Proj4 - SPAM Classification

Sean Connin & Ethan Haley

4/26/2021

Introduction

The purpose of this project is to develop a complete machine learning pipeline to classify incoming email as either ham or spam. And to evaluate/critique the model's performance. Our approach to this task includes substantive steps to process our data, engineer new features (with possible predictive value), and then develop a classification scheme using logistic regression and random forest models.

We employed tidymodels for our model development - which included model parameterization, training, cross-validation, and testing. We also reviewed our results to distinguish elements of the spam/ham emails that were important to the classification results.

Our project steps included the following:

1. Collect relevant folder and files for processing
2. Hand-engineer features based primarily on email headers
3. Build, train, and analyze results of logistic regression classifier
4. Process email text to facilitate modeling
5. Build, cross-validate, and analyze results of random forest classifier
6. Compare results to null model

With that sequence in mind, let's build our libraries.

```r
library(xml2)
library(tidymodels)
```

```
## ── Attaching packages ────────────────────────────────────── tidymodels 0.1.2 ──
```

```
## ✓ broom     0.7.3      ✓ recipes   0.1.15
## ✓ dials     0.0.9      ✓ rsample   0.0.9
## ✓ dplyr     1.0.3      ✓ tibble    3.1.0
## ✓ ggplot2   3.3.3      ✓ tidyr     1.1.2
## ✓ infer     0.5.4      ✓ tune      0.1.3
## ✓ modeldata 0.1.0      ✓ workflows 0.2.2
## ✓ parsnip   0.1.5      ✓ yardstick 0.0.8
## ✓ purrr     0.3.4
```

```
## ── Conflicts ───────────────────────────────────────── tidymodels_conflicts() ──
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
```

```r
library(magrittr)
```

```
## 
## Attaching package: 'magrittr'
```

```
## The following object is masked from 'package:tidyr':
## 
##     extract
```

```
## The following object is masked from 'package:purrr':
## 
##     set_names
```

```r
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──
```

```
## ✓ readr   1.4.0     ✓ forcats 0.5.0
## ✓ stringr 1.4.0
## ── Conflicts ──────────────────────────────────────────── tidyverse_conflicts() ──
## x readr::col_factor()   masks scales::col_factor()
## x purrr::discard()      masks scales::discard()
## x magrittr::extract()   masks tidyr::extract()
## x dplyr::filter()       masks stats::filter()
## x stringr::fixed()      masks recipes::fixed()
## x dplyr::lag()          masks stats::lag()
## x magrittr::set_names() masks purrr::set_names()
## x readr::spec()         masks yardstick::spec()
```

```
library(glue)
```

```
##
## Attaching package: 'glue'
```

```
## The following object is masked from 'package:dplyr':
##
##     collapse
```

```
library(textrecipes)
library(discrim)
```

```
##
## Attaching package: 'discrim'
```

```
## The following object is masked from 'package:dials':
##
##     smoothness
```

```
library(tidytext)
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(ranger)
```

```
##
## Attaching package: 'ranger'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```
library(themis)
```

```
## Registered S3 methods overwritten by 'themis':
##   method                  from   
##   bake.step_downsample    recipes
##   bake.step_upsample      recipes
##   prep.step_downsample    recipes
##   prep.step_upsample      recipes
##   tidy.step_downsample    recipes
##   tidy.step_upsample      recipes
##   tunable.step_downsample recipes
##   tunable.step_upsample   recipes
```

```
## 
## Attaching package: 'themis'
```

```
## The following objects are masked from 'package:recipes':
## 
##     step_downsample, step_upsample
```

```
library(skimr)
```

1. Collect relevant folder and files for processing

After cd'ing to the directory where you downloaded spam/ham corpus on your computer

```
dir()
```

```
##  [1] "easy_ham"            "easy_ham_2"          "emails.csv"         
##  [4] "hard_ham"            "headersBodies.csv"   "Proj4.html"         
##  [7] "Proj4.rmd"           "RFspamconfusion.png" "spam"               
## [10] "spam_2"
```

Collect all email filepaths in one vector

```
filepath <- c()

for (d in c("spam", "easy_ham", "easy_ham_2", "hard_ham", "spam_2")) {
  for (f in dir(d)) {
    path <- file.path(d, f)
    filepath <- c(filepath, path)
  }
}
filepath[1:3]
```

```
## [1] "spam/00001.7848dde101aa985090474a91ec93fcf0"
## [2] "spam/00002.d94f1b97e48ed3b553b3508d116e6a09"
## [3] "spam/00003.2ee33bc6eacdb11f38d052c44819ba6c"
```

Make a function to split header and body, store filenames, and classify as spam/ham

```r
filepaths <- c()
headers <- c()
bodies <- c()
spams <- c()

header_and_body <- function(filestring) {
  linelist <- read_lines(filestring)
  len <- length(linelist)
  for (line in 1:len) {
    if (linelist[line] == '') {
      header <- glue_collapse(linelist[1:(line-1)], sep='\n')
      body <- glue_collapse(linelist[line:len], sep='\n')
      return(c(filestring, header, body, str_starts(filestring, 's')))
    }
  }
}
#Use the vector of filepaths from the previous chunk
for (fp in filepath) {
  parsed <- header_and_body(fp)
  filepaths <- c(filepaths, parsed[1])
  headers <- c(headers, parsed[2])
  bodies <- c(bodies, parsed[3])
  spams <- c(spams, parsed[4])
}

d.f <- data.frame('filepaths'=filepaths, 'headers'=headers,
                  'bodies'=bodies, 'is_spam'=spams)

#write.csv(d.f, 'headersBodies.csv')
```

2. Hand-engineer features based primarily on email headers

```r
get_sender <- function(string) {
  # 'John@aol.com'
  str_replace(str_match(string, 'From:? \\S+'), 'From:? ', '')
}
get_angles <- function(string) {
  # How many left angle brackets are there
  str_count(string, '<')
}
get_uppers <- function(string) {
  # How many caps chars
  str_count(string, '[A-Z]')
}
get_lowers <- function(string) {
  # How many lower chars
  str_count(string, '[a-z]')
}
get_return_path <- function(string) {
  # should be similar to get_sender(string)
  str_replace_all(str_match(string, 'Return-Path: <\\S+>'),
              c('Return-Path: <' = '', '>' = ''))
}
get_ips <- function(string) {
  # list of all IP addresses
  str_match_all(string, "[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}")
}
get_subject <- function(string) {
  str_remove(str_match(string, 'Subject: [^\\n]+'), 'Subject: ')
}
get_zs <- function(string) {
  str_count(string, 'z{3,6}')
}
get_ys <- function(string) {
  str_count(string, 'y{3,6}')
}
get_xs <- function(string) {
  str_count(string, 'x{3,6}')
}
get_ws <- function(string) {
  str_count(string, 'w{3,6}')
}
get_domains <- function(string) {
  str_count(string, '\\S@\\S+\\.\\S')
}
has_yahoo_groups <- function(string) {
  str_detect(string, '@yahoogroups')
}
re_subject <- function(string) {
  str_detect(string, '(?i)\\bre: ')
}
has_localhost <- function(ip_list) {
  '127.0.0.1' %in% ip_list[[1]]
}
get_excl <- function(string) {
  str_count(string, '!')
}
has_bad_html <- function(string) {
  is.logical(possibly(read_html, TRUE)(string))
}
get_colors <- function(string) {
  str_count(string, '(?i)colou?r')
}
get_3Ds <- function(string) {
  str_count(string, '(?i)3d')
}
get_fonts <- function(string) {
  str_count(string, '(?i)font')
}
get_sizes <- function(string) {
  str_count(string, '(?i)size')
}
get_aligns <- function(string) {
  str_count(string, '(?i)align')
}
has_sex_words <- function(string) {
  str_detect(tolower(string), 'sex|penis|penile|viagra')
}
```

Apply the above functions to add features to the data.frame

```
d.f$bad_header <- unlist(lapply(d.f$headers, has_bad_html))
d.f$bad_body <- unlist(lapply(d.f$bodies, has_bad_html))
d.f$senders <- unlist(lapply(d.f$headers, get_sender))
d.f$subjects <- unlist(lapply(d.f$headers, get_subject))
d.f$noSub <- is.na(d.f$subjects)
d.f$re_subj <- unlist(lapply(d.f$subjects, re_subject))
d.f$headercolors <- unlist(lapply(d.f$headers, get_colors))
d.f$bodycolors <- unlist(lapply(d.f$bodies, get_colors))
d.f$header3Ds <- unlist(lapply(d.f$headers, get_3Ds))
d.f$body3Ds <- unlist(lapply(d.f$bodies, get_3Ds))
d.f$headerfonts <- unlist(lapply(d.f$headers, get_fonts))
d.f$bodyfonts <- unlist(lapply(d.f$bodies, get_fonts))
d.f$headersizes <- unlist(lapply(d.f$headers, get_sizes))
d.f$bodysizes <- unlist(lapply(d.f$bodies, get_sizes))
d.f$headeraligns <- unlist(lapply(d.f$headers, get_aligns))
d.f$bodyaligns <- unlist(lapply(d.f$bodies, get_aligns))
d.f$headerWs <- unlist(lapply(d.f$headers, get_ws))
d.f$headerXs <- unlist(lapply(d.f$headers, get_xs))
d.f$headerYs <- unlist(lapply(d.f$headers, get_ys))
d.f$headerZs <- unlist(lapply(d.f$headers, get_zs))
d.f$bodyWs <- unlist(lapply(d.f$bodies, get_ws))
d.f$bodyXs <- unlist(lapply(d.f$bodies, get_xs))
d.f$bodyYs <- unlist(lapply(d.f$bodies, get_ys))
d.f$bodyZs <- unlist(lapply(d.f$bodies, get_zs))
d.f$headerExcl <- unlist(lapply(d.f$headers, get_excl))
d.f$subjectExcl <- unlist(lapply(d.f$subjects, get_excl))
d.f$headerChars <- unlist(lapply(d.f$headers, possibly(nchar, 1)))
d.f$bodyChars <- unlist(lapply(d.f$bodies, possibly(nchar, 1)))
d.f$yahoos <- unlist(lapply(d.f$headers, has_yahoo_groups))

ipLists <- lapply(d.f$headers, get_ips)
d.f$hasLocals <- unlist(lapply(ipLists, has_localhost))

d.f$domains <- unlist(lapply(d.f$headers, get_domains))
d.f$subjectCAPS <- unlist(lapply(d.f$subjects, get_uppers))
d.f$headerBodyRatio <- d.f$headerChars / d.f$bodyChars
d.f$headerAngles <- unlist(lapply(d.f$headers, get_angles))
d.f$bodyAngles <- unlist(lapply(d.f$bodies, get_angles))
d.f$hAngleRatio <- d.f$headerAngles / d.f$headerChars
d.f$bAngleRatio <- d.f$bodyAngles / d.f$bodyChars
d.f$bodyCAPS <- unlist(lapply(d.f$bodies, get_uppers)) / d.f$bodyChars
d.f$bodyLowers <- unlist(lapply(d.f$bodies, get_lowers)) / d.f$bodyChars
d.f$exclRatio <- unlist(lapply(d.f$bodies, get_excl)) / d.f$bodyChars
d.f$subjectSex <- unlist(lapply(d.f$subjects, has_sex_words))
```

```
head(d.f, 2)
```

```
##                                  filepaths
## 1 spam/00001.7848dde101aa985090474a91ec93fcf0
## 2 spam/00002.d94f1b97e48ed3b553b3508d116e6a09
##
headers
## 1
From 12a1mailbot1@web.de  Thu Aug 22 13:17:22 2002\nReturn-Path: <12a1mailbot1@web.de>\nDelivered-To: zzzz@localhost.spamass
assin.taint.org\nReceived: from localhost (localhost [127.0.0.1])\n\tby phobos.labs.spamassassin.taint.org (Postfix) with ES
MTP id 136B943C32\n\tfor <zzzz@localhost>; Thu, 22 Aug 2002 08:17:21 -0400 (EDT)\nReceived: from mail.webnote.net [193.120.2
11.219]\n\tby localhost with POP3 (fetchmail-5.9.0)\n\tfor zzzz@localhost (single-drop); Thu, 22 Aug 2002 13:17:21 +0100 (IS
T)\nReceived: from dd_it7 ([210.97.77.167])\n\tby webnote.net (8.9.3/8.9.3) with ESMTP id NAA04623\n\tfor <zzzz@spamassassi
n.taint.org>; Thu, 22 Aug 2002 13:09:41 +0100\nFrom: 12a1mailbot1@web.de\nReceived: from r-smtp.korea.com - 203.122.2.197 by
dd_it7  with Microsoft SMTPSVC(5.5.1775.675.6);\n\t Sat, 24 Aug 2002 09:42:10 +0900\nTo: <dcek1a1@netsgo.com>\nSubject: Life
Insurance - Why Pay More?\nDate: Wed, 21 Aug 2002 20:31:57 -1600\nMIME-Version: 1.0\nMessage-ID: <0103c1042001882DD_IT7@dd_i
t7>\nContent-Type: text/html; charset="iso-8859-1"\nContent-Transfer-Encoding: quoted-printable
## 2 From ilug-admin@linux.ie  Thu Aug 22 13:27:39 2002\nReturn-Path: <ilug-admin@linux.ie>\nDelivered-To: zzzz@localhost.sp
amassassin.taint.org\nReceived: from localhost (localhost [127.0.0.1])\n\tby phobos.labs.spamassassin.taint.org (Postfix) wi
th ESMTP id A7FD7454F6\n\tfor <zzzz@localhost>; Thu, 22 Aug 2002 08:27:38 -0400 (EDT)\nReceived: from phobos [127.0.0.1]\n\t
by localhost with IMAP (fetchmail-5.9.0)\n\tfor zzzz@localhost (single-drop); Thu, 22 Aug 2002 13:27:38 +0100 (IST)\nReceive
d: from lugh.tuatha.org (root@lugh.tuatha.org [194.125.145.45]) by\n    dogma.slashnull.org (8.11.6/8.11.6) with ESMTP id g7
MCJiZ06043 for\n    <zzzz-ilug@jmason.org>; Thu, 22 Aug 2002 13:19:44 +0100\nReceived: from lugh (root@localhost [127.0.0.
1]) by lugh.tuatha.org\n    (8.9.3/8.9.3) with ESMTP id NAA29323; Thu, 22 Aug 2002 13:18:52 +0100\nReceived: from email.qve
s.com ([67.104.83.251]) by lugh.tuatha.org\n    (8.9.3/8.9.3) with ESMTP id NAA29282 for <ilug@linux.ie>; Thu,\n    22 Aug 2
002 13:18:37 +0100\nX-Authentication-Warning: lugh.tuatha.org: Host [67.104.83.251] claimed to\n    be email.qves.com\nRecei
ved: from qvp0091 ([169.254.6.22]) by email.qves.com with Microsoft\n    SMTPSVC(5.0.2195.2966); Thu, 22 Aug 2002 06:18:18 -
0600\nFrom: "Slim Down" <taylor@s3.serveimage.com>\nTo: <ilug@linux.ie>\nDate: Thu, 22 Aug 2002 06:18:18 -0600\nMessage-Id:
<59e6301c249d5$ffb7ea20$1606fea9@freeyankeedom.com>\nMIME-Version: 1.0\nContent-Type: text/plain; charset="iso-8859-1"\nCont
ent-Transfer-Encoding: 7bit\nX-Mailer: Microsoft CDO for Windows 2000\nThread-Index: AcJJ1f+3FWdz11AmR6uWbmQN5gGxxw==\nConte
nt-Class: urn:content-classes:message\nX-Mimeole: Produced By Microsoft MimeOLE V6.00.2462.0000\nX-Originalarrivaltime: 22 A
ug 2002 12:18:18.0699 (UTC) FILETIME=[FFB949B0:01C249D5]\nSubject: [ILUG] Guaranteed to lose 10-12 lbs in 30 days 10.206\nSe
nder: ilug-admin@linux.ie\nErrors-To: ilug-admin@linux.ie\nX-Mailman-Version: 1.1\nPrecedence: bulk\nList-Id: Irish Linux Us
ers' Group <ilug.linux.ie>\nX-Beenthere: ilug@linux.ie
##
bodies
## 1 \n<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">\n<HTML><HEAD>\n<META content=3D"text/html; charset=3Dw
indows-1252" http-equiv=3DContent-T=\nype>\n<META content=3D"MSHTML 5.00.2314.1000" name=3DGENERATOR></HEAD>\n<BODY><!-- Ins
erted by Calypso -->\n<TABLE border=3D0 cellPadding=3D0 cellSpacing=3D2 id=3D_CalyPrintHeader_ r=\nules=3Dnone \nstyle=3D"CO
LOR: black; DISPLAY: none" width=3D"100%">\n  <TBODY>\n  <TR>\n    <TD colSpan=3D3>\n      <HR color=3Dblack noShade SIZE=3D
1>\n    </TD></TR></TD></TR>\n  <TR>\n    <TD colSpan=3D3>\n      <HR color=3Dblack noShade SIZE=3D1>\n    </TD></TR></TBODY
></TABLE><!-- End Calypso --><!-- Inserted by Calypso=\n --><FONT \ncolor=3D#000000 face=3DVERDANA,ARIAL,HELVETICA size=3D-2
><BR></FONT></TD><=\n/TR></TABLE><!-- End Calypso --><FONT color=3D#ff0000 \nface=3D"Copperplate Gothic Bold" size=3D5 PTSIZ
E=3D"10">\n<CENTER>Save up to 70% on Life Insurance.</CENTER></FONT><FONT color=3D#ff=\n0000 \nface=3D"Copperplate Gothic Bo
ld" size=3D5 PTSIZE=3D"10">\n<CENTER>Why Spend More Than You Have To?\n<CENTER><FONT color=3D#ff0000 face=3D"Copperplate Got
hic Bold" size=3D5 PT=\nSIZE=3D"10">\n<CENTER>Life Quote Savings\n<CENTER>\n<P align=3Dleft></P>\n<P align=3Dleft></P></FONT
></U></I></B><BR></FONT></U></B></U></I>\n<P></P>\n<CENTER>\n<TABLE border=3D0 borderColor=3D#111111 cellPadding=3D0 cellSpa
cing=3D0 wi=\ndth=3D650>\n  <TBODY></TBODY></TABLE>\n<TABLE border=3D0 borderColor=3D#111111 cellPadding=3D5 cellSpacing=3D0
wi=\ndth=3D650>\n  <TBODY>\n  <TR>\n    <TD colSpan=3D2 width=3D"35%"><B><FONT face=3DVerdana size=3D4>Ensurin=\ng your \n
family's financial security is very important. Life Quote Savings ma=\nkes \n      buying life insurance simple and affordab
le. We Provide FREE Access =\nto The \n      Very Best Companies and The Lowest Rates.</FONT></B></TD></TR>\n  <TR>\n    <TD
align=3Dmiddle vAlign=3Dtop width=3D"18%">\n      <TABLE borderColor=3D#111111 width=3D"100%">\n        <TBODY>\n        <TR
>\n          <TD style=3D"PADDING-LEFT: 5px; PADDING-RIGHT: 5px" width=3D"100=\n%"><FONT \n            face=3DVerdana size=3
D4><B>Life Quote Savings</B> is FAST, EAS=\nY and \n            SAVES you money! Let us help you get started with the best v
al=\nues in \n            the country on new coverage. You can SAVE hundreds or even tho=\nusands \n            of dollars b
y requesting a FREE quote from Lifequote Savings. =\nOur \n            service will take you less than 5 minutes to complet
e. Shop an=\nd \n            compare. SAVE up to 70% on all types of Life insurance! \n</FONT></TD></TR>\n        <TR><BR><B
R>\n          <TD height=3D50 style=3D"PADDING-LEFT: 5px; PADDING-RIGHT: 5px" \n          width=3D"100%">\n            <P al
ign=3Dcenter><B><FONT face=3DVerdana size=3D5><A \n            href=3D"http://website.e365.cc/savequote/">Click Here For You
r=\n \n            Free Quote!</A></FONT></B></P></TD>\n          <P><FONT face=3DVerdana size=3D4><STRONG>\n          <CENT
ER>Protecting your family is the best investment you'll eve=\nr \n          make!<BR></B></TD></TR>\n        <TR><BR><BR></S
TRONG></FONT></TD></TR></TD></TR>\n        <TR></TR></TBODY></TABLE>\n      <P align=3Dleft><FONT face=3D"Arial, Helvetica,
sans-serif" size=3D2=\n></FONT></P>\n      <P></P>\n      <CENTER><BR><BR><BR>\n      <P></P>\n      <P align=3Dleft><BR></B
><BR><BR><BR><BR></P>\n      <P align=3Dcenter><BR></P>\n      <P align=3Dleft><BR></B><BR><BR></FONT>If you are in receipt
of this=\n email \n      in error and/or wish to be removed from our list, <A \n      href=3D"mailto:coins@btamail.net.cn">P
LEASE CLICK HERE</A> AND TYPE =\nREMOVE. If you \n      reside in any state which prohibits e-mail solicitations for insuran
=\nce, \n      please disregard this \n      email.<BR></FONT><BR><BR><BR><BR><BR><BR><BR><BR><BR><BR><BR><BR><BR=\n><BR><BR
><BR></FONT></P></CENTER></CENTER></TR></TBODY></TABLE></CENTER></=\nCENTER></CENTER></CENTER></CENTER></BODY></HTML>\n\n\n
## 2
\n1) Fight The Risk of Cancer!\nhttp://www.adclick.ws/p.cfm?o=315&s=pk007\n\n2) Slim Down - Guaranteed to lose 10-12 lbs in
30 days\nhttp://www.adclick.ws/p.cfm?o=249&s=pk007\n\n3) Get the Child Support You Deserve - Free Legal Advice\nhttp://www.a
dclick.ws/p.cfm?o=245&s=pk002\n\n4) Join the Web's Fastest Growing Singles Community\nhttp://www.adclick.ws/p.cfm?o=259&s=pk
007\n\n5) Start Your Private Photo Album Online!\nhttp://www.adclick.ws/p.cfm?o=283&s=pk007\n\nHave a Wonderful Day,\nOffer
Manager\nPrizeMama\n\n\n\n\n\n\n\n\n\n\n\n\n\nIf you wish to leave this list please use the link below.\nhttp://www.qves.co
m/trim/?ilug@linux.ie%7C17%7C114258\n\n\n-- \nIrish Linux Users' Group: ilug@linux.ie\nhttp://www.linux.ie/mailman/listinfo/
ilug for (un)subscription information.\nList maintainer: listmaster@linux.ie\n
##   is_spam bad_header bad_body             senders
## 1    TRUE      FALSE    FALSE 12a1mailbot1@web.de
## 2    TRUE      FALSE     TRUE ilug-admin@linux.ie
##                                              subjects noSub re_subj
## 1                        Life Insurance - Why Pay More? FALSE   FALSE
## 2 [ILUG] Guaranteed to lose 10-12 lbs in 30 days 10.206 FALSE   FALSE
##   headercolors bodycolors header3Ds body3Ds headerfonts bodyfonts headersizes
## 1            0         10         0      74           0        21           0
## 2            0          0         0       0           0         0           0
##   bodysizes headeraligns bodyaligns headerWs headerXs headerYs headerZs bodyWs
## 1        14            0          9        0        0        0        4      0
## 2         0            0          0        0        0        0        4      7
##   bodyXs bodyYs bodyZs headerExcl subjectExcl headerChars bodyChars yahoos
## 1      0      0      0          0           0        1060      3866  FALSE
## 2      0      0      0          0           0        1987       781  FALSE
##   hasLocals domains subjectCAPS headerBodyRatio headerAngles bodyAngles
## 1      TRUE       6           5       0.2741852            5        183
## 2      TRUE      12           5       2.5441741            8          0
##   hAngleRatio bAngleRatio   bodyCAPS bodyLowers   exclRatio subjectSex
## 1 0.004716981  0.04733575 0.22374547  0.3577341 0.002327988      FALSE
## 2 0.004026170  0.00000000 0.05377721  0.6197183 0.002560819      FALSE
```

Create a summary of feature statistics and categories

```
skim(d.f)
```

Data summary

Name	d.f
Number of rows	6046
Number of columns	45
_______________________	
Column type frequency:	
character	6
logical	7
numeric	32
________________________	
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
filepaths	0	1	43	49	0	6046	0
headers	0	1	234	15169	0	6046	0
bodies	0	1	32	299384	0	5875	0
is_spam	0	1	4	5	0	2	0
senders	0	1	1	84	0	1814	0
subjects	19	1	2	242	0	4323	0

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
bad_header	0	1	0.01	FAL: 5995, TRU: 51
bad_body	0	1	0.38	FAL: 3737, TRU: 2309
noSub	0	1	0.00	FAL: 6027, TRU: 19
re_subj	19	1	0.39	FAL: 3689, TRU: 2338
yahoos	0	1	0.02	FAL: 5912, TRU: 134
hasLocals	0	1	0.83	TRU: 5047, FAL: 999
subjectSex	19	1	0.01	FAL: 5964, TRU: 63

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
headercolors	0	1	0.00	0.03	0	0.00	0.00	0.00	1	▇▁▁▁▁

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
bodycolors	0	1	4.12	13.80	0	0.00	0.00	0.00	277	▇▁▁▁▁
header3Ds	0	1	0.23	0.59	0	0.00	0.00	0.00	6	▇▁▁▁▁
body3Ds	0	1	13.44	86.47	0	0.00	0.00	0.00	4297	▇▁▁▁▁
headerfonts	0	1	0.00	0.04	0	0.00	0.00	0.00	2	▇▁▁▁▁
bodyfonts	0	1	9.67	43.87	0	0.00	0.00	0.00	1628	▇▁▁▁▁
headersizes	0	1	0.00	0.07	0	0.00	0.00	0.00	2	▇▁▁▁▁
bodysizes	0	1	3.83	14.56	0	0.00	0.00	0.00	447	▇▁▁▁▁
headeraligns	0	1	0.00	0.01	0	0.00	0.00	0.00	1	▇▁▁▁▁
bodyaligns	0	1	2.37	8.24	0	0.00	0.00	0.00	144	▇▁▁▁▁
headerWs	0	1	0.29	0.72	0	0.00	0.00	0.00	12	▇▁▁▁▁
headerXs	0	1	0.01	0.20	0	0.00	0.00	0.00	9	▇▁▁▁▁
headerYs	0	1	1.00	0.91	0	0.00	1.00	1.00	8	▇▂▁▁▁
headerZs	0	1	1.59	3.65	0	0.00	0.00	0.00	17	▇▁▁▁▁
bodyWs	0	1	3.82	24.61	0	0.00	1.00	2.00	1491	▇▁▁▁▁
bodyXs	0	1	0.01	0.36	0	0.00	0.00	0.00	22	▇▁▁▁▁
bodyYs	0	1	0.01	0.19	0	0.00	0.00	0.00	7	▇▁▁▁▁
bodyZs	0	1	0.06	1.15	0	0.00	0.00	0.00	67	▇▁▁▁▁
headerExcl	0	1	0.25	1.29	0	0.00	0.00	0.00	84	▇▁▁▁▁
subjectExcl	19	1	0.15	0.76	0	0.00	0.00	0.00	42	▇▁▁▁▁
headerChars	0	1	1865.51	867.64	1	1091.00	1985.00	2391.75	15169	▇▁▁▁▁
bodyChars	0	1	2935.76	8102.87	1	522.00	1086.00	2318.00	299384	▇▁▁▁▁
domains	0	1	14.47	9.07	2	7.00	15.00	18.00	317	▇▁▁▁▁
subjectCAPS	19	1	5.34	5.76	0	2.00	4.00	6.00	81	▇▁▁▁▁
headerBodyRatio	0	1	148.99	572.57	0	0.69	1.77	3.64	15169	▇▁▁▁▁
headerAngles	0	1	11.09	7.90	1	5.00	9.00	16.00	316	▇▁▁▁▁
bodyAngles	0	1	48.66	156.96	0	0.00	0.00	2.00	2299	▇▁▁▁▁
hAngleRatio	0	1	0.07	0.85	0	0.00	0.01	0.01	19	▇▁▁▁▁
bAngleRatio	0	1	5.37	55.61	0	0.00	0.00	0.00	2224	▇▁▁▁▁
bodyCAPS	0	1	48.66	825.85	0	0.03	0.04	0.08	50543	▇▁▁▁▁
bodyLowers	0	1	332.41	2718.28	0	0.55	0.64	0.71	117614	▇▁▁▁▁
exclRatio	0	1	0.48	4.26	0	0.00	0.00	0.00	80	▇▁▁▁▁

Group features by type prior to modeling

```
d.f$is_spam <- as.logical(d.f$is_spam)
cl <- lapply(d.f, class)
numericals <- names(d.f)[cl == 'numeric' | cl == 'integer']
logicals <- names(d.f)[cl == 'logical']
d.f[logicals] <- lapply(d.f[logicals], factor)
characters <- names(d.f)[cl == 'character']
# check status of target column, which has to be factor
is.factor(d.f$is_spam)
```

```
## [1] TRUE
```

For numerical features, we recode missing values (NA), which some models refuse to work with.

```
colMeans(d.f[numericals])
```

```
##    headercolors      bodycolors       header3Ds         body3Ds     headerfonts
##    8.269931e-04    4.116606e+00    2.277539e-01    1.344062e+01    9.923917e-04
##       bodyfonts     headersizes       bodysizes    headeraligns      bodyaligns
##    9.665729e+00    4.961958e-03    3.829805e+00    1.653986e-04    2.373139e+00
##        headerWs        headerXs        headerYs        headerZs          bodyWs
##    2.861396e-01    1.075091e-02    1.002977e+00    1.594773e+00    3.819550e+00
##          bodyXs          bodyYs          bodyZs      headerExcl     subjectExcl
##    1.339729e-02    1.422428e-02    6.185908e-02    2.543831e-01              NA
##     headerChars       bodyChars         domains     subjectCAPS headerBodyRatio
##    1.865510e+03    2.935759e+03    1.447337e+01              NA    1.489867e+02
##    headerAngles      bodyAngles     hAngleRatio     bAngleRatio        bodyCAPS
##    1.108650e+01    4.865729e+01    6.972050e-02    5.369588e+00    4.866319e+01
##      bodyLowers       exclRatio
##    3.324073e+02    4.813868e-01
```

```
# find the NA subjects and replace them with 0
sum(d.f$noSub == T)
```

```
## [1] 19
```

```
sum(is.na(d.f$subjectCAPS))
```

```
## [1] 19
```

```
sum(is.na(d.f$subjectExcl))
```

```
## [1] 19
```

```
d.f$subjectCAPS[d.f$noSub == T] <- 0
d.f$subjectExcl[d.f$noSub == T] <- 0
colMeans(d.f[numericals])
```

```
##    headercolors      bodycolors       header3Ds         body3Ds     headerfonts
##    8.269931e-04    4.116606e+00    2.277539e-01    1.344062e+01    9.923917e-04
##       bodyfonts     headersizes       bodysizes    headeraligns      bodyaligns
##    9.665729e+00    4.961958e-03    3.829805e+00    1.653986e-04    2.373139e+00
##        headerWs        headerXs        headerYs        headerZs          bodyWs
##    2.861396e-01    1.075091e-02    1.002977e+00    1.594773e+00    3.819550e+00
##          bodyXs          bodyYs          bodyZs      headerExcl     subjectExcl
##    1.339729e-02    1.422428e-02    6.185908e-02    2.543831e-01    1.538207e-01
##     headerChars       bodyChars         domains     subjectCAPS headerBodyRatio
##    1.865510e+03    2.935759e+03    1.447337e+01    5.326662e+00    1.489867e+02
##    headerAngles      bodyAngles     hAngleRatio     bAngleRatio        bodyCAPS
##    1.108650e+01    4.865729e+01    6.972050e-02    5.369588e+00    4.866319e+01
##      bodyLowers       exclRatio
##    3.324073e+02    4.813868e-01
```

```
# make sure all factors have at least 2 levels
lapply(d.f[logicals], unique)
```

```
## $is_spam
## [1] TRUE  FALSE
## Levels: FALSE TRUE
## 
## $bad_header
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $bad_body
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $noSub
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $re_subj
## [1] FALSE TRUE  <NA>
## Levels: FALSE TRUE
## 
## $yahoos
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $hasLocals
## [1] TRUE  FALSE
## Levels: FALSE TRUE
## 
## $subjectSex
## [1] FALSE TRUE  <NA>
## Levels: FALSE TRUE
```

For subject features we recode so that missing values (NA) = FALSE.

```
d.f$re_subj[is.na(d.f$re_subj)] <- FALSE
d.f$subjectSex[is.na(d.f$subjectSex)] <- FALSE
# and change the subjects themselves too
d.f$subjects[is.na(d.f$subjects)] <- "no subject"
lapply(d.f[logicals], unique)
```

```
## $is_spam
## [1] TRUE  FALSE
## Levels: FALSE TRUE
## 
## $bad_header
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $bad_body
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $noSub
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $re_subj
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $yahoos
## [1] FALSE TRUE
## Levels: FALSE TRUE
## 
## $hasLocals
## [1] TRUE  FALSE
## Levels: FALSE TRUE
## 
## $subjectSex
## [1] FALSE TRUE
## Levels: FALSE TRUE
```

3. Build, train, and analyze results of logistic regression classifier

Split dataset into training and testing groups

```r
# set the random seed, for reproducibility, and split the data 80/20.
set.seed(607)

traintest <- initial_split(d.f, prop = .80)
train_data <- training(traintest)
test_data  <- testing(traintest)
```

Make a recipe to scale numerical features for logistic regression

```r
rec <- recipe(is_spam ~ ., data = train_data) %>%

  # Keep string features out of the modeling, but keep them around.
  update_role(all_of(characters), new_role = "ID") %>%
  step_normalize(all_of(numericals)) %>%

  # One-hot encode all logicals except for target
  step_dummy(all_of(logicals), -is_spam)

# prep() fits the scaler to the training data
scaler <- prep(rec, training = train_data)
# and bake() transforms all data using the statistics learned by prep()
scaled_train <- bake(scaler, train_data)
scaled_test <- bake(scaler, test_data)
```

Now the scaled data can be used to train models. Or at least that shows conceptually how the scaler will be fit to and transform the data. But instead of exiting the pipeline so soon, recipes can fit inside of a larger pipeline called a `workflow()` , which manages all the scaling steps as part of a model parameter fitting and prediction process. It just needs the data, the recipe, and a model.

One natural model choice to start with might be a logistic regression classifier, such as `logistic_reg()` from the `parsnip` package.

```r
lr_mod <- logistic_reg() %>%
  set_engine('glm') # barebones log_reg

spam_workflow <- workflow() %>%
  add_model(lr_mod) %>%
  add_recipe(rec)

spam_workflow
```

```
## ══ Workflow ════════════════════════════════════════════════════════════════════
## Preprocessor: Recipe
## Model: logistic_reg()
##
## ── Preprocessor ────────────────────────────────────────────────────────────────
## 2 Recipe Steps
##
## ● step_normalize()
## ● step_dummy()
##
## ── Model ───────────────────────────────────────────────────────────────────────
## Logistic Regression Model Specification (classification)
##
## Computational engine: glm
```

Fit the model to the training set.

```r
spam_fit <- spam_workflow %>%
  fit(data = train_data)
```

See which features had most influence making email look like HAM

```r
spam_fit %>%
  pull_workflow_fit() %>%
  tidy() %>%
  # estimate == coefficient
  arrange(estimate) %>%
  select(c(term, estimate, p.value))
```

```
## # A tibble: 40 x 3
##    term             estimate  p.value
##    <chr>               <dbl>    <dbl>
##  1 yahoos_TRUE.      -18.1   9.55e- 1
##  2 re_subj_TRUE.      -3.23  1.26e-72
##  3 bodyWs             -2.59  2.87e-31
##  4 headerAngles       -1.38  6.31e-19
##  5 hasLocals_TRUE.    -1.31  3.61e-21
##  6 bad_header_TRUE.   -0.681 1.59e- 1
##  7 bodyLowers         -0.631 2.40e- 3
##  8 bodyChars          -0.350 3.18e- 4
##  9 bodycolors         -0.341 1.06e- 2
## 10 bodysizes          -0.321 2.41e- 1
## # … with 30 more rows
```

Results: The YahooGroups coefficient has too high of a p-value to be worth anything, but after that, the biggest identifiers of legit email (Are you listening, all you spammers out there?) are emails that have subjects showing "RE:" in them, email bodies with websites in them ('bodyWs'), emails with a high ratio of lower case letters in the email, with a lot of angle brackets in the header, and with a localhost IP address (127.0.0.1) in the header.

Now we assess the highest SPAM indicators:

```
spam_fit %>%
  pull_workflow_fit() %>%
  tidy() %>%
  arrange(-estimate) %>%
  select(c(term, estimate, p.value))
```

```
## # A tibble: 40 x 3
##    term             estimate  p.value
##    <chr>               <dbl>    <dbl>
##  1 subjectSex_TRUE.    2.54  6.25e- 7
##  2 bodyfonts           1.82  4.41e- 7
##  3 domains             1.59  1.36e-17
##  4 noSub_TRUE.         1.57  1.52e- 2
##  5 bodyCAPS            1.05  1.72e- 2
##  6 subjectExcl         1.02  7.81e-16
##  7 (Intercept)         1.00  4.34e- 1
##  8 bodyaligns          0.869 7.73e-14
##  9 subjectCAPS         0.623 6.69e-26
## 10 exclRatio           0.565 2.53e- 6
## # … with 30 more rows
```

Results: A lot of capital letters and exclamation points in the email body won't help it get past the spam filter, unsurprisingly, nor will mentioning words like viagra and sex organs. If there are many fonts or other html markup, if a subject field is left out, or if many domains are in the header (think email blasts), those are other good indicators of spamminess.

Now we evaluate how well the trained model performs on the test data predictions.

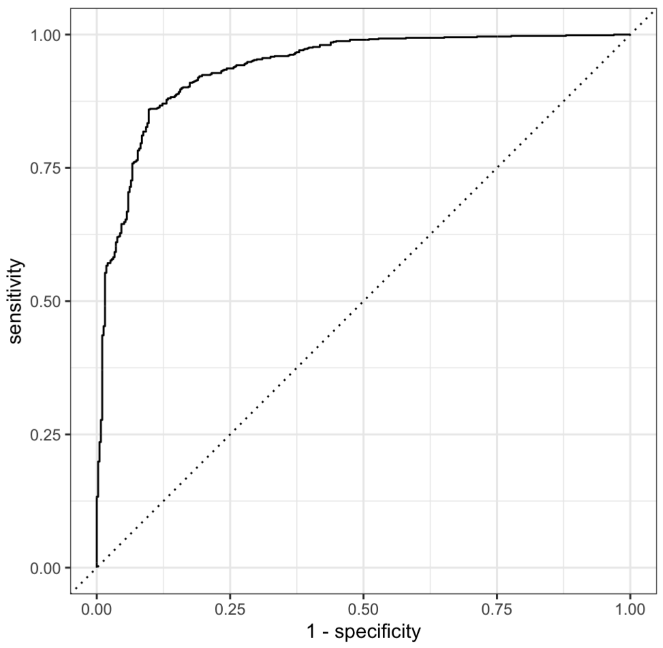
Note: We can set the logistic regression model to output its predictions in probabilities, rather than just binary 'yes'/'no' predictions, and that way we can look at the area under the ROC as a means of evaluating the model over all choices of threshold, rather than just .50.

```
spam_pred <-
  predict(spam_fit, test_data, type = "prob") %>%
  bind_cols(test_data %>% select(is_spam))
  # bound the true values for visual inspection of predictions
spam_pred
```

```
## # A tibble: 1,209 x 3
##    .pred_FALSE .pred_TRUE is_spam
##          <dbl>      <dbl> <fct>
##  1    2.49e- 3      0.998 TRUE
##  2    1.15e- 1      0.885 TRUE
##  3    3.00e- 1      0.700 TRUE
##  4    2.33e- 1      0.767 TRUE
##  5    3.19e- 1      0.681 TRUE
##  6    2.05e- 2      0.979 TRUE
##  7    3.60e- 3      0.996 TRUE
##  8    1.17e- 2      0.988 TRUE
##  9    7.14e- 2      0.929 TRUE
## 10    4.40e-13      1.00  TRUE
## # … with 1,199 more rows
```

Plot our model roc_curve

```
spam_pred %>%
  roc_curve(truth = is_spam, .pred_FALSE) %>%
  autoplot()
```



```
spam_pred %>%
  roc_auc(truth = is_spam, .pred_FALSE)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.938
```

We have a reasonable ROC-AUC for a starter model. We should also assess our model's accuracy.

```
accu <- sum((spam_pred$.pred_TRUE > 0.5) == (spam_pred$is_spam)) / nrow(spam_pred)
accu
```

```
## [1] 0.8751034
```

Note: If you change the threshold from the default 0.5 used above, the accuracy improves slightly as you lower the classification threshold, down to 0.4 and even 0.3. However, there are reasons to avoid this step:

- It doesn't make sense to tweak parameters after seeing test results, if the goal is to measure the PREdictive strength of a model.
- Lowering the threshold improves accuracy by causing the filter to send more emails, both spam and ham, to the spam folder. While in some classification applications, such as perhaps medical diagnostics, improving predictive accuracy at the expense of more false positives might be a good tradeoff, it's less likely that someone would want to see less spam in her inbox at the expense of having to constantly check her spam folder to make sure she didn't miss important ham.

87.5% accuracy isn't too bad, since about 32% of the training data is spam, meaning that a useless spam filter that let every email through would score 68% accuracy. This spam/ham imbalance brings up another idea, about training the model on a balanced spam/ham dataset, by undersampling the majority (ham) group.

The `themis` package is one of many that can perform this duty for us.

```
# Modify the existing recipe and then update the workflow with it
email_recipe <- rec %>%
    themis::step_downsample(is_spam, under_ratio = 1) # 1 is equal spam/ham
spam_workflow <- spam_workflow %>%
  update_recipe(email_recipe)
# Re-train the workflow model
spam_fit <- spam_workflow %>%
  fit(data = train_data)
```

After retraining the same model on balanced data, we check its metrics:

```
spam_pred <-
  predict(spam_fit, test_data, type = "prob") %>%
  bind_cols(test_data %>% select(is_spam))
accu <- sum((spam_pred$.pred_TRUE > 0.5) == (spam_pred$is_spam)) / nrow(spam_pred)
glue('The test accuracy after fitting the same model on a balanced training set is
{round(100 * accu, 1)}%, vs 87.5% from imbalanced training.')
```

```
## The test accuracy after fitting the same model on a balanced training set is
## 88.1%, vs 87.5% from imbalanced training.
```

Now that we've gotten a glimpse of what a barebones regression model can learn from some roughly handcrafted features, it's tempting to find an even more powerful modern model, feed it more data, in the form of email text, and see what sorts of patterns and relationships it can find and use to make predictions. So why wait?

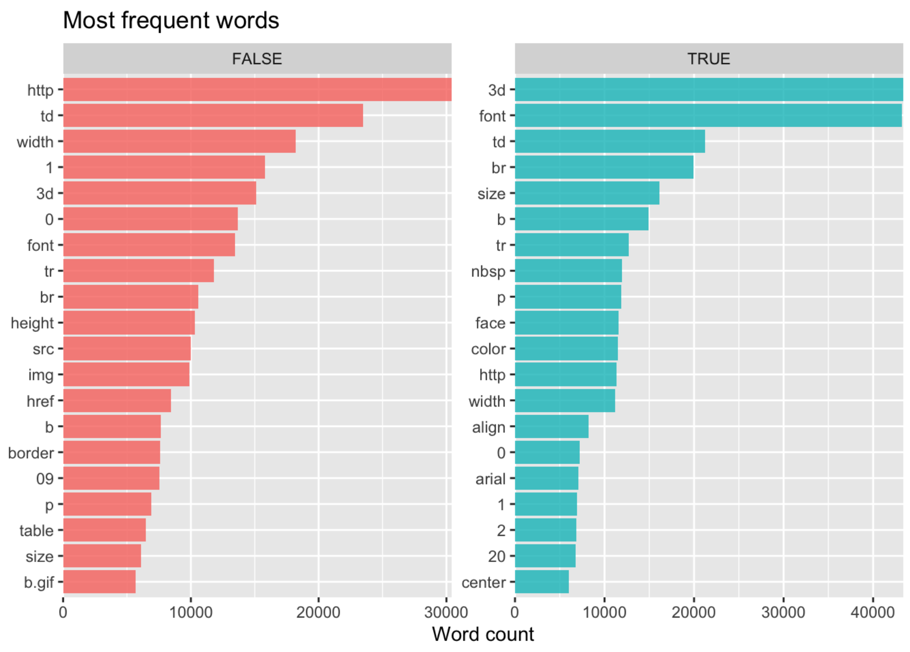
4. Process email text to facilitate modeling

First, a look at what sorts of words the language processor will be dealing with

```
# review data in tidy form
tidy<-d.f %>%
    unnest_tokens(word, bodies) %>%
    group_by(word) %>%
    filter(n()>20) %>%
        ungroup()
tidy %>%
  count(is_spam, word, sort = TRUE) %>%
  anti_join(get_stopwords()) %>%
  group_by(is_spam) %>%
  top_n(20) %>%
  ungroup() %>%
  ggplot(aes(reorder_within(word, n, is_spam), n,
    fill = is_spam
  )) +
  geom_col(alpha = 0.8, show.legend = FALSE) +
  scale_x_reordered() +
  coord_flip() +
  facet_wrap(~is_spam, scales = "free") +
  scale_y_continuous(expand = c(0, 0)) +
  labs(
    x = NULL, y = "Word count",
    title = "Most frequent words")
```

```
## Joining, by = "word"
```

```
## Selecting by n
```



Before constructing a random forest model, we'll update the recipe to calculate and select important email message tokens.

```
# reminder: "bodies" is a feature column in the d.frame, which holds the email text
email_recipe <- email_recipe %>%
    step_tokenize(bodies) %>%
    step_tokenfilter(bodies, max_tokens = 1e3) %>%
    step_tfidf(bodies)
```

5. Build, cross-validate, and analyze results of random forest classifier

Initialize a random forest model to plug into the workflow

```
rf<- rand_forest(trees = 200, mtry = 20, min_n = 3) %>%
    set_engine("ranger") %>%
    set_mode("classification")
```

Make cross-validation folds

```
set.seed(607)

email_folds <- vfold_cv(data = train_data, strata = is_spam, v = 5)
```

Update workflow to incorporate these changes

Note: This would be an appropriate place to build a sparse matrix for the data to facilitate model speed. However, we were unable to coerce a matrix into the df.

```
spam_workflow<- spam_workflow %>%
    update_recipe(email_recipe) %>%
    update_model(rf)
#Fit model (Random Forest) with cross validation. #this code comes from https://rpubs.com/shampjeff/tidy_spam_clf
#note: was unable to coerce sparse matrix into df with random forest model. Not clear why.
rf_model_fit <- fit_resamples(
    spam_workflow,
    email_folds,
    metrics = metric_set(roc_auc, accuracy),
    control = control_resamples(save_pred=TRUE))
```

```
## x Fold4: preprocessor 1/1, model 1/1: Error: Missing data in columns: headeraligns.
```

```
collect_metrics(rf_model_fit)  #roc_auc = .997, accuracy = .977
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n  std_err .config
##   <chr>    <chr>      <dbl> <int>    <dbl> <chr>
## 1 accuracy binary     0.977     4 0.00234  Preprocessor1_Model1
## 2 roc_auc  binary     0.997     4 0.000753 Preprocessor1_Model1
```

While it's possible that the model has overfit to produce these highly accurate numbers, random forests are relatively resilient to overfitting, by limiting the pool of features to choose from for each fit (we set the `mtry` argument to 20 possible features each split, when initializing the rf model), and by using many different trees (we used 200 here). But let's train the model on all 5 folds now and see how it does on the held out test data.

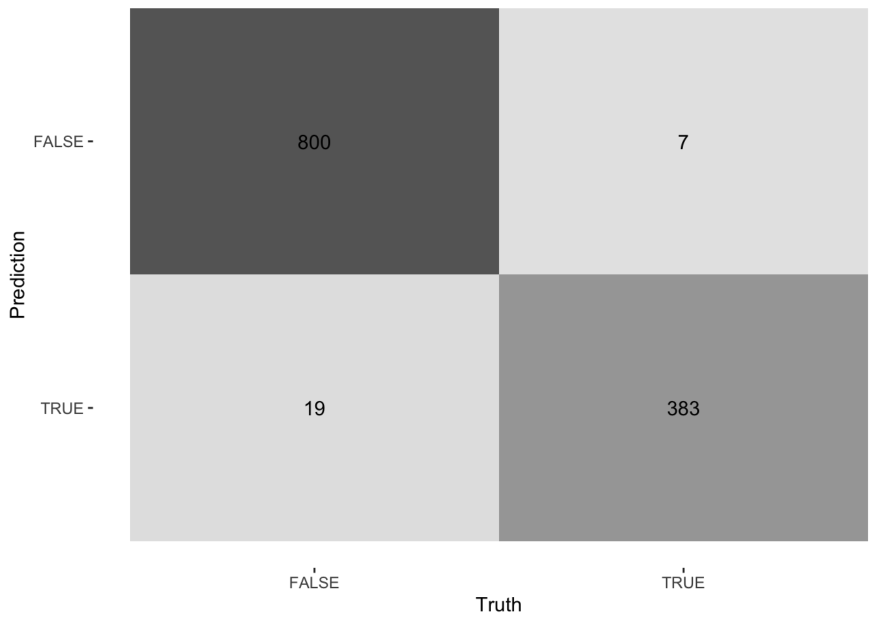
```
set.seed(607)
# Use last_fit to fit to the full (downsampled) training set and then evaluate the full test set
rf_final<-
    spam_workflow %>%
    last_fit(traintest)

cmat <- rf_final %>%
    collect_predictions() %>%
    conf_mat(truth = is_spam, estimate = .pred_class)
summary(cmat)
```

```
## # A tibble: 13 x 3
##    .metric              .estimator .estimate
##    <chr>                <chr>          <dbl>
##  1 accuracy             binary         0.978
##  2 kap                  binary         0.951
##  3 sens                 binary         0.977
##  4 spec                 binary         0.982
##  5 ppv                  binary         0.991
##  6 npv                  binary         0.953
##  7 mcc                  binary         0.951
##  8 j_index              binary         0.959
##  9 bal_accuracy         binary         0.979
## 10 detection_prevalence binary         0.667
## 11 precision            binary         0.991
## 12 recall               binary         0.977
## 13 f_meas               binary         0.984
```

The accuracy is actually a little higher now, perhaps from having 25% more data to train on. Let’s see what sorts of errors it’s still making.

```
autoplot(cmat, type = "heatmap")
```



Those results indicate a high accuracy level, but if this model were used in the real world, we might want to raise the classification threshold from 0.5 in order to reduce the 19 false positives (lower left) at the expense of the 7 false negatives (upper right).

```
preds <- rf_final$.predictions

false_negs <- preds[[1]] %>%
  filter(.pred_class == FALSE) %>%
  filter(is_spam == TRUE)

false_pos <- preds[[1]] %>%
  filter(.pred_class == TRUE) %>%
  filter(is_spam == FALSE)

false_pos
```

```
## # A tibble: 19 x 6
##    .pred_FALSE .pred_TRUE  .row .pred_class is_spam .config
##          <dbl>      <dbl> <int> <fct>       <fct>   <chr>
##  1       0.380      0.620   565 TRUE        FALSE   Preprocessor1_Model1
##  2       0.342      0.658   567 TRUE        FALSE   Preprocessor1_Model1
##  3       0.434      0.566   666 TRUE        FALSE   Preprocessor1_Model1
##  4       0.431      0.569  3514 TRUE        FALSE   Preprocessor1_Model1
##  5       0.469      0.531  3571 TRUE        FALSE   Preprocessor1_Model1
##  6       0.476      0.524  4318 TRUE        FALSE   Preprocessor1_Model1
##  7       0.351      0.649  4390 TRUE        FALSE   Preprocessor1_Model1
##  8       0.423      0.578  4401 TRUE        FALSE   Preprocessor1_Model1
##  9       0.422      0.578  4405 TRUE        FALSE   Preprocessor1_Model1
## 10       0.465      0.535  4526 TRUE        FALSE   Preprocessor1_Model1
## 11       0.27       0.73   4551 TRUE        FALSE   Preprocessor1_Model1
## 12       0.360      0.64   4552 TRUE        FALSE   Preprocessor1_Model1
## 13       0.408      0.592  4556 TRUE        FALSE   Preprocessor1_Model1
## 14       0.344      0.656  4557 TRUE        FALSE   Preprocessor1_Model1
## 15       0.372      0.628  4566 TRUE        FALSE   Preprocessor1_Model1
## 16       0.349      0.651  4589 TRUE        FALSE   Preprocessor1_Model1
## 17       0.138      0.862  4624 TRUE        FALSE   Preprocessor1_Model1
## 18       0.499      0.501  4626 TRUE        FALSE   Preprocessor1_Model1
## 19       0.391      0.609  4627 TRUE        FALSE   Preprocessor1_Model1
```

```
hard_hams <- c()
for (row in 1:length(d.f$filepaths)) {
  if (str_starts(d.f$filepaths[row], 'hard')) {
    hard_hams <- c(hard_hams, row)
  }
}
glue('Hard hams start at row {min(hard_hams)} and go to row {max(hard_hams)}.
There are {length(hard_hams)} of them.')
```

```
## Hard hams start at row 4401 and go to row 4650.
## There are 250 of them.
```

So 12 of the 19 false positives were from the “hard ham” folder. If you look at one of these false positives (row 4566, for example, since it had a 62.8% prediction of being spam, which was approximately median for this tough group), you can see the difficulty:

```
d.f$bodies[4566]
```

```
## [1] "\nYou are receiving this email because you signed up to \nreceive one of our free reports. If you would prefer \nnot
to receive messages of this type, please \nunsubscribe by following the instructions at the \nbottom of this message.\n\nDea
r Fool,\n\nJust before Enron plunged...\n\n...16 of 17 Wall Street analysts were still urging you \nto buy Enron's stock.  W
hile the company piled up debt \nand manufactured imaginary earnings, auditors gave the \nbalance sheet a big thumbs-up.  As
investors lost \neverything, Enron insiders pocketed millions.\n\nAND THAT should make every investor's blood boil!\n\nFrank
ly, the Enrons and Global Crossings of the world \ninfuriate me.  All the backroom manipulations dirty \ndealing and cover-u
ps really make it hard to know who \nyou can trust.   And that is why we've created a NEW \ninvesting service, \"David & Tom
Gardner's Motley Fool \nStock Advisor\" -- to bring the Motley Fool co-founders' \nextraordinary commitment to honest stock
analysis, \nfair dealing and complete disclosure to investors like \nyou on a regular and timely basis.\n\nThe Motley Fool,
which David and Tom founded back in \n1993, is dedicated to the principle that given the \nright tools...the average guy can
find great success in \nevery aspect of his financial life.\n\nAnd now, David and Tom are applying this principle -- and \nt
heir extensive investing experience -- to the new \n\"Motley Fool Stock Advisor.\"  \n\nBut as you'll see, this service is
n't just about \n\"buying more stocks.\"  OUR GOAL in publishing the \n\"Motley Fool Stock Advisor\" is twofold:  To help yo
u \ngain the knowledge, confidence and resolve to make \nmoney in the U.S. stock market...while avoiding the \nbiggest mista
kes, as well.\n\nThe Wall Street analysts touting Global Crossing said \nits worldwide fiber-optics network would make it th
e \nking of telecom.  Now these shares are WORTH NOTHING.  \nMicrostrategy was the poster child of the technology \nbubble-a
fter earnings restatements, the stock now sits \n99% OFF its high!  Kmart's lousy inventory management \nand marketing incom
petence helped drive the company \ninto BANKRUPTCY.\n\nWHAT ABOUT Nortel and Lucent-two of America's most \nwidely owned sto
cks.  Will they ever come back?  How \nabout AOL?  Does its merger with Time-Warner mean the \nGLORY DAYS are over -- or hav
e they just begun?   Are 90% \nof all telecommunications companies truly in danger of \ndisappearing? If so, which will surv
ive? Will great \nold names like JP Morgan, Disney and Motorola recover?  \n\nWell, those are the sorts of questions David &
Tom \nGardner can help you answer at the \"Motley Fool Stock \nAdvisor.\"  \n\nAs they say, \"Show us the money.\"  When it
comes to \nmaking promises, fast-talkers can fudge just about \nanything.  But CASH FLOW doesn't lie.\n\nYOU CAN avoid a lot
of big mistakes by simply making \nsure a company has the cash coming in to pay its \nbills.  And watching cash flow is also
one of the best \nways to identify superstar companies -- far more reliable \nthan \"earnings\" that can be manipulated 8 wa
ys to \nSunday.  Once you eliminate big losers from the \nequation -- and target the true superstars -- you'll find \ninvest
ing more fun...less stressful...and much more \nprofitable, as well.\n\nSo forget about Wall Street's \"damaged goods\"-\nco
mpanies that have a boatload of debt and accounting \nwoes, as well.  Keep it SIMPLE instead.\n\n\"Simple\" means investing
in companies that we all -- with \na little effort -- can understand.  Companies where we \nknow what businesses they're rea
lly in.  Companies \nwhere the financial statements actually mean \nsomething.  And companies easily recognizable as the \n
\"best of breed.\"\n\nIn the free report you received, \"The One Stock That \nKeeps Wall Street BUZZING,\" you read about on
e of \nDavid & Tom's very favorite investments: Starbucks.  \nBefore this company came along, most folks just took \ncoffee
for granted.  And certainly, no one proposed \nany far-reaching business model built around it.  Yet \nthese innovators took
a low-margin commodity and built \na near $3 BILLION company with more than 5000 \nretail outfits all around the world.\n\nS
tarbucks has been able to open new outlets based on \nmoney it has earned from existing ones. That is an \nastounding testam
ent to its economic power.  At The \nMotley Fool Stock Advisor we'll help you find and \ntruly understand the great investme
nts, the companies \nwith solid balance sheets and real growth prospects.    \nRight now we're following companies such as:\n
\n*America's most prestigious jeweler -- its brand name \nhas no rival.  With net margins of 59% on sales of \n$1.6 Billion,
this company is a great pick for solid \nreturns with low risk.   \n\n*This software company commands more than 25% of the
\nvideo game market.  Its shares have returned 6,000% \nsince its public debut in 1989.  An investment of \n$5,000 in 1989 i
s now worth over $300,000.  The video \ngame market is expected to double by 2005 with this \ncompany enjoying the lion's sh
are of the growth.\n\n*This premier credit rating company has locked-in \ndemand for its reports on the creditworthiness of
\nborrowers holding $30 trillion of the world's debt.  \nIt has a 20-year record of consistent sales and income \ngrowth and
is poised for another stellar year.\n\nYOU CAN TRY the \"Motley Fool Stock Advisor\" for six \nfull months 100% RISK-FREE.
If we don't prove its \nworth to you, it doesn't cost you a dime.\n\nAs a Charter Subscriber, you get the \"Motley Fool \nSt
ock Advisor\" delivered to your home each month; a \nmonthly, between-issue, e-mail Fool Flash to help you \ntake full advan
tage of breaking news; full use of the \nMotley Fool Stock Advisor subscriber-only Web site, \nfeaturing current & back news
letter issues...full \nupdates on all selected stocks...Q & A...and more; \nplus introductory Special Reports exclusively fo
r new \nsubscribers.\n\nTo join RISK-FREE, as a Charter Subscriber to \"David & \nTom Gardner's Motley Fool Stock Advisor,\"
simply click \nhere now:\nhttp://www.ppi-orders.com/index.htm?promo_code=1UE733\n\nSincerely yours,\n\nMike Bell, Publisher
\n\"David & Tom Gardner's Motley Fool Stock Advisor\"\n\nP.S. Do you like BIG PROFITS?  In 1994, investment \npros sniffed a
t America Online's high-flying stock and \nscreamed, \"Overvalued.\"  But the Gardners saw three \nthings they overlooked:
1) AOL was building the \npremier stop in cyberspace; 2) They had a great \nretention strategy; and 3) Millions of members p
aying \n$$$ every month.\n\nSo Tom and David bought it.  And today they're sitting \non whopping 4,120% gains.  So you see,
while they do \nstructure their investing strategy to avoid big \nblunders...they do make a lot of money, too.  To try the
\n\"Motley Fool Stock Advisor\" RISK-FREE, click here now:\nhttp://www.ppi-orders.com/index.htm?promo_code=1UE733\n\n~~~~~~~
~~~~~~~~~~~~~~~~~~~~~~~~~\nHOW TO UNSUBSCRIBE\n\nWe hope this email message is of value to you. If \nhowever, you do not wis
h to receive any of our future \nmessages, please unsubscribe by going to the following \nweb address:\n\nhttp://www.investo
rplace.com/newunsubscribe.php?q=66527235-1\n\n**Note if you unsubscribe by replying to this message \nplease use Unsubscribe
or Remove in the subject line. \n\nMon Jul 29 17:04:26 2002\n\n\n\n\n\n"
```

How about those 7 sneaky spams that got through the defenses?

```
false_negs
```

```
## # A tibble: 7 x 6
##   .pred_FALSE .pred_TRUE  .row .pred_class is_spam .config
##         <dbl>      <dbl> <int> <fct>       <fct>   <chr>
## 1       0.534      0.466  5351 FALSE       TRUE    Preprocessor1_Model1
## 2       0.529      0.471  5398 FALSE       TRUE    Preprocessor1_Model1
## 3       0.748      0.252  5816 FALSE       TRUE    Preprocessor1_Model1
## 4       0.560      0.440  5937 FALSE       TRUE    Preprocessor1_Model1
## 5       0.504      0.496  5940 FALSE       TRUE    Preprocessor1_Model1
## 6       0.551      0.449  5967 FALSE       TRUE    Preprocessor1_Model1
## 7       0.577      0.423  5983 FALSE       TRUE    Preprocessor1_Model1
```

6 of them had a pretty even chance of being filtered out, but let’s look at that one that was only given a 25% chance of being spam:

```
d.f$bodies[5816]
```

```
## [1] "\nGAMING ADVISOR  \n\nNEWS July 16, 2002   i2corp Enters into Research and Development Agreement with Nevada Gaming 
Equipment Supplier For Live Remote Bingo.      \n \nThe general feeling of the Top Internet Lawyers and Analysts is that \"o
nline gambling and sports betting is here to stay.\x94 Attempts at trying to stop it would be largely ineffective! Banning I
nternet gambling and sports betting presents technical and legal problems that Governments are ill equipped to enforce. \n\n
What\x92s more, there are NO competitors for the Company\x92s method or technology for remote wagering. i2corp is the Only C
ompany in the World that can legally license its method and technology for remote wagering. i2corp is currently negotiating 
with\nMajor Casinos Globally.    \n \nMichael Pollock of the Pollock Gaming Resource Group (PRRG) has issued a report studyi
ng the effects of live wagering from remote locations. \x93 The study concludes that live wagering from remote locations doe
s more than create a new source of revenue for sponsoring casinos. Additionally, it creates a marketing opportunity, a means 
to find and cultivate thousands of new customers who can be encouraged to become on-site patrons.\x94 A copy of the full rep
ort in pdf format can be downloaded at http://www.gamingobserver.com/.\n \ni2corp is the holding company for Home Gambling N
etwork, Inc. HGN holds U.S. Patent 5,800,268, which covers remote wagering on live games and events with electronic financia
l transactions. The Patent is comprised of three primary actions: the gambling is live; the player is remote and physically 
away from the actual game or event; and the winnings and losses are transacted electronically in real time. They include, bu
t are not limited to, horse racing, soccer, bingo, poker, roulette and many other Las Vegas Casino Games. \n\nFACT: Currentl
y there is an overwhelming opinion that remote wagering in real-time will become one of the most profitable/prolific industr
ies worldwide. i2corp recently won judgment against giant UUNET a subsidiary of MCI/WorldCom in a U.S. Federal Court for Pat
ent infringement. Just another compelling reason why i2corp has orphaned their competition! Therefore we conclude i2corp\x92
s common stock to be overlooked and undervalued! \n\nRECOMMENDATION: STRONG BUY\n\n     i2corp   \nOTCBB Symbol        
(ITOO)\nRecent Price     \t          .05 - .08\n52-Week Hi-Lo \t          .05 - .48\nShort Term Target Price:     .60\n\nThi
s news release contains forward-looking statements as defined by the Private Securities Litigation Reform Act of 1995. Forwa
rd-looking statements include statements concerning plans, objectives, goals, strategies, future events or performance and u
nderlying assumptions, and all statements that are other than statements of historical facts. These statements are subject t
o uncertainties and risks including, but not limited to, product and service demand and acceptance, changes in technology, e
conomic conditions, the impact of competition and pricing, government regulation, and other risks defined in this document. 
These cautionary statements expressly qualify all such forward-looking statements made by or on behalf of Gaming Advisor. In 
addition, Gaming Advisor disclaims any obligation to update any forward-looking statements to reflect events or circumstance
s after the date hereof. The information herein has been obtained from reputable sources and therefore we assume its v\n ali
dity. Gaming Advisor has been c\nompensated $5,000 for the dissemination of this information and may at anytime buy or sell 
the securities of i2corp\x92s common stock.\n\nnywguempturgwthionjpsqdw\nhttp://xent.com/mailman/listinfo/fork\n\n"
```

It seems one person's ham is another's spam. As a final check on our Random Forest model results we check output from a null model

6. Compare results to a null model

```
#Evaluate null model results as a check on ourselves
null_classification <- null_model() %>%
  set_engine("parsnip") %>%
  set_mode("classification")
null_wf <- workflow() %>%
  add_recipe(email_recipe) %>%
  add_model(null_classification)
null_model_fit <- fit_resamples(
    null_wf,
    email_folds,
    metrics = metric_set(roc_auc, accuracy),
    control = control_resamples(save_pred=TRUE))
collect_metrics(null_model_fit)  #--> roc_auc .50, accuracy = .68
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n   std_err .config
##   <chr>    <chr>      <dbl> <int>     <dbl> <chr>
## 1 accuracy binary     0.689     5 0.0000779 Preprocessor1_Model1
## 2 roc_auc  binary     0.5       5 0         Preprocessor1_Model1
```

The null model provides us with an roc_auc and accuracy that we might expect by guessing the type of email (spam vs. ham). Comparing to our null results, it's clear that our logistic and random forest models have performed well.

Future considerations

We could try different models on the same data, to see how they compare. We tried a Naive Bayes classifier, and excluded the unimpressive results from this report, but if we'd had more time to figure out how to work with a sparse version of our features, we likely could have seen nice results from an XGBoost model. We also could've squeezed a little more accuracy out of the Logistic Regression model, by adding regularization. And one more next step, if we spent a few more days on this project, would be to search a hyperparameter grid to find the most effective combination on our cross-folds, before testing.