Lab 13 26-10-2022

October 26, 2022

1 Multiple Regression

[9]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

1.1 Load Dataset

[5]:
```
df = pd.read_csv('FuelConsumption.csv')
df.info
```

[5]:
```
<bound method DataFrame.info of       MODELYEAR   MAKE       MODEL
VEHICLECLASS  ENGINESIZE  CYLINDERS  \
0          2014  ACURA         ILX         COMPACT        2.0          4
1          2014  ACURA         ILX         COMPACT        2.4          4
2          2014  ACURA  ILX HYBRID         COMPACT        1.5          4
3          2014  ACURA     MDX 4WD     SUV - SMALL        3.5          6
4          2014  ACURA     RDX AWD     SUV - SMALL        3.5          6
…         …    …         …             …         …        …
1062       2014  VOLVO    XC60 AWD     SUV - SMALL        3.0          6
1063       2014  VOLVO    XC60 AWD     SUV - SMALL        3.2          6
1064       2014  VOLVO    XC70 AWD     SUV - SMALL        3.0          6
1065       2014  VOLVO    XC70 AWD     SUV - SMALL        3.2          6
1066       2014  VOLVO    XC90 AWD  SUV - STANDARD        3.2          6

     TRANSMISSION FUELTYPE  FUELCONSUMPTION_CITY  FUELCONSUMPTION_HWY  \
0             AS5        Z                   9.9                  6.7
1              M6        Z                  11.2                  7.7
2             AV7        Z                   6.0                  5.8
3             AS6        Z                  12.7                  9.1
4             AS6        Z                  12.1                  8.7
…           …      …                   …                  …
1062          AS6        X                  13.4                  9.8
1063          AS6        X                  13.2                  9.5
1064          AS6        X                  13.4                  9.8
1065          AS6        X                  12.9                  9.3
```

```
1066          AS6         X                  14.9                 10.2

      FUELCONSUMPTION_COMB  FUELCONSUMPTION_COMB_MPG  CO2EMISSIONS
0                      8.5                        33           196
1                      9.6                        29           221
2                      5.9                        48           136
3                     11.1                        25           255
4                     10.6                        27           244
…                      …                        …           …
1062                  11.8                        24           271
1063                  11.5                        25           264
1064                  11.8                        24           271
1065                  11.3                        25           260
1066                  12.8                        22           294

[1067 rows x 13 columns]>
```

1.2 Feature Selection

[11]:

```
sdf = df[['ENGINESIZE', 'CYLINDERS', 'FUELCONSUMPTION_CITY',␣
 ↪'FUELCONSUMPTION_HWY',
          'FUELCONSUMPTION_COMB', 'CO2EMISSIONS' ]]
sdf.head()
```

[11]:

```
   ENGINESIZE  CYLINDERS  FUELCONSUMPTION_CITY  FUELCONSUMPTION_HWY  \
0         2.0          4                   9.9                  6.7
1         2.4          4                  11.2                  7.7
2         1.5          4                   6.0                  5.8
3         3.5          6                  12.7                  9.1
4         3.5          6                  12.1                  8.7

   FUELCONSUMPTION_COMB  CO2EMISSIONS
0                   8.5           196
1                   9.6           221
2                   5.9           136
3                  11.1           255
4                  10.6           244
```

1.3 Extraction of variables

[12]:

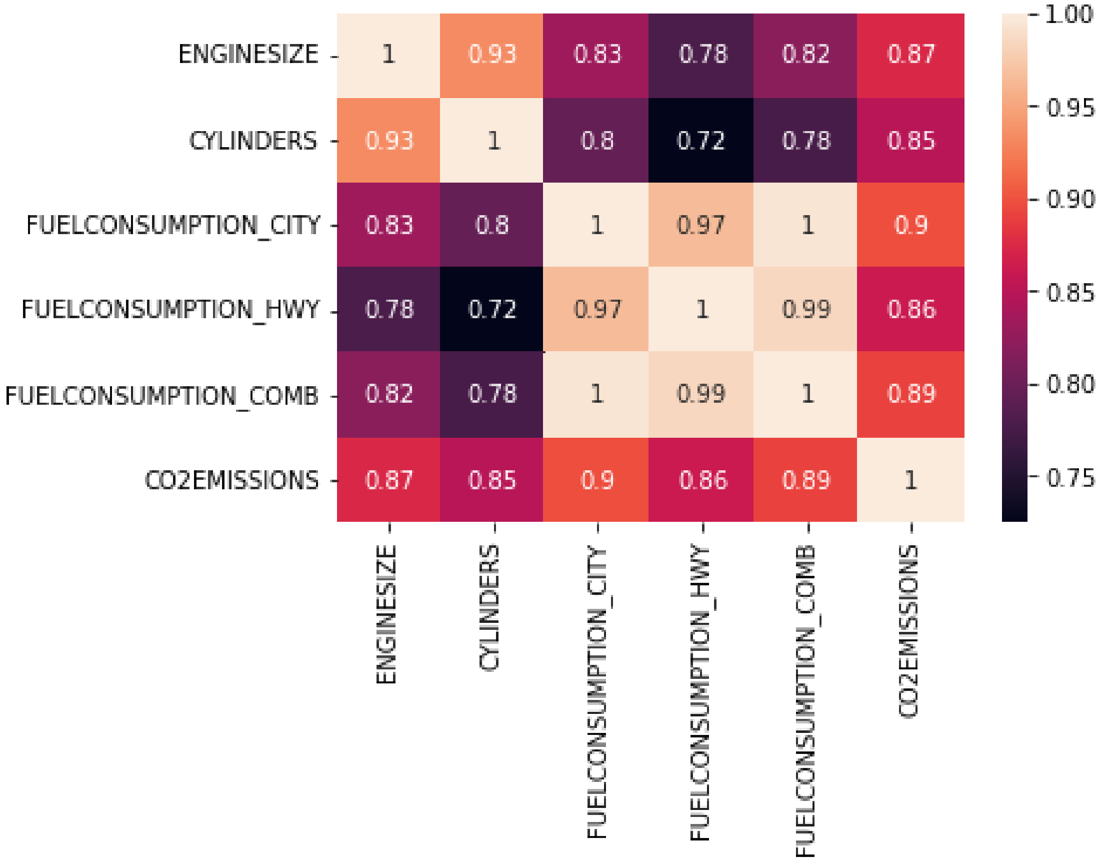
```
x = sdf.iloc[:, :-1].values
y = sdf.iloc[:, -1].values
print(x)
print(y)
```

```
[[ 2.   4.   9.9  6.7  8.5]
 [ 2.4  4.  11.2  7.7  9.6]
 [ 1.5  4.   6.   5.8  5.9]
 …
 [ 3.   6.  13.4  9.8 11.8]
 [ 3.2  6.  12.9  9.3 11.3]
 [ 3.2  6.  14.9 10.2 12.8]]
[196 221 136 … 271 260 294]
```

1.4 Correlation using heatmap

[13]:
```
sns.heatmap(sdf.corr(),annot=True)
```

[13]:
```
<AxesSubplot:>
```



1.5 Splitting Dataset

[14]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=1/3,␣
 ↪random_state=0)
```

1.6 Training

[15]:

```
from sklearn.linear_model import LinearRegression
regression = LinearRegression()
regression.fit(x_train, y_train)
```

[15]:

```
LinearRegression()
```

1.7 Prediction

[16]:

```
y_pred= regression.predict(x_test)
x_pred= regression.predict(x_train)
```

1.7.1 finding regression coefficients

[18]:

```
print("Coefficient:", regression.coef_)
print("Intercept:",regression.intercept_)
```

```
Coefficient: [ 10.8547316    7.65914224 -11.21595269 -10.47231908  31.14100197]
Intercept: 64.88135054065862
```

1.8 Visualisation

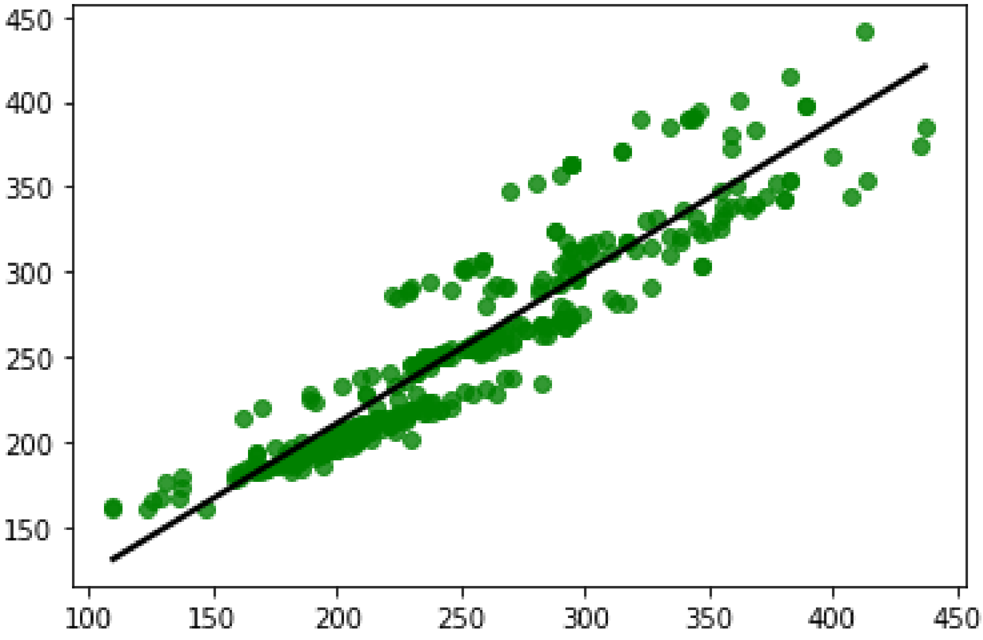
1.8.1 Regression Plot

[36]:

```
sns.regplot(x=y_test, y=y_pred, ci=None, scatter_kws={"color": "g"},␣
 ↪line_kws={"color": "black"})
```

[36]:

```
<AxesSubplot:>
```



1.8.2 kde plot

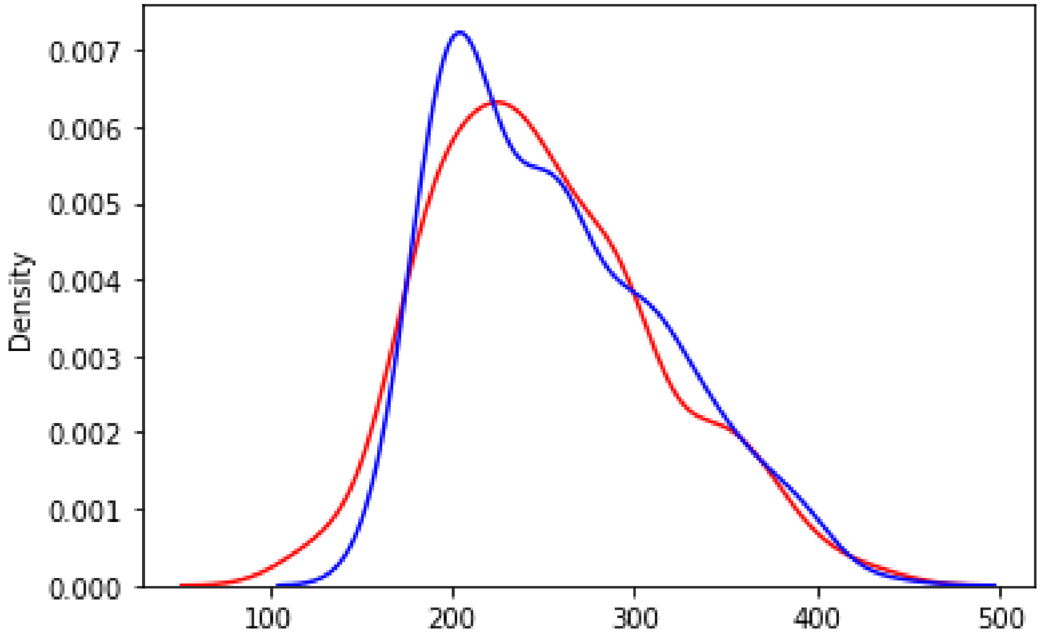
[27]:

```
ax1 = sns.distplot(y_test, hist=False, color="r", label="Actual Value")
sns.distplot(y_pred, hist=False, color="b", label="Fitted Values" , ax=ax1)
```

```
/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `kdeplot` (an axes-level function for
kernel density plots).
  warnings.warn(msg, FutureWarning)
/opt/anaconda3/lib/python3.9/site-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed in a
future version. Please adapt your code to use either `displot` (a figure-level
function with similar flexibility) or `kdeplot` (an axes-level function for
kernel density plots).
  warnings.warn(msg, FutureWarning)
```

[27]:

```
<AxesSubplot:ylabel='Density'>
```



1.9 Evaluation

[23]:

```
print('Residual sum squares %.2f' % np.mean((y_pred - y_test) **2 ))
```

```
Residual sum squares 617.32
```

[29]:

```
print(regression.score(x_test, y_test))
```

```
0.8426468641660053
```