STATS 415 HW 7

Jiahao Cheng

March 2025

Problem 1

(a) As Figure 1 shows, 9.13% emails in the test set were misclassified by the optimal tree. 12.62% spam emails in the test set were misclassified, 6.77% non-spam emails in the test set were misclassified.

```
> train <- read.csv("spam-train.txt", header = FALSE)
> test <- read.csv("spam-test.txt", header = FALSE)
> library(tree)
> train$V58 <- as.factor(train$V58)
> tree.train = tree(V58~., train)
> summary(tree.train)

Classification tree:
tree(formula = V58 ~ ., data = train)
Variables actually used in tree construction:
[1] "V52" "V7"  "V24" "V16" "V23" "V27" "V53" "V55" "V25"
Number of terminal nodes:  13
Residual mean deviance:  0.4927 = 1505 / 3054
Misclassification error rate: 0.08771 = 269 / 3067
> Spam.test= test$V58
> tree.pred = predict(tree.train, test ,type="class")
> table(tree.pred,Spam.test)
         Spam.test
tree.pred   0   1
        0 854  78
        1  62 540
> (78 + 62) / (854 + 78 + 62 + 540)
[1] 0.09126467
> 78 / (78 + 540)
[1] 0.1262136
> 62 / (854 + 62)
[1] 0.06768559
```

Figure 1: Classification Tree Result

(b) Figure 2 shows the code, Figure 3 shows the sub-tree with 8 terminal nodes. Variables shown below are used in tree construction.
word_freq_remove(V7),
word_freq_free(V16),
word_freq_money(V24),
char_freq_!(V52),
char_freq_$(V53),
capital_run_length_average(V55)

```
> set.seed(7)
> cv.spam = cv.tree(tree.train, FUN=prune.misclass)
> #names(cv.spam)
> #cv.spam
> #par(mfrow=c(1,2))
> #plot(cv.spam$size,cv.spam$dev / length(train),ylab="cv error", xlab="size",type="b")
> #plot(cv.spam$k, cv.spam$dev / length(train),ylab="cv error", xlab="k",type="b")
> prune.spam = prune.misclass(tree.train, best=8)
> par(mfrow=c(1,1))
> plot(prune.spam)
> text(prune.spam,pretty=0)
```

Figure 2: Sub-tree code

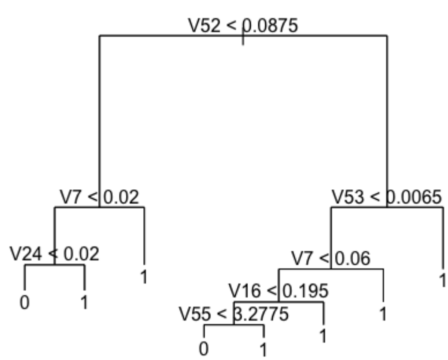

Figure 3: Sub-tree plot

(c) Figure 4 shows the code result. 3.46% emails in the test set were misclassified. 3.24% spam emails in the test set were misclassified, 3.60% non-spam emails in the test set were misclassified.
Figure 5 shows the code to determine which variables are most important. Figure 6 shows the result plot. From the plot, we can say that word_freq_remove(V7) and char_freq_!(V52) are the most important variables.
A larger value of m will incur a larger computational burden, and also create a model with low bias because it imposes few restrictions on the trees.
A small value of m introduces more bias, but it reduces variance by decreasing the similarity (correlation) between trees.

```
> library(randomForest)
> rf <- randomForest(
+   V58~., # Model formula
+   data=train, # Training data
+   mtry=ncol(train)-1, # Use all columns
+   importance=TRUE) # Return feature importance measures
> rf_preds <- predict(rf, newdata=test)
> table(rf_preds, Spam.test)
        Spam.test
rf_preds   0   1
       0 883  20
       1  33 598
> (20 + 33) / (883 + 20 + 33 + 598)
[1] 0.0345502
> 33 / (883 + 33)
[1] 0.0360262
> 20 / (598 + 20)
[1] 0.03236246
```

Figure 4: RandomeForest

```
> df <- as.data.frame(importance(rf))
> df$Variable <- row.names(df)
> df_sorted <- df[order(df$MeanDecreaseGini, decreasing = TRUE), ]
> head(df_sorted, 10)
            0        1 MeanDecreaseAccuracy MeanDecreaseGini Variable
V52  72.21696 57.63386             88.20588        375.33004      V52
V7  103.01546 51.63853            112.92336        204.23425       V7
V53  35.33228 26.95347             37.38934        192.84728      V53
V25  28.54003 85.34892             71.23328         97.50733      V25
V16  55.32040 51.70929             68.67131         72.38767      V16
V55  38.32835 25.93708             38.61886         64.26254      V55
V57  31.45054 19.99643             36.02775         58.07273      V57
V56  28.51065 29.15934             38.63348         44.76727      V56
V24  27.92409 31.20139             37.52922         40.61251      V24
V5   34.24882 36.98341             44.89756         27.72236       V5
> varImpPlot(rf, n.var=10)
```

Figure 5: Variable Importance

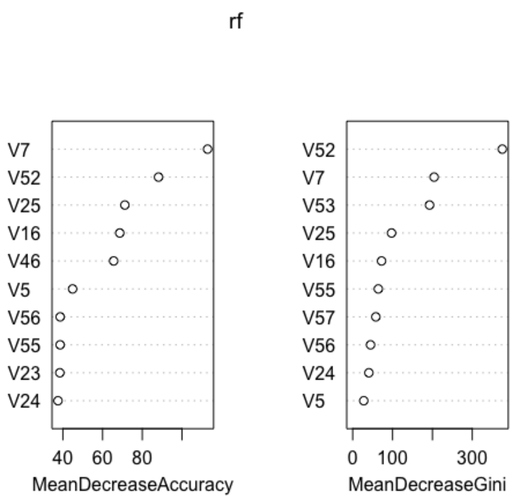

Figure 6: Variable Importance Plot