BMI 651

Joshua Burkhart

February 22, 2016

HW4: Writeup

(100 pts)

For this assignment, you will utilize the spam dataset (58 columns total, with 57 being features and last a factor : email or spam, and 4601 rows - representing emails)

You can download data at: http://statweb.stanford.edu/~tibs/ElemStatLearn/datasets/spam.data)

See details at: http://statweb.stanford.edu/~tibs/ElemStatLearn/datasets/spam.info.txt

Develop an R script that does the following:

1.

- Allows you to assess classifier you develop to a constant classifier (naïve case- always predicts the same class, no matter what the input features).
- For the constant classifier, it should output what fraction are actually spam, what the constant classifier predicts and the error rate of the constant classifier
- Must provide assessment of which approaches below out-performs the constant classifier

Our dataset contains 1813 rows labelled 1 (spam) and 2788 rows labelled 0 (not spam) for a 60.6% (non spam) to 39.4% (spam) split. As a constant (naive) classifier could achieve a 39.4% misclassification rate by outputting 0 (non spam) for each example, we'll hope to develop a classifier with a lower misclassification rate. [APPENDIX 1]

2.

- Splits the dataset at random into a training set and testing set (1:1 split)
- checks to ensure they are non-overlapping
- split requested is correct
- compares percentage of spam emails in each set.

We'll perform a 1:1 split to separate our training and test datasets. [APPENDIX 2A]

Let's assure the training and test datasets are non-overlapping. Interestingly, the original dataset appears to have 391 duplicated rows. We'll filter these out and resplit our data into training and test datasets. We'll also check the class (spam / non spam) distribution of the filtered spam dataset and observe it changes only slightly to 60.1% non spam, 39.9% spam. Finally, we see that our training and test datasets contain no intersecting (overlapping) rows. [APPENDIX 2B]

We'll make sure we split our training and test data correctly. [APPENDIX 2C]

We'll compare the percentage of spam in training & test datasets. [APPENDIX 2D]

3.

- Classification tree for training data that you prune by cross-validation.
- Script must generate plots for CV error versus tree size
- plot best tree and provide its error rate on the testing data.
- It should also summarize the variables on the tree.

Build classification tree for training dataset pruned following cross validation. Interestingly, K=10 fold cross validation shows deviance decreasing for each additional split, up to 11 (for a tree with 12 terminal nodes), the same number of splits used to build the original tree. Thus, nothing is actually pruned. That the tree stops after 11 splits is likely an artifact of the built-in heuristics of the R 'tree' library. [APPENDIX 3A]

Generate plots for cv error vs tree size. The deviance decreases for each of the 11 splits (resulting in 12 terminal nodes).[APPENDIX 3B]

Plot the tree [APPENDIX 3C]

Getting misclassification error on test dataset. Using only 11 splits, we were able to achieve a 10.1% misclassification rate using our 'pruned' tree. [APPENDIX 3D]

Summarize variables in 'pruned' tree. [APPENDIX 3E]

4.

- Bagging for ensemble of 100 trees for training data.
- Script must generate a plot of the importance of the variables, (for ensemble)
- provide error rate (ensemble) on the testing data.

Bagging for ensemble of 100 trees for training data [APPENDIX 4A]

plot of the importance of the variables, (for ensemble) [APPENDIX 4B]

Calculating test error rate... we see a slight reduction in test error after bagging, 10.1% to 9.36% error, a reduction of 0.74%. [APPENDIX 4C]

5.

- Boosting for ensemble of 100 trees for training data.
- Script must generate a plot of importance of the variables
- provide error rate (ensemble) on the testing data.

Boosting for ensemble of 100 trees for training data [APPENDIX 5A]

plot of the importance of the variables, (for ensemble) [APPENDIX 5B]

Calculating test error rate... we see our test error rate nearly cut in half after boosting, 9.36% to 5.13% error, a reduction of 4.23%. [APPENDIX 5C]