STATS 765: Lab 6

Task 1

Build a classification tree using the variables in wordmatrix as predictors and the is_spam variable in df as the outcome. Comment on the shape of the tree.

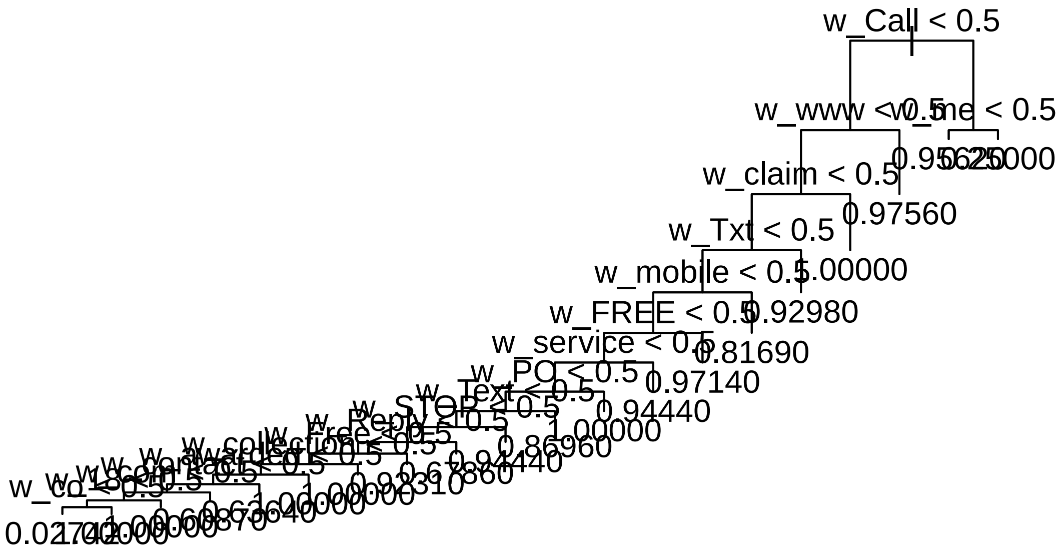
```
library(tree)
library(rpart)
load("data/spam.rda")

spam_data = data.frame(df$is_spam, wordmatrix)
names(spam_data)[1] = "is_spam"
rpart(is_spam~. , data=spam_data)
```

```
## n= 5574
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##      1) root 5574 646.8907000 0.13401510
##        2) w_Call< 0.5 5417 542.0834000 0.11279310
##          4) w_www< 0.5 5335 478.1488000 0.09953140
##            8) w_claim< 0.5 5269 423.9628000 0.08825204
##             16) w_Txt< 0.5 5212 379.4321000 0.07904835
##               32) w_mobile< 0.5 5141 329.6242000 0.06885820
##                 64) w_FREE< 0.5 5106 299.9452000 0.06267137
##                  128) w_service< 0.5 5070 269.8667000 0.05641026
##                    256) w_PO< 0.5 5049 251.0913000 0.05248564
##                      512) w_Text< 0.5 5026 233.0571000 0.04874652
##                       1024) w_STOP< 0.5 5008 217.6198000 0.04552716
##                         2048) w_Reply< 0.5 4980 200.2287000 0.04196787
##                           4096) w_Free< 0.5 4967 189.1866000 0.03966177
##                             8192) w_collection< 0.5 4956 179.0194000 0.03753027
##                              16384) w_awarded< 0.5 4946 169.7372000 0.03558431
##                                32768) w_contact< 0.5 4924 156.6702000 0.03290008
##                                  65536) w_com< 0.5 4901 143.5307000 0.03019792
##                                   131072) w_co< 0.5 4894 136.9377000 0.02881079
##                                     262144) w_18< 0.5 4887 130.3258000 0.02741968 *
##                                     262145) w_18>=0.5 7   0.0000000 1.00000000 *
##                                   131073) w_co>=0.5 7   0.0000000 1.00000000 *
##                                  65537) w_com>=0.5 23   5.4782610 0.60869570 *
##                                32769) w_contact>=0.5 22   5.0909090 0.63636360 *
##                              16385) w_awarded>=0.5 10   0.0000000 1.00000000 *
##                             8193) w_collection>=0.5 11   0.0000000 1.00000000 *
##                           4097) w_Free>=0.5 13   0.9230769 0.92307690 *
##                         2049) w_Reply>=0.5 28   6.1071430 0.67857140 *
##                       1025) w_STOP>=0.5 18   0.9444444 0.94444440 *
##                      513) w_Text>=0.5 23   2.6086960 0.86956520 *
##                    257) w_PO>=0.5 21   0.0000000 1.00000000 *
##                  129) w_service>=0.5 36   1.8888890 0.94444440 *
##                 65) w_FREE>=0.5 35   0.9714286 0.97142860 *
##               33) w_mobile>=0.5 71  10.6197200 0.81690140 *
##              17) w_Txt>=0.5 57   3.7192980 0.92982460 *
##             9) w_claim>=0.5 66   0.0000000 1.00000000 *
##           5) w_www>=0.5 82   1.9512200 0.97560980 *
##         3) w_Call>=0.5 157  18.1910800 0.86624200  
##           6) w_me>=0.5 20   3.7500000 0.25000000 *
##           7) w_me< 0.5 137   5.7372260 0.95620440 *
```

```
tree = tree(is_spam ~ ., data = spam_data)
summary(tree)
```

```
## 
## Regression tree:
## tree(formula = is_spam ~ ., data = spam_data)
## Variables actually used in tree construction:
##  [1] "w_Call"       "w_www"        "w_claim"      "w_Txt"        "w_mobile"    
##  [6] "w_FREE"       "w_service"    "w_PO"         "w_Text"       "w_STOP"      
## [11] "w_Reply"      "w_Free"       "w_collection" "w_awarded"    "w_contact"   
## [16] "w_com"        "w_18"         "w_co"         "w_me"        
## Number of terminal nodes:  20 
## Residual mean deviance:  0.03243 = 180.1 / 5554 
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
## -0.97560 -0.02742 -0.02742  0.00000 -0.02742  0.97260
```

```
{
plot(tree)
text(tree)
}
```



The tree has 20 nodes and appears to have a staircase structure. It is probably overfitted to the data as it appears to focus on only a couple of observations at a time.

Task 2

Look at the description of the data sources. Would you expect dividing spam and non-spam to be easier or harder in this dataset than in real life, and why?

Likely that the spam data will not be coming from Singapore and the UK, therefore we would need to train models on local messages for a particular country and it should be more useful in that country. Our real-world data will look very different to the training data. The model is also trained on UK and Singapore English, which isn't representative of the English for the rest of the world. The spam data is from the UK, and the non-spam data is from Singapore. This introduces bias into our data as the model will learn the differences between Singapore and UK English to identify whether a given message is spam or not. Data is also imbalanced, so we get bias towardds the UK data.