Random Forest

October 23, 2021

1 Random Forest

1.1 Importing the libraries

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

[2]:
```
from sklearn.model_selection import train_test_split
```

[3]:
```
from sklearn.ensemble import RandomForestRegressor
```

1.2 Importing the data set

[4]:
```
dataset = pd.read_csv("petrol_consumption.csv")
```

[5]:
```
x = dataset.iloc[:,:-1].values
y = dataset.iloc[:,-1].values
```

[6]:
```
dataset.head()
```

[6]:

	Petrol_tax	Average_income	Paved_Highways	Population_Driver_licence(%)	\
0	9.0	3571	1976	0.525	
1	9.0	4092	1250	0.572	
2	9.0	3865	1586	0.580	
3	7.5	4870	2351	0.529	
4	8.0	4399	431	0.544	

	Petrol_Consumption
0	541
1	524
2	561
3	414
4	410

[7]:
```
print(x)
```

```
[[9.0000e+00 3.5710e+03 1.9760e+03 5.2500e-01]
 [9.0000e+00 4.0920e+03 1.2500e+03 5.7200e-01]
```

```
[9.0000e+00 3.8650e+03 1.5860e+03 5.8000e-01]
[7.5000e+00 4.8700e+03 2.3510e+03 5.2900e-01]
[8.0000e+00 4.3990e+03 4.3100e+02 5.4400e-01]
[1.0000e+01 5.3420e+03 1.3330e+03 5.7100e-01]
[8.0000e+00 5.3190e+03 1.1868e+04 4.5100e-01]
[8.0000e+00 5.1260e+03 2.1380e+03 5.5300e-01]
[8.0000e+00 4.4470e+03 8.5770e+03 5.2900e-01]
[7.0000e+00 4.5120e+03 8.5070e+03 5.5200e-01]
[8.0000e+00 4.3910e+03 5.9390e+03 5.3000e-01]
[7.5000e+00 5.1260e+03 1.4186e+04 5.2500e-01]
[7.0000e+00 4.8170e+03 6.9300e+03 5.7400e-01]
[7.0000e+00 4.2070e+03 6.5800e+03 5.4500e-01]
[7.0000e+00 4.3320e+03 8.1590e+03 6.0800e-01]
[7.0000e+00 4.3180e+03 1.0340e+04 5.8600e-01]
[7.0000e+00 4.2060e+03 8.5080e+03 5.7200e-01]
[7.0000e+00 3.7180e+03 4.7250e+03 5.4000e-01]
[7.0000e+00 4.7160e+03 5.9150e+03 7.2400e-01]
[8.5000e+00 4.3410e+03 6.0100e+03 6.7700e-01]
[7.0000e+00 4.5930e+03 7.8340e+03 6.6300e-01]
[8.0000e+00 4.9830e+03 6.0200e+02 6.0200e-01]
[9.0000e+00 4.8970e+03 2.4490e+03 5.1100e-01]
[9.0000e+00 4.2580e+03 4.6860e+03 5.1700e-01]
[8.5000e+00 4.5740e+03 2.6190e+03 5.5100e-01]
[9.0000e+00 3.7210e+03 4.7460e+03 5.4400e-01]
[8.0000e+00 3.4480e+03 5.3990e+03 5.4800e-01]
[7.5000e+00 3.8460e+03 9.0610e+03 5.7900e-01]
[8.0000e+00 4.1880e+03 5.9750e+03 5.6300e-01]
[9.0000e+00 3.6010e+03 4.6500e+03 4.9300e-01]
[7.0000e+00 3.6400e+03 6.9050e+03 5.1800e-01]
[7.0000e+00 3.3330e+03 6.5940e+03 5.1300e-01]
[8.0000e+00 3.0630e+03 6.5240e+03 5.7800e-01]
[7.5000e+00 3.3570e+03 4.1210e+03 5.4700e-01]
[8.0000e+00 3.5280e+03 3.4950e+03 4.8700e-01]
[6.5800e+00 3.8020e+03 7.8340e+03 6.2900e-01]
[5.0000e+00 4.0450e+03 1.7782e+04 5.6600e-01]
[7.0000e+00 3.8970e+03 6.3850e+03 5.8600e-01]
[8.5000e+00 3.6350e+03 3.2740e+03 6.6300e-01]
[7.0000e+00 4.3450e+03 3.9050e+03 6.7200e-01]
[7.0000e+00 4.4490e+03 4.6390e+03 6.2600e-01]
[7.0000e+00 3.6560e+03 3.9850e+03 5.6300e-01]
[7.0000e+00 4.3000e+03 3.6350e+03 6.0300e-01]
[7.0000e+00 3.7450e+03 2.6110e+03 5.0800e-01]
[6.0000e+00 5.2150e+03 2.3020e+03 6.7200e-01]
[9.0000e+00 4.4760e+03 3.9420e+03 5.7100e-01]
[7.0000e+00 4.2960e+03 4.0830e+03 6.2300e-01]
[7.0000e+00 5.0020e+03 9.7940e+03 5.9300e-01]]
```

[8]:
```
print(y)
```

```
[541 524 561 414 410 457 344 467 464 498 580 471 525 508 566 635 603 714
 865 640 649 540 464 547 460 566 577 631 574 534 571 554 577 628 487 644
 640 704 648 968 587 699 632 591 782 510 610 524]
```

1.3 Splitting the dataset into the Training set and Test set

[9]:
```
x_train,x_test,y_train,y_test =  train_test_split(x,y, test_size = 0.2,␣
↪random_state = 0)
```

1.4 Training the Random Forest Regression model on the training dataset

[10]:
```
regressor = RandomForestRegressor(n_estimators = 10, random_state = 0)
regressor.fit(x_train, y_train)
```

[10]:
```
RandomForestRegressor(n_estimators=10, random_state=0)
```

1.5 Predict Test Result

[11]:
```
y_predict = regressor.predict(x_test)
```

[12]:
```
df=pd.DataFrame({'Actual':y_test, 'Predicted':y_predict, 'diffrence' : y_test -␣
↪y_predict})
df
```

[12]:

	Actual	Predicted	diffrence
0	534	573.3	-39.3
1	410	537.7	-127.7
2	577	595.7	-18.7
3	571	587.1	-16.1
4	577	623.9	-46.9
5	704	601.1	102.9
6	487	593.8	-106.8
7	587	577.7	9.3
8	467	457.2	9.8
9	580	566.9	13.1

1.6 Visualising the Decision Tree Regression result

1.7 Evaluating the Algorithm

[13]:
```
from sklearn import metrics
print('Mean Absolute Error:', metrics.mean_absolute_error(y_test, y_predict))
print('Mean Squared Error:', metrics.mean_squared_error(y_test, y_predict))
print('Root Mean Squared Error:', np.sqrt(metrics.mean_squared_error(y_test,␣
↪y_predict)))
```

```
Mean Absolute Error: 49.059999999999995
Mean Squared Error: 4300.907999999999
Root Mean Squared Error: 65.58130831265872
```