Bootstrap_Random_Forest

May 8, 2021

1 Application of Bootstrap samples in Random Forest

[1]:

```python
import numpy as np
from sklearn.datasets import load_boston
from sklearn.metrics import mean_squared_error
import random
from sklearn.tree import DecisionTreeRegressor
import math
```

Load the boston house dataset

[2]:

```python
boston = load_boston()
x=boston.data #independent variables
y=boston.target #target variable
```

[3]:

```python
y.shape
```

[3]:

```
(506,)
```

[7]:

```python
def generate_sample(x):
    x_60 = random.sample(range(x.shape[0]),round(x.shape[0]*0.6))
    x_40 = random.sample(x_60,round(x.shape[0]*0.4))
    sample_index = x_60 + x_40

    oob_index = list(set(range(x.shape[0])) - set(sample_index))

    return sample_index,oob_index
```

[9]:

```python
## get the 30 sample data
list_input_data=[]
list_output_data = []
list_input_oob_data =[]
list_output_oob_data = []
for i in range(0,30):
    sample_index,oob_index= generate_sample(x)
    list_input_data.append(x[sample_index])
    list_output_data.append(y[sample_index])
    list_input_oob_data.append(x[oob_index])
    list_output_oob_data.append(y[oob_index])
```

[10]:

```
## Trainning model on each sample
all_selected_models = []
for i in range(30):
    model = DecisionTreeRegressor(max_depth=None)
    model.fit(list_input_data[i],list_output_data[i])
    all_selected_models.append(model)
```

[11]:

```
list_of_y_pred = []
for i in range(30):

    y_pred = all_selected_models[i].predict(list_input_data[i])
    list_of_y_pred.append(y_pred)
```

[37]:

```
## Finding mean squared error for every model
from sklearn.metrics import mean_squared_error
list_of_mse= []
for i in range(30):
    mse = mean_squared_error(list_output_data[i],list_of_y_pred[i])
    list_of_mse.append(mse)
```

[17]:

```
## Mean of the MSE:
print("Mean of the MSE of all sample is {}".format(np.mean(list_of_mse)))
```

```
Mean of the MSE of all sample is 8.106870962745576e-32
```

[21]:

```
## Finding Y_pred for oob data
list_of_oob_y_pred = []
for i in range(30):

    y_pred = all_selected_models[i].predict(list_input_oob_data[i])
    list_of_oob_y_pred.append(y_pred)
```

[24]:

```
list_of_oob_mse= []
for i in range(30):
    mse_oob = mean_squared_error(list_output_oob_data[i],list_of_oob_y_pred[i])
    list_of_oob_mse.append(mse_oob)
```

[25]:

```
## Mean Squared error of oob data
np.mean(list_of_oob_mse)
```

[25]:

```
24.20633168316832
```

2 Task 2

[39]:

```
list_input_data=[]
list_output_data = []
list_input_oob_data =[]
list_output_oob_data = []
for i in range(0,35):
    sample_index,oob_index= generate_sample(x)
    list_input_data.append(x[sample_index])
    list_output_data.append(y[sample_index])
    list_input_oob_data.append(x[oob_index])
    list_output_oob_data.append(y[oob_index])
```

[40]:

```
all_selected_models = []
for i in range(35):
    model = DecisionTreeRegressor(max_depth=None)
    model.fit(list_input_data[i],list_output_data[i])
    all_selected_models.append(model)
```

[41]:

```
list_of_y_pred = []
for i in range(35):
    y_pred = all_selected_models[i].predict(list_input_data[i])
    list_of_y_pred.append(y_pred)
```

[43]:

```
list_of_mse= []
for i in range(35):
    mse = mean_squared_error(list_output_data[i],list_of_y_pred[i])
    list_of_mse.append(mse)
```

[44]:

```
## Mean changed
print("Mean of the mse : {}".format(np.mean(list_of_mse)))
```

```
Mean of the mse : 6.574583264292565e-32
```

[49]:

```
list_of_oob_y_pred = []
for i in range(35):
    y_pred = all_selected_models[i].predict(list_input_oob_data[i])
    list_of_oob_y_pred.append(y_pred)
```

[50]:

```
list_of_oob_mse= []
for i in range(35):
    mse = np.square(np.subtract(list_output_oob_data[i],list_of_oob_y_pred[i])).
 ↪mean()
    list_of_oob_mse.append(mse)
```

[51]:

```
np.mean(list_of_oob_mse)
```

[51]:

```
23.627012729844413
```

[218]:

```
## Here we dont have the standard deviation of mean_squared_error  population␣
 ↪so we are going to use
## second method in which we find the standard deviation of our sample
```

[52]:

```
list_of_mse = np.array(list_of_mse)
list_of_oob_mse = np.array(list_of_oob_mse)
mean_of_mse = list_of_mse.mean()
mean_of_oob_mse = list_of_oob_mse.mean()
standard_dev_mse = np.std(list_of_mse)
standard_dev_oob_mse = np.std(list_of_oob_mse)
```

[53]:

```
standard_dev_oob_mse
```

[53]:

```
5.938947806171344
```

[54]:

```
def find_confidence_limit(mean,std,samp_size):
    upper_limit = mean-1.96*std/math.sqrt(samp_size)
    lower_limit = mean+1.96*std/math.sqrt(samp_size)
    return upper_limit,lower_limit
```

[55]:

```
u_mse,l_mse = find_confidence_limit(mean_of_mse,standard_dev_mse,35)
```

[56]:

```
print("Upper Confidence limit is {} \nLower confidence limit is {}".
 ↪format(u_mse,l_mse))
```

```
Upper Confidence limit is 4.876399638797741e-32
Lower confidence limit is 8.272766889787388e-32
```

[57]:

```
u_oob_mse,l_oob_mse =␣
 ↪find_confidence_limit(mean_of_oob_mse,standard_dev_oob_mse,35)
```

[58]:

```
print("Upper Confidence limit is {} \nLower confidence limit is {}".
 ↪format(u_oob_mse,l_oob_mse))
```

```
Upper Confidence limit is 21.65943654310186
Lower confidence limit is 25.594588916586964
```

3 Task 3

[59]:

```
xq= [0.18,20.0,5.00,0.0,0.421,5.60,72.2,7.95,7.0,30.0,19.1,372.13,18.60]
xq = np.array(xq)
```

[64]:

```
y_pred_of_query= []
for i in range(len(all_selected_models)):
    ypred = all_selected_models[i].predict((xq).reshape(1,-1))
    y_pred_of_query.append(ypred)
```

[65]:

```
Y_median = np.median(y_pred_of_query)
print("Median of the query point is {}".format(Y_median))
```

```
Median of the query point is 18.5
```