ML EXP 12 - YASH ASHOK SHIRSATH TE AIDS-69

```
import pandas as yash
import numpy as shirsath
import seaborn as train
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score,
mean_absolute_error

link =
"https://raw.githubusercontent.com/amankharwal/Website-data/master/
CarPrice.csv"
carkadata = yash.read_csv(link)
```

Exploratory Data Analysis

```
carkadata.head()
```

```
   car_ID  symboling                   CarName fueltype aspiration
doornumber  \
0       1          3        alfa-romero giulia      gas        std
two
1       2          3       alfa-romero stelvio      gas        std
two
2       3          1  alfa-romero Quadrifoglio      gas        std
two
3       4          2               audi 100 ls      gas        std
four
4       5          2                audi 100ls      gas        std
four

       carbody drivewheel enginelocation  wheelbase  ...
enginesize  \
0  convertible        rwd          front       88.6  ...         130

1  convertible        rwd          front       88.6  ...         130

2    hatchback        rwd          front       94.5  ...         152

3        sedan        fwd          front       99.8  ...         109

4        sedan        4wd          front       99.4  ...         136
```

```
   fuelsystem  boreratio  stroke compressionratio horsepower  peakrpm
citympg  \
0        mpfi       3.47    2.68              9.0        111     5000
21
1        mpfi       3.47    2.68              9.0        111     5000
21
2        mpfi       2.68    3.47              9.0        154     5000
19
3        mpfi       3.19    3.40             10.0        102     5500
24
4        mpfi       3.19    3.40              8.0        115     5500
18

   highwaympg    price
0          27  13495.0
1          27  16500.0
2          26  16500.0
3          30  13950.0
4          22  17450.0

[5 rows x 26 columns]
```

```
carkadata.tail()
```

```
     car_ID  symboling          CarName fueltype aspiration doornumber
\
200     201         -1  volvo 145e (sw)      gas        std       four

201     202         -1      volvo 144ea      gas      turbo       four

202     203         -1      volvo 244dl      gas        std       four

203     204         -1        volvo 246   diesel      turbo       four

204     205         -1      volvo 264gl      gas      turbo       four


    carbody drivewheel enginelocation  wheelbase  ...  enginesize
fuelsystem  \
200   sedan        rwd          front      109.1  ...         141
mpfi
201   sedan        rwd          front      109.1  ...         141
mpfi
202   sedan        rwd          front      109.1  ...         173
mpfi
203   sedan        rwd          front      109.1  ...         145
idi
204   sedan        rwd          front      109.1  ...         141
mpfi

     boreratio  stroke compressionratio horsepower  peakrpm citympg  \
200       3.78    3.15              9.5        114     5400      23
201       3.78    3.15              8.7        160     5300      19
202       3.58    2.87              8.8        134     5500      18
203       3.01    3.40             23.0        106     4800      26
204       3.78    3.15              9.5        114     5400      19

     highwaympg    price
200          28  16845.0
201          25  19045.0
202          23  21485.0
203          27  22470.0
204          25  22625.0

[5 rows x 26 columns]
```

```
carkadata.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
```

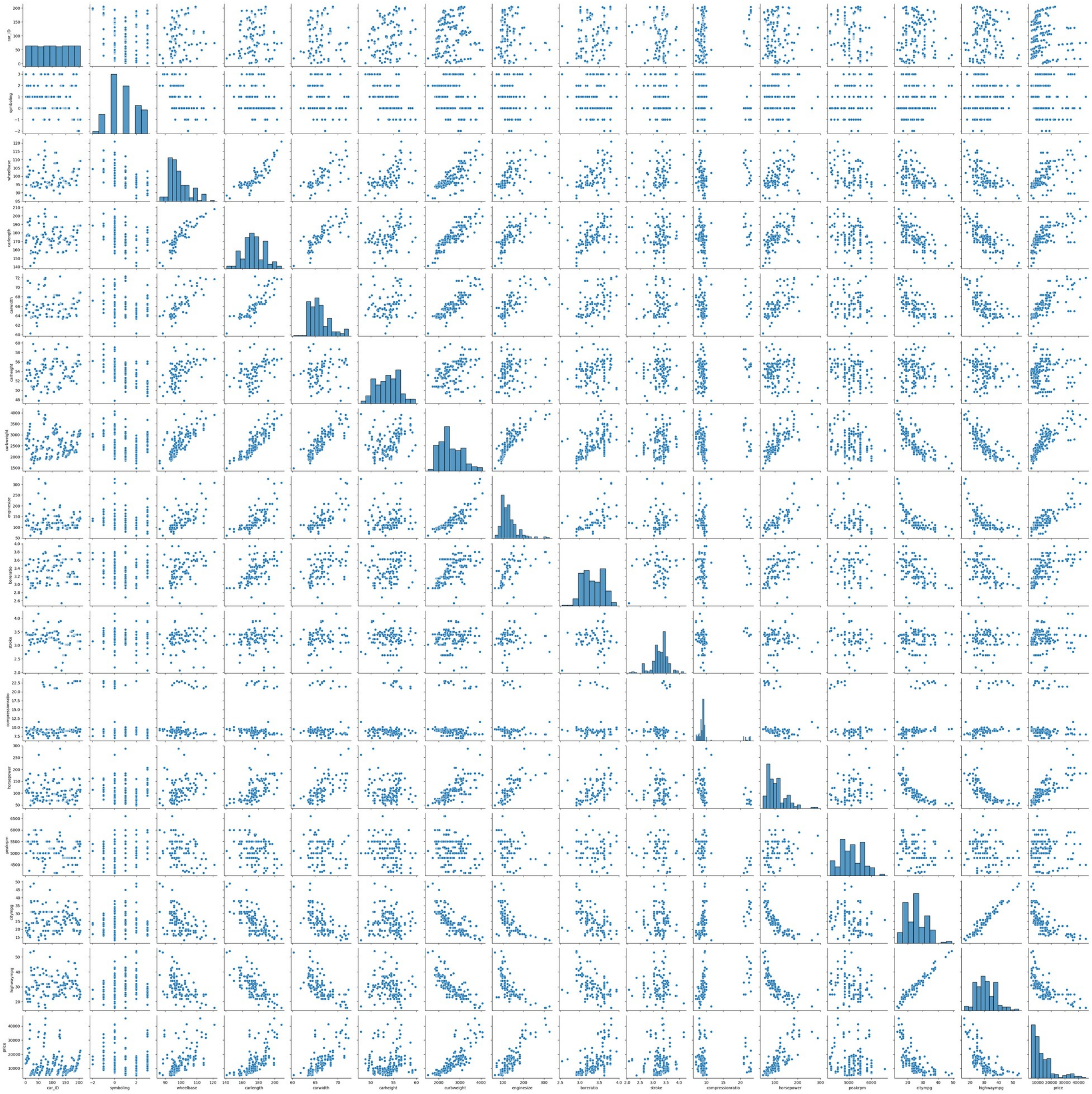
```
carkadata.describe()

           car_ID   symboling   wheelbase   carlength    carwidth
carheight  \
count  205.000000  205.000000  205.000000  205.000000  205.000000
205.000000
mean   103.000000    0.834146   98.756585  174.049268   65.907805
53.724878
std     59.322565    1.245307    6.021776   12.337289    2.145204
2.443522
min      1.000000   -2.000000   86.600000  141.100000   60.300000
47.800000
25%     52.000000    0.000000   94.500000  166.300000   64.100000
52.000000
50%    103.000000    1.000000   97.000000  173.200000   65.500000
54.100000
75%    154.000000    2.000000  102.400000  183.100000   66.900000
55.500000
max    205.000000    3.000000  120.900000  208.100000   72.300000
59.800000

        curbweight  enginesize   boreratio      stroke
compressionratio  \
count   205.000000  205.000000  205.000000  205.000000
205.000000
mean   2555.565854  126.907317    3.329756    3.255415
10.142537
std     520.680204   41.642693    0.270844    0.313597
3.972040
min    1488.000000   61.000000    2.540000    2.070000
7.000000
25%    2145.000000   97.000000    3.150000    3.110000
8.600000
50%    2414.000000  120.000000    3.310000    3.290000
9.000000
75%    2935.000000  141.000000    3.580000    3.410000
9.400000
max    4066.000000  326.000000    3.940000    4.170000
23.000000

       horsepower      peakrpm     citympg  highwaympg         price
count  205.000000   205.000000  205.000000  205.000000    205.000000
mean   104.117073  5125.121951   25.219512   30.751220  13276.710571
std     39.544167   476.985643    6.542142    6.886443   7988.852332
min     48.000000  4150.000000   13.000000   16.000000   5118.000000
25%     70.000000  4800.000000   19.000000   25.000000   7788.000000
50%     95.000000  5200.000000   24.000000   30.000000  10295.000000
75%    116.000000  5500.000000   30.000000   34.000000  16503.000000
max    288.000000  6600.000000   49.000000   54.000000  45400.000000
```

```
train.pairplot(carkadata)
plt.show()
```



Linear Regrssion Modelling

```
# Identify & Remove Non Numeric Data
binanumberwalekolam =
carkadata.select_dtypes(include=['object']).columns
numberwaladata = carkadata.drop(columns=binanumberwalekolam)
```

```python
Xash = numberwaladata.drop(columns=['price']) # Xash is Predictor
Yash = numberwaladata['price']  # Yash is Target

Xash_train, Xash_test, Yash_train, Yash_test = train_test_split(Xash,
Yash, test_size=0.2, random_state=42)

Yash_Cha_Model = LinearRegression()
Yash_Cha_Model.fit(Xash_train, Yash_train)
```

```
LinearRegression()
```

```python
Yash_Cha_Prediction = Yash_Cha_Model.predict(Xash_test) # TeSP

# Mean Squared Error measures how close a regression line is to a set
of data points.
mse = mean_squared_error(Yash_test, Yash_Cha_Prediction)
print("Mean Squared Error:-", mse)
```

```
Mean Squared Error:- 11710105.078807332
```

```python
# one of the two main performance indicators for a regression model.
rmse = shirsath.sqrt(mse)
print("Root Mean Squared Error:-", rmse)
```

```
Root Mean Squared Error:- 3422.0030798944836
```

```python
# MAE is a common metric used in statistics and machine learning to
assess the performance of regression models
mae = mean_absolute_error(Yash_test, Yash_Cha_Prediction)
print("Mean Absolute Error:-", mae)
```

```
Mean Absolute Error:- 2411.093962409612
```

```python
# how well the model explains the variability in the dependent
variable based on the independent variables.
r_squared = r2_score(Yash_test, Yash_Cha_Prediction)
print("R-Squared:-", r_squared)
```

```
R-Squared:- 0.8516657126363221
```

In the context of linear regression or other regression models, accuracy is not typically used as a performance metric. Instead, we use metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R^2) to evaluate the model's performance.