Email Classification Using Classification Trees

Jasmine Barrera, Joe Sanchez, Aditya Ranade

March 25, 2019

1 Introduction

These days people get a lot of spam emails (the emails which have advertisements, money making schemes, chain letters, etc.) which can be a serious problem for the email company. If the email company does not put a system in place to filter out the spam emails, it can lose its customers and that could have serious implications on the company. In this project, we employ the classification and regression tree (CART) approach to classify a given email as spam or non spam based on various attributes like percentage of the word "free" in the email, percentage of exclamation marks in the email, etc. on the spam base data set available on the UCI Machine Learning Repository. In the database, we have 4601 observations on 58 variables out of which 1813 were considered spam and 2788 were considered as non spam. The data is readily available to us, hence is a secondary dataset with a mix of continuous and categorical variable.

2 Method

The selection of the best tree is done through the cross validation error. The tree with the lowest cross validation error is considered the best tree. However, the best tree might have too many splits and might be very difficult to understand. Hence, we find the optimal tree such that the cross validation error is slightly higher than the optimal tree but the splits are comparatively less. The process to select the optimal tree based on testing data set with 10 cross fold validation and the one standard error rule as follows.

Initially we consider the full tree and the list of sub trees based on training data set incorporating 10 fold cross validation. The optimal subtree is selected as the smallest subtree which has the cross validation error less than one standard error greater than the minimum cross validated error. Then we pass the testing sample through the optimal tree and calculate the misclassification error rate. An ideal tree model should have a low misclassification error rate on the test sample.

3 Analysis and Results

3.1 Classifier 1

For classifier 1, the optimal tree is tree 10 with 11 splits and 12 terminal nodes and the corresponding misclassification rate of the optimal tree is 10.87%. The optimal tree has 53 false positive cases which means 53 emails out of 1535 test samples were predicted to be spam but were observed to be not spam in reality. The tree has 114 false negative cases which means 114 out of 1535 test samples were predicted to be not spam but were observed to be spam in reality. Hence the optimal tree for classifier 1 has 3.45% false positive error rate and 7.43% false negative error rate. Please note if we follow the 1 SE rule strictly, the optimal tree is tree 14 with 19 splits and 20 terminal nodes with misclassification rate of 9.7%, false positive and false negative error rates 3.65% and 6.05% respectively.

For the first classifier we plot a subtree of the optimal tree with 7 splits and 8 terminal nodes which can be seen in figure 1 in the figures section. This subtree has total misclassification error rate 12.18% with false positive and false negative error rates 3.45% and 8.73% respectively.

Some of the variables used in the construction of the optimal tree for the first classifier are cfexc, crllongest, crlaverage, cfdollar, wffree, wfyour, wfremove, crltotal, wfhp, wfmoney, wfall, wfhpl, wfgeorge, wf000, wfinternet, wflabs, wf857.

The cross validation estimates of errors and training errors of sequence of pruned trees against the trees' complexity for the first classifier can be found in figure 3 in the figures section.

3.2 Classifier 2

For classifier 2, the optimal tree is tree 10 with 12 splits and 13 terminal nodes and the corresponding misclassification rate of the optimal tree is 15.30%. The optimal tree has 14 false positive cases which means 14 emails out of 1535 test samples were predicted to be spam but were observed to be not spam in reality. The tree has 221 false negative cases which means 221 out of 1535 test samples were predicted to be not spam but were observed to be spam in reality. Hence the optimal tree for classifier 2 has 0.91% false positive error rate and 14.39% false negative error rate. Please note if we follow the 1 SE rule strictly, the optimal tree is tree 15 with 24 splits and 25 terminal nodes with misclassification rate of 13.16%, false positive and false negative error rates 1.04% and 12.12% respectively.

For the second classifier we plot a subtree of the optimal tree with 7 splits and 8 terminal nodes which can be seen in figure 2 in the figures section. This subtree has total misclassification error rate 17% with false positive and false negative error rates 0.78% and 16.22% respectively.

Some of the variables used in the construction of the optimal tree for the second classifier are wfremove, crllongest, wf000, wfyour, crltotal, cfexc, crlaverage, wfgeorge, wfyou, wfhp, wfhpl, wfall, wfmoney, wfaddresses, wf415, wf857, cfdollar, wfcredit, wf85, wforder, wffree, wfmake, wfmail, wfre, wfreceive.

The cross validation estimates of errors and training errors of sequence of pruned trees against the trees' complexity for the second classifier can be found in figure 4 in the figures section.

4 Conclusion

Before comparing the cross validation estimates of errors of sequence of pruned trees against the trees' complexity, it is worthy to note that for classifier 1 (figure 3 in Figures section), the crossvalidation error is slightly more than training error which is on expected lines. Howerver for classifier 2 (figure 4 in Figures section), the crossvalidation error is considerably higher than the training error, which indicates classifier 2 model is overfitted.

Comparing the cross validation estimates of errors of sequence of pruned trees against the trees' complexity for the first and second classifier from figure 3 and 4, indicates the cross validation error for the first classifier has a smooth decreasing curve as compared to the cross validation error for the second classifier. Also, the overall curve for the first classifier is smoother than the second classifier. However, a huge decrease in the cross validation error is observed as for the second classifier as compared to the first classifier. The overall cross validation error is significantly lower for the first classifier as compared to the second classifier for trees with all the sizes.

Comparing the training errors of sequence of pruned trees against the trees' complexity for the first and second classifier from figure 3 and 4, it appears the training error for the first classifier has a big drop as compared to the training error for the second classifier. However, when we observe the scale, we can see the change is approximately same for both the classifiers.

If we go by just the misclassification error, we will use classifier 1 since it has lower misclassification error. However, if we are concerned with avoiding false positive cases, which means we want to avoid cases where a non spam email is classified as a spam email as that might cause us to miss out on important and authentic emails, we introduce a high penalty for the false positive case. This classifier then reduces the false positive cases but, in the process, the false negative cases increase. This also makes the model overfit. So, the overall misclassification error increases by some amount and hence the second classifier has overall misclassification error greater than the first classifier. Depending on the demands, we can we can choose classifier 1 or 2 accordingly.

5 Figures

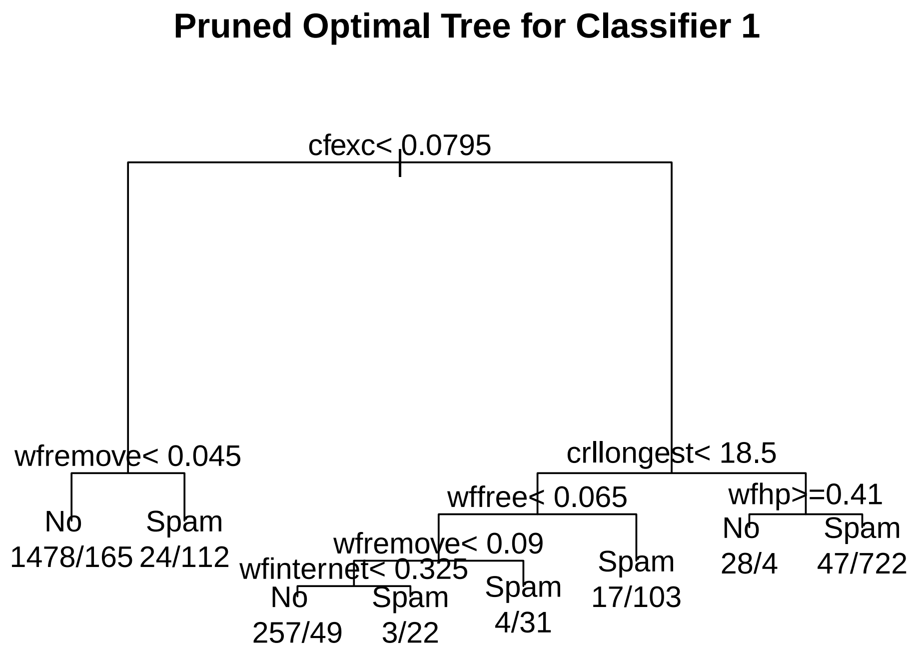

Figure 1: Subtree of the optimal tree for first classifier

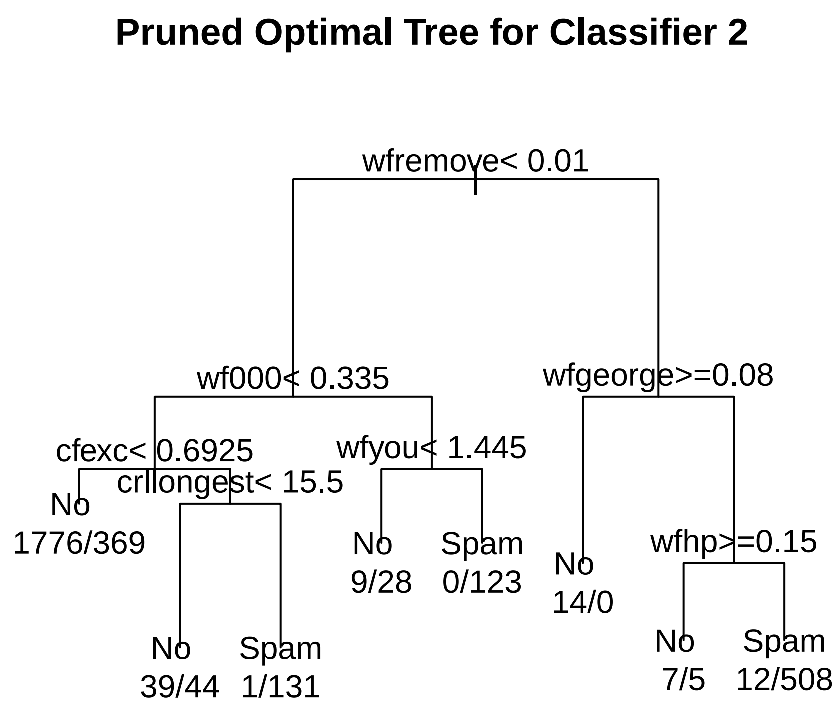

Figure 2: Subtree of the optimal tree for second classifier

Figure 3: Crossvalidation estimates of errors and training errors for first classifier

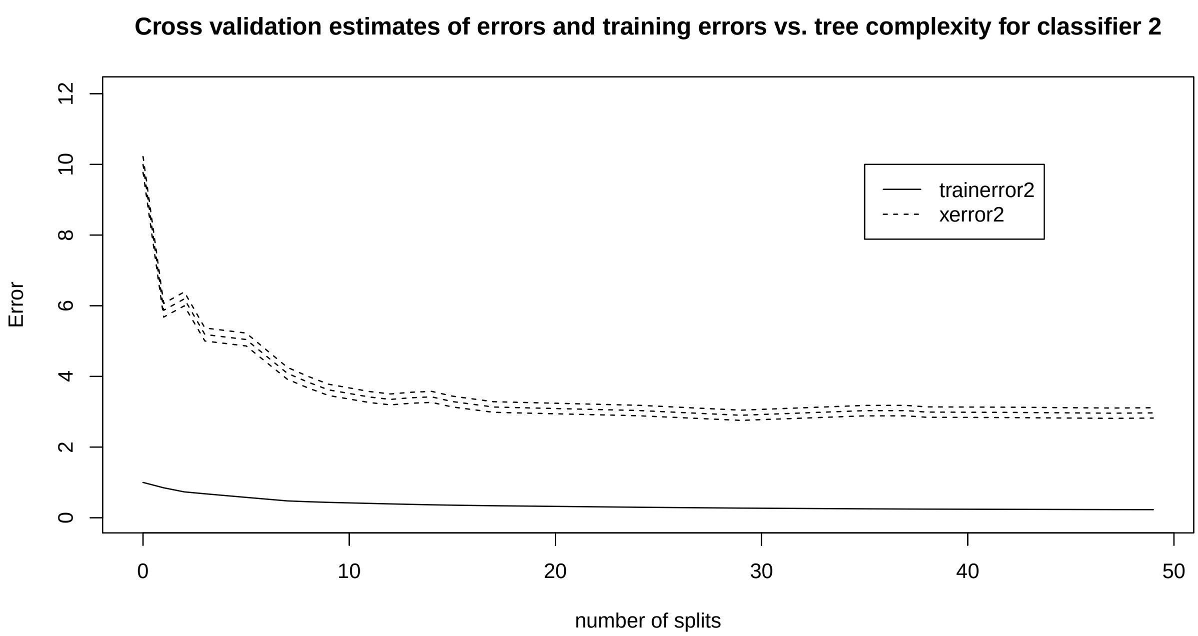

Figure 4: Crossvalidation estimates of errors and training errors for second classifier

Appendix A: Contribution

Basically all three of us worked on the entire project, however the majority work done is as follows

Coding: Jasmine Barrera and Joe Sanchez

Report writing: Aditya Ranade

Appendix B: R Code

```r
library(rpart)
library(rpart.plot)
spam.data <- read.table("https://edoras.sdsu.edu/~jjfan/sta702/spamdata.txt",
                        sep=",", na.strings="NA")
dim(spam.data)  # 4601   58
names(spam.data) <- c("wfmake", "wfaddress", "wfall", "wf3d", "wfour",
                      "wfover", "wfremove", "wfinternet", "wforder", "wfmail",
                      "wfreceive", "wfwill", "wfpeople", "wfreport", "wfaddresses",
                      "wffree", "wfbusiness", "wfemail", "wfyou", "wfcredit", "wfyour",
                      "wffont", "wf000", "wfmoney", "wfhp", "wfhpl", "wfgeorge", "wf650",
                      "wflab", "wflabs", "wftelnet", "wf857", "wfdata", "wf415", "wf85",
                      "wftechnology", "wf1999", "wfparts", "wfpm", "wfdirect", "wfcs",
                      "wfmeeting", "wforiginal", "wfproject", "wfre", "wfedu", "wftable",
                      "wfconference", "cfsc", "cfpar", "cfbrack", "cfexc", "cfdollar",
                      "cfpound", "crlaverage", "crllongest", "crltotal", "spam")
table(spam.data$spam)  #0:2788 (nonspam); 1:1813 (spam) = data for spam column
spam.data$spam <- factor(spam.data$spam, levels=0:1,
                         labels=c("No", "Spam"))


# split data into a training sample and a test sample, using stratified sampling
set1<-spam.data[spam.data$spam=="Spam",] #spam dataset
set0<-spam.data[spam.data$spam=="No",] #nonspam dataset
dim(set1)  # 1813   58
dim(set0)  # 2788   58
1813*2/3 #1208.667 emails = spam training
2788*2/3 #1858.667 emails = nonspam training

set.seed(858)
training1<-sample(1:1813,1208) #samples w/out replacement 1208 items from 1:1813
test1<-(1:1813)[-training1] #takes (1813*1/3) items from 1:1813
sum((1:1813)==sort(c(training1,test1))) # sort data to see if sum is same as 1:1813

training0<-sample(1:2788,1858)
test0<-(1:2788)[-training0]
sum((1:2788==sort(c(training0,test0)))) #2788


train<-rbind(set1[training1,],set0[training0,]) #combines training data for 0 and 1
test<-rbind(set1[test1,], set0[test0,])
dim(train)  # 3066   58
dim(test)   # 1535   58
3066+1535 #4601 = total
1208+1858 #3066 = training total
4601 - 3066 #1535 = testing total

###### First Classifier #######
# tree growing and pruning with training data (aka fit classification tree to spam data using 10fold xvalidation)
my.control <- rpart.control(cp=0, xval=10) #cp = 0 means form largest tree; 10-fold xvalid
fit1<- rpart(spam ~ ., data=train, method="class",
             control=my.control)  #this makes our largest tree and gives subtree sequence

```

```
# plot the tree that corresponds to cp=0 (the largest, unpruned tree):
plot(fit1, margin=0.1)
text(fit1, use.n=T) #gives text to tree
title("Largest Tree for Classifier1")
printcp(fit1)  #recall, cp = cost complexity parameter
plotcp(fit1)

#tree 16 has absolutely smallest xerror (0.22765)
#0.22765 + 0.013098 (get 0.240748; look for xerror smaller than this)
#strictly following the 1SE rule, the optimal tree is tree 14
#but if we want simpler tree, we could go with tree 10.


#plot of xval estimates of error and training error against tree complexity
numsplits <- fit1$cptable[,2] #assigns the respective cptable values to variables names
trainerror <- fit1$cptable[,3]
xerror <- fit1$cptable[,4]
xstd <- fit1$cptable[,5]

plot(numsplits, trainerror, ylim=c(.05, 1.2), type="l") #plot training error (solid line)
lines(numsplits, xerror, lty =2) #plot xvalid error (dashed lines)
lines(numsplits, xerror-xstd, lty=2) #lower error bar for xvalid error
lines(numsplits, xerror+xstd, lty=2) #higher error bar for xvalid error
title("Cross-validation Error Estimates and Training Error for Classifier1")
legend(40, 1, c("trainerror","xerror"), lty=c(1,2)) #coordinates are for top right corner of legend

#get optimal tree by pruning (tree 10)
fit1pruned <- prune(fit1, cp=0.007)
print(fit1pruned)
plot(fit1pruned, margin=0.1)
text(fit1pruned, use.n=T)
title("Optimal Tree for Classifier1")
summary(fit1pruned)
summary(fit1, cp=0.007) #gives same result as above!

#################misclassification error rate for tree 7########################
#get subtree of optimal tree by pruning (tree 7)
fit1bpruned <- prune(fit1, cp=0.0135)
printcp(fit1bpruned)
plot(fit1bpruned, margin=0.1)
text(fit1bpruned, use.n=T)
title("Pruned Optimal Tree for Classifier1")
summary(fit1bpruned)
summary(fit1, cp=0.0135) #gives same result as above!

#running test data down pruned optimal tree (tree 7)
pred1<-predict(fit1bpruned,newdata=test,type="class")
error1<-table(test$spam,pred1)[1,2]+table(test$spam,pred1)[2,1]
error1 #total misclassification error
errorrate1<-error1/length(test$spam)
errorrate1 # total misclassification error rate=12.18%


#details of the misclassification error table (extra)
falserror1 <- table(test$spam,pred1) #table with number of false positive and false negative
falserror1 #930 non and 605 spam (test); 1011 non and 524 spam (pred1), where test and pred1 have total size 1535
falserrorate1 <- falserror1/length(test$spam)
falserrorate1 #divided false errors above by sample size 1535
#0.03452769 is the false positive error rate
#0.08729642 is the false negative error rate

#################misclassification error rate for tree 10########################
```

```
#running test data down pruned optimal tree
pred1a<-predict(fit1pruned,newdata=test,type="class")
error1a<-table(test$spam,pred1a)[1,2]+table(test$spam,pred1a)[2,1]
error1a #total misclassification error
errorrate1a <-error1a/length(test$spam)
errorrate1a # total misclassification error rate=10.87%


#details of the misclassification error table (extra)
falserror1a <- table(test$spam,pred1a) #table with number of false positive and false
    negative
falserror1a #930 non and 605 spam (test); 877 non and 491 spam (pred1a), where test and
    pred1 have total size 1535
falserrorate1a <- falserror1a/length(test$spam)
falserrorate1a #divided false errors above by sample size 1535
#0.03452769 is the false positive error rate
#0.07426710 is the false negative error rate

#################misclassification error rate for tree 14##########################
fit1cpruned <- prune(fit1, cp=0.003)
#running test data down pruned optimal tree
pred1c<-predict(fit1cpruned,newdata=test,type="class")
error1c<-table(test$spam,pred1c)[1,2]+table(test$spam,pred1c)[2,1]
error1c #total misclassification error
errorrate1c <-error1c/length(test$spam)
errorrate1c # total misclassification error rate=9.70%


#details of the misclassification error table (extra)
falserror1c <- table(test$spam,pred1c) #table with number of false positive and false
    negative
falserror1c #930 non and 605 spam (test); 874 non and 512 spam (pred1a), where test and
    pred1 have total size 1535
falserrorate1c <- falserror1c/length(test$spam)
falserrorate1c #divided false errors above by sample size 1535
#0.03648208 is the false positive error rate
#0.06058632 is the false negative error rate



###### Second Classifier #######
# tree growing and pruning with training data (aka fit classification tree to spam data
    using 10fold xvalidation)
my.control2 <- rpart.control(cp=0, xval=10) #cp = 0 means form largest tree; 10-fold xvalid
lmat <- matrix(c(0,10,1,0), byrow=T, nrow=2) #loss matrix, where we are penalize false
    positive 10 times more than false negative errors
fit2<- rpart(spam ~ ., data=train, method="class",
             parm=list(loss=lmat),
             control=my.control2)  #this makes our largest tree and gives subtree sequence

# plot the tree that corresponds to cp=0 (the largest, unpruned tree):
plot(fit2, margin=0.1)
title("Largest Tree for Classifier2")
text(fit2, use.n=T) #gives text to tree
printcp(fit2)  #recall, cp = cost complexity parameter
plotcp(fit2)
#tree 16 has absolutely smallest xerror (2.9007)
#2.9007 + 0.14490 (get 3.0456; look for xerror smaller than this)
#strictly following the 1SE rule, the optimal tree is tree 15; but if want smaller, could
    go with tree 10

#plot of xval estimates of error and training error against tree complexity
numsplits2 <- fit2$cptable[,2] #assigns the respective cptable values to variables names
trainerror2 <- fit2$cptable[,3]
xerror2 <- fit2$cptable[,4]
```

```r
xstd2 <- fit2$cptable[,5]

plot(numsplits2, trainerror2, ylim=c(.05, 12), type="l") #plot training error (solid line)
lines(numsplits2, xerror2, lty =2) #plot xvalid error (dashed lines)
lines(numsplits2, xerror2-xstd2, lty=2) #lower error bar for xvalid error
lines(numsplits2, xerror2+xstd2, lty=2) #higher error bar for xvalid error
title("Cross-validation Error Estimates and Training Error for Classifier2")
legend(35, 10, c("trainerror2","xerror2"), lty=c(1,2)) #coordinates are for top right
    corner of legend

#get optimal tree by pruning (tree 10)
fit2pruned <- prune(fit2, cp=0.0135) #tree 10 with 12 splits and 13 terminal nodes
print(fit2pruned)
plot(fit2pruned, margin=0.1)
text(fit2pruned, use.n=T)
title("Optimal Tree for Classifier2")
summary(fit2pruned)
summary(fit2, cp=0.0135) #gives same result as above!

#get subtree of optimal tree by pruning (tree 6)
fit2bpruned <- prune(fit2, cp=0.035)
print(fit2bpruned)
plot(fit2bpruned, margin=0.1)
text(fit2bpruned, use.n=T)
title("Pruned Optimal Tree for Classifier2")
summary(fit2bpruned)
summary(fit2, cp=0.035) #gives same result as above!

#################misclassification error rate for tree 6#########################
#running test data down pruned optimal tree
pred2<-predict(fit2bpruned,newdata=test,type="class")
error2<-table(test$spam,pred2)[1,2]+table(test$spam,pred2)[2,1]
error2 #total misclassification error
errorrate2<-error2/length(test$spam)
errorrate2 #total misclassification error rate = 17%

#details of the misclassification error table (extra)
falserror2 <- table(test$spam,pred2) #table with number of false positive and false
    negative
falserror2 #930 non and 605 spam (test); 1167 non and 368 spam (pred1), where test and
    pred1 have total size 1535
falserrorate2 <- falserror2/length(test$spam)
falserrorate2 #divided false errors above by sample size 1535
#0.00781759 is the false positive error rate
#0.16221498 is the false negative error rate




#################misclassification error rate for tree 10#########################
#running test data down pruned optimal tree
pred2b<-predict(fit2pruned,newdata=test,type="class")
error2b<-table(test$spam,pred2b)[1,2]+table(test$spam,pred2b)[2,1]
error2b #total misclassification error
errorrate2b<-error2b/length(test$spam)
errorrate2b #total misclassification error rate

#details of the misclassification error table (extra)
falserror2b <- table(test$spam,pred2b) #table with number of false positive and false
    negative
falserror2b #930 non and 605 spam (test); 916 non and 384 spam (pred2b), where test and
    pred1 have total size 1535
falserrorate2b <- falserror2b/length(test$spam)
falserrorate2b #divided false errors above by sample size 1535
#0.00781759 is the false positive error rate
```

```
#0.16221498 is the false negative error rate

#################misclassification error rate for tree 15#########################
fit2cpruned <- prune(fit2 , cp=0.005)
printcp(fit2cpruned)
#running test data down pruned optimal tree
pred2c<-predict(fit2cpruned,newdata=test,type="class")
error2c<-table(test$spam,pred2c)[1,2]+table(test$spam,pred2c)[2,1]
error2c #total misclassification error
errorrate2c<-error2c/length(test$spam)
errorrate2c #total misclassification error rate = 13.16%

#details of the misclassification error table (extra)
falserror2c <- table(test$spam,pred2c) #table with number of false positive and false
    negative
falserror2c #930 non and 605 spam (test); 914 non and 419 spam (pred2c), where test and
    pred1 have total size 1535
falserrorate2c <- falserror2c/length(test$spam)
falserrorate2c #divided false errors above by sample size 1535
#0.01042345 is the false positive error rate
#0.12117264 is the false negative error rate
```

Listing 1: The R code used for analysis