STAT6620-01- Homework3

**Chinki**

**April 10, 2017**

1. **Naive Bayes for SMS spam filtering**

**Step 1**-**Collecting the data**

The data is adapted from the SMS spam Collection at <http://www.dt.fee.unicamp.br/~tiago/> smsspamcollection/.

**Step 2- Exploring and prepering the Data**

# read the sms data into the sms data frame  
sms\_raw <- read.csv("sms\_spam.csv", stringsAsFactors = FALSE)  
  
# Examine the structure of the sms Data  
str(sms\_raw)

## 'data.frame': 5559 obs. of 2 variables:  
## $ type: chr "ham" "ham" "ham" "spam" ...  
## $ text: chr "Hope you are having a good week. Just checking in" "K..give back my thanks." "Am also doing in cbe only. But have to pay." "complimentary 4 STAR Ibiza Holiday or Â£10,000 cash needs your URGENT collection. 09066364349 NOW from Landline not to lose out"| \_\_truncated\_\_ ...

# convert spam/ham to factor.  
sms\_raw$type=factor(sms\_raw$type)  
  
#Examine the type of variable  
str(sms\_raw)

## 'data.frame': 5559 obs. of 2 variables:  
## $ type: Factor w/ 2 levels "ham","spam": 1 1 1 2 2 1 1 1 2 1 ...  
## $ text: chr "Hope you are having a good week. Just checking in" "K..give back my thanks." "Am also doing in cbe only. But have to pay." "complimentary 4 STAR Ibiza Holiday or Â£10,000 cash needs your URGENT collection. 09066364349 NOW from Landline not to lose out"| \_\_truncated\_\_ ...

#create a table of sms\_raw$type  
table(sms\_raw$type)

##   
## ham spam   
## 4812 747

# build a corpus using the text mining (tm) package:  
#install.packages("tm") (If tm package is not installed )  
library(tm)

## Loading required package: NLP

sms\_corpus=VCorpus(VectorSource(sms\_raw$text))

# examine the sms corpus  
print(sms\_corpus)

## <<VCorpus>>  
## Metadata: corpus specific: 0, document level (indexed): 0  
## Content: documents: 5559

inspect(sms\_corpus[1:2])

## <<VCorpus>>  
## Metadata: corpus specific: 0, document level (indexed): 0  
## Content: documents: 2  
##   
## [[1]]  
## <<PlainTextDocument>>  
## Metadata: 7  
## Content: chars: 49  
##   
## [[2]]  
## <<PlainTextDocument>>  
## Metadata: 7  
## Content: chars: 23

# make sms\_corpus as a charater   
as.character(sms\_corpus[[1]])

## [1] "Hope you are having a good week. Just checking in"

#Apply charater to 1 to 2   
lapply(sms\_corpus[1:2], as.character)

## $`1`  
## [1] "Hope you are having a good week. Just checking in"  
##   
## $`2`  
## [1] "K..give back my thanks."

# clean up the corpus using tm\_map()  
sms\_corpus\_clean <- tm\_map(sms\_corpus, content\_transformer(tolower))

# show the difference between sms\_corpus and corpus\_clean  
as.character(sms\_corpus[[1]])

## [1] "Hope you are having a good week. Just checking in"

as.character(sms\_corpus\_clean[[1]])

## [1] "hope you are having a good week. just checking in"

# Remove numbers   
sms\_corpus\_clean <- tm\_map(sms\_corpus\_clean, removeNumbers)  
#Remove stop words  
sms\_corpus\_clean <- tm\_map(sms\_corpus\_clean, removeWords, stopwords())  
# Remove punctuvation  
sms\_corpus\_clean <- tm\_map(sms\_corpus\_clean, removePunctuation)

# illustration of word stemming  
#install.packages("SnowballC") (If snowballc package is not installed )  
library(SnowballC)  
wordStem(c("learn", "learned", "learning", "learns"))

## [1] "learn" "learn" "learn" "learn"

sms\_corpus\_clean <- tm\_map(sms\_corpus\_clean, stemDocument)  
# eliminate unneeded whitespace  
sms\_corpus\_clean <- tm\_map(sms\_corpus\_clean, stripWhitespace)

# examine the final clean corpus  
lapply(sms\_corpus[1:3], as.character)

## $`1`  
## [1] "Hope you are having a good week. Just checking in"  
##   
## $`2`  
## [1] "K..give back my thanks."  
##   
## $`3`  
## [1] "Am also doing in cbe only. But have to pay."

lapply(sms\_corpus\_clean[1:3], as.character)

## $`1`  
## [1] "hope good week just check"  
##   
## $`2`  
## [1] "kgive back thank"  
##   
## $`3`  
## [1] "also cbe pay"

# create a document-term sparse matrix  
sms\_dtm <- DocumentTermMatrix(sms\_corpus\_clean)

str(sms\_dtm)

## List of 6  
## $ i : int [1:42173] 1 1 1 1 1 2 2 2 3 3 ...  
## $ j : int [1:42173] 962 2278 2578 2945 6224 430 2997 5636 183 911 ...  
## $ v : num [1:42173] 1 1 1 1 1 1 1 1 1 1 ...  
## $ nrow : int 5559  
## $ ncol : int 6576  
## $ dimnames:List of 2  
## ..$ Docs : chr [1:5559] "1" "2" "3" "4" ...  
## ..$ Terms: chr [1:6576] "â<U+0082>å<U+0093>""| \_\_truncated\_\_ "â<U+0082>å<U+0093>harri""| \_\_truncated\_\_ "â<U+0082>â<U+0080><U+009C>""| \_\_truncated\_\_ "â<U+0080><U+0098>morrow""| \_\_truncated\_\_ ...  
## - attr(\*, "class")= chr [1:2] "DocumentTermMatrix" "simple\_triplet\_matrix"  
## - attr(\*, "weighting")= chr [1:2] "term frequency" "tf"

#Alternative solution   
sms\_dtm2=DocumentTermMatrix(sms\_corpus,control = list(tolower=TRUE,removeNumbers=TRUE,stopwords=TRUE,removePunctuation=TRUE,stemming=TRUE))  
str(sms\_dtm2)

## List of 6  
## $ i : int [1:43231] 1 1 1 1 1 2 2 2 3 3 ...  
## $ j : int [1:43231] 994 2352 2689 3086 6572 452 3142 5927 194 943 ...  
## $ v : num [1:43231] 1 1 1 1 1 1 1 1 1 1 ...  
## $ nrow : int 5559  
## $ ncol : int 6965  
## $ dimnames:List of 2  
## ..$ Docs : chr [1:5559] "1" "2" "3" "4" ...  
## ..$ Terms: chr [1:6965] "â<U+0082>å<U+0093>""| \_\_truncated\_\_ "â<U+0082>å<U+0093>harri""| \_\_truncated\_\_ "â<U+0082>â<U+0080><U+009C>""| \_\_truncated\_\_ "â<U+0080><U+0098>morrow""| \_\_truncated\_\_ ...  
## - attr(\*, "class")= chr [1:2] "DocumentTermMatrix" "simple\_triplet\_matrix"  
## - attr(\*, "weighting")= chr [1:2] "term frequency" "tf"

# Alternative solution:  
sms\_dtm3=DocumentTermMatrix(sms\_corpus,control = list(tolower=TRUE,removeNumbers=TRUE,stopwords=function(x){removeWords(x,stopwords())},removePunctuation=TRUE,stemming=TRUE))  
str(sms\_dtm3)

## List of 6  
## $ i : int [1:42173] 1 1 1 1 1 2 2 2 3 3 ...  
## $ j : int [1:42173] 962 2278 2578 2945 6224 430 2997 5636 183 911 ...  
## $ v : num [1:42173] 1 1 1 1 1 1 1 1 1 1 ...  
## $ nrow : int 5559  
## $ ncol : int 6576  
## $ dimnames:List of 2  
## ..$ Docs : chr [1:5559] "1" "2" "3" "4" ...  
## ..$ Terms: chr [1:6576] "â<U+0082>å<U+0093>""| \_\_truncated\_\_ "â<U+0082>å<U+0093>harri""| \_\_truncated\_\_ "â<U+0082>â<U+0080><U+009C>""| \_\_truncated\_\_ "â<U+0080><U+0098>morrow""| \_\_truncated\_\_ ...  
## - attr(\*, "class")= chr [1:2] "DocumentTermMatrix" "simple\_triplet\_matrix"  
## - attr(\*, "weighting")= chr [1:2] "term frequency" "tf"

#Comare sms\_dtm,sms\_dtm2 and sms\_dtm3  
sms\_dtm

## <<DocumentTermMatrix (documents: 5559, terms: 6576)>>  
## Non-/sparse entries: 42173/36513811  
## Sparsity : 100%  
## Maximal term length: 40  
## Weighting : term frequency (tf)

sms\_dtm2

## <<DocumentTermMatrix (documents: 5559, terms: 6965)>>  
## Non-/sparse entries: 43231/38675204  
## Sparsity : 100%  
## Maximal term length: 40  
## Weighting : term frequency (tf)

sms\_dtm3

## <<DocumentTermMatrix (documents: 5559, terms: 6576)>>  
## Non-/sparse entries: 42173/36513811  
## Sparsity : 100%  
## Maximal term length: 40  
## Weighting : term frequency (tf)

# creating training and test datasets  
sms\_dtm\_train <- sms\_dtm[1:4169, ]  
sms\_dtm\_test <- sms\_dtm[4170:5559, ]  
# also save the labels  
sms\_train\_labels <- sms\_raw[1:4169, ]$type  
sms\_test\_labels <- sms\_raw[4170:5559, ]$type  
  
# check that the proportion of spam is similar  
prop.table(table(sms\_train\_labels))

## sms\_train\_labels  
## ham spam   
## 0.8647158 0.1352842

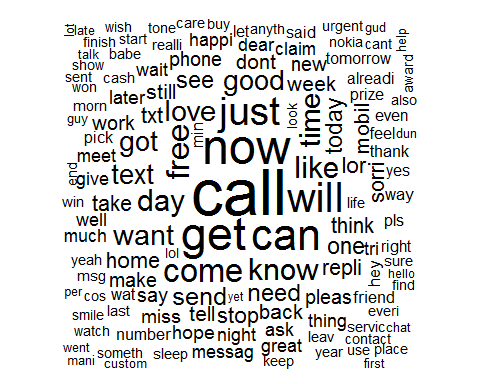
prop.table(table(sms\_test\_labels))

## sms\_test\_labels  
## ham spam   
## 0.8683453 0.1316547

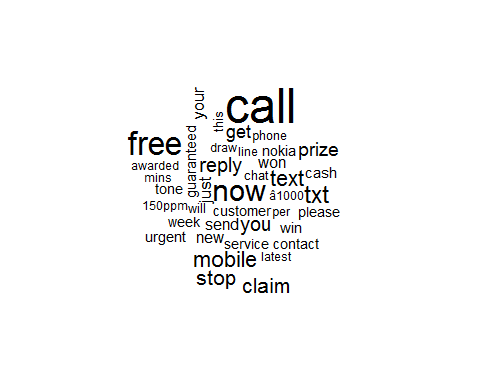
# word cloud visualization  
#install.packages("wordcloud") (If wordcloud package is not installed )  
library(wordcloud)

## Loading required package: RColorBrewer

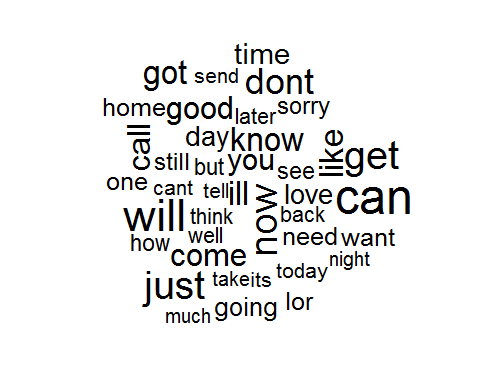
wordcloud(sms\_corpus\_clean, min.freq = 50, random.order = FALSE)



# subset the training data into spam and ham groups  
spam <- subset(sms\_raw, type == "spam")  
ham <- subset(sms\_raw, type == "ham")  
wordcloud(spam$text, max.words = 40, scale = c(3, 0.5))



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



sms\_dtm\_freq\_train <- removeSparseTerms(sms\_dtm\_train, 0.999)  
sms\_dtm\_freq\_train

## <<DocumentTermMatrix (documents: 4169, terms: 1104)>>  
## Non-/sparse entries: 24827/4577749  
## Sparsity : 99%  
## Maximal term length: 19  
## Weighting : term frequency (tf)

# indicator features for frequent words  
findFreqTerms(sms\_dtm\_train, 5)

## [1] "â<U+0082>â<U+0080><U+009C>" "abiola" "abl"   
## [4] "abt" "accept" "access"   
## [7] "account" "across" "act"   
## [10] "activ" "actual" "add"   
## [13] "address" "admir" "adult"   
## [16] "advanc" "aft" "afternoon"   
## [19] "age" "ago" "aha"   
## [22] "ahead" "aight" "aint"   
## [25] "air" "aiyo" "alex"   
## [28] "almost" "alon" "alreadi"   
## [31] "alright" "also" "alway"   
## [34] "angri" "announc" "anoth"   
## [37] "answer" "anymor" "anyon"   
## [40] "anyth" "anytim" "anyway"   
## [43] "apart" "app" "appli"   
## [46] "appreci" "arcad" "ard"   
## [49] "area" "argu" "argument"   
## [52] "armand" "around" "arrang"   
## [55] "arriv" "asap" "ask"   
## [58] "askd" "attempt" "auction"   
## [61] "avail" "ave" "avoid"   
## [64] "await" "awak" "award"   
## [67] "away" "awesom" "âwk"   
## [70] "babe" "babi" "back"   
## [73] "bad" "bag" "bank"   
## [76] "bare" "basic" "bath"   
## [79] "batteri" "bcoz" "bday"   
## [82] "beauti" "becom" "bed"   
## [85] "bedroom" "beer" "begin"   
## [88] "believ" "best" "better"   
## [91] "bid" "big" "bill"   
## [94] "bird" "birthday" "bit"   
## [97] "black" "blank" "bless"   
## [100] "blue" "bluetooth" "bold"   
## [103] "bonus" "boo" "book"   
## [106] "boost" "bore" "boss"   
## [109] "bother" "bout" "box"   
## [112] "boy" "boytoy" "break"   
## [115] "breath" "bring" "brother"   
## [118] "bslvyl" "btnationalr" "buck"   
## [121] "bus" "busi" "buy"   
## [124] "cabin" "call" "caller"   
## [127] "callertun" "camcord" "came"   
## [130] "camera" "campus" "can"   
## [133] "cancel" "cancer" "cant"   
## [136] "car" "card" "care"   
## [139] "carlo" "case" "cash"   
## [142] "cashbal" "catch" "caus"   
## [145] "celebr" "cell" "centr"   
## [148] "chanc" "chang" "charg"   
## [151] "chat" "cheap" "cheaper"   
## [154] "check" "cheer" "chennai"   
## [157] "chikku" "childish" "children"   
## [160] "choic" "choos" "christma"   
## [163] "claim" "class" "clean"   
## [166] "clear" "close" "club"   
## [169] "code" "coffe" "cold"   
## [172] "colleagu" "collect" "colleg"   
## [175] "colour" "come" "comin"   
## [178] "comp" "compani" "competit"   
## [181] "complet" "complimentari" "comput"   
## [184] "condit" "confirm" "congrat"   
## [187] "congratul" "connect" "contact"   
## [190] "content" "contract" "cook"   
## [193] "cool" "copi" "correct"   
## [196] "cos" "cost" "costa"   
## [199] "costâpm" "coupl" "cours"   
## [202] "cover" "coz" "crave"   
## [205] "crazi" "creat" "credit"   
## [208] "cri" "cross" "cuddl"   
## [211] "cum" "cup" "current"   
## [214] "custcar" "custom" "cut"   
## [217] "cute" "cuz" "dad"   
## [220] "daddi" "darl" "darlin"   
## [223] "darren" "dat" "date"   
## [226] "day" "dead" "deal"   
## [229] "dear" "decid" "decim"   
## [232] "decis" "deep" "definit"   
## [235] "del" "deliv" "deliveri"   
## [238] "den" "depend" "detail"   
## [241] "didnt" "die" "diet"   
## [244] "differ" "difficult" "digit"   
## [247] "din" "dinner" "direct"   
## [250] "dis" "discount" "discuss"   
## [253] "disturb" "dnt" "doc"   
## [256] "doctor" "doesnt" "dog"   
## [259] "doin" "don" "done"   
## [262] "dont" "door" "doubl"   
## [265] "download" "draw" "dream"   
## [268] "drink" "drive" "drop"   
## [271] "drug" "dude" "due"   
## [274] "dun" "dunno" "dvd"   
## [277] "earli" "earlier" "earth"   
## [280] "easi" "eat" "eatin"   
## [283] "egg" "either" "els"   
## [286] "email" "embarass" "end"   
## [289] "energi" "england" "enjoy"   
## [292] "enough" "enter" "entitl"   
## [295] "entri" "envelop" "etc"   
## [298] "euro" "eve" "even"   
## [301] "ever" "everi" "everybodi"   
## [304] "everyon" "everyth" "exact"   
## [307] "exam" "excel" "excit"   
## [310] "excus" "expect" "experi"   
## [313] "expir" "extra" "eye"   
## [316] "face" "facebook" "fact"   
## [319] "fall" "famili" "fanci"   
## [322] "fantasi" "fantast" "far"   
## [325] "fast" "fat" "father"   
## [328] "fault" "feb" "feel"   
## [331] "felt" "fetch" "fight"   
## [334] "figur" "file" "fill"   
## [337] "film" "final" "find"   
## [340] "fine" "finger" "finish"   
## [343] "first" "fix" "flag"   
## [346] "flat" "flight" "flower"   
## [349] "follow" "fone" "food"   
## [352] "forev" "forget" "forgot"   
## [355] "forward" "found" "freak"   
## [358] "free" "freemsg" "freephon"   
## [361] "fren" "fri" "friday"   
## [364] "friend" "friendship" "frm"   
## [367] "frnd" "frnds" "full"   
## [370] "fullonsmscom" "fun" "funni"   
## [373] "futur" "gal" "game"   
## [376] "gap" "gas" "gave"   
## [379] "gay" "gentl" "get"   
## [382] "gettin" "gift" "girl"   
## [385] "girlfrnd" "give" "glad"   
## [388] "god" "goe" "goin"   
## [391] "gone" "gonna" "good"   
## [394] "goodmorn" "goodnight" "got"   
## [397] "goto" "gotta" "great"   
## [400] "grin" "guarante" "gud"   
## [403] "guess" "guy" "gym"   
## [406] "haf" "haha" "hai"   
## [409] "hair" "half" "hand"   
## [412] "handset" "hang" "happen"   
## [415] "happi" "hard" "hate"   
## [418] "hav" "havent" "head"   
## [421] "hear" "heard" "heart"   
## [424] "heavi" "hee" "hell"   
## [427] "hello" "help" "hey"   
## [430] "hgsuiteland" "hit" "hiya"   
## [433] "hmm" "hmmm" "hmv"   
## [436] "hol" "hold" "holder"   
## [439] "holiday" "home" "hook"   
## [442] "hop" "hope" "horni"   
## [445] "hospit" "hot" "hotel"   
## [448] "hour" "hous" "how"   
## [451] "howev" "howz" "hrs"   
## [454] "httpwwwurawinnercom" "hug" "huh"   
## [457] "hungri" "hurri" "hurt"   
## [460] "iâ<U+0082>ë<U+009C>m" "ice" "idea"   
## [463] "identifi" "ignor" "ill"   
## [466] "immedi" "import" "inc"   
## [469] "includ" "india" "info"   
## [472] "inform" "insid" "instead"   
## [475] "interest" "invit" "ipod"   
## [478] "irrit" "ish" "island"   
## [481] "issu" "ive" "izzit"   
## [484] "januari" "jay" "job"   
## [487] "john" "join" "joke"   
## [490] "joy" "jst" "jus"   
## [493] "just" "juz" "kate"   
## [496] "keep" "kept" "kick"   
## [499] "kid" "kill" "kind"   
## [502] "kinda" "king" "kiss"   
## [505] "knew" "know" "knw"   
## [508] "ladi" "land" "landlin"   
## [511] "laptop" "lar" "last"   
## [514] "late" "later" "latest"   
## [517] "laugh" "lazi" "ldn"   
## [520] "lead" "learn" "least"   
## [523] "leav" "lect" "left"   
## [526] "leh" "lei" "less"   
## [529] "lesson" "let" "letter"   
## [532] "liao" "librari" "lie"   
## [535] "life" "lift" "light"   
## [538] "like" "line" "link"   
## [541] "list" "listen" "littl"   
## [544] "live" "lmao" "load"   
## [547] "loan" "local" "locat"   
## [550] "log" "lol" "london"   
## [553] "long" "longer" "look"   
## [556] "lookin" "lor" "lose"   
## [559] "lost" "lot" "lovabl"   
## [562] "love" "lover" "loyalti"   
## [565] "ltd" "luck" "lucki"   
## [568] "lunch" "luv" "mad"   
## [571] "made" "mah" "mail"   
## [574] "make" "malaria" "man"   
## [577] "mani" "march" "mark"   
## [580] "marri" "match" "mate"   
## [583] "matter" "maxim" "maxmin"   
## [586] "may" "mayb" "meal"   
## [589] "mean" "meant" "med"   
## [592] "medic" "meet" "meetin"   
## [595] "meh" "member" "men"   
## [598] "merri" "messag" "met"   
## [601] "mid" "midnight" "might"   
## [604] "min" "mind" "mine"   
## [607] "minut" "miracl" "miss"   
## [610] "mistak" "moan" "mob"   
## [613] "mobil" "mobileupd" "mode"   
## [616] "mom" "moment" "mon"   
## [619] "monday" "money" "month"   
## [622] "morn" "mother" "motorola"   
## [625] "move" "movi" "mrng"   
## [628] "mrt" "mrw" "msg"   
## [631] "msgs" "mths" "much"   
## [634] "mum" "murder" "music"   
## [637] "must" "muz" "nah"   
## [640] "nake" "name" "nation"   
## [643] "natur" "naughti" "near"   
## [646] "need" "net" "network"   
## [649] "neva" "never" "new"   
## [652] "news" "next" "nice"   
## [655] "nigeria" "night" "nite"   
## [658] "nobodi" "noe" "nokia"   
## [661] "noon" "nope" "normal"   
## [664] "normpton" "noth" "notic"   
## [667] "now" "num" "number"   
## [670] "nyt" "obvious" "offer"   
## [673] "offic" "offici" "okay"   
## [676] "oki" "old" "omg"   
## [679] "one" "onlin" "onto"   
## [682] "oop" "open" "oper"   
## [685] "opinion" "opt" "optout"   
## [688] "orang" "orchard" "order"   
## [691] "oredi" "oso" "other"   
## [694] "otherwis" "outsid" "pack"   
## [697] "page" "paid" "pain"   
## [700] "paper" "parent" "park"   
## [703] "part" "parti" "partner"   
## [706] "pass" "passion" "password"   
## [709] "past" "pay" "peopl"   
## [712] "per" "person" "pete"   
## [715] "phone" "photo" "pic"   
## [718] "pick" "pictur" "pin"   
## [721] "piss" "pix" "pizza"   
## [724] "place" "plan" "play"   
## [727] "player" "pleas" "pleasur"   
## [730] "plenti" "pls" "plus"   
## [733] "plz" "pmin" "pmsg"   
## [736] "pobox" "point" "poli"   
## [739] "polic" "poor" "pop"   
## [742] "possess" "possibl" "post"   
## [745] "pound" "power" "ppm"   
## [748] "pray" "present" "press"   
## [751] "pretti" "previous" "price"   
## [754] "princess" "privat" "prize"   
## [757] "prob" "probabl" "problem"   
## [760] "project" "promis" "pub"   
## [763] "put" "qualiti" "question"   
## [766] "quick" "quit" "quiz"   
## [769] "quot" "rain" "random"   
## [772] "rang" "rate" "rather"   
## [775] "rcvd" "reach" "read"   
## [778] "readi" "real" "reali"   
## [781] "realli" "reason" "receipt"   
## [784] "receiv" "recent" "record"   
## [787] "refer" "regard" "regist"   
## [790] "relat" "relax" "remain"   
## [793] "rememb" "remind" "remov"   
## [796] "rent" "rental" "repli"   
## [799] "repres" "request" "respond"   
## [802] "respons" "rest" "result"   
## [805] "return" "reveal" "review"   
## [808] "reward" "right" "ring"   
## [811] "rington" "rite" "road"   
## [814] "rock" "role" "room"   
## [817] "roommat" "rose" "round"   
## [820] "rowwjhl" "rpli" "rreveal"   
## [823] "run" "rush" "sad"   
## [826] "sae" "safe" "said"   
## [829] "sale" "sat" "saturday"   
## [832] "savamob" "save" "saw"   
## [835] "say" "sch" "school"   
## [838] "scream" "sea" "search"   
## [841] "sec" "second" "secret"   
## [844] "see" "seem" "seen"   
## [847] "select" "self" "sell"   
## [850] "semest" "send" "sens"   
## [853] "sent" "serious" "servic"   
## [856] "set" "settl" "sex"   
## [859] "sexi" "shall" "share"   
## [862] "shd" "ship" "shirt"   
## [865] "shop" "short" "show"   
## [868] "shower" "sick" "side"   
## [871] "sigh" "sight" "sign"   
## [874] "silent" "simpl" "sinc"   
## [877] "singl" "sipix" "sir"   
## [880] "sis" "sister" "sit"   
## [883] "situat" "skxh" "skype"   
## [886] "slave" "sleep" "slept"   
## [889] "slow" "slowli" "small"   
## [892] "smile" "smoke" "sms"   
## [895] "smth" "snow" "sofa"   
## [898] "sol" "somebodi" "someon"   
## [901] "someth" "sometim" "somewher"   
## [904] "song" "soni" "sonyericsson"   
## [907] "soon" "sorri" "sort"   
## [910] "sound" "south" "space"   
## [913] "speak" "special" "specialcal"   
## [916] "spend" "spent" "spoke"   
## [919] "spree" "stand" "start"   
## [922] "statement" "station" "stay"   
## [925] "std" "step" "still"   
## [928] "stockport" "stone" "stop"   
## [931] "store" "stori" "street"   
## [934] "student" "studi" "stuff"   
## [937] "stupid" "style" "sub"   
## [940] "subscrib" "success" "suck"   
## [943] "suit" "summer" "sun"   
## [946] "sunday" "sunshin" "sup"   
## [949] "support" "suppos" "sure"   
## [952] "surf" "surpris" "sweet"   
## [955] "swing" "system" "take"   
## [958] "talk" "tampa" "tariff"   
## [961] "tcs" "tea" "teach"   
## [964] "tear" "teas" "tel"   
## [967] "tell" "ten" "tenerif"   
## [970] "term" "test" "text"   
## [973] "thank" "thanx" "that"   
## [976] "thing" "think" "thinkin"   
## [979] "thk" "tho" "though"   
## [982] "thought" "throw" "thru"   
## [985] "tht" "thur" "tick"   
## [988] "ticket" "til" "till"   
## [991] "time" "tire" "titl"   
## [994] "tmr" "toclaim" "today"   
## [997] "togeth" "told" "tomo"   
## [1000] "tomorrow" "tone" "tonight"   
## [1003] "tonit" "took" "top"   
## [1006] "torch" "tot" "total"   
## [1009] "touch" "tough" "tour"   
## [1012] "toward" "town" "track"   
## [1015] "train" "transact" "travel"   
## [1018] "treat" "tri" "trip"   
## [1021] "troubl" "true" "trust"   
## [1024] "truth" "tscs" "ttyl"   
## [1027] "tuesday" "turn" "twice"   
## [1030] "two" "txt" "txting"   
## [1033] "txts" "type" "ufind"   
## [1036] "ugh" "ull" "uncl"   
## [1039] "understand" "unless" "unlimit"   
## [1042] "unredeem" "unsub" "unsubscrib"   
## [1045] "updat" "ure" "urgent"   
## [1048] "urself" "use" "user"   
## [1051] "usf" "usual" "uve"   
## [1054] "valentin" "valid" "valu"   
## [1057] "via" "video" "vikki"   
## [1060] "visit" "vodafon" "voic"   
## [1063] "vomit" "voucher" "wait"   
## [1066] "wake" "walk" "wan"   
## [1069] "wana" "wanna" "want"   
## [1072] "wap" "warm" "wast"   
## [1075] "wat" "watch" "water"   
## [1078] "way" "weak" "wear"   
## [1081] "weather" "wed" "wednesday"   
## [1084] "weed" "week" "weekend"   
## [1087] "welcom" "well" "wen"   
## [1090] "went" "what" "whatev"   
## [1093] "whenev" "whole" "wid"   
## [1096] "wif" "wife" "wil"   
## [1099] "will" "win" "wine"   
## [1102] "winner" "wish" "wit"   
## [1105] "within" "without" "wiv"   
## [1108] "wkli" "wks" "wnt"   
## [1111] "woke" "won" "wonder"   
## [1114] "wont" "word" "work"   
## [1117] "workin" "world" "worri"   
## [1120] "wors" "worth" "wot"   
## [1123] "wow" "write" "wrong"   
## [1126] "wwq" "wwwgetzedcouk" "xmas"   
## [1129] "xxx" "yahoo" "yar"   
## [1132] "yeah" "year" "yep"   
## [1135] "yes" "yesterday" "yet"   
## [1138] "yoga" "yup"

# save frequently-appearing terms to a character vector  
sms\_freq\_words <- findFreqTerms(sms\_dtm\_train, 5)  
str(sms\_freq\_words)

## chr [1:1139] "â<U+0082>â<U+0080><U+009C>""| \_\_truncated\_\_ "abiola" "abl" "abt" "accept" ...

# create DTMs with only the frequent terms  
sms\_dtm\_freq\_train <- sms\_dtm\_train[ , sms\_freq\_words]  
sms\_dtm\_freq\_test <- sms\_dtm\_test[ , sms\_freq\_words]

# convert counts to a factor  
convert\_counts <- function(x) {  
 x <- ifelse(x > 0, "Yes", "No")  
}

# apply() convert\_counts() to columns of train/test data  
sms\_train <- apply(sms\_dtm\_freq\_train, MARGIN = 2, convert\_counts)  
sms\_test <- apply(sms\_dtm\_freq\_test, MARGIN = 2, convert\_counts)

**Step 3- Training a model on the data**

#install.packages("e1071") (If package is not installed )  
library(e1071)  
sms\_classifier <- naiveBayes(sms\_train, sms\_train\_labels)

**Step 4- Evaluating model performance**

sms\_test\_pred <- predict(sms\_classifier, sms\_test)  
head(sms\_test\_pred)

## [1] ham ham ham ham spam ham   
## Levels: ham spam

#install.packages("gmodels") (If package is not installed )  
library(gmodels)  
CrossTable(sms\_test\_pred, 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 | 1201 | 30 | 1231 |   
## | 0.995 | 0.164 | |   
## -------------|-----------|-----------|-----------|  
## spam | 6 | 153 | 159 |   
## | 0.005 | 0.836 | |   
## -------------|-----------|-----------|-----------|  
## Column Total | 1207 | 183 | 1390 |   
## | 0.868 | 0.132 | |   
## -------------|-----------|-----------|-----------|  
##   
##

**Step 5- Improving model performance**

sms\_classifier2 <- naiveBayes(sms\_train, sms\_train\_labels, laplace = 1)  
sms\_test\_pred2 <- predict(sms\_classifier2, sms\_test)  
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 |   
## | 0.996 | 0.153 | |   
## -------------|-----------|-----------|-----------|  
## spam | 5 | 155 | 160 |   
## | 0.004 | 0.847 | |   
## -------------|-----------|-----------|-----------|  
## Column Total | 1207 | 183 | 1390 |   
## | 0.868 | 0.132 | |   
## -------------|-----------|-----------|-----------|  
##   
##

sms\_classifier3 <- naiveBayes(sms\_train, sms\_train\_labels, laplace = 2)  
sms\_test\_pred3 <- predict(sms\_classifier3, sms\_test)  
CrossTable(sms\_test\_pred3, 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 | 1204 | 34 | 1238 |   
## | 0.998 | 0.186 | |   
## -------------|-----------|-----------|-----------|  
## spam | 3 | 149 | 152 |   
## | 0.002 | 0.814 | |   
## -------------|-----------|-----------|-----------|  
## Column Total | 1207 | 183 | 1390 |   
## | 0.868 | 0.132 | |   
## -------------|-----------|-----------|-----------|  
##   
##

Accuracy is not improving by adding Laplace estimator. Accuracy is 97.62%.

1. **Naive Bayes classification for iris dataset**

**step 1- collecting dataset**

source=<https://archive.ics.uci.edu/ml/datasets/Iris> This is perhaps the best known database to be found in the pattern recognition literature. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other 2; the latter are NOT linearly separable from each other. Data set contains 5 variable named as 1. sepal length in cm 2. sepal width in cm 3. petal length in cm 4. petal width in cm 5. class: -- Iris Setosa -- Iris Versicolour -- Iris Virginica

**step 2- Exploring and preparing dataset**

#read the data  
iris=read.table("C:/Users/chink/Desktop/iris.data.txt",sep = ",",na.strings = T)  
#Applying headers  
colnames(iris)[1:5]=c("sepal\_length","sepal\_width","petal\_length","petal\_width","class")  
summary(iris)

## sepal\_length sepal\_width petal\_length petal\_width   
## Min. :4.300 Min. :2.000 Min. :1.000 Min. :0.100   
## 1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600 1st Qu.:0.300   
## Median :5.800 Median :3.000 Median :4.350 Median :1.300   
## Mean :5.843 Mean :3.054 Mean :3.759 Mean :1.199   
## 3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100 3rd Qu.:1.800   
## Max. :7.900 Max. :4.400 Max. :6.900 Max. :2.500   
## class   
## Iris-setosa :50   
## Iris-versicolor:50   
## Iris-virginica :50   
##   
##   
##

#converting as a factor to class  
iris$class=factor(iris$class)  
#Finding structure of iris data  
str(iris)

## 'data.frame': 150 obs. of 5 variables:  
## $ sepal\_length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...  
## $ sepal\_width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...  
## $ petal\_length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...  
## $ petal\_width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...  
## $ class : Factor w/ 3 levels "Iris-setosa",..: 1 1 1 1 1 1 1 1 1 1 ...

# Creating table for class variable   
table(iris$class)

##   
## Iris-setosa Iris-versicolor Iris-virginica   
## 50 50 50

As order of setosa is from 1 to 50 and versicolor is from 51 to 100. We can not take directly data from 1 to 100 in training because it will contain only two. so taking training data randomly.

#Making random sample   
sample\_iris=sample(150,110,replace = FALSE)

#creating training and test dataset  
iris\_training=iris[sample\_iris,]  
iris\_test=iris[-sample\_iris,]

#creating levels   
iris\_training\_labels=iris[sample\_iris,]$class  
iris\_test\_labels=iris[-sample\_iris,]$class

#table for training and test data  
table(iris\_training$class)

##   
## Iris-setosa Iris-versicolor Iris-virginica   
## 37 37 36

table(iris\_test$class)

##   
## Iris-setosa Iris-versicolor Iris-virginica   
## 13 13 14

**Step 3- Training a model on a data**

library(e1071)  
iris\_classifier=naiveBayes(iris\_training,iris\_training\_labels)

**Step 4- Evaluvating model performance**

iris\_test\_pred=predict(iris\_classifier,iris\_test)  
iris\_test\_pred

## [1] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [5] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [9] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [13] Iris-setosa Iris-versicolor Iris-versicolor Iris-versicolor  
## [17] Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor  
## [21] Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor  
## [25] Iris-versicolor Iris-versicolor Iris-virginica Iris-virginica   
## [29] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## [33] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## [37] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## Levels: Iris-setosa Iris-versicolor Iris-virginica

library(gmodels)  
CrossTable(iris\_test\_pred,iris\_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: 40   
##   
##   
## | actual   
## predicted | Iris-setosa | Iris-versicolor | Iris-virginica | Row Total |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-setosa | 13 | 0 | 0 | 13 |   
## | 1.000 | 0.000 | 0.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-versicolor | 0 | 13 | 0 | 13 |   
## | 0.000 | 1.000 | 0.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-virginica | 0 | 0 | 14 | 14 |   
## | 0.000 | 0.000 | 1.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Column Total | 13 | 13 | 14 | 40 |   
## | 0.325 | 0.325 | 0.350 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
##   
##

Step 5- Improving model performance

iris\_classifier2=naiveBayes(iris\_training,iris\_training\_labels,laplace = 1)  
iris\_test\_pred2=predict(iris\_classifier2,iris\_test)  
iris\_test\_pred2

## [1] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [5] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [9] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [13] Iris-setosa Iris-versicolor Iris-versicolor Iris-versicolor  
## [17] Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor  
## [21] Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor  
## [25] Iris-versicolor Iris-versicolor Iris-virginica Iris-virginica   
## [29] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## [33] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## [37] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## Levels: Iris-setosa Iris-versicolor Iris-virginica

CrossTable(iris\_test\_pred2,iris\_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: 40   
##   
##   
## | actual   
## predicted | Iris-setosa | Iris-versicolor | Iris-virginica | Row Total |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-setosa | 13 | 0 | 0 | 13 |   
## | 1.000 | 0.000 | 0.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-versicolor | 0 | 13 | 0 | 13 |   
## | 0.000 | 1.000 | 0.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-virginica | 0 | 0 | 14 | 14 |   
## | 0.000 | 0.000 | 1.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Column Total | 13 | 13 | 14 | 40 |   
## | 0.325 | 0.325 | 0.350 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
##   
##

iris\_classifier3=naiveBayes(iris\_training,iris\_training\_labels,laplace = 2)  
iris\_test\_pred3=predict(iris\_classifier3,iris\_test)  
iris\_test\_pred3

## [1] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [5] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [9] Iris-setosa Iris-setosa Iris-setosa Iris-setosa   
## [13] Iris-setosa Iris-versicolor Iris-versicolor Iris-versicolor  
## [17] Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor  
## [21] Iris-versicolor Iris-versicolor Iris-versicolor Iris-versicolor  
## [25] Iris-versicolor Iris-versicolor Iris-virginica Iris-virginica   
## [29] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## [33] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## [37] Iris-virginica Iris-virginica Iris-virginica Iris-virginica   
## Levels: Iris-setosa Iris-versicolor Iris-virginica

CrossTable(iris\_test\_pred3,iris\_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: 40   
##   
##   
## | actual   
## predicted | Iris-setosa | Iris-versicolor | Iris-virginica | Row Total |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-setosa | 13 | 0 | 0 | 13 |   
## | 1.000 | 0.000 | 0.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-versicolor | 0 | 13 | 0 | 13 |   
## | 0.000 | 1.000 | 0.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Iris-virginica | 0 | 0 | 14 | 14 |   
## | 0.000 | 0.000 | 1.000 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
## Column Total | 13 | 13 | 14 | 40 |   
## | 0.325 | 0.325 | 0.350 | |   
## ----------------|-----------------|-----------------|-----------------|-----------------|  
##   
##

Accuracy is 100% without using the laplace estimator and with laplace estimator.

**2.Naive Bayes for HouseVotes84**

**Step 1- Collecting Data**

The data is collected from package mlbench.This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the CQA. A data frame with 435 observations on 17 variables:

1 Class Name: 2 (democrat, republican) 2 handicapped-infants: 2 (y,n) 3 water-project-cost-sharing: 2 (y,n) 4 adoption-of-the-budget-resolution: 2 (y,n) 5 physician-fee-freeze: 2 (y,n) 6 el-salvador-aid: 2 (y,n) 7 religious-groups-in-schools: 2 (y,n) 8 anti-satellite-test-ban: 2 (y,n) 9 aid-to-nicaraguan-contras: 2 (y,n) 10 mx-missile: 2 (y,n) 11 immigration: 2 (y,n) 12 synfuels-corporation-cutback: 2 (y,n) 13 education-spending: 2 (y,n) 14 superfund-right-to-sue: 2 (y,n) 15 crime: 2 (y,n) 16 duty-free-exports: 2 (y,n) 17 export-administration-act-south-africa: 2 (y,n)

**Step 2- Exploring and Preparing Data**

# Reading data   
data (HouseVotes84, package="mlbench")  
#Structure of Data   
str(HouseVotes84)

## 'data.frame': 435 obs. of 17 variables:  
## $ Class: Factor w/ 2 levels "democrat","republican": 2 2 1 1 1 1 1 2 2 1 ...  
## $ V1 : Factor w/ 2 levels "n","y": 1 1 NA 1 2 1 1 1 1 2 ...  
## $ V2 : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 2 ...  
## $ V3 : Factor w/ 2 levels "n","y": 1 1 2 2 2 2 1 1 1 2 ...  
## $ V4 : Factor w/ 2 levels "n","y": 2 2 NA 1 1 1 2 2 2 1 ...  
## $ V5 : Factor w/ 2 levels "n","y": 2 2 2 NA 2 2 2 2 2 1 ...  
## $ V6 : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 2 2 2 1 ...  
## $ V7 : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 2 ...  
## $ V8 : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 2 ...  
## $ V9 : Factor w/ 2 levels "n","y": 1 1 1 1 1 1 1 1 1 2 ...  
## $ V10 : Factor w/ 2 levels "n","y": 2 1 1 1 1 1 1 1 1 1 ...  
## $ V11 : Factor w/ 2 levels "n","y": NA 1 2 2 2 1 1 1 1 1 ...  
## $ V12 : Factor w/ 2 levels "n","y": 2 2 1 1 NA 1 1 1 2 1 ...  
## $ V13 : Factor w/ 2 levels "n","y": 2 2 2 2 2 2 NA 2 2 1 ...  
## $ V14 : Factor w/ 2 levels "n","y": 2 2 2 1 2 2 2 2 2 1 ...  
## $ V15 : Factor w/ 2 levels "n","y": 1 1 1 1 2 2 2 NA 1 NA ...  
## $ V16 : Factor w/ 2 levels "n","y": 2 NA 1 2 2 2 2 2 2 NA ...

#converting as a factor  
HouseVotes84$Class=factor(HouseVotes84$Class)

table(HouseVotes84$Class)

##   
## democrat republican   
## 267 168

#Creating model  
model <- naiveBayes(Class ~ ., data = HouseVotes84)  
model

##   
## Naive Bayes Classifier for Discrete Predictors  
##   
## Call:  
## naiveBayes.default(x = X, y = Y, laplace = laplace)  
##   
## A-priori probabilities:  
## Y  
## democrat republican   
## 0.6137931 0.3862069   
##   
## Conditional probabilities:  
## V1  
## Y n y  
## democrat 0.3953488 0.6046512  
## republican 0.8121212 0.1878788  
##   
## V2  
## Y n y  
## democrat 0.4979079 0.5020921  
## republican 0.4932432 0.5067568  
##   
## V3  
## Y n y  
## democrat 0.1115385 0.8884615  
## republican 0.8658537 0.1341463  
##   
## V4  
## Y n y  
## democrat 0.94594595 0.05405405  
## republican 0.01212121 0.98787879  
##   
## V5  
## Y n y  
## democrat 0.78431373 0.21568627  
## republican 0.04848485 0.95151515  
##   
## V6  
## Y n y  
## democrat 0.5232558 0.4767442  
## republican 0.1024096 0.8975904  
##   
## V7  
## Y n y  
## democrat 0.2277992 0.7722008  
## republican 0.7592593 0.2407407  
##   
## V8  
## Y n y  
## democrat 0.1711027 0.8288973  
## republican 0.8471338 0.1528662  
##   
## V9  
## Y n y  
## democrat 0.2419355 0.7580645  
## republican 0.8848485 0.1151515  
##   
## V10  
## Y n y  
## democrat 0.5285171 0.4714829  
## republican 0.4424242 0.5575758  
##   
## V11  
## Y n y  
## democrat 0.4941176 0.5058824  
## republican 0.8679245 0.1320755  
##   
## V12  
## Y n y  
## democrat 0.8554217 0.1445783  
## republican 0.1290323 0.8709677  
##   
## V13  
## Y n y  
## democrat 0.7103175 0.2896825  
## republican 0.1392405 0.8607595  
##   
## V14  
## Y n y  
## democrat 0.64980545 0.35019455  
## republican 0.01863354 0.98136646  
##   
## V15  
## Y n y  
## democrat 0.36254980 0.63745020  
## republican 0.91025641 0.08974359  
##   
## V16  
## Y n y  
## democrat 0.06486486 0.93513514  
## republican 0.34246575 0.65753425

#Creating training and test Data set  
HouseVotes84\_training=HouseVotes84[1:350,]  
HouseVotes84\_test=HouseVotes84[351:435,]

#Creating lables for Data set  
HouseVotes84\_training\_labels=HouseVotes84[1:350,]$Class  
HouseVotes84\_test\_labels=HouseVotes84[351:435,]$Class

#Creating prob table for training and test data  
prop.table(table(HouseVotes84\_training\_labels))

## HouseVotes84\_training\_labels  
## democrat republican   
## 0.6142857 0.3857143

prop.table(table(HouseVotes84\_test\_labels))

## HouseVotes84\_test\_labels  
## democrat republican   
## 0.6117647 0.3882353

**Step 3- Training a model on the data**

library(e1071)  
houseVotes84\_classifier=naiveBayes(HouseVotes84\_training,HouseVotes84\_training\_labels)

**Step 4- Evaluating model performance**

HouseVotes84\_test\_pred=predict(houseVotes84\_classifier,HouseVotes84\_test)  
HouseVotes84\_test\_pred

## [1] democrat republican democrat republican democrat democrat   
## [7] republican republican democrat republican democrat democrat   
## [13] democrat republican republican republican democrat democrat   
## [19] democrat republican democrat democrat republican democrat   
## [25] republican republican democrat republican republican republican  
## [31] democrat democrat republican democrat republican democrat   
## [37] democrat democrat republican democrat democrat democrat   
## [43] republican republican democrat democrat democrat democrat   
## [49] democrat republican republican republican republican republican  
## [55] republican republican democrat republican democrat republican  
## [61] republican democrat republican republican democrat democrat   
## [67] republican democrat democrat democrat republican democrat   
## [73] democrat democrat democrat democrat democrat republican  
## [79] democrat democrat republican democrat republican republican  
## [85] republican  
## Levels: democrat republican

library(gmodels)  
CrossTable(HouseVotes84\_test\_pred,HouseVotes84\_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: 85   
##   
##   
## | actual   
## predicted | democrat | republican | Row Total |   
## -------------|------------|------------|------------|  
## democrat | 45 | 1 | 46 |   
## | 0.865 | 0.030 | |   
## -------------|------------|------------|------------|  
## republican | 7 | 32 | 39 |   
## | 0.135 | 0.970 | |   
## -------------|------------|------------|------------|  
## Column Total | 52 | 33 | 85 |   
## | 0.612 | 0.388 | |   
## -------------|------------|------------|------------|  
##   
##

**Step 5- Improving model performance**

houseVotes84\_classifier2=naiveBayes(HouseVotes84\_training,HouseVotes84\_training\_labels,laplace = 1)  
HouseVotes84\_test\_pred2=predict(houseVotes84\_classifier2,HouseVotes84\_test)  
HouseVotes84\_test\_pred2

## [1] democrat republican democrat republican democrat democrat   
## [7] republican republican democrat republican democrat democrat   
## [13] democrat republican republican republican democrat democrat   
## [19] democrat republican democrat democrat republican democrat   
## [25] republican republican democrat republican republican republican  
## [31] democrat democrat republican democrat republican republican  
## [37] democrat democrat republican democrat democrat democrat   
## [43] republican republican democrat democrat democrat democrat   
## [49] democrat republican republican republican republican republican  
## [55] republican republican democrat republican democrat republican  
## [61] republican democrat republican republican democrat democrat   
## [67] republican democrat democrat democrat republican democrat   
## [73] democrat democrat democrat democrat democrat republican  
## [79] democrat democrat republican democrat republican republican  
## [85] republican  
## Levels: democrat republican

library(gmodels)  
CrossTable(HouseVotes84\_test\_pred2,HouseVotes84\_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: 85   
##   
##   
## | actual   
## predicted | democrat | republican | Row Total |   
## -------------|------------|------------|------------|  
## democrat | 44 | 1 | 45 |   
## | 0.846 | 0.030 | |   
## -------------|------------|------------|------------|  
## republican | 8 | 32 | 40 |   
## | 0.154 | 0.970 | |   
## -------------|------------|------------|------------|  
## Column Total | 52 | 33 | 85 |   
## | 0.612 | 0.388 | |   
## -------------|------------|------------|------------|  
##   
##

Accuracy is decreasing by adding Laplace estimator. Hence we will use model without Laplace estimator. Accuracy is 90.59%.

1. **Sentiment Analysis**

I use add-on in Google Sheet for sentiment analysis. A basic sentiment analysis is to identify sentiment of sentence level.1st column is subjective (subjective, objective), 2nd subjective confidence (0 to 1), polarity (positive, negative), polarity confidence (0 to 1).

I use twitter data for sentiment analysis.

Link of the work –

<https://docs.google.com/spreadsheets/d/19kkqO3SlvjHQCmao5-mKHsxaRmR7jOgU773gvl-eD14/edit#gid=0>

Screenshot of sentiment analysis add-on in Google Sheet.

