R Notebook

Problem 1. Build an R Notebook of the SMS message filtering example in the textbook on pages 103 to 123. Show each step and add appropriate documentation. This is the same as Lesson 4.

```
#install.packages("tm")
library(tm)
```

```
## Loading required package: NLP
```

```
#install.packages("SnowballC")
library(SnowballC)
#install.packages("wordcloud")
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
#install.packages("e1071")
library(e1071)
library(gmodels)

#Import the data and save in a dataframe
sms_raw <- read.csv("sms_spam.csv", stringsAsFactors = FALSE)
str(sms_raw)
```

```
## 'data.frame':    5574 obs. of  2 variables:
##  $ type: chr  "ham" "ham" "spam" "ham" ...
##  $ text: chr  "Go until jurong point, crazy.. Available only in bugis n great world la e buffet... C
```

Data Preparation - Cleaning and Standardizing text data

```
# Convert the type element to factor since it is categorical
sms_raw$type <- factor(sms_raw$type)

# Verify the conversion
str(sms_raw$type)
```

```
##  Factor w/ 2 levels "ham","spam": 1 1 2 1 1 2 1 1 2 2 ...
```

```
# Get the count of ham and spam
table(sms_raw$type)
```

```
##
##  ham spam
## 4827  747
```

```
# Use the VectorSource() reader function to create a source object from the existing sms_raw$text vecto
sms_corpus <- VCorpus(VectorSource(sms_raw$text))

# Print the corpus
print(sms_corpus)
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 5574
```

```
# Get summary of the first and secong SMS
inspect(sms_corpus[1:2])
```

```
## <<VCorpus>>
## Metadata:  corpus specific: 0, document level (indexed): 0
## Content:  documents: 2
##
## [[1]]
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 111
##
## [[2]]
## <<PlainTextDocument>>
## Metadata:  7
## Content:  chars: 29
```

```
# View the actual first message
as.character(sms_corpus[[1]])
```

```
## [1] "Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there
```

```
# View the first two SMS
lapply(sms_corpus[1:2], as.character)
```

```
## $`1`
## [1] "Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there
##
## $`2`
## [1] "Ok lar... Joking wif u oni..."
```

```
# Standardize the messages to use only lowercase characters
sms_corpus_clean <- tm_map(sms_corpus, content_transformer(tolower))

# Check the first element to see if the previous command worked
as.character(sms_corpus[[1]])
```

```
## [1] "Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there
```

```
as.character(sms_corpus_clean[[1]])
```

```
## [1] "go until jurong point, crazy.. available only in bugis n great world la e buffet... cine there
```

```
# Remove numbers from the SMS
sms_corpus_clean <- tm_map(sms_corpus_clean, removeNumbers)

# Remove the stop words
sms_corpus_clean <- tm_map(sms_corpus_clean, removeWords, stopwords())

# Eliminate punctuations
sms_corpus_clean <- tm_map(sms_corpus_clean, removePunctuation)

# Apply the wordStem() function through stemDocument to return the root forms of a same vector of terms
sms_corpus_clean <- tm_map(sms_corpus_clean, stemDocument)

# Remove additional whitespace
sms_corpus_clean <- tm_map(sms_corpus_clean, stripWhitespace)
```

Data Preparation - splitting text documents into words

```r
# Creata a DTM sparse matrix
sms_dtm <- DocumentTermMatrix(sms_corpus_clean)

# Create a DTM directly from the raw, unprocessed SMS corpus
sms_dtm2 <- DocumentTermMatrix(sms_corpus, control = list(
  tolower = T,
  removeNumbers = T,
  stopwords = T,
  removePunctuation = T,
  stemming = T
))

# Comparing both the DTMs
sms_dtm
```

```
## <<DocumentTermMatrix (documents: 5574, terms: 6592)>>
## Non-/sparse entries: 42608/36701200
## Sparsity           : 100%
## Maximal term length: 40
## Weighting          : term frequency (tf)
```

```r
sms_dtm2
```

```
## <<DocumentTermMatrix (documents: 5574, terms: 6995)>>
## Non-/sparse entries: 43713/38946417
## Sparsity           : 100%
## Maximal term length: 40
## Weighting          : term frequency (tf)
```

Data preparation – creating training and test datasets

```r
sms_dtm_train <- sms_dtm[1:4169, ]
sms_dtm_test <- sms_dtm[4170:5559, ]

# Create labels for each of the rows
sms_train_labels <- sms_raw[1:4169, ]$type
sms_test_labels <- sms_raw[4170:5559, ]$type

# Compare the proportion of spam in training and testing data frames
prop.table(table(sms_train_labels))
```

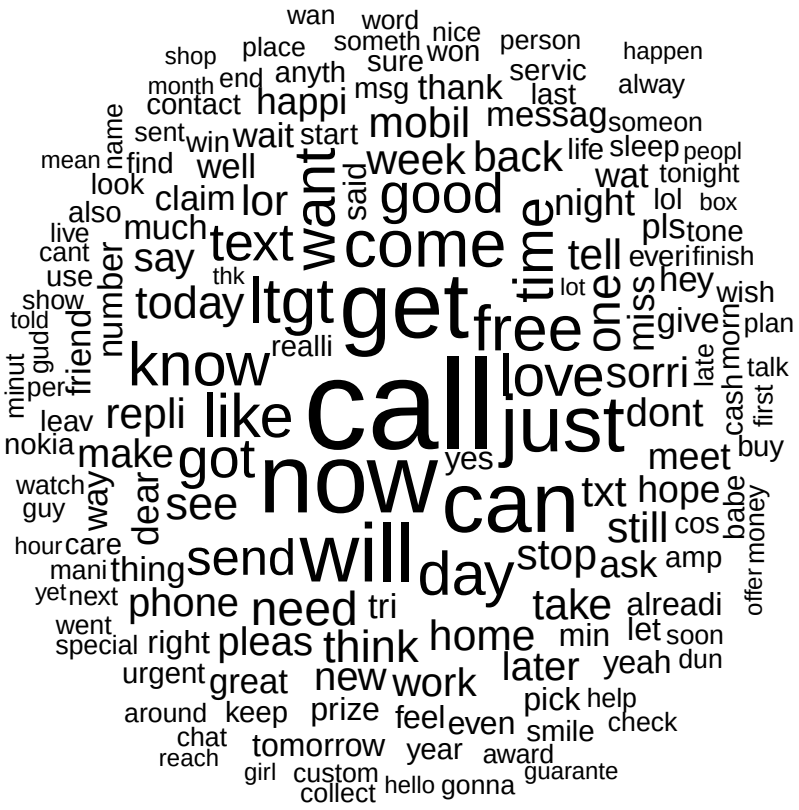
```
## sms_train_labels
##       ham      spam
## 0.8647158 0.1352842
```

```r
prop.table(table(sms_test_labels))
```

```
## sms_test_labels
##       ham      spam
## 0.8697842 0.1302158
```

Visualizing text data – word clouds

```r
# Create a word cloud directly from tm corpus object
wordcloud(sms_corpus_clean, min.freq = 50, random.order = FALSE)
```




```
# Subset the sms_raw data by the SMS type
spam <- subset(sms_raw, type == "spam")
ham <- subset(sms_raw, type == "ham")

# Create word clouds of the subsets
wordcloud(spam$text, max.words = 40, scale = c(3, 0.5))
```

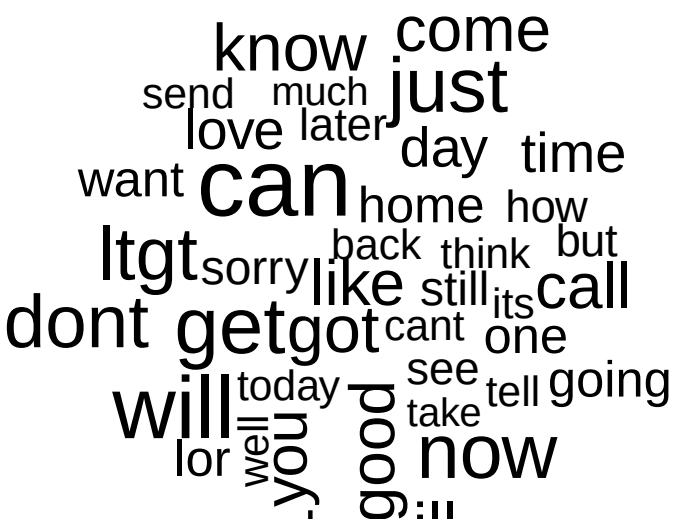
```
## Warning in tm_map.SimpleCorpus(corpus, tm::removePunctuation):
## transformation drops documents
```

```
## Warning in tm_map.SimpleCorpus(corpus, function(x) tm::removeWords(x,
## tm::stopwords())): transformation drops documents
```


```
wordcloud(ham$text, max.words = 40, scale = c(3, 0.5))
```

```
## Warning in tm_map.SimpleCorpus(corpus, tm::removePunctuation):
## transformation drops documents
```

```
## Warning in tm_map.SimpleCorpus(corpus, tm::removePunctuation):
## transformation drops documents
```



Data preparation – creating indicator features for frequent words

```
# Find words appearing at least 5 times in the sms)dtm_train matrix
findFreqTerms(sms_dtm_train, 5)
```

```
##    [1] "abiola"        "abl"           "abt"           "accept"
##    [5] "access"        "account"       "across"        "activ"
##    [9] "actual"        "add"           "address"       "admir"
##   [13] "adult"         "advanc"        "aft"           "afternoon"
##   [17] "aftr"          "age"           "ago"           "ahead"
##   [21] "aight"         "aint"          "air"           "aiyah"
##   [25] "alex"          "almost"        "alon"          "alreadi"
##   [29] "alright"       "alrit"         "also"          "alway"
##   [33] "amp"           "angri"         "announc"       "anoth"
##   [37] "answer"        "anybodi"       "anymor"        "anyon"
##   [41] "anyth"         "anytim"        "anyway"        "apart"
##   [45] "app"           "appli"         "appoint"       "appreci"
##   [49] "april"         "ard"           "area"          "argument"
##   [53] "arm"           "around"        "arrang"        "arrest"
##   [57] "arriv"         "asap"          "ask"           "askd"
##   [61] "asleep"        "ass"           "attempt"       "auction"
##   [65] "avail"         "ave"           "avoid"         "await"
##   [69] "award"         "away"          "awesom"        "babe"
##   [73] "babi"          "back"          "bad"           "bag"
##   [77] "bak"           "balanc"        "bank"          "bare"
##   [81] "bath"          "batteri"       "bcoz"          "bcum"
##   [85] "bday"          "beauti"        "becom"         "bed"
##   [89] "bedroom"       "begin"         "believ"        "belli"
##   [93] "best"          "better"        "bid"           "big"
##   [97] "bill"          "bird"          "birthday"      "bit"
##  [101] "black"         "blank"         "bless"         "blue"
##  [105] "bluetooth"     "bodi"          "bold"          "bonus"
##  [109] "boo"           "book"          "bore"          "boss"
##  [113] "bother"        "bout"          "bowl"          "box"
##  [117] "boy"           "boytoy"        "brand"         "break"
```

```
##  [121] "breath"        "brilliant"     "bring"         "brother"
##  [125] "bslvyl"        "btnationalr"   "budget"        "bugi"
##  [129] "bus"           "busi"          "buy"           "buzz"
##  [133] "cabin"         "cafe"          "cal"           "call"
##  [137] "caller"        "callertun"     "camcord"       "came"
##  [141] "camera"        "can"           "cancel"        "cant"
##  [145] "car"           "card"          "care"          "carlo"
##  [149] "case"          "cash"          "cashbal"       "catch"
##  [153] "caus"          "chanc"         "chang"         "charact"
##  [157] "charg"         "chariti"       "chat"          "cheap"
##  [161] "check"         "cheer"         "chennai"       "chikku"
##  [165] "childish"      "children"      "chines"        "choic"
##  [169] "choos"         "christma"      "cine"          "cinema"
##  [173] "claim"         "class"         "clean"         "clear"
##  [177] "click"         "clock"         "close"         "club"
##  [181] "code"          "coffe"         "coin"          "cold"
##  [185] "colleagu"      "collect"       "colleg"        "colour"
##  [189] "come"          "comin"         "comp"          "compani"
##  [193] "competit"      "complet"       "complimentari" "comput"
##  [197] "concentr"      "condit"        "confid"        "confirm"
##  [201] "congrat"       "congratul"     "connect"       "contact"
##  [205] "content"       "convey"        "cook"          "cool"
##  [209] "copi"          "correct"       "cos"           "cost"
##  [213] "countri"       "coupl"         "cours"         "cover"
##  [217] "coz"           "crave"         "crazi"         "credit"
##  [221] "cri"           "croydon"       "cuddl"         "cum"
##  [225] "cup"           "current"       "custcar"       "custom"
##  [229] "cut"           "cute"          "cuz"           "dad"
##  [233] "daddi"         "damn"          "darl"          "darlin"
##  [237] "darren"        "dat"           "date"          "day"
##  [241] "dead"          "deal"          "dear"          "decid"
##  [245] "deep"          "definit"       "del"           "delet"
##  [249] "deliv"         "deliveri"      "den"           "depend"
##  [253] "detail"        "dey"           "didnt"         "die"
##  [257] "differ"        "difficult"     "digit"         "din"
##  [261] "dinner"        "direct"        "dis"           "discount"
##  [265] "discuss"       "disturb"       "dnt"           "doctor"
##  [269] "doesnt"        "dog"           "doin"          "dollar"
##  [273] "don"           "done"          "dont"          "don‘t"
##  [277] "door"          "doubl"         "download"      "draw"
##  [281] "dream"         "drink"         "drive"         "drop"
##  [285] "drug"          "dude"          "dun"           "dunno"
##  [289] "dvd"           "earli"         "earlier"       "easi"
##  [293] "eat"           "eatin"         "either"        "els"
##  [297] "email"         "embarass"      "empti"         "end"
##  [301] "enemi"         "energi"        "england"       "enjoy"
##  [305] "enough"        "enter"         "entri"         "envelop"
##  [309] "especi"        "etc"           "euro"          "eve"
##  [313] "even"          "ever"          "everi"         "everyon"
##  [317] "everyth"       "exact"         "exam"          "excel"
##  [321] "excit"         "excus"         "expect"        "experi"
##  [325] "expir"         "extra"         "eye"           "face"
##  [329] "facebook"      "fact"          "fall"          "famili"
##  [333] "fanci"         "fantasi"       "fantast"       "far"
```

```
##  [337] "fast"          "fat"          "father"       "fault"
##  [341] "feel"          "felt"         "fetch"        "fight"
##  [345] "figur"         "file"         "fill"         "film"
##  [349] "final"         "find"         "fine"         "finger"
##  [353] "finish"        "first"        "five"         "fix"
##  [357] "flight"        "flirt"        "flower"       "follow"
##  [361] "fone"          "food"         "forev"        "forget"
##  [365] "forgot"        "forward"      "found"        "free"
##  [369] "freemsg"       "freephon"     "fren"         "fri"
##  [373] "friday"        "friend"       "friendship"   "frm"
##  [377] "frnd"          "frnds"        "fuck"         "full"
##  [381] "fullonsmscom"  "fun"          "funni"        "futur"
##  [385] "gal"           "game"         "gap"          "gas"
##  [389] "gave"          "gay"          "gentl"        "get"
##  [393] "gettin"        "gift"         "girl"         "give"
##  [397] "glad"          "god"          "goe"          "goin"
##  [401] "gone"          "gonna"        "good"         "goodmorn"
##  [405] "goodnight"     "got"          "goto"         "gotta"
##  [409] "great"         "green"        "greet"        "grin"
##  [413] "group"         "guarante"     "gud"          "guess"
##  [417] "guy"           "gym"          "haf"          "haha"
##  [421] "hai"           "hair"         "half"         "hand"
##  [425] "hang"          "happen"       "happi"        "hard"
##  [429] "hav"           "havent"       "head"         "hear"
##  [433] "heard"         "heart"        "heavi"        "hee"
##  [437] "hell"          "hello"        "help"         "hey"
##  [441] "hgsuiteland"   "high"         "hit"          "hiya"
##  [445] "hmm"           "hmmm"         "hmv"          "hol"
##  [449] "hold"          "holder"       "holiday"      "home"
##  [453] "honey"         "hook"         "hop"          "hope"
##  [457] "horni"         "hospit"       "hot"          "hotel"
##  [461] "hour"          "hous"         "housemaid"    "how"
##  [465] "howev"         "howz"         "hrs"          "hug"
##  [469] "huh"           "hungri"       "hurri"        "hurt"
##  [473] "iam"           "ice"          "idea"         "identifi"
##  [477] "ignor"         "ill"          "imagin"       "imma"
##  [481] "immedi"        "import"       "inc"          "inch"
##  [485] "includ"        "india"        "indian"       "info"
##  [489] "inform"        "instead"      "interest"     "interview"
##  [493] "invit"         "ipod"         "irrit"        "ish"
##  [497] "issu"          "ive"          "izzit"        "januari"
##  [501] "jay"           "job"          "john"         "join"
##  [505] "joke"          "joy"          "jus"          "just"
##  [509] "juz"           "kalli"        "kate"         "keep"
##  [513] "kept"          "key"          "kick"         "kid"
##  [517] "kill"          "kind"         "kinda"        "king"
##  [521] "kiss"          "knew"         "know"         "knw"
##  [525] "ladi"          "land"         "landlin"      "laptop"
##  [529] "lar"           "last"         "late"         "later"
##  [533] "latest"        "laugh"        "lazi"         "ldn"
##  [537] "lead"          "learn"        "least"        "leav"
##  [541] "lect"          "left"         "leh"          "lei"
##  [545] "lemm"          "less"         "lesson"       "let"
##  [549] "letter"        "liao"         "librari"      "lick"
```

```
##  [553] "lie"          "life"         "lift"         "light"
##  [557] "like"         "line"         "link"         "list"
##  [561] "listen"       "littl"        "live"         "load"
##  [565] "loan"         "local"        "locat"        "log"
##  [569] "login"        "lol"          "long"         "longer"
##  [573] "look"         "lor"          "lose"         "lost"
##  [577] "lot"          "lovabl"       "love"         "lover"
##  [581] "loverboy"     "loyalti"      "ltd"          "ltdecimalgt"
##  [585] "ltgt"         "lttimegt"     "luck"         "lucki"
##  [589] "lunch"        "luv"          "made"         "mah"
##  [593] "mail"         "make"         "man"          "mani"
##  [597] "march"        "mark"         "marri"        "marriag"
##  [601] "match"        "mate"         "matter"       "maxim"
##  [605] "may"          "mayb"         "mean"         "meant"
##  [609] "med"          "medic"        "meet"         "meh"
##  [613] "mell"         "member"       "men"          "menu"
##  [617] "merri"        "messag"       "met"          "mid"
##  [621] "midnight"     "might"        "min"          "mind"
##  [625] "mine"         "minut"        "miracl"       "miss"
##  [629] "mistak"       "moan"         "mob"          "mobil"
##  [633] "mobileupd"    "mode"         "mom"          "moment"
##  [637] "mon"          "monday"       "money"        "month"
##  [641] "mood"         "moon"         "morn"         "motorola"
##  [645] "move"         "movi"         "mrng"         "mrt"
##  [649] "msg"          "msgs"         "mths"         "much"
##  [653] "mum"          "murder"       "music"        "must"
##  [657] "muz"          "nah"          "nake"         "name"
##  [661] "nation"       "natur"        "naughti"      "near"
##  [665] "need"         "net"          "network"      "neva"
##  [669] "never"        "new"          "news"         "next"
##  [673] "nice"         "nigeria"      "night"        "nite"
##  [677] "nobodi"       "noe"          "nokia"        "none"
##  [681] "noon"         "nope"         "normal"       "noth"
##  [685] "notic"        "now"          "ntt"          "num"
##  [689] "number"       "nxt"          "nyt"          "offer"
##  [693] "offic"        "offici"       "okay"         "oki"
##  [697] "old"          "omw"          "one"          "onlin"
##  [701] "oop"          "open"         "oper"         "opinion"
##  [705] "opt"          "optout"       "orang"        "orchard"
##  [709] "order"        "oredi"        "oso"          "other"
##  [713] "otherwis"     "outsid"       "pack"         "page"
##  [717] "paid"         "pain"         "paper"        "parent"
##  [721] "park"         "part"         "parti"        "partner"
##  [725] "pass"         "passion"      "password"     "past"
##  [729] "pay"          "peac"         "peopl"        "per"
##  [733] "person"       "pete"         "phone"        "photo"
##  [737] "pic"          "pick"         "pictur"       "piec"
##  [741] "pix"          "pizza"        "place"        "plan"
##  [745] "plane"        "play"         "player"       "pleas"
##  [749] "pleasur"      "pls"          "plus"         "plz"
##  [753] "pmin"         "pmsg"         "pobox"        "poboxwwq"
##  [757] "point"        "poli"         "polic"        "poor"
##  [761] "pop"          "possibl"      "post"         "pound"
##  [765] "power"        "pple"         "ppm"          "practic"
```

```
##  [769] "pray"          "prefer"        "prepar"        "press"
##  [773] "pretti"        "price"         "princess"      "privat"
##  [777] "prize"         "prob"          "probabl"       "problem"
##  [781] "process"       "project"       "promis"        "pub"
##  [785] "put"           "qualiti"       "question"      "quick"
##  [789] "quit"          "quiz"          "quot"          "rain"
##  [793] "rate"          "rather"        "rcvd"          "reach"
##  [797] "read"          "readi"         "real"          "realiz"
##  [801] "realli"        "reason"        "receipt"       "receiv"
##  [805] "recent"        "record"        "refer"         "regard"
##  [809] "regist"        "remain"        "rememb"        "remind"
##  [813] "remov"         "rent"          "rental"        "repli"
##  [817] "repres"        "request"       "respond"       "respons"
##  [821] "rest"          "result"        "return"        "reveal"
##  [825] "review"        "right"         "ring"          "rington"
##  [829] "rite"          "road"          "rock"          "room"
##  [833] "roommat"       "rose"          "round"         "rowwjhl"
##  [837] "rpli"          "rreveal"       "run"           "sad"
##  [841] "sae"           "safe"          "said"          "sale"
##  [845] "sam"           "sat"           "saturday"      "savamob"
##  [849] "save"          "saw"           "say"           "sch"
##  [853] "school"        "score"         "scream"        "sea"
##  [857] "search"        "season"        "sec"           "second"
##  [861] "secret"        "see"           "seem"          "seen"
##  [865] "select"        "self"          "sell"          "semest"
##  [869] "send"          "sens"          "sent"          "serious"
##  [873] "servic"        "set"           "settl"         "sex"
##  [877] "sexi"          "shall"         "share"         "shd"
##  [881] "ship"          "shirt"         "shit"          "shop"
##  [885] "short"         "show"          "shower"        "shuhui"
##  [889] "sick"          "side"          "sigh"          "sight"
##  [893] "sign"          "silent"        "simpl"         "sinc"
##  [897] "sing"          "singl"         "sir"           "sis"
##  [901] "sister"        "sit"           "situat"        "sky"
##  [905] "slave"         "sleep"         "slept"         "slow"
##  [909] "slowli"        "small"         "smile"         "smoke"
##  [913] "sms"           "smth"          "snow"          "sofa"
##  [917] "solv"          "somebodi"      "someon"        "someth"
##  [921] "sometim"       "somewher"      "song"          "soni"
##  [925] "sonyericsson"  "soon"          "sorri"         "sort"
##  [929] "sound"         "space"         "speak"         "special"
##  [933] "specialcal"    "spend"         "spent"         "spoke"
##  [937] "sport"         "spree"         "stand"         "star"
##  [941] "start"         "statement"     "station"       "stay"
##  [945] "std"           "still"         "stock"         "stop"
##  [949] "store"         "stori"         "str"           "straight"
##  [953] "street"        "strong"        "student"       "studi"
##  [957] "stuff"         "stupid"        "style"         "sub"
##  [961] "subscrib"      "success"       "summer"        "sun"
##  [965] "sunday"        "sunshin"       "support"       "suppos"
##  [969] "sure"          "surpris"       "sweet"         "swing"
##  [973] "system"        "take"          "talk"          "tampa"
##  [977] "tcs"           "teach"         "team"          "tear"
##  [981] "teas"          "tel"           "tell"          "ten"
```

```
##  [985] "tenerif"       "term"          "test"          "text"
##  [989] "thank"         "thanx"         "that"          "thing"
##  [993] "think"         "thinkin"       "thk"           "thnk"
##  [997] "tho"           "though"        "thought"       "throw"
## [1001] "thru"          "tht"           "thur"          "ticket"
## [1005] "til"           "till"          "time"          "tire"
## [1009] "titl"          "tmr"           "tncs"          "today"
## [1013] "togeth"        "told"          "tomo"          "tomorrow"
## [1017] "tone"          "tonight"       "tonit"         "took"
## [1021] "top"           "tot"           "total"         "touch"
## [1025] "tough"         "tour"          "toward"        "town"
## [1029] "track"         "train"         "transact"      "treat"
## [1033] "tri"           "trip"          "troubl"        "true"
## [1037] "trust"         "truth"         "tscs"          "ttyl"
## [1041] "tuesday"       "turn"          "twice"         "two"
## [1045] "txt"           "txting"        "txts"          "type"
## [1049] "ufind"         "ugh"           "umma"          "uncl"
## [1053] "understand"    "unless"        "unlimit"       "unredeem"
## [1057] "unsub"         "unsubscrib"    "updat"         "ure"
## [1061] "urgent"        "urself"        "use"           "usf"
## [1065] "usual"         "uve"           "valentin"      "valid"
## [1069] "valu"          "vari"          "verifi"        "via"
## [1073] "video"         "visit"         "voic"          "voucher"
## [1077] "wait"          "wake"          "walk"          "wan"
## [1081] "wana"          "wanna"         "want"          "wap"
## [1085] "warm"          "wast"          "wat"           "watch"
## [1089] "water"         "way"           "weak"          "wear"
## [1093] "weather"       "wed"           "wednesday"     "weed"
## [1097] "week"          "weekend"       "weight"        "welcom"
## [1101] "well"          "wen"           "went"          "wer"
## [1105] "wet"           "what"          "whatev"        "whenev"
## [1109] "whole"         "wid"           "wif"           "wife"
## [1113] "wil"           "will"          "win"           "wine"
## [1117] "winner"        "wish"          "wit"           "within"
## [1121] "without"       "wiv"           "£wk"           "wkli"
## [1125] "wnt"           "woke"          "won"           "wonder"
## [1129] "wont"          "word"          "work"          "workin"
## [1133] "world"         "worri"         "worth"         "wot"
## [1137] "wow"           "write"         "wrong"         "wun"
## [1141] "wwwgetzedcouk" "xmas"          "xxx"           "yahoo"
## [1145] "yar"           "yeah"          "year"          "yep"
## [1149] "yes"           "yest"          "yesterday"     "yet"
## [1153] "yoga"          "yogasana"      "yrs"           "yun"
## [1157] "yup"
```

```
sms_freq_words <- findFreqTerms(sms_dtm_train, 5)

str(sms_freq_words)
```

```
##  chr [1:1157] "abiola" "abl" "abt" "accept" "access" "account" ...
```

```
# Filter the DTM to include only the frequently used terms
sms_dtm_freq_train<- sms_dtm_train[ , sms_freq_words]
sms_dtm_freq_test <- sms_dtm_test[ , sms_freq_words]
```

```r
# Convert counts to Yes/No strings
convert_counts <- function(x) {
x <- ifelse(x > 0, "Yes", "No")
}

# Apply it to each of the columns
sms_train <- apply(sms_dtm_freq_train, MARGIN = 2,
convert_counts)
sms_test <- apply(sms_dtm_freq_test, MARGIN = 2,
convert_counts)
```

Training a model on the data

```r
sms_classifier <- naiveBayes(sms_train, sms_train_labels)
```

Evaluating model performance

```r
# Make the predictions
sms_test_pred <- predict(sms_classifier, sms_test)

CrossTable(sms_test_pred, sms_test_labels,
prop.chisq = FALSE, prop.t = FALSE,
dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  1390
##
##
##              | actual
##    predicted |       ham |      spam | Row Total |
## -------------|-----------|-----------|-----------|
##          ham |      1200 |        20 |      1220 |
##              |     0.984 |     0.016 |     0.878 |
##              |     0.993 |     0.110 |           |
## -------------|-----------|-----------|-----------|
##         spam |         9 |       161 |       170 |
##              |     0.053 |     0.947 |     0.122 |
##              |     0.007 |     0.890 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1209 |       181 |      1390 |
##              |     0.870 |     0.130 |           |
## -------------|-----------|-----------|-----------|
##
##
```

Improving model performance

```
# We'll build a Naive Bayes model as done earlier, but this time set laplace = 1
sms_classifier2 <- naiveBayes(sms_train, sms_train_labels,
laplace = 1)

# Make predictions
sms_test_pred2 <- predict(sms_classifier2, sms_test)

# Compare the predictions to actual classes
CrossTable(sms_test_pred2, sms_test_labels,
prop.chisq = FALSE, prop.t = FALSE, prop.r = FALSE,
dnn = c('predicted', 'actual'))
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Col Total |
## |-------------------------|
##
##
## Total Observations in Table:  1390
##
##
##              | actual
##    predicted |       ham |      spam | Row Total |
## -------------|-----------|-----------|-----------|
##          ham |      1202 |        28 |      1230 |
##              |     0.994 |     0.155 |           |
## -------------|-----------|-----------|-----------|
##         spam |         7 |       153 |       160 |
##              |     0.006 |     0.845 |           |
## -------------|-----------|-----------|-----------|
## Column Total |      1209 |       181 |      1390 |
##              |     0.870 |     0.130 |           |
## -------------|-----------|-----------|-----------|
##
##
```

Problem 2. Install the requisite packages to execute the following code that classifies the built-in iris data using Naive Bayes. Build an R Notebook and explain in detail what each step does. Be sure to look up each function to understand how it is used.

```
#install.packages("klaR")
library(klaR)
```

```
## Loading required package: MASS
```

```
data(iris)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
```

```
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
# identify indexes to be in testing dataset
# every index of 5th, 10th, 15th .. will be the testing dataset
# the rest are training dataset
testidx <- which(1:length(iris[, 1]) %% 5 == 0)

# separate into training and testing datasets
iristrain <- iris[-testidx,]
iristest <- iris[testidx,]

# apply Naive Bayes
nbmodel <- NaiveBayes(Species~., data=iristrain)

# check the accuracy
prediction <- predict(nbmodel, iristest[,-5])
table(prediction$class, iristest[,5])
```

```
##
##              setosa versicolor virginica
##   setosa         10          0         0
##   versicolor      0         10         2
##   virginica       0          0         8
```

1. How would you make a prediction for a new case with the above package? If the new case was stored in a vector named "unkown", then it's prediction would be made using the following command: prediction <- predict(nbmodel, unknown)
2. How does this package deal with numeric features? For each numeric variable, this package produces a table giving, for each target class, mean and standard deviation of the (sub-)variable or a object of class density.
3. How does it specify a Laplace estimator? It has an argument fL which is the Factor for Laplace correlation,default factor is 0, i.e. no correction

Problem 3. What are Laplace estimators and why are they used in Naive Bayes classification? Provide an example of how they might be used and when.

Laplace estimators are small values which are added for each of the feature counts in the frequency table which ensures that each feature has a nonzero probability of occurring with each class. They are used in Naive Bayes classification to reflect a presumed prior probability of how the feature relates to the class. Example: We are classifying emails as spam or ham using the following set of words: "Rolex", "Princess", "Free" and "Travel". Ideally all these words would appear in the spam category. However in practice some words never appear in past for specific category and suddenly appear at later stages, which makes entire calculations as zeros. In our case, the word "Free" never occured before. Suppose it appears for the first time along with the other words in the set. The probability of "Princess" in spam will be zero which will make the entire calculation zero and thus it will not classify it in spam. However it is highly likely that it is spam. Using a Laplace estimator will ensure that the calculation is not zero and classify it as spam.