11676 – Big Data Analytics

lneves

HW 3 – Random Forest

For homework 3, I have enhanced my python code from the data preparation part. For HW2, I have used the linear interpolation available on Panda, but the result of my random forest was getting really bad when choosing the “Artificial” features, thus, creating a really high standard deviation of accuracy even on large forests.

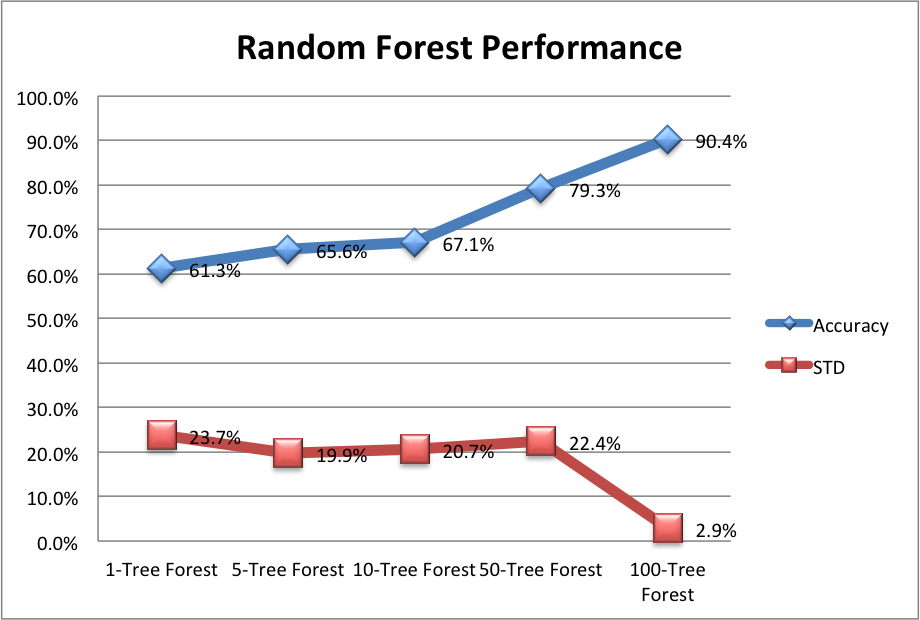
I have them implemented the idea that the professor gave in class. I have used the existing features as input for a linear regression model and tried to create the new features as a regression task. Each iteration, I would generate the values for the new feature and add them as features for the next iteration. You can check the code on my github repo under python\_code.

After having a more reliable value for my features, I have tuned my random forest implementation. As I had already built the decision tree structure, I would just create a way to shuffle data and create N decision trees (N being the input) of random samples of the data containing 2/3 of the elements. Then, I would get this tree to use only a random sample of sqrt(F) features, being F the total number of features. I would train my forest on a subset of the total data containing 230.853 instances and, after my forest was built, test it on a 57.893 instances different test set. For deciding the label, I would make the trees vote and get the label voted by the majority.

I tried with a bigger data set, but my computer ran out of memory.

For my accuracy studies, I have run the calculation 10 times on forests with 1, 5, 10, 50 and 100 trees and computed the average and standard deviation. I did that so that I could compare the accuracy with my previous approach that was only one tree with accuracy 76%.

My results were as follows:



|  |  |  |
| --- | --- | --- |
|  | Accuracy(Mean) | STD |
| 1-Tree Forest | 61.3% | 23.7% |
| 5-Tree Forest | 65.6% | 19.9% |
| 10-Tree Forest | 67.1% | 20.7% |
| 50-Tree Forest | 79.3% | 22.4% |
| 100-Tree Forest | 90.4% | 2.9% |

This result is as expected. Having less features and less data would make my accuracy worst when having only one tree, but as we grow the number of trees, the accuracy gets much higher and the standard deviation, smaller.

As of code organization, I have created a random forest class that would have an array list of decision trees. The rest of the code is pretty similar, since I had a decision tree and a decision tree node class that I am calling multiple times now. I have just fixed a bug on one of the files.