Momin Khan
Cogs 118A
Final Report

Abstract

This paper explores the usage of three supervised learning algorithms (Decision Trees, Boosting and Neural Nets) and analyzes the results to compare the classifiers amongst themselves. Previously, these classifiers have been examined in detail on similar datasets (LETTER, COV_TYPE, ADULT) and the results obtained from this experiment can be compared. Unlike previous studies, this only focuses on one performance criteria: accuracy.

1. Introduction

Learning algorithms have gained popularity with time as they have seen improvements. Out of the many learning algorithms available, I am using Boosting, Random Forests and Decision Trees (KNN) for the purposes of this experiment. Boosting and Random Forests are considerably newer supervised learning algorithms as compared to the rest and have great performance. This paper however only deals with the comparison of three supervised learning algorithms using accuracy as the sole performance criteria.

2. Method

2.1. Learning Algorithms

I try to explore how the supervised learning algorithms differ on different datasets with different parameter settings. It would be helpful to see how the different parameter settings weigh in on the classifier's performance. Whether some parameters work better for some datasets as compared to the other settings. The experiments are extensive in terms of parameter tweaking given that computation capabilities of the computer aren't exceeded.

Boosted Trees: The algorithm used for boosting trees was found at (http://www.mathworks.com/matlabcentral/fileexchange/21317-adaboost) and I considered boosting trees after 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024 and 2048 steps of boosting as described in (https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf) . The AdaBoost type of boosting was applied. The same procedure was applied on all three datasets.

Decision Trees: The function used for Decision Trees was found at (https://www.mathworks.com/help/stats/classification-trees-and-regression-trees.html), where the fitctree function can be used to classify and predict. For decision trees, the depth of the tree was controlled by manipulating the 'MinLeafSize' property of the tree.

Neural Networks: The algorithm used for training Neural Networks was found at (http://www.mathworks.com/products/neural-network/code-examples.html). The number of hidden units were varied and the following were used {1,2,4,8,32,128}.

2.2. Datasets

All the three datasets were quite huge to begin with. Running supervised learning algorithms on these datasets were taking hours to compute so to be economical, I took between 3% - 30% of dataset's that were taking a longer time to run on certain algorithms. This was also dependent on the size of the dataset such as COV_TYPE is a 581012x55 dataset, Letter.p2 is a 20000x17 dataset and Adult is a 32561x16 dataset. Even after taking 3% - 30% of the dataset, the computations still took a considerable amount of time. For COV_TYPE, in the boosting experiment only 3% of the set was used, since the boosting-tree computations were taking exceedingly long to carry out. For Decision Tree, 30% of the COV_TYPE dataset was used, and for both Boosting and Decision Tree calculations, multiple Training|Validation and Test set splits were tried.

Letter.p2 was converted into a binary classification by converting letters A-M into +1 and N-Z to -1 making it out to be a well-balanced problem.
COV TYPE has been converted to a binary problem by treating the largest class as the positive and the rest as negative. The largest class in this dataset is Lodgepole Pine and denoted by '2' in the last column of COV_TYPE. The largest class was assigned the label of +1 and the rest -1. 30% of the dataset was randomly extracted. It was then split 40% - 60%. Out of the % set aside for training, 90% of that is for training and the rest 10% as validation set with the rest of the for test set.

For boosting experiments on Letter.p2 and Adult, 30% of the original data was used while for COV_TYPE, 3% of the data was used. The results shown below for each data split variation are the best training and test error achieved after 2,4,8,16,32,64,128,256,512,1024 and 2048 steps of boosting on each dataset size variation of a particular dataset.

3. Experiment

For boosting experiments on Letter.p2 and Adult, 30% of the original data was used while for COV_TYPE, 3% of the data was used. The results shown below for each data split variation are the best training and test error achieved after 2,4,8,16,32,64,128,256,512,1024 and 2048 steps of boosting on each dataset size variation of a particular dataset.

Boosting – Letter.p2 - 30%

Training|Validation/Test Data	**Training error**	**Test error**
20-80%	60%	50%
50-50%	47%	52.1%
80-20%	62.5%	57.5%

Boosting – COV_TYPE – 3% of set used

Training|Validation/Test Data	**Training Error**	**Test Error**
80 % - 20 %	62.96	56.03
50 % - 50 %	77.78	51.72
20% - 80 %	77.04	44.14

Boosting – Adult – 30% of set used

Training|Validation/Test Data	**Training Error**	**Test Error**
80 % - 20 %	65.38 %	66.15 %

50 % - 50 %	69.23 %	70.77 %
20 % - 80 %	88.46 %	67.69 %

Boosting

Mean Training Error	**Mean Test Error**
74.89 %	57.34%

For Decision Tree experiments, 30% of the dataset was used for Letter.p2 and COV_TYPE while the whole dataset was used for ADULT. The optimum MinLeafSize parameter was used to control the depth of the tree and see how pruning affects the tree.

Decision Tree – Letter.p2 pruning- 30% of set used

Training|Validation/Test Data	**Training Error**	**Test Error**	**Optimum MinLeafSize Value**
80% - 20%	15.81	17.50	0.3551
50% - 50%	16.40	22.50	0.2021
20% - 80%	20.75	25.37	0.9661

Decision Tree – COV_TYPE pruning – 30% of set used

Training | Validation/Test Data	**Training Error**	**Test Error**	**Optimum MinLeafSize Value**
80% - 20%	23.73 %	26.07 %	33
50% - 50%	26.20 %	25.33 %	10.9210
20% - 80%	26.16 %	29.75 %	0.0671

Decision Tree – Adult pruning – Whole set

Training|Validation/Test Data	**Training Error**	**Test Error**	**Optimum MinLeafSize Value**
40% – 60 %	18.39 %	17.28 %	0.5221

Decision Tree

Mean Training Error	**Mean Test Error**
21.06 %	23.4 %

For Neural Networks, we used the entire set of ADULT AND LETTER.p2 but only used 10% of the COV_TYPE dataset due to long computation times. For the COV_TYPE dataset, I carried out a couple of different data splits since I was only using 10% of it. The Neural Network was trained at 1, 2, 4, 8, 32 and 128 hidden layers.

Neural Networks – Adult – Whole set used

Training|Validation/Test Data	**Training Error (mean squared error)**	**Test Error (mean squared error)**	**Hidden layers**
70% – 30%	0.4342	0.4402	1
70% – 30%	0.4018	0.4064	2
70% – 30%	0.3769	0.3869	4
70% – 30%	0.4161	0.4000	8
70% – 30%	0.4042	0.3976	32
70% – 30%	0.4239	0.3884	128

Neural Networks - COV_TYPE – 10% dataset used

Training|Validation/Test Data	Training Error (mean squared error)	Test Error (mean squared error)	Hidden layers
60% - 40%	2.432	2.351	1
60% - 40%	2.587	2.338	2
60% - 40%	2.176	2.309	4
60% - 40%	2.439	2.407	8
60% - 40%	2.254	2.265	32
60% - 40%	2.769	2.541	128

Training|Validation/Test Data	Training Error (mean squared error)	Test Error (mean squared error)	Hidden layers
40% - 60%	2.671	2.48	1
40% - 60%	2.586	2.483	2
40% - 60%	2.688	2.586	4
40% - 60%	2.653	2.697	8
40% - 60%	2.618	2.663	32

40% - 60%	3.646	3.283	128

Letter.p2 – Neural Networks - Whole set used

Training|Validation/Test Data	**Training Error (mean squared error)**	**Test Error (mean squared error)**	**Hidden layers**
60% - 40%	2.627	2.623	1
60% - 40%	2.319	2.157	2
60% - 40%	1.721	1.745	4
60% - 40%	1.562	1.519	8
60% - 40%	1.003	0.8835	32
60% - 40%	0.7241	0.6799	128

Neural Networks

Mean Training Error	**Mean Test Error**
1.830	1.669

	Letter.p2 Mean Training | Test Error	**ADULT Mean Training | Test Error**	**COV_TYPE Mean Training | Test Error**
Boosting	56.50% | 53.16%	50.63% | 74.35 %	73.59% | 50.63%
Decision Trees	17.65% | 21.79%	18.39% | 17.28%	25.36% | 27.05%
Neural Networks	1.6593% | 1.601%	0.4095% | 0.4032%	2.626% | 2.533%

4. Conclusion

The boosted trees performed poorly on the three chosen datasets and this could be due to a several number of reasons. Firstly, when calibrating boosted trees, only the steps of boosting were varied and no calibration with either Platt's method or Isotonic Regression was carried out. It is also important to note that the datasets could not be analyzed in their entirety due to time constraints and the results may also be affected due to that. I was expecting boosted trees to perform better than Decision Trees, with Neural Networks performing the best of the three. As also mentioned in the paper, out of the three learning algorithms Neural Nets performed the best and calibrating Neural Nets with different values of hidden layers did not result in a significant change in the Training and Testing error. Also keeping in mind that the score for each algorithm mentioned in the paper is normalized over eight metrics while in this paper only one metric is being observed, which could also produce dissimilar results i.e. boosting scores are less efficient than decision trees. In general, we know this not to be true but as mentioned in the paper, some models that perform poorly on average, but seem to be better models than boosted trees for the datasets used. To sum it up, neural networks were the best learning algorithm on the given dataset while boosting was the worst. It could also be that only ADABOOST was tried for boosting and that resulted in poor results. Other boosting algorithms might have provided us with more consistent results with the paper(**https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf)**.

5. References

http://www.mathworks.com/matlabcentral/fileexchange/21317-adaboost)

https://www.mathworks.com/help/stats/classification-trees-and-regression-trees.html

http://www.mathworks.com/products/neural-network/code-examples.html

https://www.cs.cornell.edu/~caruana/ctp/ct.papers/caruana.icml06.pdf

http://archive.ics.uci.edu/ml/datasets/Adult

https://archive.ics.uci.edu/ml/machine-learning-databases/covtype/

https://archive.ics.uci.edu/ml/datasets/letter+recognition