CS 7646 Machine Learning For Trading
Report for Assess Learners

Jiawei Zhang
jzhang950

Abstract

There are four learner files we created for this project:

1) DTLearner: While building the decision tree, I actually find the correlation of all the X respect to Y. After that, I select the index of maximum abs value's column number and use it to find the median of the column, which is the split value. Then, I use the recursive method to separate the data and put them in the left tree and right tree. While I cannot split the data or the Y values are all same or the number of rows of data is less than the leaf size, I return the tree. Finally, we use the 60% of data from csv file to train the tree and 40% of data from csv file to test the tree and calculate the RMSE and correlation.
2) RTLearner: Similar to the DTLearner, I build the random tree. The difference between decision tree and random tree is that I used the random vanlue to select the the column index.
3) BagLearner: The BagLearner takes parameter: types of learner, number of bagg and kwargs. It will random choose the data from tain data and put them in each bag so that it will generate the corresponding number of bags with random data with the specified learner that we choose.
4) InsaneLearner: The insane learner is actually the linear regression learner. I used the least square regression expression to train the data.

Questions and Explanations

Q1: Does overfitting occur with respect to leaf_size? Consider the dataset istanbul.csv with DTLearner. For which values of leaf_size does overfitting occur? Use RMSE as your metric for assessing overfitting.

By definition, the overfitting occurs when the in sample RMSE is decreasing and out of sample RMSE is increasing according to the degree of freedom. In order to find the corresponding area of RMSE value, I test the DTLearner by changing the leaf_size and get the following result:

When leaf_size = 1

```
In sample results
RMSE:  0.000163429863094
corr:  0.999903595936

Out of sample results
RMSE:  0.00729929173437
corr:  0.689521381567
```

When leaf_size = 20

```
In sample results
RMSE:  0.00655589110794
corr:  0.830514796523

Out of sample results
RMSE:  0.00616769385252
corr:  0.722964996589
```

When leaf_size = 30

```
In sample results
RMSE:  0.00689816525811
corr:  0.810255652679

Out of sample results
RMSE:  0.00619969531675
corr:  0.704048383737
```

When leaf_size = 40

```
In sample results
RMSE:  0.00764593143351
corr:  0.76026955565

Out of sample results
RMSE:  0.00632656493321
corr:  0.680158837956
```

When leaf_size = 50

```
In sample results
RMSE:  0.00794230848153
corr:  0.738011359164

Out of sample results
RMSE:  0.00645340974511
corr:  0.657004837057
```

When leaf_size = 100

```
In sample results
RMSE:  0.00913766501258
corr:  0.630305442462

Out of sample results
RMSE:  0.00622621803059
corr:  0.665099691542
```

The data is overftting when the leaf_size = 1.
Since when I vary the leaf_size from 1 to 100, I found that when the leaf_size equals to 1, the RMSE of in sample of data equals to 0.000163, which nearly to 0. Here we can say that the data in perfectly overfit. As we increase the leaf_size, the RMSE of in sample is increasing a lot. It means that the model we trained is becoming more general and not as precisely fit to the data. The data will not becoming more overfitting. Also, when we increase the leaf_size, we may find that the RMSE of out sample in decreasing a little bit. The lower of RMSE of out of sample means that the data is in a good fit and it takes mean value of the data test for approxiamtion.

Q2: Can bagging reduce or eliminate overfitting with respect to leaf_size? Again consider the dataset istanbul.csv with DTLearner. To investigate this choose a fixed number of bags to use and vary leaf_size to evaluate.

In order to get the corresponding RMSE value compare to the Q1, I vary the leaf_size from 1 to 100, keep the number of bags equals to 20, and get the following data for the bagging:

When leaf_size = 1:

```
In sample results
RMSE:  0.00268757583251
corr:  0.977937237603

Out of sample results
RMSE:  0.00484394915222
corr:  0.817040671676
```

When leaf_size = 20

```
In sample results
RMSE:  0.00575039493526
corr:  0.882743357939

Out of sample results
RMSE:  0.00467022302206
corr:  0.824266628769
```

When leaf_size = 30

```
In sample results
RMSE:  0.00604854938457
corr:  0.867540107556

Out of sample results
RMSE:  0.00492045391704
corr:  0.802453016061
```

When leaf_size = 40

```
In sample results
RMSE:  0.00671783212546
corr:  0.833713394134

Out of sample results
RMSE:  0.0048889972582
corr:  0.808312816515
```

When leaf_size = 50

```
In sample results
RMSE:  0.00705734386328
corr:  0.81754874661

Out of sample results
RMSE:  0.00500242761058
corr:  0.795158531341
```

When leaf_size = 100

```
In sample results
RMSE:  0.00843508158538
corr:  0.720622543044

Out of sample results
RMSE:  0.00521846852095
corr:  0.777095124237
```

In Q1, the RMSE value of in sample when the leaf_size is 1 equals to 0.000163. While using the bagging learner, the RMSE of in sample when the leaf_size = 1 is 0.0026876. When I use the bagging learner, the RMSE value becomes 20 times of the original one without bagging. In some ways that we may say the bagging can elimiante the overfitting problems.

Q3: Quantitatively compare "classic" decision trees (DTLearner) versus random trees (RTLearner). In which ways is one method better than the other?

When the leaf_size = 1:

DTL:

```
[[  7.00000000e+00   5.50359000e-04   1.00000000e+00   3.20000000e+02]
 [  1.00000000e+00  -8.66932600e-03   1.00000000e+00   1.60000000e+02]
 [  7.00000000e+00  -1.11146640e-02   1.00000000e+00   8.20000000e+01]
 ...,
 [  0.00000000e+00   2.38314220e-02   1.00000000e+00   2.00000000e+00]
 [ -1.00000000e+00   4.78045310e-02              nan              nan]
 [ -1.00000000e+00   3.85061930e-02              nan              nan]]
jzhang950

In sample results
RMSE:  0.000163429863094
corr:  0.999903595936

Out of sample results
RMSE:  0.00727936694313
corr:  0.690720895061
```

RTL:

```
[[  7.00000000e+00   5.50359000e-04   1.00000000e+00   3.20000000e+02]
 [  1.00000000e+00  -8.66932600e-03   1.00000000e+00   1.60000000e+02]
 [  7.00000000e+00  -1.11146640e-02   1.00000000e+00   8.20000000e+01]
 ...,
 [  0.00000000e+00   2.38314220e-02   1.00000000e+00   2.00000000e+00]
 [ -1.00000000e+00   4.78045310e-02              nan              nan]
 [ -1.00000000e+00   3.85061930e-02              nan              nan]]
jzhang950

In sample results
RMSE:  0.000163429863094
corr:  0.999903595936

Out of sample results
RMSE:  0.00706057290896
corr:  0.709872604019
```

When the leaf_size = 10:

DTL:

```
In sample results
RMSE:  0.00579515550131
corr:  0.870389655207

Out of sample results
RMSE:  0.00564566289549
corr:  0.768862653669
```

RTL:

```
In sample results
RMSE:  0.00579515550131
corr:  0.870389655207

Out of sample results
RMSE:  0.00564566289549
corr:  0.768862653669
```

1) From the data, when the leaf_size = 1, the in sample RMSE and correlation value are the same for both DTL and RTL. However, the out of sample values are different. Under this condition, the RTL gets a smaller value in RMSE but greater value in correlation. According to analysis, I found that when I set the leaf_size = 1, which means in overfitting

condition, the RTL takes smaller value of RMSE in out of sample results and greater value of correlation. It means that the RTL gives the test data fits better than the DTL gives the results under the overfitting condition.

2) In order to compare the cost of DTLearner and RTlearner, I import the time and choose the simple.csv and recorded the two time as below:

DTL:

```
(20, 2)
(20,)
[[  1.    2.    1.    6. ]
 [  0.    2.    1.    4. ]
 [  1.    1.5   1.    2. ]
 [ -1.    5.    nan   nan]
 [ -1.    6.    nan   nan]
 [ -1.    6.    nan   nan]
 [  1.    4.    1.    2. ]
 [ -1.   10.    nan   nan]
 [ -1.   11.    nan   nan]]
jzhang950

In sample results
RMSE:  0.0
corr:  1.0

Out of sample results
RMSE:  0.0
corr:  1.0
--- 1.19209289551e-06 seconds ---
```

RTL:

```
(20, 2)
(20,)
[[  1.    2.    1.    6. ]
 [  0.    2.    1.    4. ]
 [  0.    1.5   1.    2. ]
 [ -1.    6.    nan   nan]
 [ -1.    5.    nan   nan]
 [ -1.    6.    nan   nan]
 [  0.    1.5   1.    2. ]
 [ -1.   11.    nan   nan]
 [ -1.   10.    nan   nan]]
jzhang950

In sample results
RMSE:  0.0
corr:  1.0

Out of sample results
RMSE:  0.0
corr:  1.0
--- 9.53674316406e-07 seconds ---
```

From the data, we could fine the running time for DTL is 1.192e-6 seconds which is greater than the running time of RTL, 9.54e-7 since we calculate the correlation and select the maximum of abs value for correlation and then take the corresponding i for finding the split value in the DTLearner. In conclusion, in the running cost aspect, the RTLearner is greater than DTLearner. If we using the BagLearner and increase the bag size, the difference between them in running time will be more obviously.