Spambase

```
library("tree")

library("adabag")

library("randomForest")

spam <- read.csv("spambase.data",header = F)

str(spam)

names(spam)  <-   c( "word_freq_make","word_freq_address","word_freq_all",
"word_freq_3d","word_freq_our","word_freq_over","word_freq_remove","word_freq
_internet","word_freq_order","word_freq_mail","word_freq_receive","word_freq_
will","word_freq_people","word_freq_report","word_freq_addresses","word_freq_
free","word_freq_business","word_freq_email","word_freq_you",word_freq_credit
","word_freq_your","word_freq_font","word_freq_000","word_freq_money","word_f
req_hp","word_freq_hpl","word_freq_george","word_freq_650","word_freq_lab","w
ord_freq_labs","word_freq_telnet","word_freq_857","word_freq_data","word_freq
_415","word_freq_85","word_freq_technology","word_freq_1999","word_freq_parts
","word_freq_pm","word_freq_direct","word_freq_cs","word_freq_meeting","word_
freq_original","word_freq_project","word_freq_re","word_freq_edu","word_freq_
table","word_freq_conference","char_freq_;","char_freq_(","char_freq_[","char
_freq_!","char_freq_$","char_freq_#","capital_run_length_average","capital_ru
n_length_longest","capital_run_length_total","spam")

spam$spam <- as.factor(spam$spam)

spam <- data.frame(spam)
```

(a)

1) What fraction of the e-mails are actually spam?

39.4% of the e-mails are actually spam

```
prop.table(table(spam$spam))
      email(0)     spam(1)
      0.6059552 0.3940448
```

2) What should the constant classifier predict?

Email.

3) What is the error rate of the constant classifier?

39.4 %

b) Divide the data set at random into a training set of 2301 rows and a testing set of 2300 rows. Check that the two halves do not overlap (use intersect() function), and that they have the right number of rows. What fraction of each half is spam?

```
indx <- sample(1:nrow(spam),2301,replace = FALSE)

TrainData_spam <- spam[indx,]

TestData_spam <- spam[-indx,]

intersect(TrainData_spam,TestData_spam)

  data frame with 0 columns and 0 rows
```

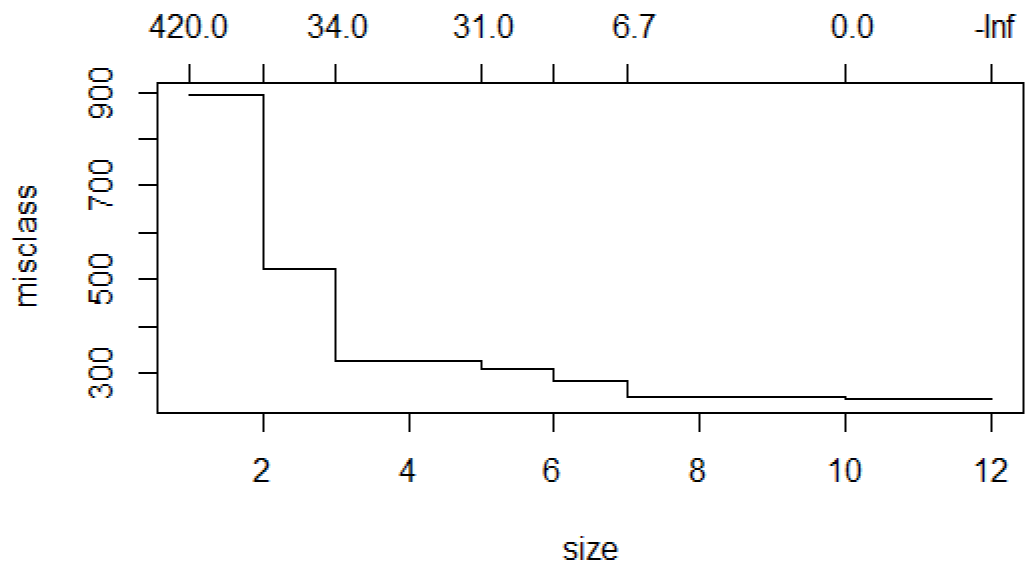
```
prop.table(table(TrainData_spam$spam))
      email(0)   spam(1)
      0.603216 0.396784

prop.table(table(TestData_spam$spam))
      email(0)   spam(1)
      0.6086957 0.3913043
```

c) 1) Fit a classification tree to the training data. Prune the tree by cross-validation.Include a plot of the CV error versus tree size, a plot of the best tree, and its error rate on the testing data. Which variables appear in the tree?

```
spam.tree <- tree(spam ~ ., data=TrainData_spam)
plot(spam.tree)
text(spam.tree,cex = 0.6)


spam.tree.cv <- cv.tree(spam.tree , FUN = prune.misclass)
```

```
> spam.tree.cv
$size
[1] 12 10  7  6  5  3  2  1

$dev
[1] 242 242 247 283 310 326 523 893

$k
[1]       -Inf   0.000000   6.666667  22.000000  31.000000  33.500000 129.000000 425.000000

$method
[1] "misclass"

attr(,"class")
[1] "prune"         "tree.sequence"
```

```
plot(spam.tree.cv)
```




```
size <- spam.tree.cv$size[which.min(spam.tree.cv$dev)]
size
[1] 12
spam.tree.prune <- prune.tree(spam.tree,best = size)
plot(spam.tree.prune)
text(spam.tree.prune,cex=0.5)
summary(spam.tree.prune)
```

Decision after Pruning

```
summary(spam.tree.prune)

predict.tree <-
predict(spam.tree.prune,newdata=TestData_spam,type="class")
mean(predict.tree != TestData_spam$spam)
  0.09956522
```

Error Rate – 9.956%

(2) Use bagging and random forest model to fit an ensemble of 100 trees to the training data. Report the error rate of these methods on the testing data. Include a plot of the importance of the variables, according to the ensemble.

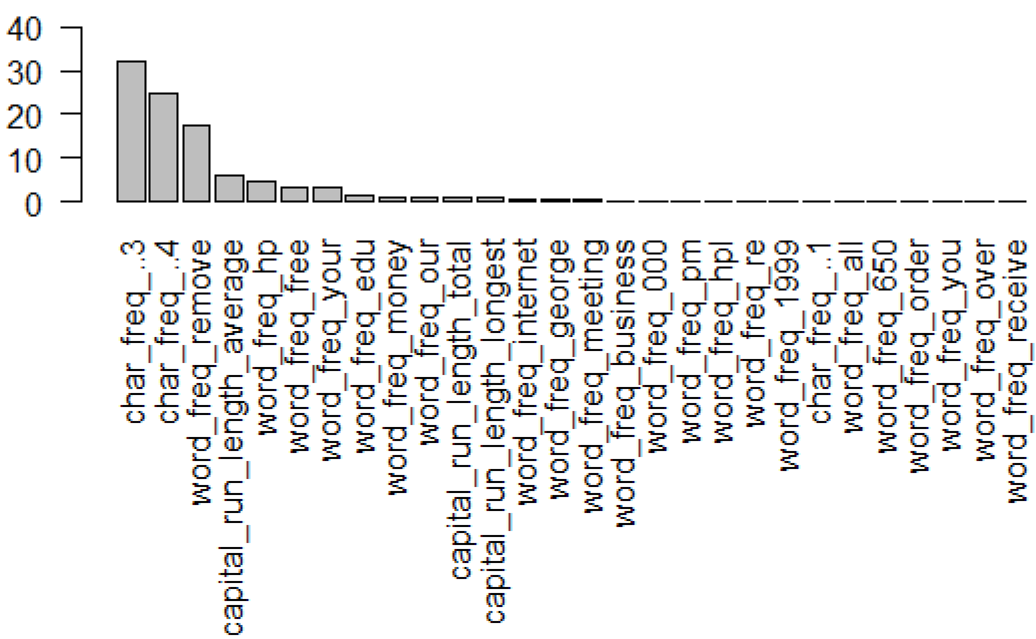
Bagging

```
spam.bag <- bagging(spam~., data=TrainData_spam,mfinal =
100,importance= TRUE)
```

Importance of the variables (by Gain of the Gini index)

```
> sort(spam.bag$importance,decreasing = TRUE)
            char_freq_..3             char_freq_..4          word_freq_remove capital_run_length_average
              32.24289581               24.54316004               17.50238713                 5.96902735
             word_freq_hp            word_freq_free            word_freq_your              word_freq_edu
               4.76480718                3.47278106                3.10028557                 1.37475309
          word_freq_money             word_freq_our  capital_run_length_total capital_run_length_longest
               1.19223266                1.10858891                1.06227061                 1.02533486
       word_freq_internet          word_freq_george         word_freq_meeting         word_freq_business
               0.65707000                0.54469354                0.50456707                 0.19174344
            word_freq_000              word_freq_pm             word_freq_hpl               word_freq_re
               0.17393071                0.10998417                0.10134395                 0.07567185
           word_freq_1999             char_freq_..1             word_freq_all              word_freq_650
               0.07231312                0.04520603                0.03951956                 0.03693937
          word_freq_order             word_freq_you            word_freq_over          word_freq_receive
               0.02664988                0.02258448                0.02066300                 0.01859555
              char_freq_.             char_freq_..2             char_freq_..5               word_freq_3d
               0.00000000                0.00000000                0.00000000                 0.00000000
            word_freq_415              word_freq_85             word_freq_857          word_freq_address
               0.00000000                0.00000000                0.00000000                 0.00000000
      word_freq_addresses      word_freq_conference          word_freq_credit               word_freq_cs
               0.00000000                0.00000000                0.00000000                 0.00000000
           word_freq_data          word_freq_direct           word_freq_email             word_freq_font
               0.00000000                0.00000000                0.00000000                 0.00000000
            word_freq_lab            word_freq_labs            word_freq_mail             word_freq_make
               0.00000000                0.00000000                0.00000000                 0.00000000
       word_freq_original           word_freq_parts          word_freq_people          word_freq_project
               0.00000000                0.00000000                0.00000000                 0.00000000
         word_freq_report           word_freq_table      word_freq_technology           word_freq_telnet
               0.00000000                0.00000000                0.00000000                 0.00000000
           word_freq_will
               0.00000000
```

```
imp <- sort(spam.bag$importance,decreasing = TRUE)
par(mar=c(12,2,1,1)+.1)
barplot(imp[imp>0],las=2,ylim=c(0,40))
```




```
predict.bag <- predict.bagging(spam.bag,newdata = TestData_spam)
predict.bag$confusion
```

```
              Observed Class
Predicted Class    0    1
              0 1316  121
              1   64  799
```

```
predict.bag$error
```

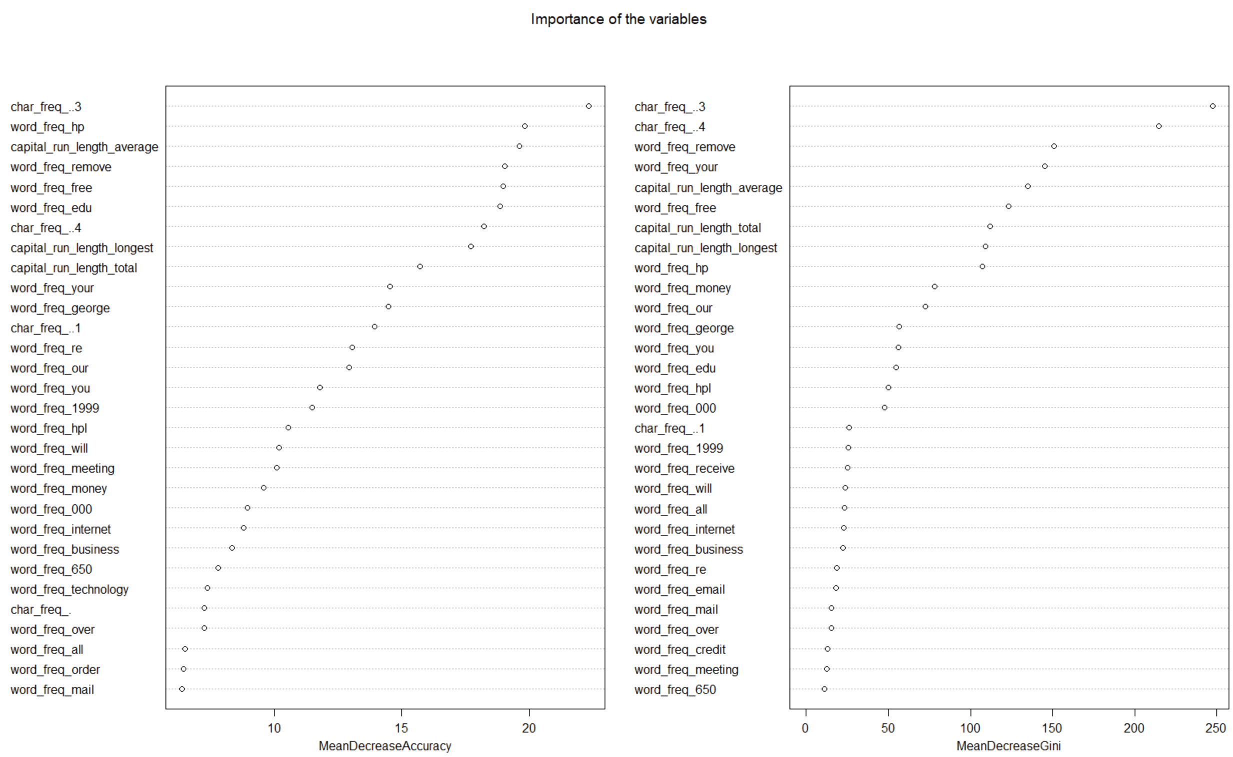
```
0.08043478
```

Random Forest

```
spam.rf <- randomForest(spam ~ ., data=spam,mtry = sqrt(ncol(spam)-1),
ntree=100,importance=TRUE, proximity=TRUE)
```

```
> spam.rf

Call:
 randomForest(formula = spam ~ ., data = TrainData_spam, mtry = sqrt(ncol(spam) -      1), ntree = 100, importance =
 TRUE, proximity = TRUE)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 8

        OOB estimate of  error rate: 5.26%
Confusion matrix:
     0   1 class.error
0 1361  47  0.03338068
1   74 819  0.08286674
```

```
varImpPlot(spam.rf)
```




```
varImpPlot(spam.rf,main="Importance of the variables")
predict.rf <- predict(spam.rf,newdata = TestData_spam)
```

(3) Which (if any) of these methods out-performs the constant classifier?

	Constant Classifier	Decision Tree (Pruned after Cross Validation)	Random Forest	Bagging
Error Rate(%)	39.6784	9.956522	5.26	8.043478

By comparing the error rate , it can be concluded that all of the three methods out-performs the constant classifier.

d) (1) What fraction of the spam e-mails in the training set did it not classify as spam?

```
          Prediction
Actual       0    1
      0   1361   47
      1     74  819
```

74/(74+819) = 74/893 = 0.08286 = 8.286%

(2) What fraction of the genuine e-mails in the testing set did it classify as spam?

```
confusionMatrix(predict.rf, TestData_spam$spam)

           Reference
Prediction    0    1
         0 1335   75
         1   45  845
```

45/(1335+45) = 45/1380 = 0.0326 = 3.26%

(3) What fraction of e-mails it classified as spam were actually spam?

845/(845+75) = 845/920 = 0.91847 = 91.847%