Assignment 12

Max Wagner

November 10, 2015

Looking at the data

The spambase email data is already past the corpus phase, and instead gives a dtm like structure, with a few additional columns with capital letter information and the spam/ham indicator. I'll make the last column a factor for later use in the model. The table at the end shows there are 2788 ham emails, and 1813 spam emails.

```
library(caret)
```

```
## Loading required package: lattice
## Loading required package: ggplot2
```

```
library(kernlab)
spambase <- read.csv("spambase.data", header = FALSE)
spambase$V58 <- as.factor(spambase$V58)
head(spambase)
```

```
##     V1   V2   V3 V4   V5   V6   V7   V8   V9  V10  V11  V12  V13  V14  V15
## 1 0.00 0.64 0.64  0 0.32 0.00 0.00 0.00 0.00 0.00 0.00 0.64 0.00 0.00 0.00
## 2 0.21 0.28 0.50  0 0.14 0.28 0.21 0.07 0.00 0.94 0.21 0.79 0.65 0.21 0.14
## 3 0.06 0.00 0.71  0 1.23 0.19 0.19 0.12 0.64 0.25 0.38 0.45 0.12 0.00 1.75
## 4 0.00 0.00 0.00  0 0.63 0.00 0.31 0.63 0.31 0.63 0.31 0.31 0.31 0.00 0.00
## 5 0.00 0.00 0.00  0 0.63 0.00 0.31 0.63 0.31 0.63 0.31 0.31 0.31 0.00 0.00
## 6 0.00 0.00 0.00  0 1.85 0.00 0.00 1.85 0.00 0.00 0.00 0.00 0.00 0.00 0.00
##    V16  V17  V18  V19  V20  V21 V22  V23  V24 V25 V26 V27 V28 V29 V30 V31
## 1 0.32 0.00 1.29 1.93 0.00 0.96   0 0.00 0.00   0   0   0   0   0   0   0
## 2 0.14 0.07 0.28 3.47 0.00 1.59   0 0.43 0.43   0   0   0   0   0   0   0
## 3 0.06 0.06 1.03 1.36 0.32 0.51   0 1.16 0.06   0   0   0   0   0   0   0
## 4 0.31 0.00 0.00 3.18 0.00 0.31   0 0.00 0.00   0   0   0   0   0   0   0
## 5 0.31 0.00 0.00 3.18 0.00 0.31   0 0.00 0.00   0   0   0   0   0   0   0
## 6 0.00 0.00 0.00 0.00 0.00 0.00   0 0.00 0.00   0   0   0   0   0   0   0
##   V32 V33 V34 V35 V36  V37 V38 V39  V40 V41 V42  V43 V44  V45  V46 V47 V48
## 1   0   0   0   0   0 0.00   0   0 0.00   0   0 0.00   0 0.00 0.00   0   0
## 2   0   0   0   0   0 0.07   0   0 0.00   0   0 0.00   0 0.00 0.00   0   0
## 3   0   0   0   0   0 0.00   0   0 0.06   0   0 0.12   0 0.06 0.06   0   0
## 4   0   0   0   0   0 0.00   0   0 0.00   0   0 0.00   0 0.00 0.00   0   0
## 5   0   0   0   0   0 0.00   0   0 0.00   0   0 0.00   0 0.00 0.00   0   0
## 6   0   0   0   0   0 0.00   0   0 0.00   0   0 0.00   0 0.00 0.00   0   0
##    V49   V50 V51   V52   V53   V54   V55 V56  V57 V58
## 1 0.00 0.000   0 0.778 0.000 0.000 3.756  61  278   1
## 2 0.00 0.132   0 0.372 0.180 0.048 5.114 101 1028   1
## 3 0.01 0.143   0 0.276 0.184 0.010 9.821 485 2259   1
## 4 0.00 0.137   0 0.137 0.000 0.000 3.537  40  191   1
```

```
## 5 0.00 0.135   0 0.135 0.000 0.000 3.537  40   191   1
## 6 0.00 0.223   0 0.000 0.000 0.000 3.000  15    54   1
```

```
table(spambase$V58)
```

```
##
##    0    1
## 2788 1813
```

Spliting the data

We need to split the data into two different sets. One section will be for training, and the other section for testing. We'll need to split the original set into spam and ham first. Then recombine it into a smaller set.

```
ham <- subset(spambase, V58 == 0)
spam <- subset(spambase, V58 == 1)
training <- rbind(ham[1:600,], spam[1:400,])
testing <- rbind(ham[601:1200,], spam[401:800,])
```

A model or two

The next step is to try to fit it to a model. I'll try out a couple different ones to see how they compare to each other. I used SVM and random forests to test out how well it fit.

```
svm <- train(training$V58 ~ ., data = training, method = "svmRadial")
pred <- predict(svm, testing)
confu <- confusionMatrix(pred, testing$V58); confu
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 576  84
##          1  24 316
##
##                Accuracy : 0.892
##                  95% CI : (0.8711, 0.9106)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.7692
##  Mcnemar's Test P-Value : 1.369e-08
##
##             Sensitivity : 0.9600
##             Specificity : 0.7900
##          Pos Pred Value : 0.8727
##          Neg Pred Value : 0.9294
##              Prevalence : 0.6000
##          Detection Rate : 0.5760
```

```
##    Detection Prevalence : 0.6600
##       Balanced Accuracy : 0.8750
##
##        'Positive' Class : 0
##
```

```r
rf <- train(training$V58 ~ ., data = training, method = "rf")
```

```
## Loading required package: randomForest
## randomForest 4.6-12
## Type rfNews() to see new features/changes/bug fixes.
```

```r
pred <- predict(rf, testing)
confu <- confusionMatrix(pred, testing$V58); confu
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 584  53
##          1  16 347
##
##                Accuracy : 0.931
##                  95% CI : (0.9135, 0.9459)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.854
##  Mcnemar's Test P-Value : 1.465e-05
##
##             Sensitivity : 0.9733
##             Specificity : 0.8675
##          Pos Pred Value : 0.9168
##          Neg Pred Value : 0.9559
##              Prevalence : 0.6000
##          Detection Rate : 0.5840
##    Detection Prevalence : 0.6370
##       Balanced Accuracy : 0.9204
##
##        'Positive' Class : 0
##
```

From the two models, we can see that SVM gave an accuracy of 89.3%, and random forests gave an accuracy of 93.4%. The big caveat with the entire project is that I am unsure of how "statisically sound" the entire process was. The exclusion of a traditional corpus and instead having a percentages document made the process confusing to me at first.