STAT 542 HW #3

Max Candocia

November 5, 2014

The dataset is a collection of emails, with 3,601 training emails and 1,000 testing emails. The variables refer to summary statistics of text in documents as well as punctuation and key words used in emails. The response $Y = 1$ refers to spam, and the response $Y = 0$ refers to non-spam.

Note that all of the errors and deviances are displayed in the final section for consiceness.

Problem 1

For the logistic regression models, there is an assumption that all of the variables measured are error-free, and that the response variable is a function of independent variables. Also, each observation is independent and the response variables are also independent of each other. Additionally, it is assumed that a linear combination of predictors is equal to the log odds of the probability of a given point belonging to a particular class.

The AIC model and BIC model tend to select finance-related words more. LASSO includes all variables except $W857$, $W415$, $Wparts$, and $Cparen$.

Here are the quantities of variables for each model.

- Full model: 57 variables
- AIC model: 41 variables
- BIC model: 26 variables
- LASSO: 53 variables

As expected, BIC is the most restrictive, and LASSO barely removes any variables, which is expected since it uses an L_1 penalty instead of an L_0 penalty.

Problem 2

For LDA, there is an assumption that members from each class are distributed according to a multivariate normal distribution with zero values for covariances between variables.

For RDA, there is an assumption that members from each class are distributed according to amultivariate normal distribution with possibly nonzero values for covariance matrices, but it also assumes that a method like QDA would be too biased, and tends to shrink the covariance matrix towards a diagonal one, making the assumption that variance bias is less of an issue.

For a parametric Naive Bayes model, the assumption is similar to that of LDA, but the covariance matrix has the same variances along the diagonal. This reduces the parameterization significantlly, but, as we will see, oversimpifies the model at the same time.

For a nonparametric Naive Bayes model, the requirement of the data being normally distributed is no longer used, but the other assumptions of independence are still used.

Problem 3

For the linear kernel, cost (C) values of 1 through 5 are tested, and 5-fold cross-validation is used for each of them to determin the optimal value. A value of $C = 1$ was used for both of them.

For the quadratic kernel, the same cost values are test, but the scale parameter is also tested on a range from 1 to 5, using **expand.grid()** to produce the different combinations. 5-fold cross-validation is also used to determine these parameters. Both the cost and scale parameter were set to 1.

For the gaussian kernel, the same criteria are used as for the linear kernel, except values of 1 through 15 are tested instead of 1 through 5. Both gaussian kernels used $C = 8$ for the cost parameter.

Below are the number of support vectors for each model

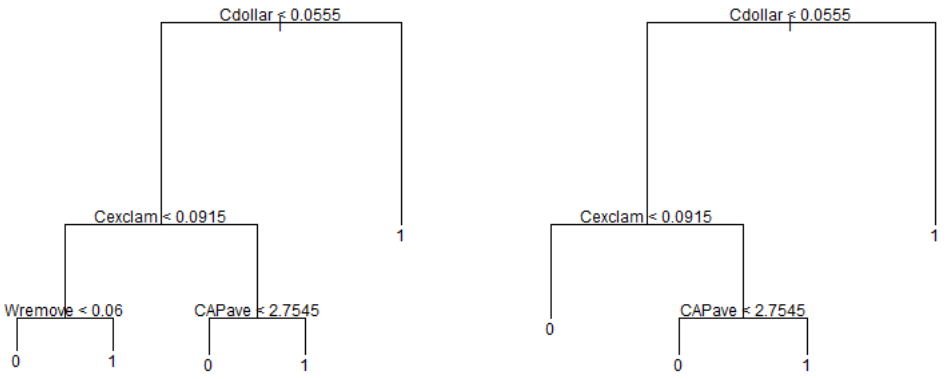
	Model	Number of Support Vectors
1	Linear Full	727
2	Linear Selection	728
3	Quadratic Full	617
4	Quadratic Selection	606
5	Gaussian Full	1031
6	Gaussian Selection	1039

With SVMs, the probabilities are derived from a trained sigmoid function after the SVM is trained. In this case, a Laplacian (double-exponential) distribution is used. The estimation procedure of the sigmoid curve can be especially complicated for non-linear SVMs, but the general idea is that the probability is generated from a distribution calculated in parallel to the SVM. The values for the parameters in the SVM can be found in the **prob.model** attribute of the SVM object in R. The values used in the sigmoidal function are a bit more complicated than those described in Pratt's 1999 paper, as normally only $p + 1$ points are on a margin (i.e., support vectors), but the algorithm utilized by R treats the dimensionality differently. 3-fold cross-validation is used in order to determine the probabilistic model.

Problem 4

The measure for impurity in the classification tree is deviance, or negative log likelihood. When pruning the tree, it is also used, but with 5-fold cross-validation determining the cutoff level. The value at which deviance (and BIC, incidentally) is optimal is 15 leaves.

When cutting down to a tree size of 4 from 5, the node with the lowest alpha is on a split with the $Wremove$ variable. The value of this alpha is 203.0072. Below is a plot of both of them side-by-side.

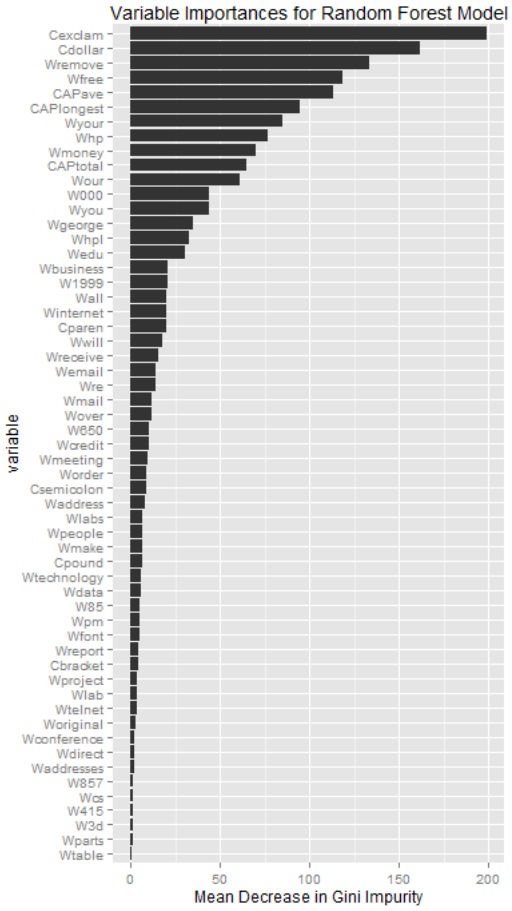


Classification trees of sizes 5 and 4, respectively

Problem 5

m is chosen such that $m = \lfloor \sqrt{p} \rfloor$, where p is the number of predictor variables. In this case, $m = \sqrt{57}$ rounded down, or 7.

Below is a graph of variable importances.



Variable importances in the random forest model

While the correlation between important variables in each model seems positive, random forest does not exclude any variables (granted enough trees have been grown), thus giving marginal importance to the less significant variable. Additionally, it shows which variables are the most important, and given that random forest's predictions tend to be more accurate (as will be seen later in the results section), there is more significance in the importances.

Problem 6

With AdaBoost, I initially use 5,000 trees. In order to find the optimal number of boosting iterations, I had to play around with the training fraction of the dataset, as well as the shrinkage parameter, which were ultimately 0.7 and 0.01, respectively. In order to find the optimal number, I used 5-fold cross-validation, and then found the optimal value using **gbm.perf**. The below graph shows the cross-validation error on the green curve.

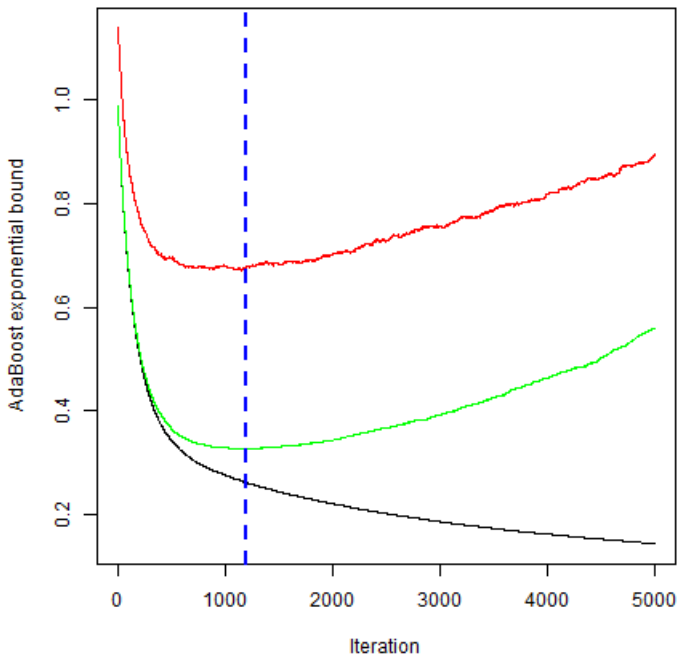


Figure 1: Error curve for AdaBoost, with the red curve representing hold-out error, and green representing cross-validation error

Problem 7

The number of layers in this model is 1, and the chosen number of neurons is 15, with a decay rate of 0.12. These values were obtained from 2 grid searches. Also note that all models assume the response variable is a class variable with two possible values.

Misclassification and Log-Likelihoods

Below is the table of misclassification rates and log-likelihoods. A few things to note:

- Naive Bayes is not very good for this model and its assumptions oversimplify the actual nature of the data.
- Among logistic models, the full logistic had the best misclassification rate, but the second worst log-likelihood. The latter is not surprising, although it suggests that the BIC model was underfitting.
- RDA and Naive Bayes are not well-described by log-likelihood. This may be due partly to the high misclassification rates and incorrect assumptions, leading to probabilities of 1 and 0.
- AdaBoost and neural networks work alright, but their performance is not particularly noticeable in terms of misclassification. AdaBoost's log-likelihood is the second-highest, though, which is notable.
- Random forest is the best algorithm in terms of misclassification and log-likelihood among all models. Additionally, it is among the fastest, next to the full logistic model and LDA.

Model	Misclassification Rate	Log-Likelihood
AIC Logistic	0.0700	-266.3993
BIC Logistic	0.0820	-278.0324
Lasso Logistic	0.0750	-255.3922
Full Logistic	0.0670	-274.7648
LDA w/o Selection	0.1200	-287.7993
LDA with Selection	0.1220	-287.4724
RDA w/o Selection	0.1220	-Inf
RDA with Selection	0.3400	-Inf
Bayes NP w/o Selection	0.4210	-Inf
Bayes NP with Selection	0.3760	-Inf
Bayes Parametric w/o Selection	0.2820	-Inf
Bayes Parametric with Selection	0.2610	-Inf
Linear SVM w/o Selection	0.0820	-250.2101
Linear SVM with Selection	0.0750	-250.7703
Quadratic SVM w/o Selection	0.1310	-434.4725
Quadratic SVM with Selection	0.1320	-Inf
Gaussian SVM w/o Selection	0.0720	-214.9323
Gaussian SVM with Selection	0.0760	-219.6775
Tree Model (5 leaves)	0.1480	-391.4712
Tree Model (15 leaves)	0.0930	-263.4563
Random Forest	0.0540	-170.6398
Adaboost	0.0710	-181.6480
Neural Network (1 layer, 15 Nodes)	0.0740	-252.6968