Building SPAM filter in R

EST 530 Big Data Project 2

Group members: Shawn, Peter, Karen, Abhinav, Neil

Structure

1. Data preparation

2. Extracting spam predictors

3. Creating data frame

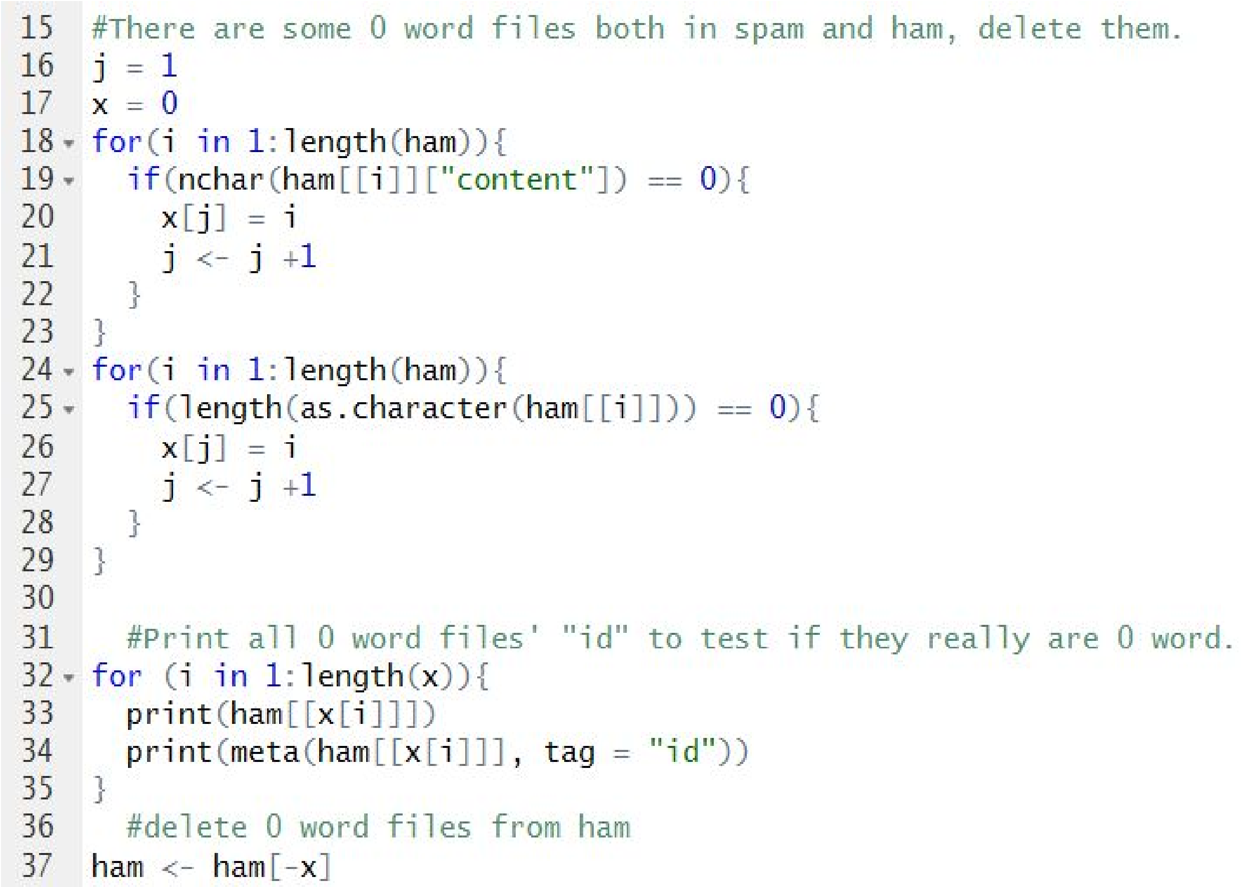
4. Machine learning models

5. Conclusion

Data Preparation

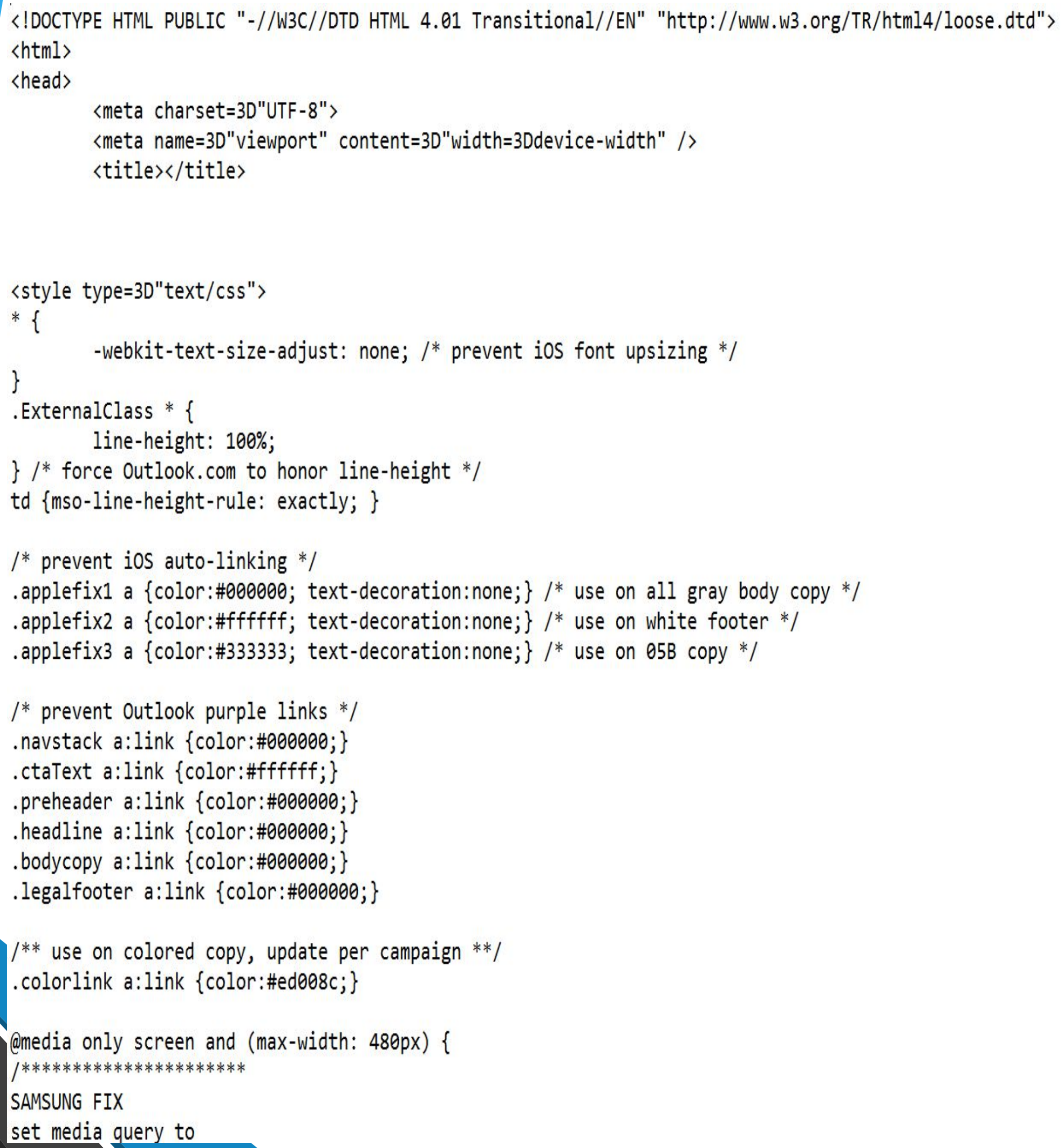
Preparation: Delete empty files

- spam (14/433)
- ham (5/210)

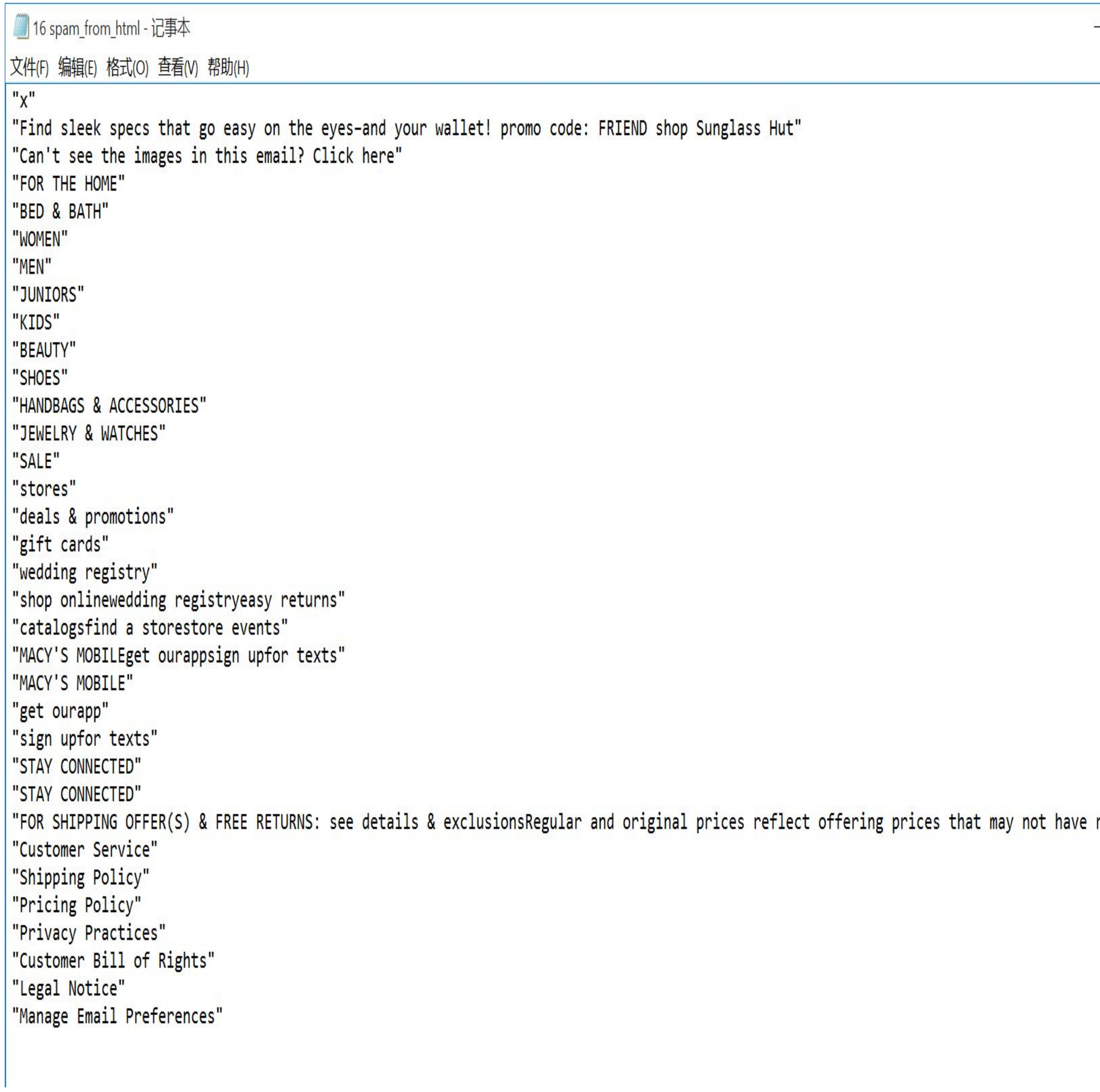


```
#There are some 0 word files both in spam and ham, delete them.
j = 1
x = 0
for(i in 1:length(ham)){
  if(nchar(ham[[i]]["content"]) == 0){
    x[j] = i
    j <- j +1
  }
}
for(i in 1:length(ham)){
  if(length(as.character(ham[[i]])) == 0){
    x[j] = i
    j <- j +1
  }
}

  #Print all 0 word files' "id" to test if they really are 0 word.
for (i in 1:length(x)){
  print(ham[[x[i]]])
  print(meta(ham[[x[i]]], tag = "id"))
}
  #delete 0 word files from ham
ham <- ham[-x]
```

Preparation: “html” to “txt” (“XML” package)



```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
        <meta charset=3D"UTF-8">
        <meta name=3D"viewport" content=3D"width=3Ddevice-width" />
        <title></title>



<style type=3D"text/css">
* {
        -webkit-text-size-adjust: none; /* prevent iOS font upsizing */
}
.ExternalClass * {
        line-height: 100%;
} /* force Outlook.com to honor line-height */
td {mso-line-height-rule: exactly; }

/* prevent iOS auto-linking */
.applefix1 a {color:#000000; text-decoration:none;} /* use on all gray body copy */
.applefix2 a {color:#ffffff; text-decoration:none;} /* use on white footer */
.applefix3 a {color:#333333; text-decoration:none;} /* use on 05B copy */

/* prevent Outlook purple links */
.navstack a:link {color:#000000;}
.ctaText a:link {color:#ffffff;}
.preheader a:link {color:#000000;}
.headline a:link {color:#000000;}
.bodycopy a:link {color:#000000;}
.legalfooter a:link {color:#000000;}

/** use on colored copy, update per campaign **/
.colorlink a:link {color:#ed008c;}

@media only screen and (max-width: 480px) {
/**********************
SAMSUNG FIX
set media query to
```

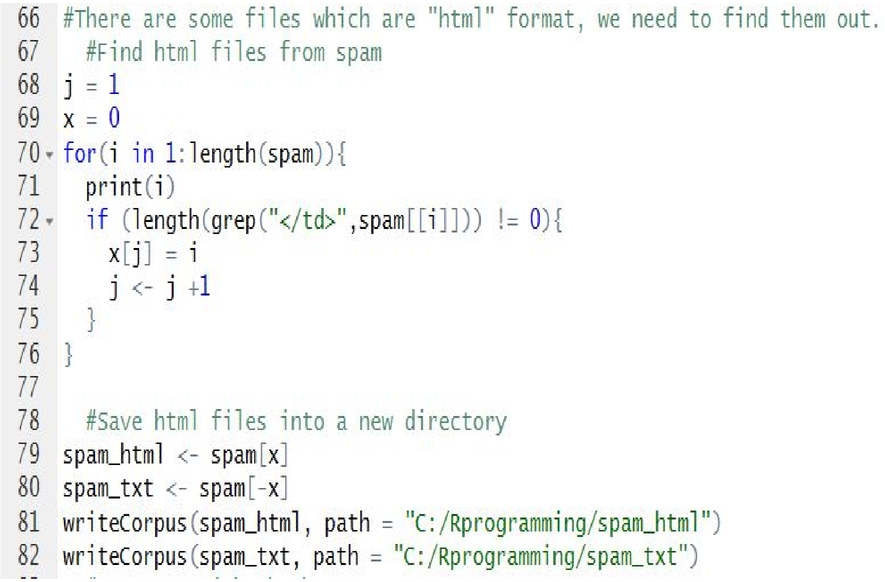


16 spam_from_html - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
"x"
"Find sleek specs that go easy on the eyes-and your wallet! promo code: FRIEND shop Sunglass Hut"
"Can't see the images in this email? Click here"
"FOR THE HOME"
"BED & BATH"
"WOMEN"
"MEN"
"JUNIORS"
"KIDS"
"BEAUTY"
"SHOES"
"HANDBAGS & ACCESSORIES"
"JEWELRY & WATCHES"
"SALE"
"stores"
"deals & promotions"
"gift cards"
"wedding registry"
"shop onlinewedding registryeasy returns"
"catalogsfind a storestore events"
"MACY'S MOBILEget ourappsign upfor texts"
"MACY'S MOBILE"
"get ourapp"
"sign upfor texts"
"STAY CONNECTED"
"STAY CONNECTED"
"FOR SHIPPING OFFER(S) & FREE RETURNS: see details & exclusionsRegular and original prices reflect offering prices that may not have
"Customer Service"
"Shipping Policy"
"Pricing Policy"
"Privacy Practices"
"Customer Bill of Rights"
"Legal Notice"
"Manage Email Preferences"
```

Preparation: “html” to “txt” (“XML” package)

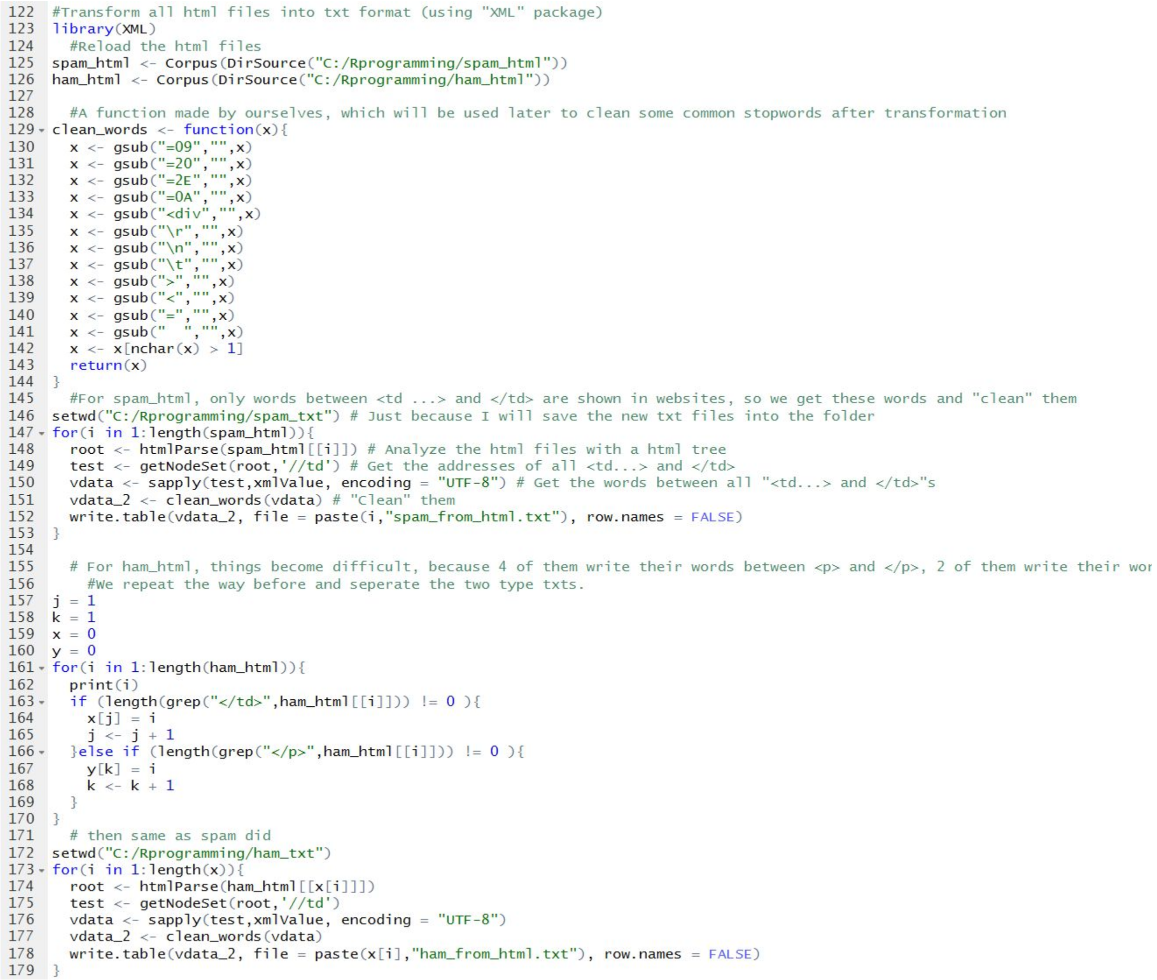
- Find HTML files



```r
#There are some files which are "html" format, we need to find them out.
  #Find html files from spam
j = 1
x = 0
for(i in 1:length(spam)){
  print(i)
  if (length(grep("</td>",spam[[i]])) != 0){
    x[j] = i
    j <- j +1
  }
}

  #Save html files into a new directory
spam_html <- spam[x]
spam_txt <- spam[-x]
writeCorpus(spam_html, path = "C:/Rprogramming/spam_html")
writeCorpus(spam_txt, path = "C:/Rprogramming/spam_txt")
```

- spam(147/433)
- ham(6/210)

- Transform “HTML” to “txt”

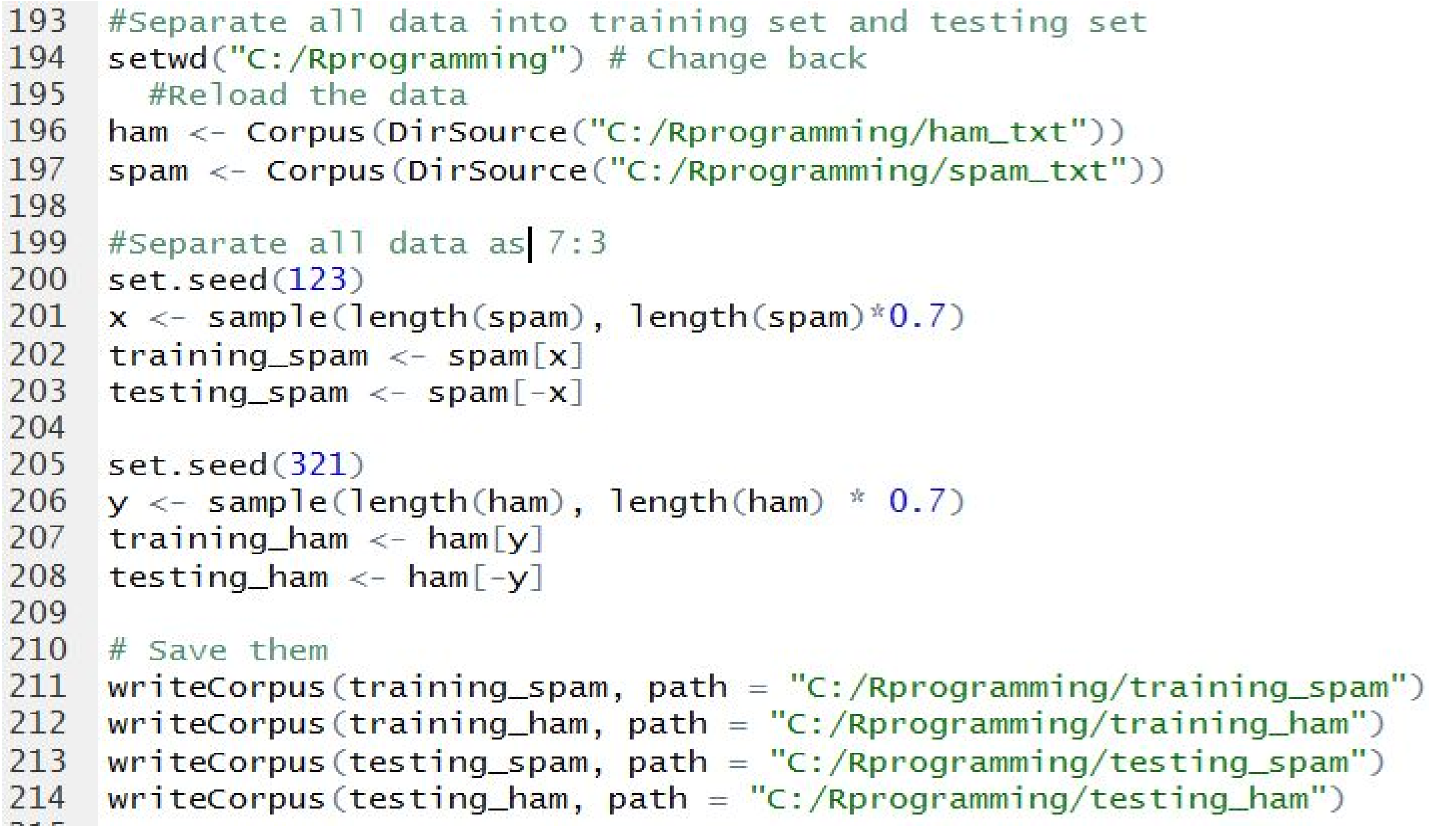


```r
#Transform all html files into txt format (using "XML" package)
library(XML)
  #Reload the html files
spam_html <- Corpus(DirSource("C:/Rprogramming/spam_html"))
ham_html <- Corpus(DirSource("C:/Rprogramming/ham_html"))

  #A function made by ourselves, which will be used later to clean some common stopwords after transformation
clean_words <- function(x){
  x <- gsub("=09","",x)
  x <- gsub("=20","",x)
  x <- gsub("=2E","",x)
  x <- gsub("=0A","",x)
  x <- gsub("<div","",x)
  x <- gsub("\r","",x)
  x <- gsub("\n","",x)
  x <- gsub("\t","",x)
  x <- gsub(">","",x)
  x <- gsub("<","",x)
  x <- gsub("=","",x)
  x <- gsub("  ","",x)
  x <- x[nchar(x) > 1]
  return(x)
}
  #For spam_html, only words between <td ...> and </td> are shown in websites, so we get these words and "clean" them
setwd("C:/Rprogramming/spam_txt") # Just because I will save the new txt files into the folder
for(i in 1:length(spam_html)){
  root <- htmlParse(spam_html[[i]]) # Analyze the html files with a html tree
  test <- getNodeSet(root,'//td') # Get the addresses of all <td...> and </td>
  vdata <- sapply(test,xmlValue, encoding = "UTF-8") # Get the words between all "<td...> and </td>"s
  vdata_2 <- clean_words(vdata) # "Clean" them
  write.table(vdata_2, file = paste(i,"spam_from_html.txt"), row.names = FALSE)
}

  # For ham_html, things become difficult, because 4 of them write their words between <p> and </p>, 2 of them write their wor
    #We repeat the way before and seperate the two type txts.
j = 1
k = 1
x = 0
y = 0
for(i in 1:length(ham_html)){
  print(i)
  if (length(grep("</td>",ham_html[[i]])) != 0 ){
    x[j] = i
    j <- j + 1
  }else if (length(grep("</p>",ham_html[[i]])) != 0 ){
    y[k] = i
    k <- k + 1
  }
}
  # then same as spam did
setwd("C:/Rprogramming/ham_txt")
for(i in 1:length(x)){
  root <- htmlParse(ham_html[[x[i]]])
  test <- getNodeSet(root,'//td')
  vdata <- sapply(test,xmlValue, encoding = "UTF-8")
  vdata_2 <- clean_words(vdata)
  write.table(vdata_2, file = paste(x[i],"ham_from_html.txt"), row.names = FALSE)
}
```

Extracting spam predictors

Extracting spam predictors: Pre processing

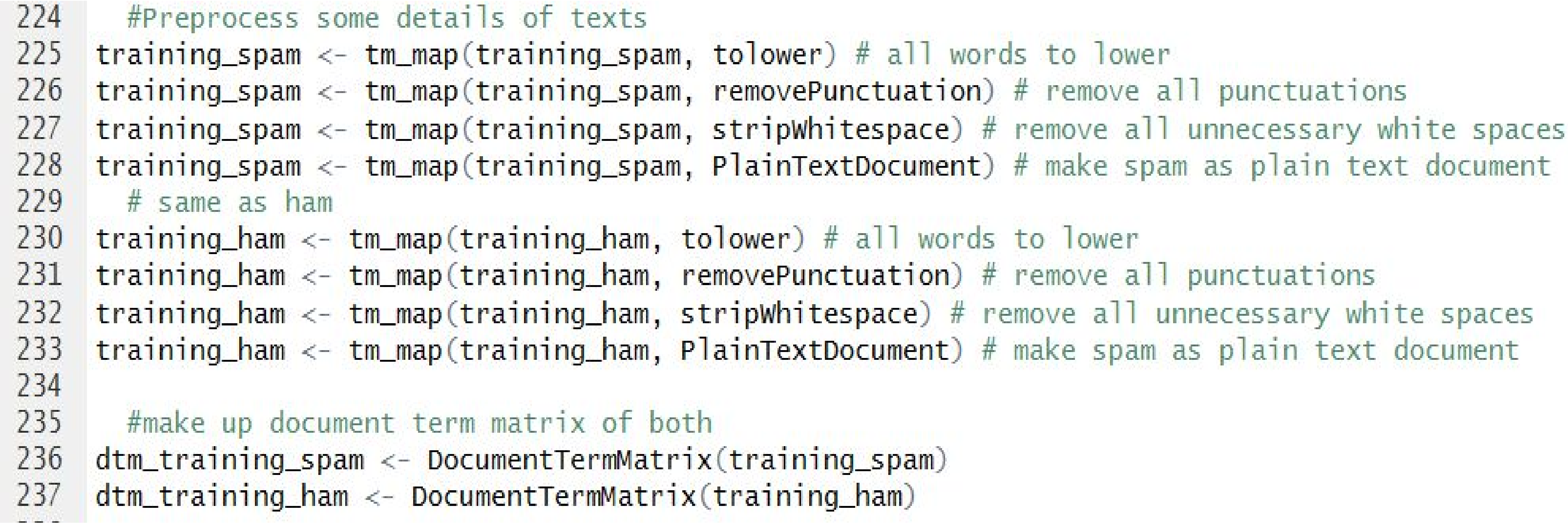
- 70% of data in training set
- 30% of data in testing set



```
#Separate all data into training set and testing set
setwd("C:/Rprogramming") # Change back
  #Reload the data
ham <- Corpus(DirSource("C:/Rprogramming/ham_txt"))
spam <- Corpus(DirSource("C:/Rprogramming/spam_txt"))

#Separate all data as 7:3
set.seed(123)
x <- sample(length(spam), length(spam)*0.7)
training_spam <- spam[x]
testing_spam <- spam[-x]

set.seed(321)
y <- sample(length(ham), length(ham) * 0.7)
training_ham <- ham[y]
testing_ham <- ham[-y]

# Save them
writeCorpus(training_spam, path = "C:/Rprogramming/training_spam")
writeCorpus(training_ham, path = "C:/Rprogramming/training_ham")
writeCorpus(testing_spam, path = "C:/Rprogramming/testing_spam")
writeCorpus(testing_ham, path = "C:/Rprogramming/testing_ham")
```

Extracting spam predictors:

Creating DTM from Training Group

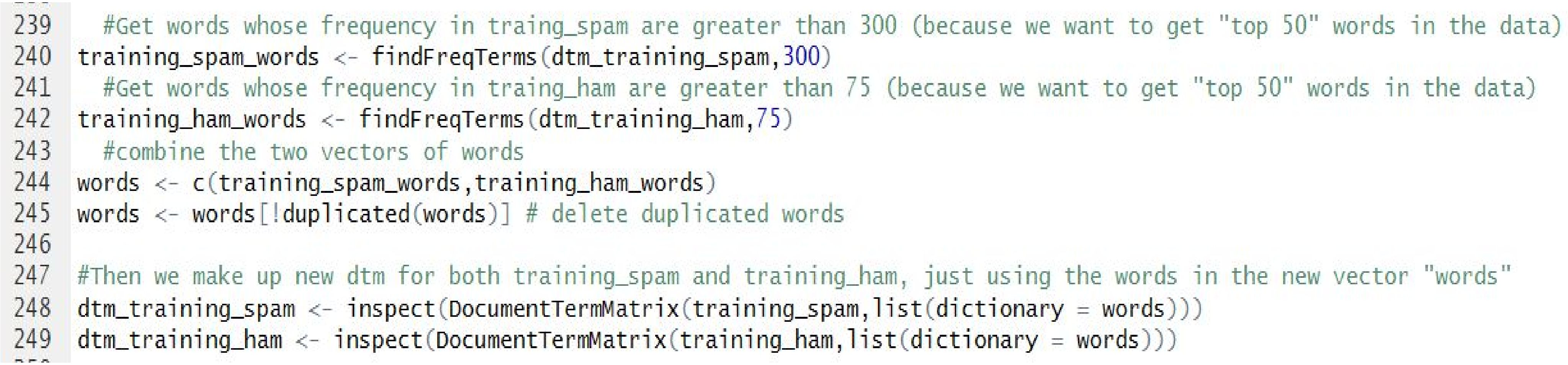


```
  #Preprocess some details of texts
training_spam <- tm_map(training_spam, tolower) # all words to lower
training_spam <- tm_map(training_spam, removePunctuation) # remove all punctuations
training_spam <- tm_map(training_spam, stripWhitespace) # remove all unnecessary white spaces
training_spam <- tm_map(training_spam, PlainTextDocument) # make spam as plain text document
  # same as ham
training_ham <- tm_map(training_ham, tolower) # all words to lower
training_ham <- tm_map(training_ham, removePunctuation) # remove all punctuations
training_ham <- tm_map(training_ham, stripWhitespace) # remove all unnecessary white spaces
training_ham <- tm_map(training_ham, PlainTextDocument) # make spam as plain text document

  #make up document term matrix of both
dtm_training_spam <- DocumentTermMatrix(training_spam)
dtm_training_ham <- DocumentTermMatrix(training_ham)
```

Extracting spam predictors: Principles

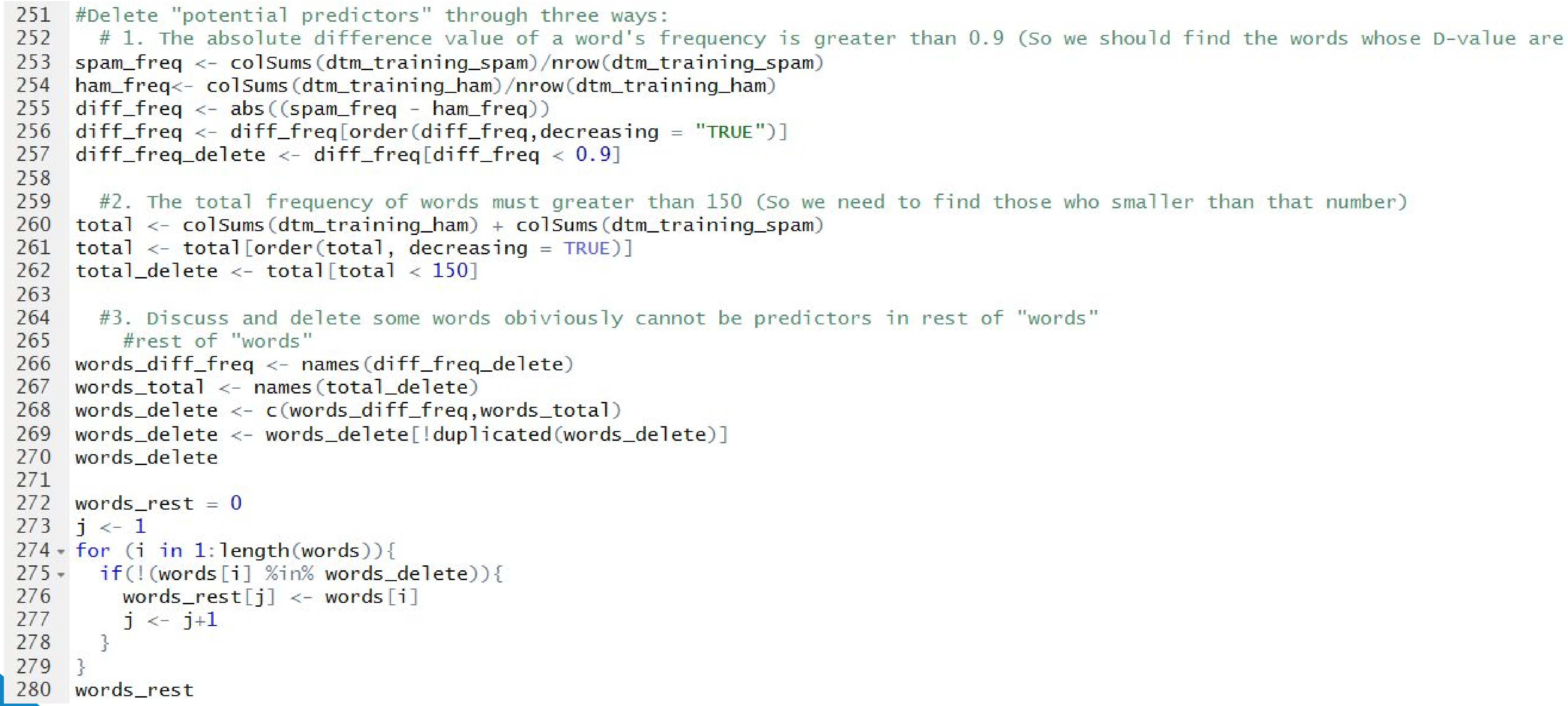
- Top 50 words in training group (spam and ham)
- Mean of Word Frequency difference between spam and ham greater than 0.9
- Sum of Word Frequency in training group greater than 150

Extracting spam predictors: Creating DTM with “top-100” words



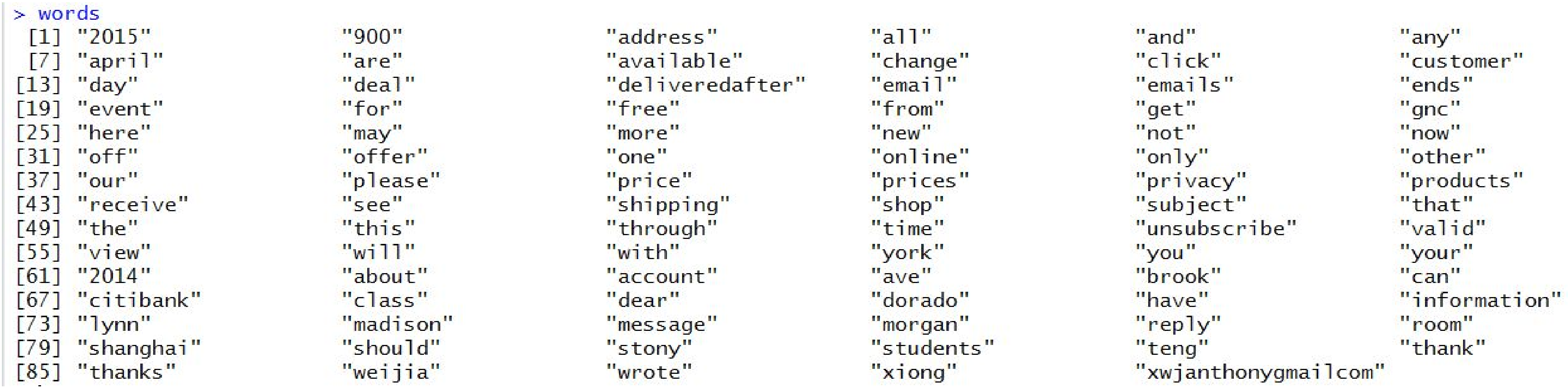
```
  #Get words whose frequency in traing_spam are greater than 300 (because we want to get "top 50" words in the data)
training_spam_words <- findFreqTerms(dtm_training_spam,300)
  #Get words whose frequency in traing_ham are greater than 75 (because we want to get "top 50" words in the data)
training_ham_words <- findFreqTerms(dtm_training_ham,75)
  #combine the two vectors of words
words <- c(training_spam_words,training_ham_words)
words <- words[!duplicated(words)] # delete duplicated words

#Then we make up new dtm for both training_spam and training_ham, just using the words in the new vector "words"
dtm_training_spam <- inspect(DocumentTermMatrix(training_spam,list(dictionary = words)))
dtm_training_ham <- inspect(DocumentTermMatrix(training_ham,list(dictionary = words)))
```

Extracting spam predictors: Delete unsatisfied words

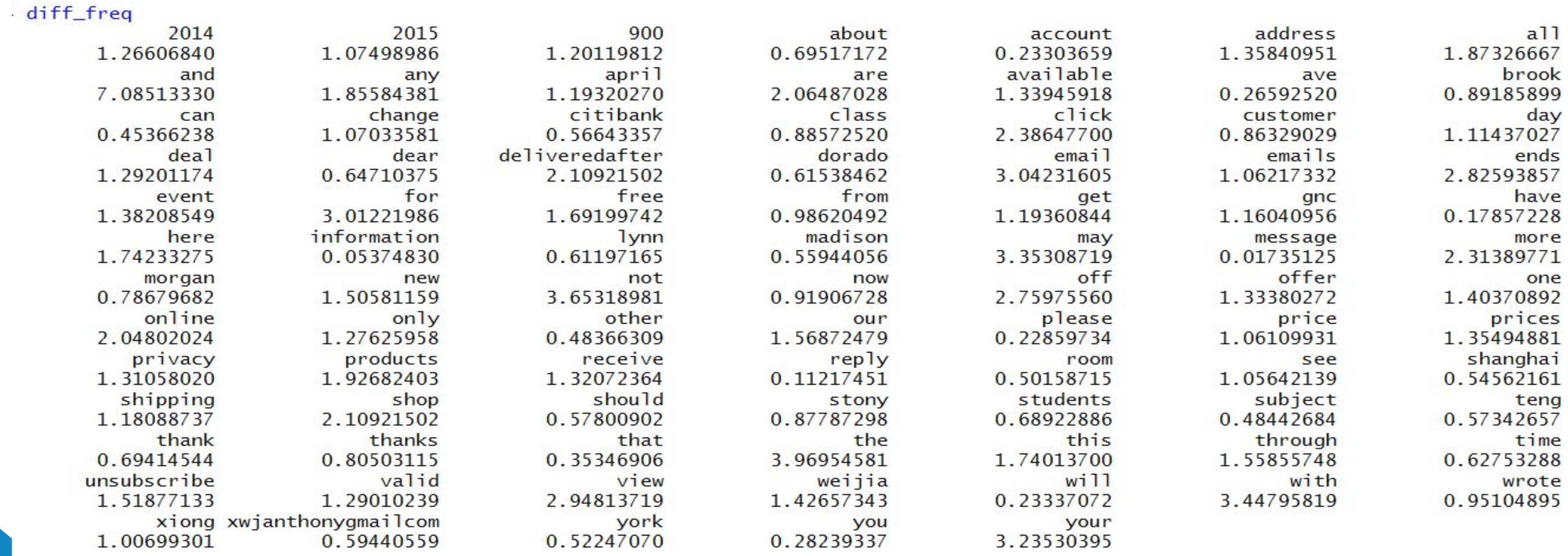


```r
#Delete "potential predictors" through three ways:
  # 1. The absolute difference value of a word's frequency is greater than 0.9 (So we should find the words whose D-value are
spam_freq <- colSums(dtm_training_spam)/nrow(dtm_training_spam)
ham_freq<- colSums(dtm_training_ham)/nrow(dtm_training_ham)
diff_freq <- abs((spam_freq - ham_freq))
diff_freq <- diff_freq[order(diff_freq,decreasing = "TRUE")]
diff_freq_delete <- diff_freq[diff_freq < 0.9]

  #2. The total frequency of words must greater than 150 (So we need to find those who smaller than that number)
total <- colSums(dtm_training_ham) + colSums(dtm_training_spam)
total <- total[order(total, decreasing = TRUE)]
total_delete <- total[total < 150]

  #3. Discuss and delete some words obiviously cannot be predictors in rest of "words"
    #rest of "words"
words_diff_freq <- names(diff_freq_delete)
words_total <- names(total_delete)
words_delete <- c(words_diff_freq,words_total)
words_delete <- words_delete[!duplicated(words_delete)]
words_delete

words_rest = 0
j <- 1
for (i in 1:length(words)){
  if(!(words[i] %in% words_delete)){
    words_rest[j] <- words[i]
    j <- j+1
  }
}
words_rest
```

Extracting spam predictors: Results

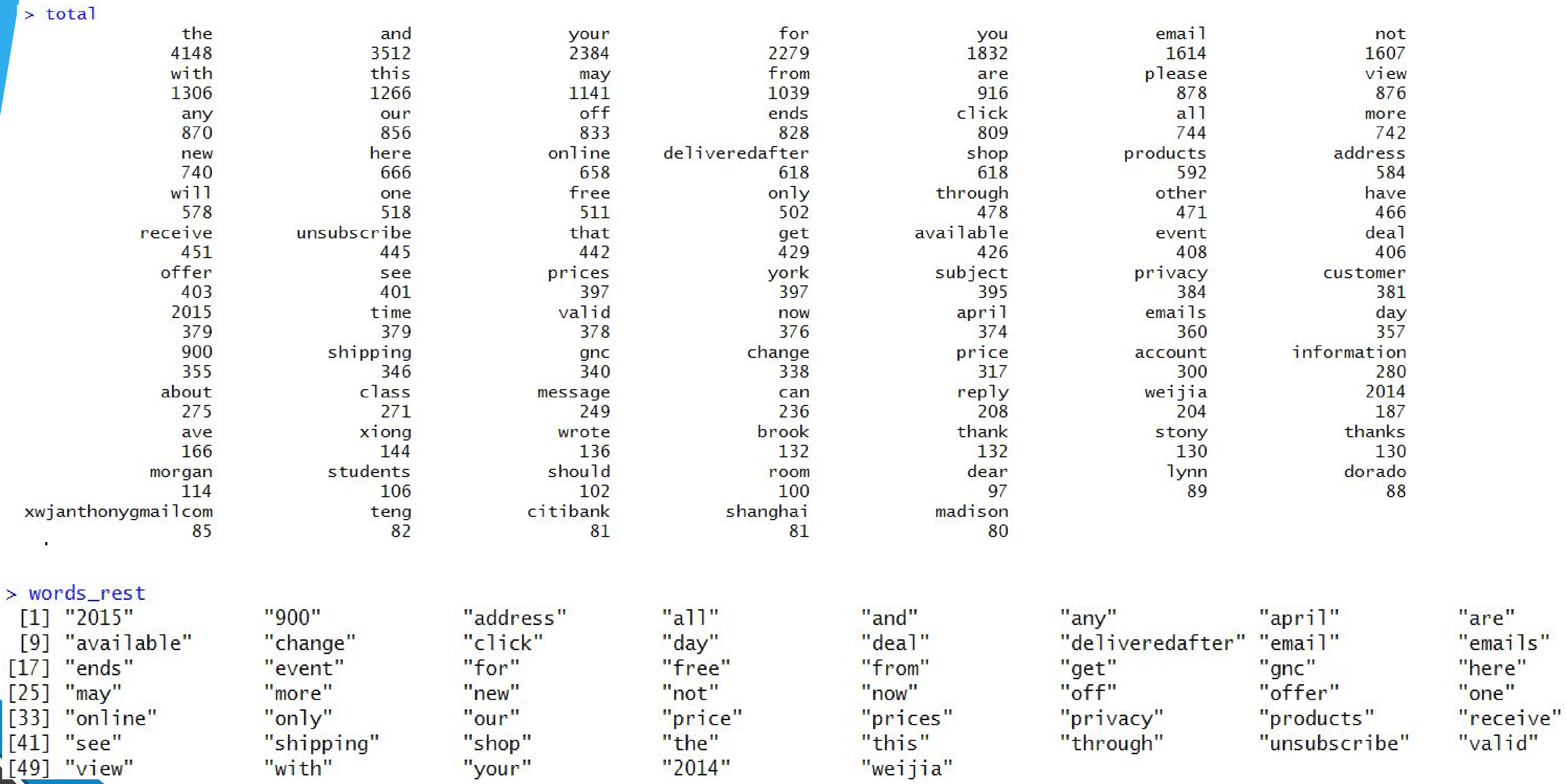


```
> words
 [1] "2015"               "900"                "address"            "all"                "and"                "any"
 [7] "april"              "are"                "available"          "change"             "click"              "customer"
[13] "day"                "deal"               "deliveredafter"     "email"              "emails"             "ends"
[19] "event"              "for"                "free"               "from"               "get"                "gnc"
[25] "here"               "may"                "more"               "new"                "not"                "now"
[31] "off"                "offer"              "one"                "online"             "only"               "other"
[37] "our"                "please"             "price"              "prices"             "privacy"            "products"
[43] "receive"            "see"                "shipping"           "shop"               "subject"            "that"
[49] "the"                "this"               "through"            "time"               "unsubscribe"        "valid"
[55] "view"               "will"               "with"               "york"               "you"                "your"
[61] "2014"               "about"              "account"            "ave"                "brook"              "can"
[67] "citibank"           "class"              "dear"               "dorado"             "have"               "information"
[73] "lynn"               "madison"            "message"            "morgan"             "reply"              "room"
[79] "shanghai"           "should"             "stony"              "students"           "teng"               "thank"
[85] "thanks"             "weijia"             "wrote"              "xiong"              "xwjanthonygmailcom"
```



```
diff_freq
              2014               2015                900              about            account            address                all
        1.26606840         1.07498986         1.20119812         0.69517172         0.23303659         1.35840951         1.87326667
               and                any              april                are          available                ave              brook
        7.08513330         1.85584381         1.19320270         2.06487028         1.33945918         0.26592520         0.89185899
               can             change           citibank              class              click           customer                day
        0.45366238         1.07033581         0.56643357         0.88572520         2.38647700         0.86329029         1.11437027
              deal               dear     deliveredafter             dorado              email             emails               ends
        1.29201174         0.64710375         2.10921502         0.61538462         3.04231605         1.06217332         2.82593857
             event                for               free               from                get                gnc               have
        1.38208549         3.01221986         1.69199742         0.98620492         1.19360844         1.16040956         0.17857228
              here        information               lynn            madison                may            message               more
        1.74233275         0.05374830         0.61197165         0.55944056         3.35308719         0.01735125         2.31389771
            morgan                new                not                now                off              offer                one
        0.78679682         1.50581159         3.65318981         0.91906728         2.75975560         1.33380272         1.40370892
            online               only              other                our             please              price             prices
        2.04802024         1.27625958         0.48366309         1.56872479         0.22859734         1.06109931         1.35494881
           privacy           products            receive              reply               room                see           shanghai
        1.31058020         1.92682403         1.32072364         0.11217451         0.50158715         1.05642139         0.54562161
          shipping               shop             should              stony           students            subject               teng
        1.18088737         2.10921502         0.57800902         0.87787298         0.68922886         0.48442684         0.57342657
             thank             thanks               that                the               this            through               time
        0.69414544         0.80503115         0.35346906         3.96954581         1.74013700         1.55855748         0.62753288
       unsubscribe              valid               view             weijia               will               with              wrote
        1.51877133         1.29010239         2.94813719         1.42657343         0.23337072         3.44795819         0.95104895
             xiong xwjanthonygmailcom               york                you               your
        1.00699301         0.59440559         0.52247070         0.28239337         3.23530395
```

Extracting spam predictors: Results



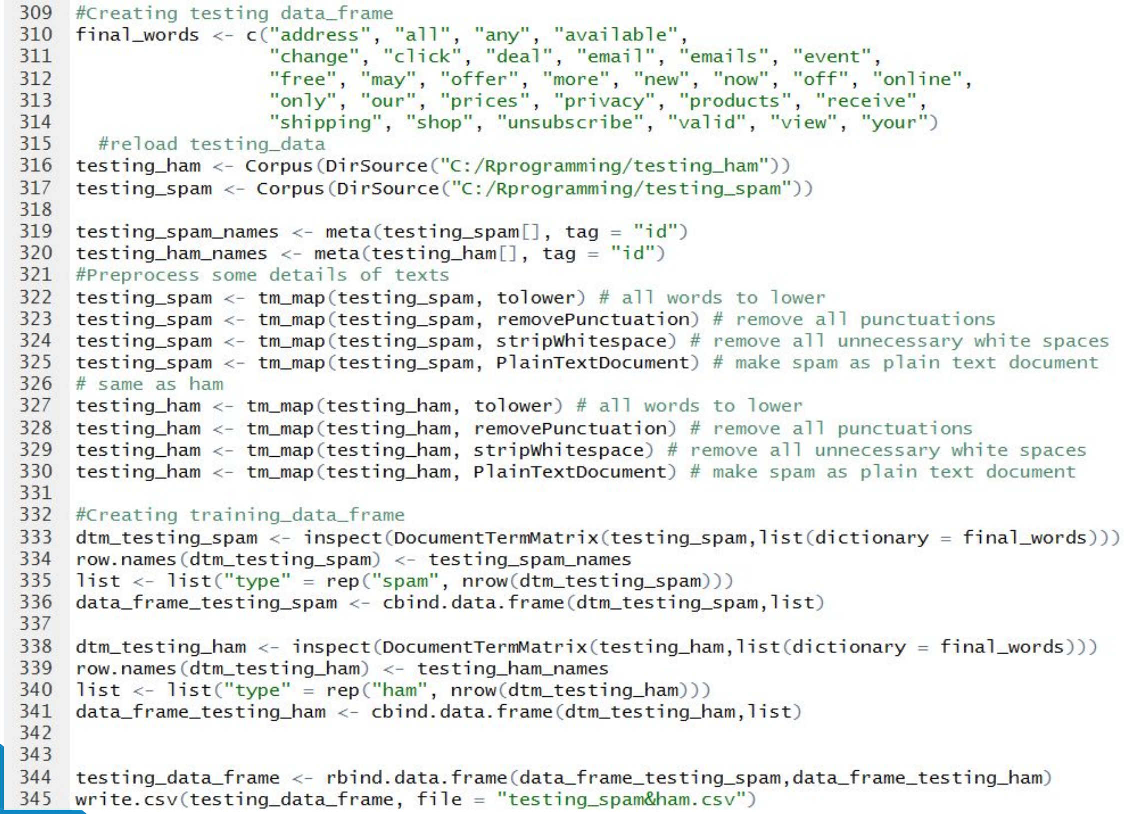
```
> total
               the              and            your             for              you            email              not 
              4148             3512            2384            2279             1832             1614             1607 
              with             this             may            from              are           please             view 
              1306             1266            1141            1039              916              878              876 
               any              our             off            ends            click              all             more 
               870              856             833             828              809              744              742 
               new             here          online   deliveredafter            shop          products          address 
               740              666             658             618              618              592              584 
              will              one            free            only          through            other             have 
               578              518             511             502              478              471              466 
           receive      unsubscribe            that             get        available            event             deal 
               451              445             442             429              426              408              406 
             offer              see          prices            york          subject          privacy         customer 
               403              401             397             397              395              384              381 
              2015             time           valid             now            april           emails              day 
               379              379             378             376              374              360              357 
               900         shipping             gnc          change            price          account      information 
               355              346             340             338              317              300              280 
             about            class         message             can            reply           weijia             2014 
               275              271             249             236              208              204              187 
               ave            xiong           wrote           brook            thank            stony           thanks 
               166              144             136             132              132              130              130 
            morgan         students          should            room             dear             lynn           dorado 
               114              106             102             100               97               89               88 
xwjanthonygmailcom             teng        citibank        shanghai          madison 
                85               82              81              81               80 
```

```
> words_rest
 [1] "2015"           "900"            "address"        "all"            "and"            "any"            "april"          "are"           
 [9] "available"      "change"         "click"          "day"            "deal"           "deliveredafter" "email"          "emails"        
[17] "ends"           "event"          "for"            "free"           "from"           "get"            "gnc"            "here"          
[25] "may"            "more"           "new"            "not"            "now"            "off"            "offer"          "one"           
[33] "online"         "only"           "our"            "price"          "prices"         "privacy"        "products"       "receive"       
[41] "see"            "shipping"       "shop"           "the"            "this"           "through"        "unsubscribe"    "valid"         
[49] "view"           "with"           "your"           "2014"           "weijia"        
```
]

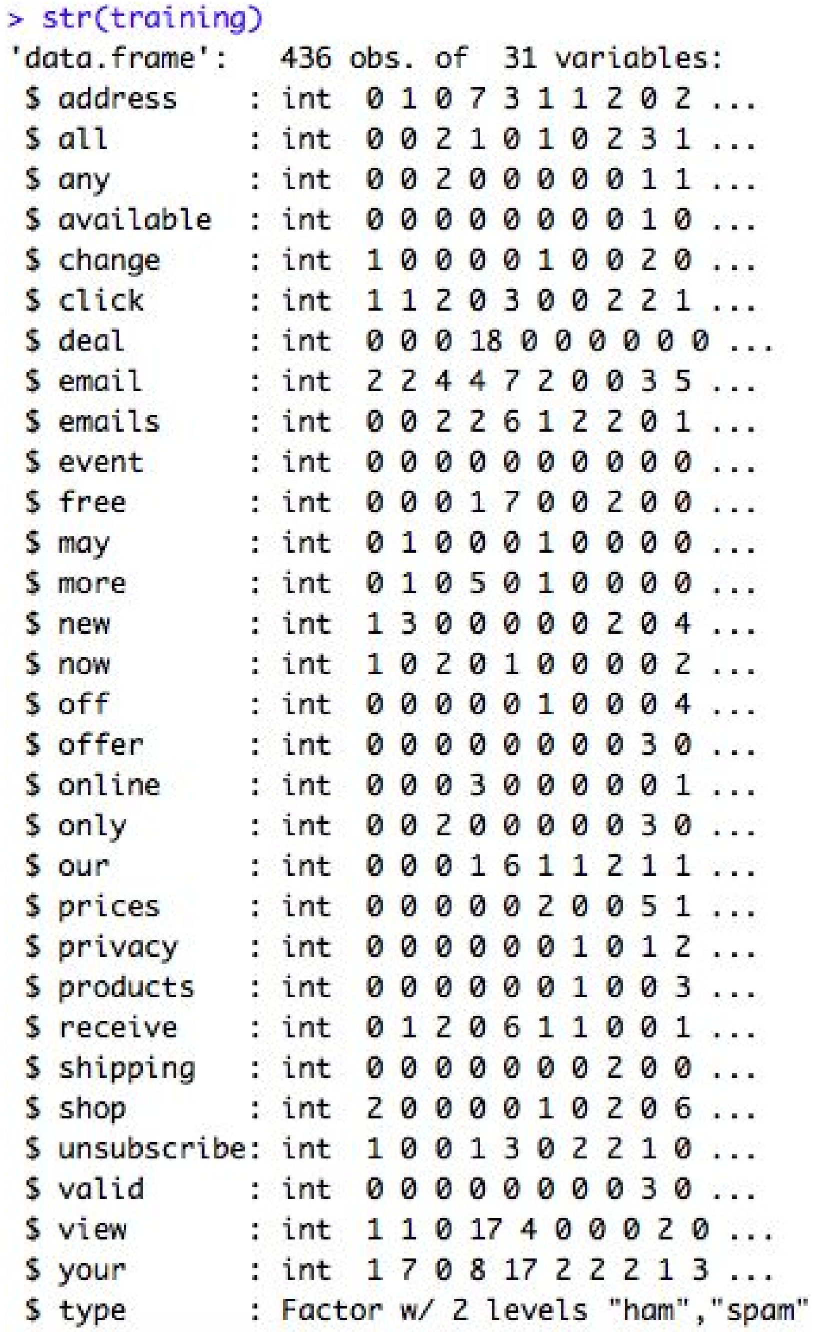
Extracting spam predictors

- "address","all","any","available","change","click","deal","email", "emails", "event", "free", "may", "more", "new", "now", "off", "offer", "online", "only", "our", "prices", "privacy", "products", "receive", "shipping", "shop", "unsubscribe", "valid", "view", "your", "type”

Creating data frame



```r
#Creating testing data_frame
final_words <- c("address", "all", "any", "available",
                 "change", "click", "deal", "email", "emails", "event",
                 "free", "may", "offer", "more", "new", "now", "off", "online",
                 "only", "our", "prices", "privacy", "products", "receive",
                 "shipping", "shop", "unsubscribe", "valid", "view", "your")
  #reload testing_data
testing_ham <- Corpus(DirSource("C:/Rprogramming/testing_ham"))
testing_spam <- Corpus(DirSource("C:/Rprogramming/testing_spam"))

testing_spam_names <- meta(testing_spam[], tag = "id")
testing_ham_names <- meta(testing_ham[], tag = "id")
#Preprocess some details of texts
testing_spam <- tm_map(testing_spam, tolower) # all words to lower
testing_spam <- tm_map(testing_spam, removePunctuation) # remove all punctuations
testing_spam <- tm_map(testing_spam, stripWhitespace) # remove all unnecessary white spaces
testing_spam <- tm_map(testing_spam, PlainTextDocument) # make spam as plain text document
# same as ham
testing_ham <- tm_map(testing_ham, tolower) # all words to lower
testing_ham <- tm_map(testing_ham, removePunctuation) # remove all punctuations
testing_ham <- tm_map(testing_ham, stripWhitespace) # remove all unnecessary white spaces
testing_ham <- tm_map(testing_ham, PlainTextDocument) # make spam as plain text document

#Creating training_data_frame
dtm_testing_spam <- inspect(DocumentTermMatrix(testing_spam,list(dictionary = final_words)))
row.names(dtm_testing_spam) <- testing_spam_names
list <- list("type" = rep("spam", nrow(dtm_testing_spam)))
data_frame_testing_spam <- cbind.data.frame(dtm_testing_spam,list)

dtm_testing_ham <- inspect(DocumentTermMatrix(testing_ham,list(dictionary = final_words)))
row.names(dtm_testing_ham) <- testing_ham_names
list <- list("type" = rep("ham", nrow(dtm_testing_ham)))
data_frame_testing_ham <- cbind.data.frame(dtm_testing_ham,list)


testing_data_frame <- rbind.data.frame(data_frame_testing_spam,data_frame_testing_ham)
write.csv(testing_data_frame, file = "testing_spam&ham.csv")
```

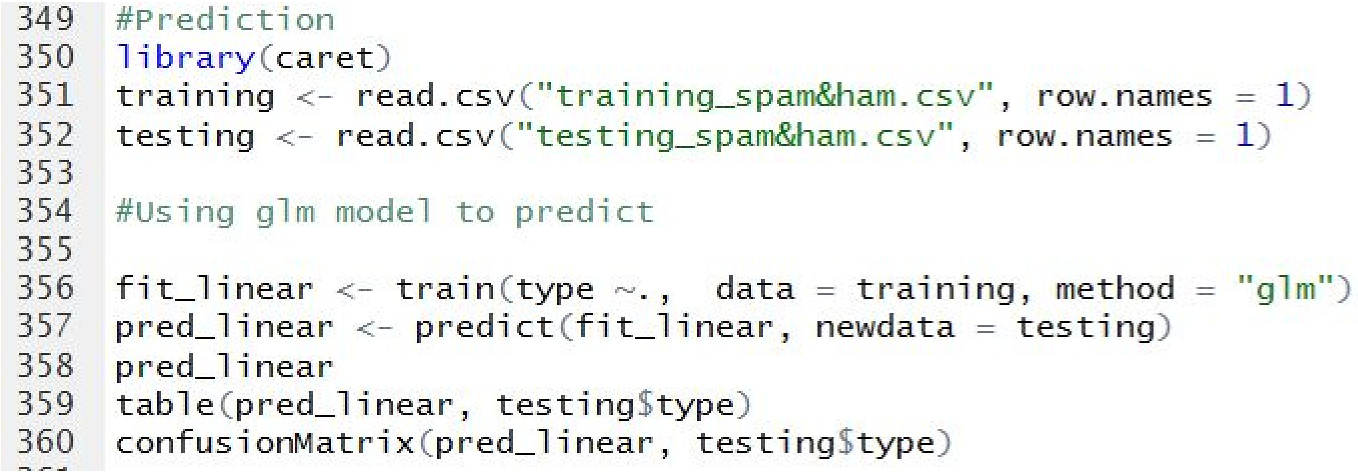


```
> str(training)
'data.frame':   436 obs. of  31 variables:
 $ address    : int  0 1 0 7 3 1 1 2 0 2 ...
 $ all        : int  0 0 2 1 0 1 0 2 3 1 ...
 $ any        : int  0 0 2 0 0 0 0 0 1 1 ...
 $ available  : int  0 0 0 0 0 0 0 0 1 0 ...
 $ change     : int  1 0 0 0 0 1 0 0 2 0 ...
 $ click      : int  1 1 2 0 3 0 0 2 2 1 ...
 $ deal       : int  0 0 0 18 0 0 0 0 0 0 ...
 $ email      : int  2 2 4 4 7 2 0 0 3 5 ...
 $ emails     : int  0 0 2 2 6 1 2 2 0 1 ...
 $ event      : int  0 0 0 0 0 0 0 0 0 0 ...
 $ free       : int  0 0 0 1 7 0 0 2 0 0 ...
 $ may        : int  0 1 0 0 0 1 0 0 0 0 ...
 $ more       : int  0 1 0 5 0 1 0 0 0 0 ...
 $ new        : int  1 3 0 0 0 0 0 2 0 4 ...
 $ now        : int  1 0 2 0 1 0 0 0 0 2 ...
 $ off        : int  0 0 0 0 0 1 0 0 0 4 ...
 $ offer      : int  0 0 0 0 0 0 0 0 3 0 ...
 $ online     : int  0 0 0 3 0 0 0 0 0 1 ...
 $ only       : int  0 0 2 0 0 0 0 0 3 0 ...
 $ our        : int  0 0 0 1 6 1 1 2 1 1 ...
 $ prices     : int  0 0 0 0 0 2 0 0 5 1 ...
 $ privacy    : int  0 0 0 0 0 0 1 0 1 2 ...
 $ products   : int  0 0 0 0 0 0 1 0 0 3 ...
 $ receive    : int  0 1 2 0 6 1 1 0 0 1 ...
 $ shipping   : int  0 0 0 0 0 0 0 2 0 0 ...
 $ shop       : int  2 0 0 0 0 1 0 2 0 6 ...
 $ unsubscribe: int  1 0 0 1 3 0 2 2 1 0 ...
 $ valid      : int  0 0 0 0 0 0 0 0 3 0 ...
 $ view       : int  1 1 0 17 4 0 0 0 2 0 ...
 $ your       : int  1 7 0 8 17 2 2 2 1 3 ...
 $ type       : Factor w/ 2 levels "ham","spam"
```

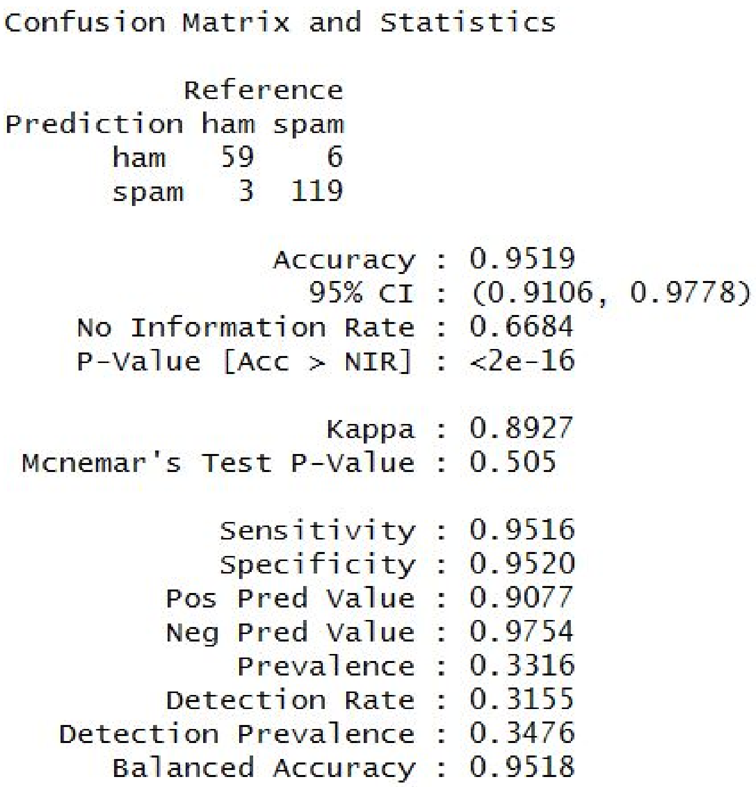
Machine Learning Models

Training and Testing

- Generalized Linear Model



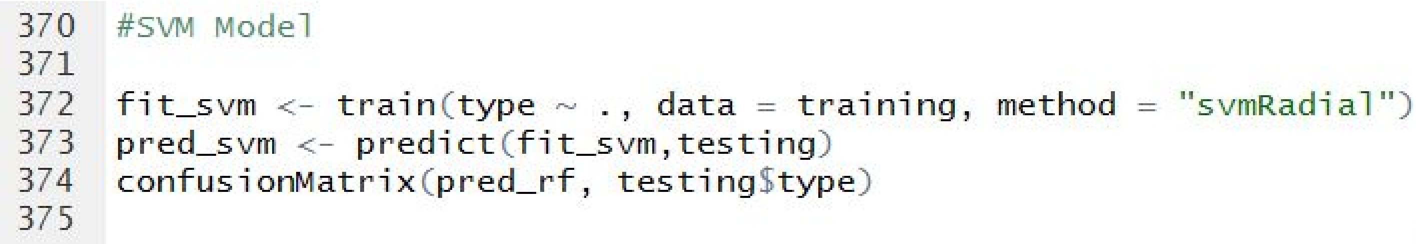
```
#Prediction
library(caret)
training <- read.csv("training_spam&ham.csv", row.names = 1)
testing <- read.csv("testing_spam&ham.csv", row.names = 1)

#Using glm model to predict

fit_linear <- train(type ~.,  data = training, method = "glm")
pred_linear <- predict(fit_linear, newdata = testing)
pred_linear
table(pred_linear, testing$type)
confusionMatrix(pred_linear, testing$type)
```



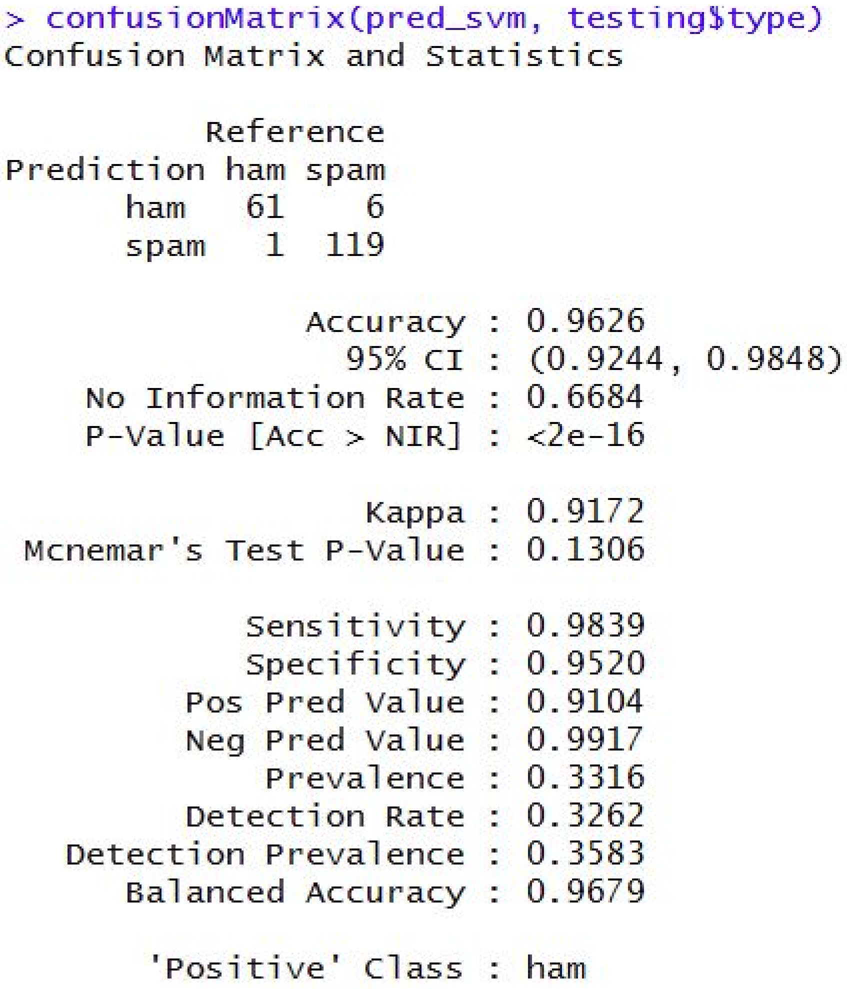
```
Confusion Matrix and Statistics

          Reference
Prediction ham spam
      ham   59    6
      spam   3  119

               Accuracy : 0.9519
                 95% CI : (0.9106, 0.9778)
    No Information Rate : 0.6684
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.8927
 Mcnemar's Test P-Value : 0.505

            Sensitivity : 0.9516
            Specificity : 0.9520
         Pos Pred Value : 0.9077
         Neg Pred Value : 0.9754
             Prevalence : 0.3316
         Detection Rate : 0.3155
   Detection Prevalence : 0.3476
      Balanced Accuracy : 0.9518
```

Training and Testing

- SVM



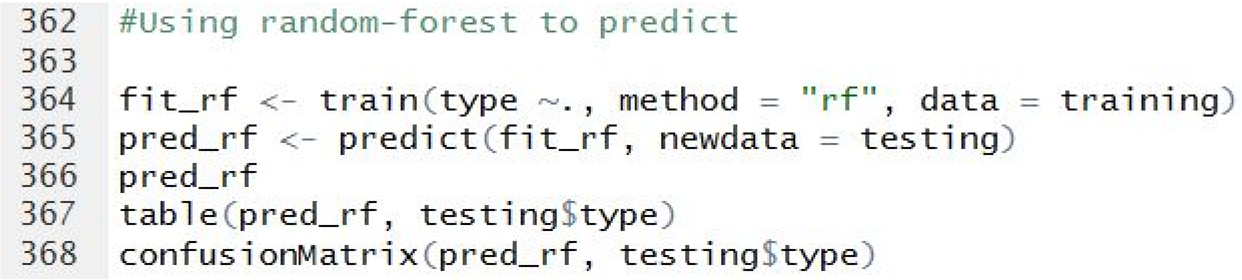
```r
#SVM Model

fit_svm <- train(type ~ ., data = training, method = "svmRadial")
pred_svm <- predict(fit_svm,testing)
confusionMatrix(pred_rf, testing$type)

```



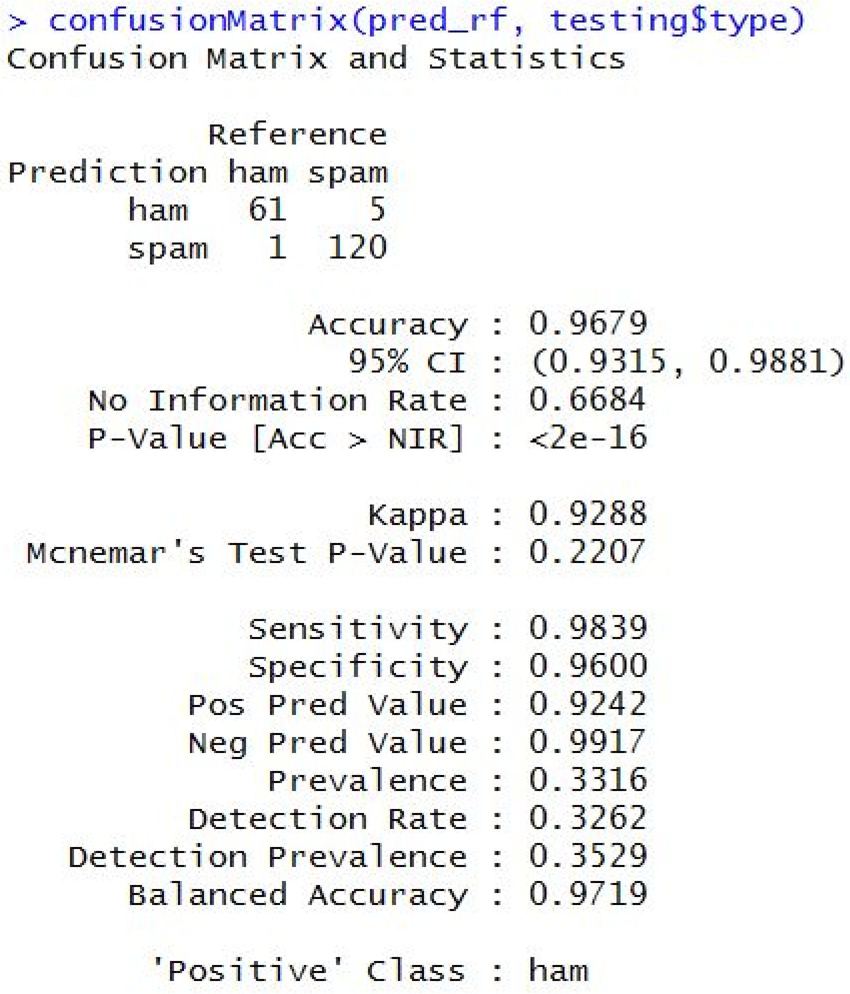
```
> confusionMatrix(pred_svm, testing$type)
Confusion Matrix and Statistics

          Reference
Prediction ham spam
      ham   61    6
      spam   1  119
                                          
               Accuracy : 0.9626          
                 95% CI : (0.9244, 0.9848)
    No Information Rate : 0.6684          
    P-Value [Acc > NIR] : <2e-16          
                                          
                  Kappa : 0.9172          
 Mcnemar's Test P-Value : 0.1306          
                                          
            Sensitivity : 0.9839          
            Specificity : 0.9520          
         Pos Pred Value : 0.9104          
         Neg Pred Value : 0.9917          
             Prevalence : 0.3316          
         Detection Rate : 0.3262          
   Detection Prevalence : 0.3583          
      Balanced Accuracy : 0.9679          
                                          
       'Positive' Class : ham             
```

Training and Testing

- Random Forest



```
#Using random-forest to predict

fit_rf <- train(type ~., method = "rf", data = training)
pred_rf <- predict(fit_rf, newdata = testing)
pred_rf
table(pred_rf, testing$type)
confusionMatrix(pred_rf, testing$type)
```



```
> confusionMatrix(pred_rf, testing$type)
Confusion Matrix and Statistics

          Reference
Prediction ham spam
      ham   61    5
      spam   1  120

               Accuracy : 0.9679
                 95% CI : (0.9315, 0.9881)
    No Information Rate : 0.6684
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9288
 Mcnemar's Test P-Value : 0.2207

            Sensitivity : 0.9839
            Specificity : 0.9600
         Pos Pred Value : 0.9242
         Neg Pred Value : 0.9917
             Prevalence : 0.3316
         Detection Rate : 0.3262
   Detection Prevalence : 0.3529
      Balanced Accuracy : 0.9719

       'Positive' Class : ham
```

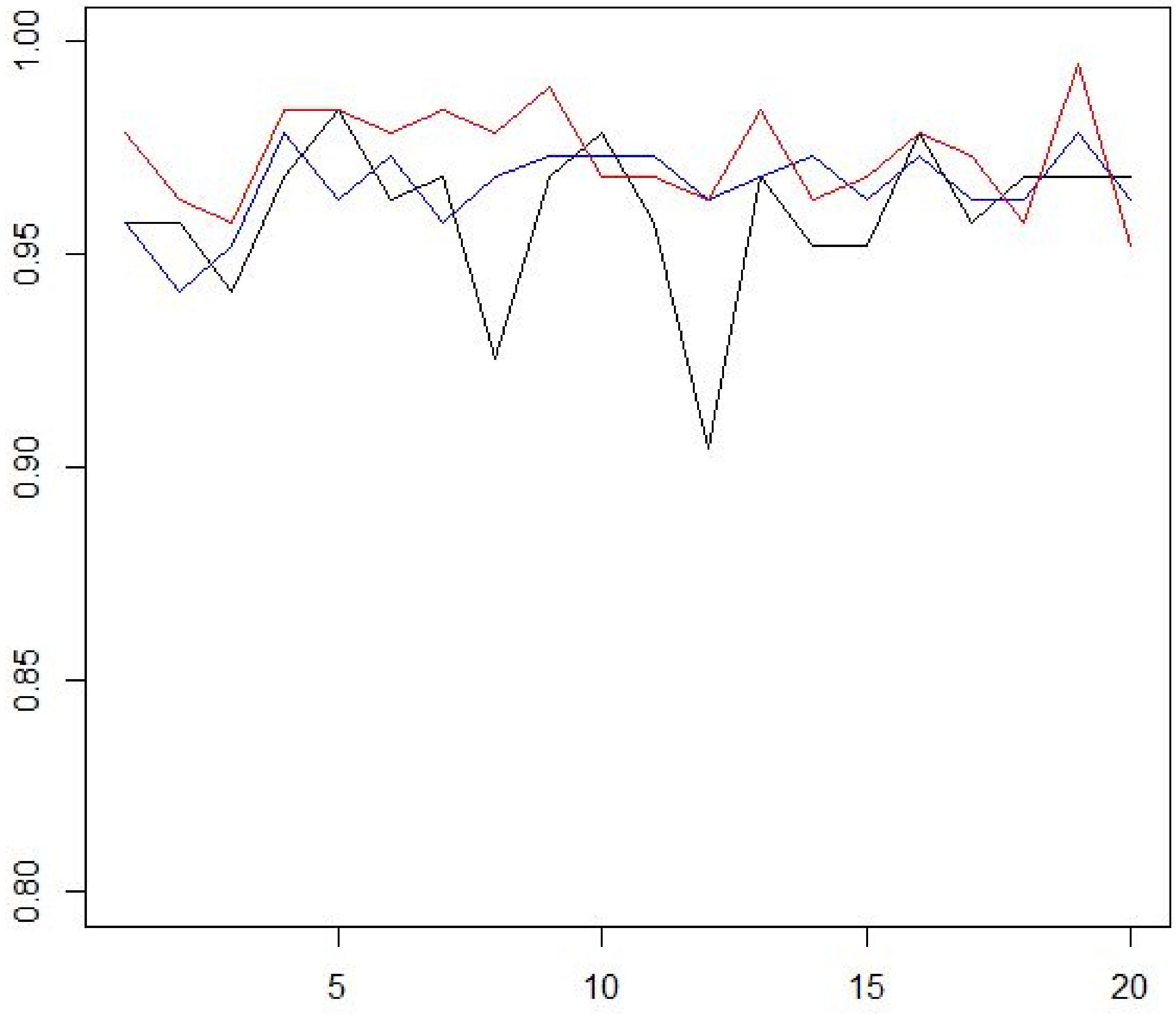
Conclusion

Conclusion

Comparing 3 models

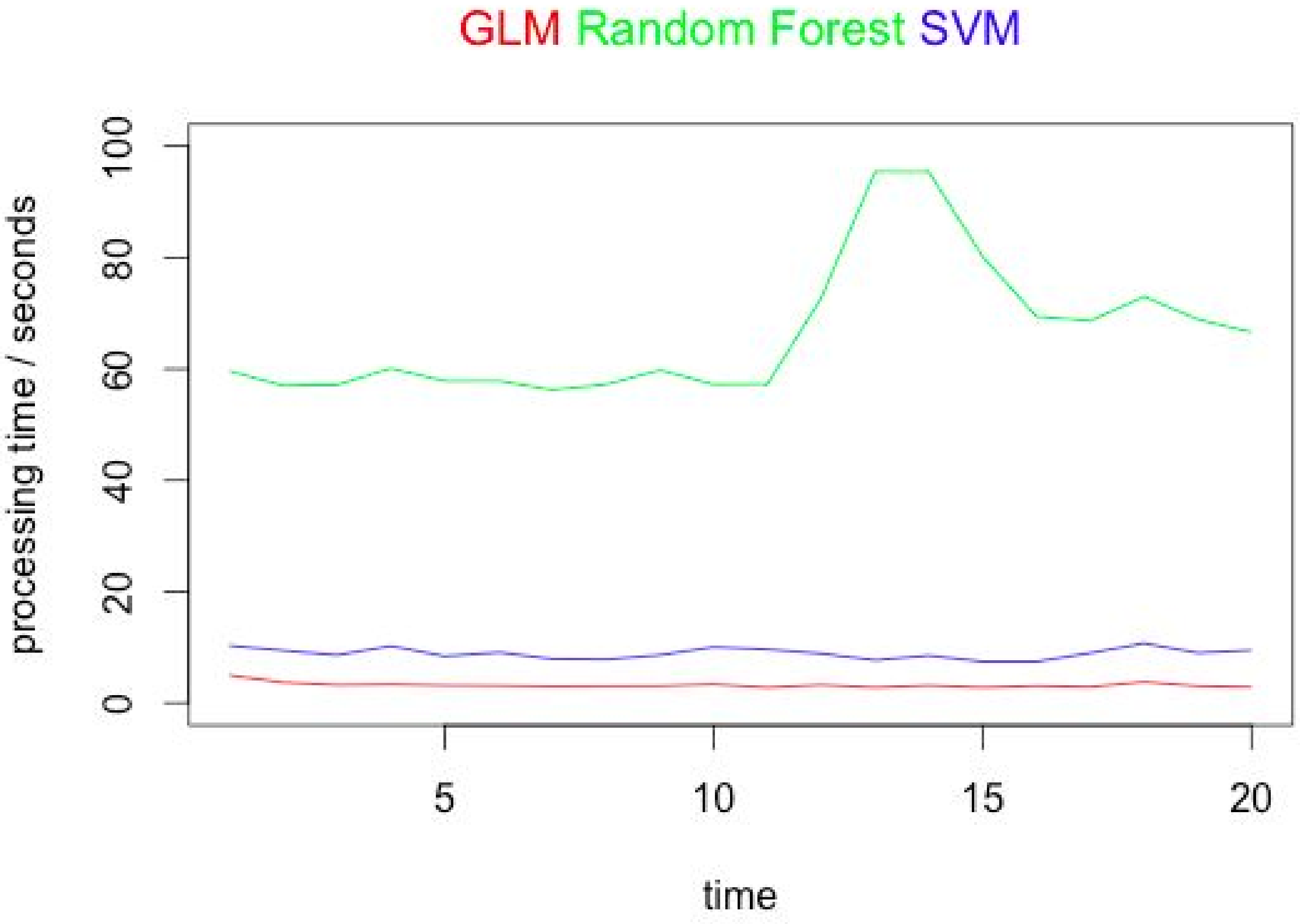
GLM SVM Random Forest

Accuracy



Comparing 3 models

Processing time



False recognition

Negative spam (7 emails)

- 3 order emails
- 2 emails asking for feedback
- 1 email from China
- 1 email too many words

Positive ham (3 emails)

- “email”, “receive”
- “email”, “receive”
- Too many replies which increase some word frequency (your)

Suggestion

- Add more predictors related with “order confirmation emails”
- Consider email sender’s background and recipient’s preference
- Consider different languages