(), image="YlOrRd", annot=True, lw=0.5)
plt.show()
```



Training the Model

Importing required machine learning libraries for training the model

In [24]:
```
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
from sklearn.pipeline import make_pipeline
```

In [25]:
```
X = car.iloc[:, :18]
y = car.iloc[:, -1]

print(X.shape)
print(y.shape)
```

```
(205, 18)
(205,)
```

In [26]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)
```

OneHotEncoding

In [27]:
```
ohe = OneHotEncoder()
ohe.fit(X[cat_features])

trf1 = make_column_transformer(
    (OneHotEncoder(categories=ohe.categories_), cat_features),
    remainder='passthrough'
)
```

Linear Regression

In [28]:
```
from sklearn.linear_model import LinearRegression

lin_reg = LinearRegression()
pipe_1 = make_pipeline(trf1, lin_reg)

pipe_1.fit(X_train, y_train)
```

Out[28]:
```
Pipeline(steps=[('columntransformer',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('onehotencoder',
                                                  OneHotEncoder(categories=[array(['diesel', 'gas'], dtype=object),
                                                                            array(['std', 'turbo'], dtype=object),
                                                                            array(['four', 'two'], dtype=object),
                                                                            array(['convertible', 'hardtop', 'hatchback', 'sedan', 'wagon'],
                                                                                  dtype=object),
                                                                            array(['fwd', 'rwd'], dtype=object),
                                                                            array(['front', 'rear'], dtype=object),
                                                                            array(['dohc', 'dohcv', 'l', 'ohc', 'ohcf', 'ohcv',
                                                                                   'rotor'], dtype=object),
                                                                            array(['eight', 'five', 'four', 'six', 'three',
                                                                                   'twelve', 'two'], dtype=object),
                                                                            array(['1bbl', '2bbl', '4bbl', 'idi', 'mfi', 'mpfi',
                                                                                   'spdi', 'spfi'], dtype=object),
                                                                            array(['Nissan versa', 'alfa-romero giulia',
                                                                                   'alfa-romero stelvio',
                                                                                   'alfa-romero Quadrifoglio',
                                                                                   'audi 100 ls', 'audi 100ls',
                                                                                   'audi 4000', 'audi 5000',
                                                                                   'audi fox', 'bmw 320i', 'bmw 320...
                                                                                   'volkswagen 1131 deluxe sedan',
                                                                                   'volkswagen model 111',
                                                                                   'volkswagen rabbit',
                                                                                   'volkswagen rabbit custom',
                                                                                   'volkswagen super beetle',
                                                                                   'volvo 144ea', 'volvo 145e (sw)',
                                                                                   'volvo 244dl', 'volvo 245',
                                                                                   'volvo 246', 'volvo 264gl',
                                                                                   'volvo diesel', 'volvo turbo'],
                                                                                  dtype=object)]),
                                                  ['fueltype', 'aspiration',
                                                   'doornumber', 'carbody',
                                                   'drivewheel',
                                                   'enginelocation',
                                                   'wheelbase', 'carlength',
                                                   'carwidth', 'curbweight',
                                                   'enginetype',
                                                   'cylindernumber',
                                                   'enginesize',
                                                   'fuelsystem', 'CarName',
                                                   'boreratio', 'horsepower',
                                                   'citympg', 'highwaympg',
                                                   'price'])])),
                ('linearregression', LinearRegression())])
```

In [29]:
```
y_predLR = pipe_1.predict(X_test)
```

In [30]:
```
r2_LR = metrics.r2_score(y_test, y_predLR).round(4)
r2_LR
```

Out[30]:
```
0.619
```

In [31]:
```
RMSE_LR = metrics.mean_absolute_error(y_test, y_predLR).round(3)
MAE_LR = np.sqrt(metrics.mean_squared_error(y_test, y_predLR)).round(3)
print("MAE = {MAE_LR} \nRMSE = {RMSE_LR}")
```

```
MAE = 5484.144
RMSE = 3457.464
```

Decision Tree Regressor

In [32]:
```
from sklearn.tree import DecisionTreeRegressor

dt = DecisionTreeRegressor()
pipe_dt = make_pipeline(trf1, dt)

pipe_dt.fit(X_train, y_train)
```

Out[32]:
```
Pipeline(steps=[('columntransformer',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('onehotencoder',
                                                  OneHotEncoder(categories=[array(['diesel', 'gas'], dtype=object),
                                                                            array(['std', 'turbo'], dtype=object),
                                                                            array(['four', 'two'], dtype=object),
                                                                            array(['convertible', 'hardtop', 'hatchback', 'sedan', 'wagon'],
                                                                                  dtype=object),
                                                                            array(['fwd', 'rwd'], dtype=object),
                                                                            array(['front', 'rear'], dtype=object),
                                                                            array(['dohc', 'dohcv', 'l', 'ohc', 'ohcf', 'ohcv',
                                                                                   'rotor'], dtype=object),
                                                                            array(['eight', 'five', 'four', 'six', 'three',
                                                                                   'twelve', 'two'], dtype=object),
                                                                            array(['1bbl', '2bbl', '4bbl', 'idi', 'mfi', 'mpfi',
                                                                                   'spdi', 'spfi'], dtype=object),
                                                                            array(['Nissan versa', 'alfa-romero giulia',
                                                                                   'alfa-romero stelvio',
                                                                                   'alfa-romero Quadrifoglio',
                                                                                   'audi 100 ls', 'audi 100ls',
                                                                                   'audi 4000', 'audi 5000',
                                                                                   'audi fox', 'bmw 320i', 'bmw 320...
                                                                                   'volkswagen 1131 deluxe sedan',
                                                                                   'volkswagen model 111',
                                                                                   'volkswagen rabbit',
                                                                                   'volkswagen rabbit custom',
                                                                                   'volkswagen super beetle',
                                                                                   'volvo 144ea', 'volvo 145e (sw)',
                                                                                   'volvo 244dl', 'volvo 245',
                                                                                   'volvo 246', 'volvo 264gl',
                                                                                   'volvo diesel', 'volvo turbo'],
                                                                                  dtype=object)]),
                                                  ['fueltype', 'aspiration',
                                                   'doornumber', 'carbody',
                                                   'drivewheel',
                                                   'enginelocation',
                                                   'wheelbase', 'carlength',
                                                   'carwidth', 'curbweight',
                                                   'enginetype',
                                                   'cylindernumber',
                                                   'enginesize',
                                                   'fuelsystem', 'CarName',
                                                   'boreratio', 'horsepower',
                                                   'citympg', 'highwaympg',
                                                   'price'])])),
                ('decisiontreeregressor', DecisionTreeRegressor())])
```

In [33]:
```
y_predDT = pipe_dt.predict(X_test)
```

In [34]:
```
r2_DT = metrics.r2_score(y_test, y_predDT).round(4)
r2_DT
```

Out[34]:
```
0.8729
```

In [35]:
```
RMSE_DT = metrics.mean_absolute_error(y_test, y_predDT).round(3)
MAE_DT = np.sqrt(metrics.mean_squared_error(y_test, y_predDT)).round(3)
print("MAE = {MAE_DT} \nRMSE = {RMSE_DT}")
```

```
MAE = 3187.395
RMSE = 1999.345
```

Random Forest

In [36]:
```
from sklearn.ensemble import RandomForestRegressor

rf = RandomForestRegressor(random_state=0)
pipe_rf = make_pipeline(trf1, rf)

pipe_rf.fit(X_train, y_train)
```

Out[36]:
```
Pipeline(steps=[('columntransformer',
                 ColumnTransformer(remainder='passthrough',
                                   transformers=[('onehotencoder',
                                                  OneHotEncoder(categories=[array(['diesel', 'gas'], dtype=object),
                                                                            array(['std', 'turbo'], dtype=object),
                                                                            array(['four', 'two'], dtype=object),
                                                                            array(['convertible', 'hardtop', 'hatchback', 'sedan', 'wagon'],
                                                                                  dtype=object),
                                                                            array(['fwd', 'rwd'], dtype=object),
                                                                            array(['front', 'rear'], dtype=object),
                                                                            array(['dohc', 'dohcv', 'l', 'ohc', 'ohcf', 'ohcv',
                                                                                   'rotor'], dtype=object),
                                                                            array(['eight', 'five', 'four', 'six', 'three',
                                                                                   'twelve', 'two'], dtype=object),
                                                                            array(['1bbl', '2bbl', '4bbl', 'idi', 'mfi', 'mpfi',
                                                                                   'spdi', 'spfi'], dtype=object),
                                                                            array(['Nissan versa', 'alfa-romero giulia',
                                                                                   'alfa-romero stelvio',
                                                                                   'alfa-romero Quadrifoglio',
                                                                                   'audi 100 ls', 'audi 100ls',
                                                                                   'audi 4000', 'audi 5000',
                                                                                   'audi fox', 'bmw 320i', 'bmw 320...
                                                                                   'volkswagen 1131 deluxe sedan',
                                                                                   'volkswagen model 111',
                                                                                   'volkswagen rabbit',
                                                                                   'volkswagen rabbit custom',
                                                                                   'volkswagen super beetle',
                                                                                   'volvo 144ea', 'volvo 145e (sw)',
                                                                                   'volvo 244dl', 'volvo 245',
                                                                                   'volvo 246', 'volvo 264gl',
                                                                                   'volvo diesel', 'volvo turbo'],
                                                                                  dtype=object)]),
                                                  ['fueltype', 'aspiration',
                                                   'doornumber', 'carbody',
                                                   'drivewheel',
                                                   'enginelocation',
                                                   'wheelbase', 'carlength',
                                                   'carwidth', 'curbweight',
                                                   'enginetype',
                                                   'cylindernumber',
                                                   'enginesize',
                                                   'fuelsystem', 'CarName',
                                                   'boreratio', 'horsepower',
                                                   'citympg', 'highwaympg',
                                                   'price'])])),
                ('randomforestregressor',
                 RandomForestRegressor(random_state=0))])
```

In [37]:
```
y_predRF = pipe_rf.predict(X_test)
```

In [38]:
```
r2_rf = metrics.r2_score(y_test, y_predRF).round(4)
r2_rf
```

Out[38]:
```
0.9531
```

In [39]:
```
RMSE_rf = metrics.mean_absolute_error(y_test, y_predRF).round(3)
MAE_rf = np.sqrt(metrics.mean_squared_error(y_test, y_predRF)).round(3)
print("MAE = {MAE_rf} \nRMSE = {RMSE_rf}")
```

```
MAE = 1924.679
RMSE = 1386.174
```

Conclusion

In [40]:
```
pd.DataFrame(data=[r2_lm, MAE_lm, RMSE_lm], [r2_dt, MAE_dt, RMSE_dt], [r2_rf, MAE_rf, RMSE_rf],
             index=['Logistic Regression','Decision Tree', "Random Forest"],
             columns=["R2 Score", "MAE", "RMSE"])
```

Out[40]:

	R2 Score	MAE	RMSE
Logistic Regression	0.6190	5484.144	3457.464
Decision Tree	0.8729	3187.395	1999.345
Random Forest	0.9531	1924.679	1386.174