# **ALY-6020 Project- Week 2: Classification Using Naive Bayes**

# 

# 

## **Assignment Title: ALY-6020 Week 2 Project**

## Course Name: ALY6020 20941 Predictive Analytics SEC 01 Winter 2018 CPS [VTL-A-OL] ALY6020.20941.201825

## Academic Term: Course Number: 20941

## Submission Date: 01/21/2018

## Student: Srinivasu Narahari

## Professor: Steward Huang

**Introduction:**

**(Lantz, B.)**

Classifiers based on Bayesian methods utilize training data to calculate an observed probability of each outcome based on the evidence provided by feature values. When the classifier is later applied to unlabeled data, it uses the observed probabilities to predict the most likely class for the new features.

**Concepts Bayesian method**

**Probability:**

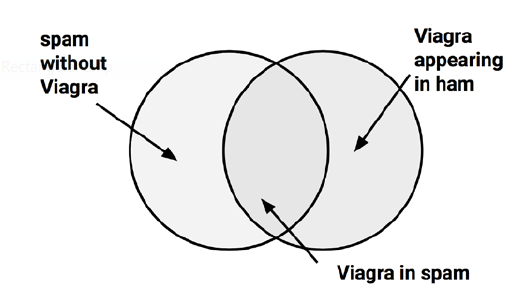
The possibility of occurring event is called probability.

**Event: Heads Result**

**Trial: Coin Flip**

The probability of event is calculated by the dividing the number of trials the event occurs divided by total number of trails**.**

**Joint Probability:**



The probability that both P(spam) and P(Viagra) occur, can be written as P (spam ∩ Viagra).

Calculating the probability of as P (spam ∩ Viagra) depends on both probability of P(spam) and P(Viagra). If these two evets are not dependent on other they are called **independent events**. Independent events do not affect the output of another independent event. They cannot provide any details about other independent event.

**Conditional Probability:**

P(A|B**):** Denotes the probability of occurrence of A, if event B occurred. This called conditional probability.



*P(A)*, the probability for occurrence of A. This estimate is known as P(A). This estimate is known as the

**prior probability**. The probability for occurrence of event A, if event B occurred is called **likelihood.**

**The Naive Bayes algorithm:**

The Naive Bayes algorithm is named as such because it makes some "naive" assumptions about the data. Naive Bayes assumes that all the features in the dataset are equally important and independent.

**Analysis:**

**PART A:**

**Filtering mobile phone spam with the Naive Bayes algorithm:** This example helps in finding the spam messages received to mobile phone. Developing a classification algorithm that could filter SMS spam would provide a useful tool for cellular phone providers. Junk messages are labeled spam, while

legitimate messages are labeled ham. The following steps needs to perform on the data set achieve the results.

**Step 1 – collecting data**

1. http://www.dt.fee.unicamp.br/~tiago/smsspamcollection/
2. smsNaive\_raw <- read.csv("C:\\narahariTransactions\AddmissionsInfo\\ALY6020 20941 Predictive Analytics SEC 01\\Week2\\sms\_spam.csv", stringsAsFactors = FALSE)
3. smsNaive\_raw$type <- factor(smsNaive\_raw$type)
4. str(smsNaive\_raw$type)
5. table(smsNaive\_raw$type)

**Step 2 – exploring and preparing the data**

1. Data preparation – cleaning and standardizing text data
2. install.packages("tm")
3. smsNaive\_corpus <- VCorpus(VectorSource(smsNaive\_raw$text))
4. print(smsNaive\_corpus)
5. inspect(smsNaive\_corpus[1:2])
6. as.character(smsNaive\_corpus[[1]])
7. lapply(smsNaive\_corpus[1:2], as.character)
8. smsNaive\_corpus\_clean <- tm\_map(smsNaive\_corpus,content\_transformer(tolower))
9. as.character(smsNaive\_corpus[[1]])
10. smsNaive\_corpus\_clean <- tm\_map(smsNaive\_corpus\_clean, removeNumbers)
11. smsNaive\_corpus\_clean <- tm\_map(smsNaive\_corpus\_clean,removeWords, stopwords())
12. smsNaive\_corpus\_clean <- tm\_map(smsNaive\_corpus\_clean, removePunctuation)
13. removePunctuation("hello...world")
14. install.packages("SnowballC")
15. library(SnowballC)
16. wordStem(c("learn", "learned", "learning", "learns"))
17. smsNaive\_corpus\_clean <- tm\_map(smsNaive\_corpus\_clean, stemDocument)
18. smsNaive\_corpus\_clean <- tm\_map(smsNaive\_corpus\_clean, stripWhitespace)
19. SMS messages before cleaning
20. as.character(smsNaive\_corpus[1:3])
21. SMS messages after cleaning
22. as.character(smsNaive\_corpus\_clean[1:3])

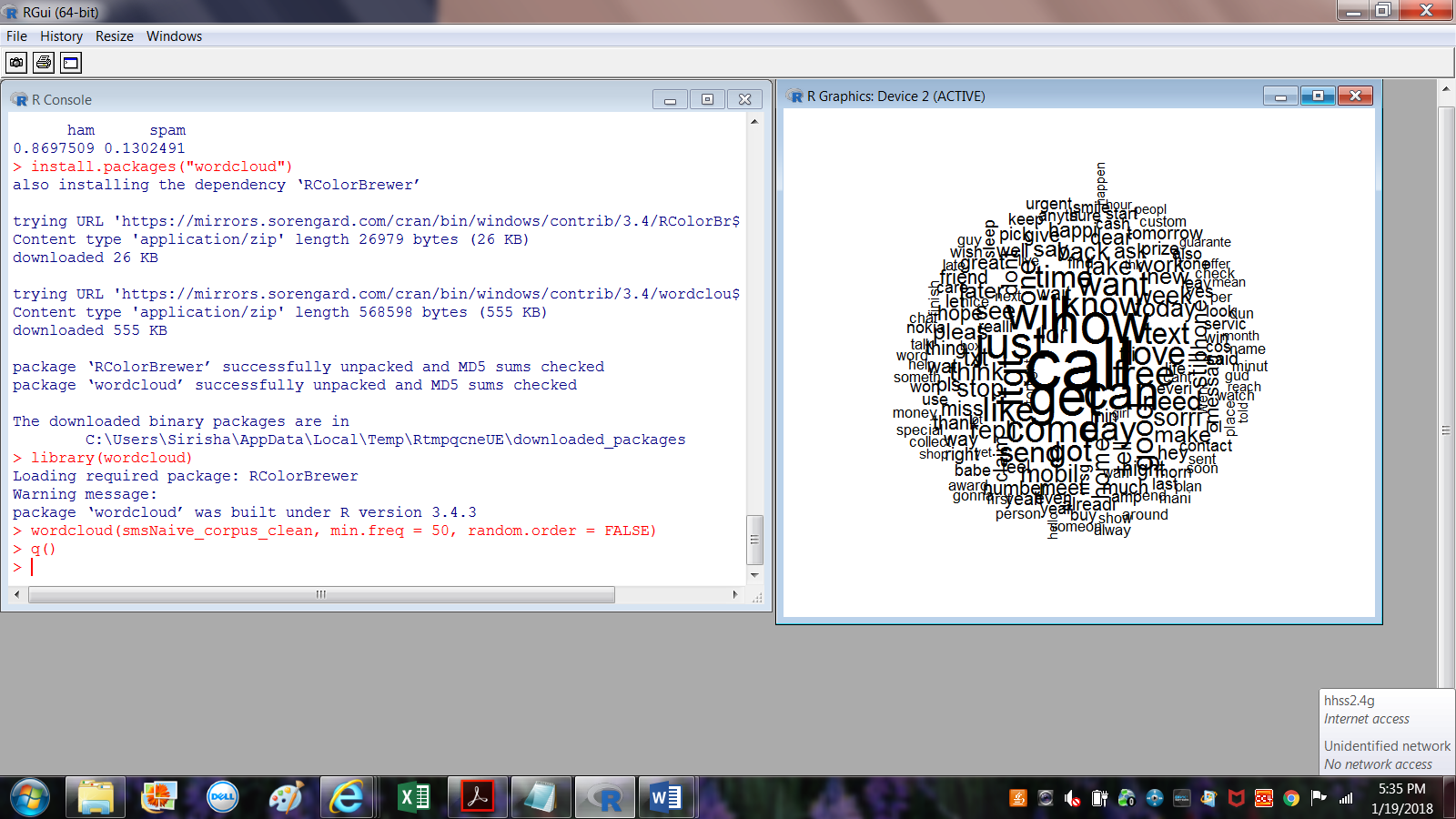
**Data preparation – splitting text documents into words**

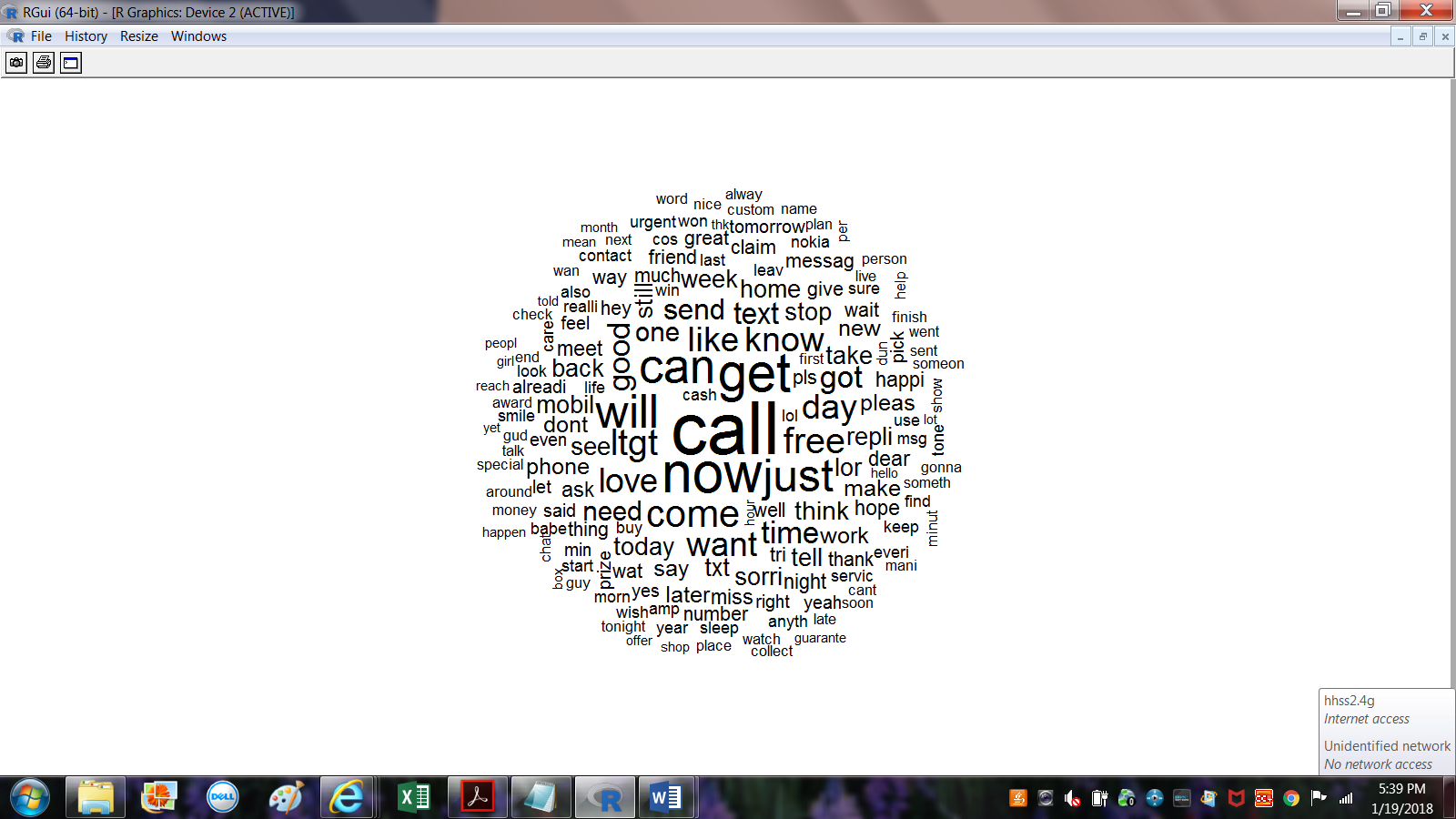
1. smsNaive\_dtm <- DocumentTermMatrix(smsNaive\_corpus\_clean)
2. smsNaive\_dtm2 <- DocumentTermMatrix(smsNaive\_corpus, control = list(tolower = TRUE,removeNumbers = TRUE,stopwords = TRUE,removePunctuation = TRUE,stemming = TRUE))

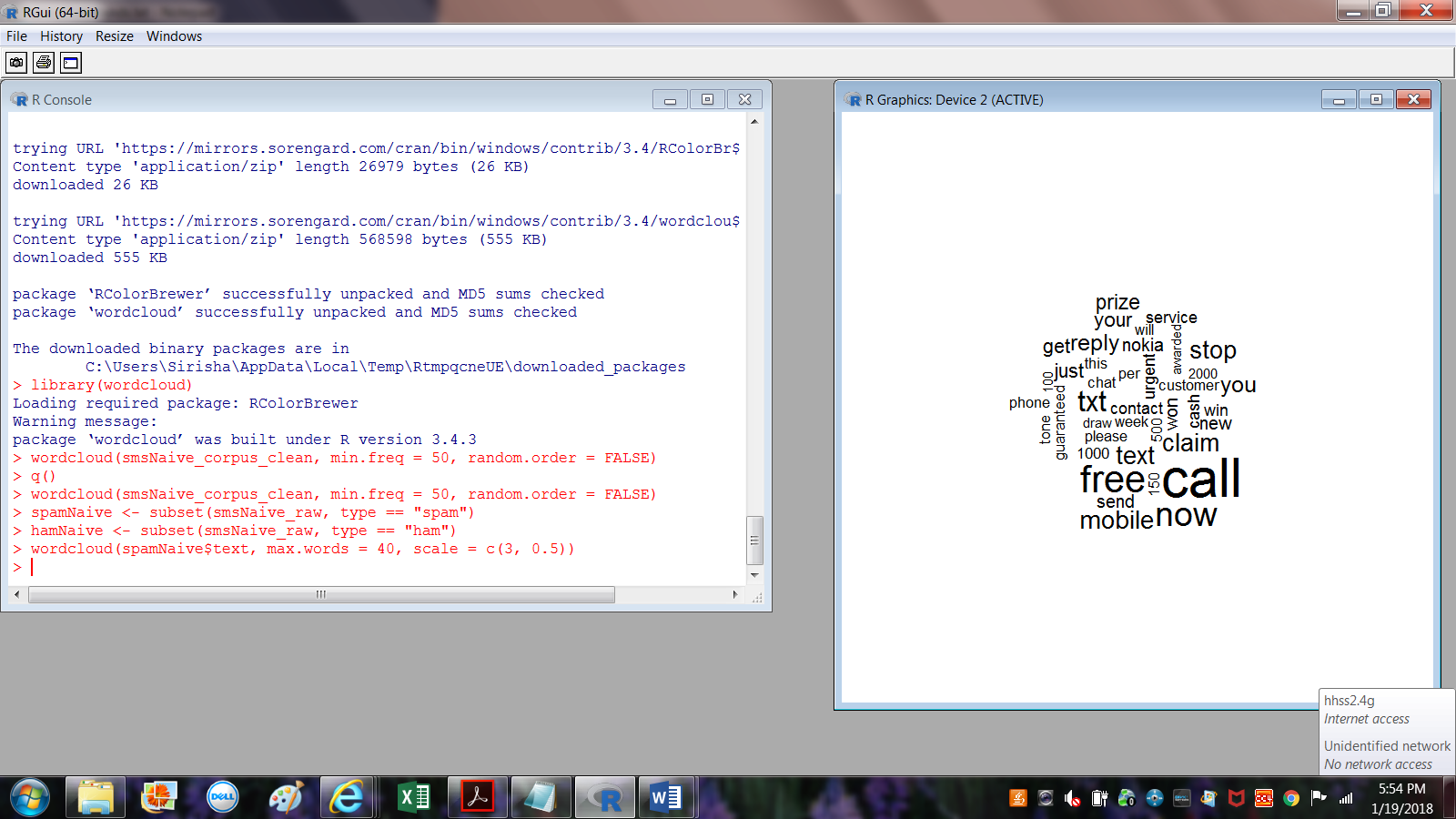
**Data preparation – creating training and test datasets**

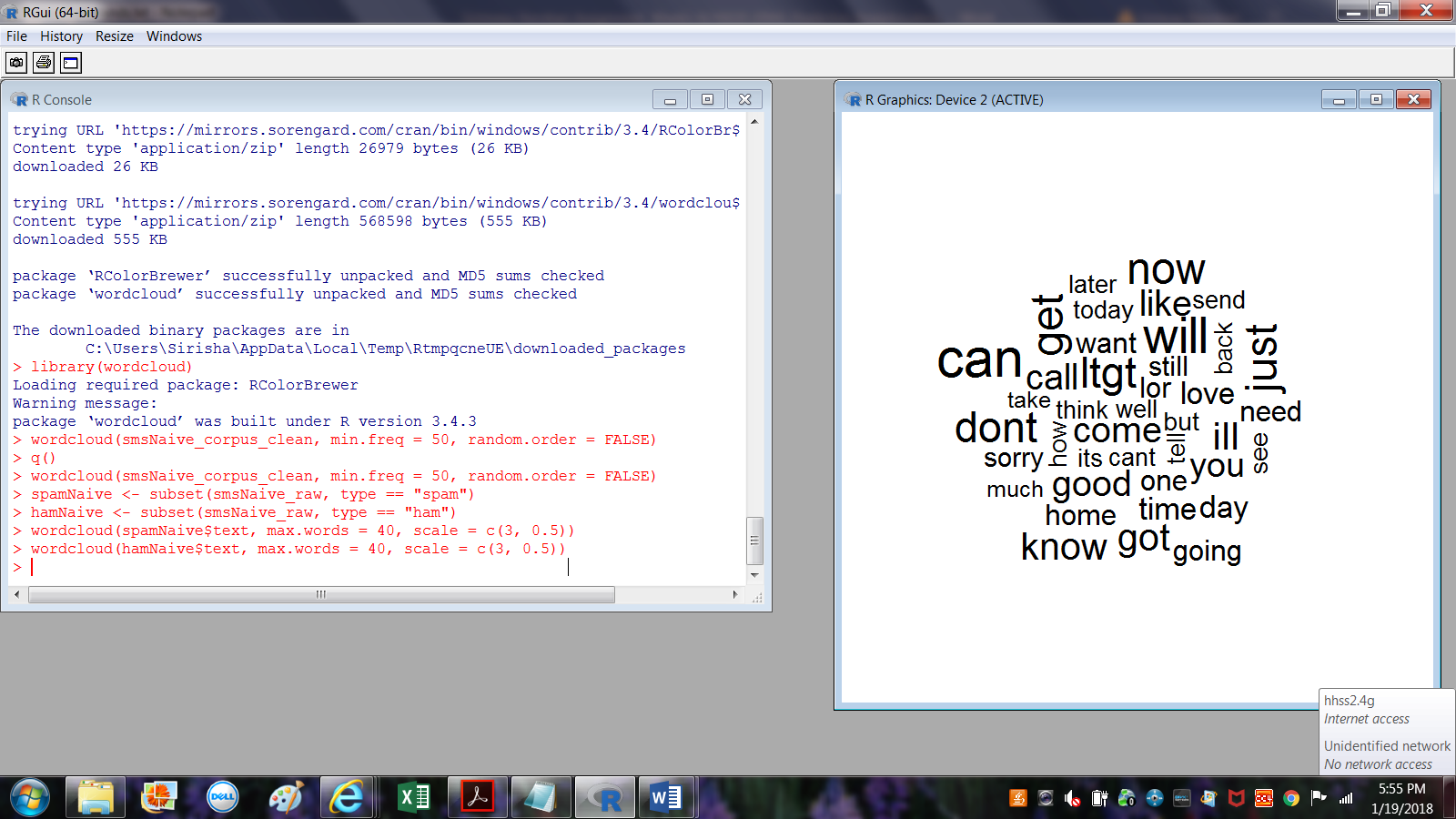
1. smsNaive\_dtm\_train <- smsNaive\_dtm[1:4169, ]
2. smsNaive\_dtm\_test <- smsNaive\_dtm[4170:5559, ]
3. smsNaive\_train\_labels <- smsNaive\_raw[1:4169, ]$type
4. smsNaive\_test\_labels <- smsNaive\_raw[4170:5559, ]$type
5. prop.table(table(smsNaive\_train\_labels))
6. prop.table(table(smsNaive\_test\_labels))

**Visualizing text data – word clouds**





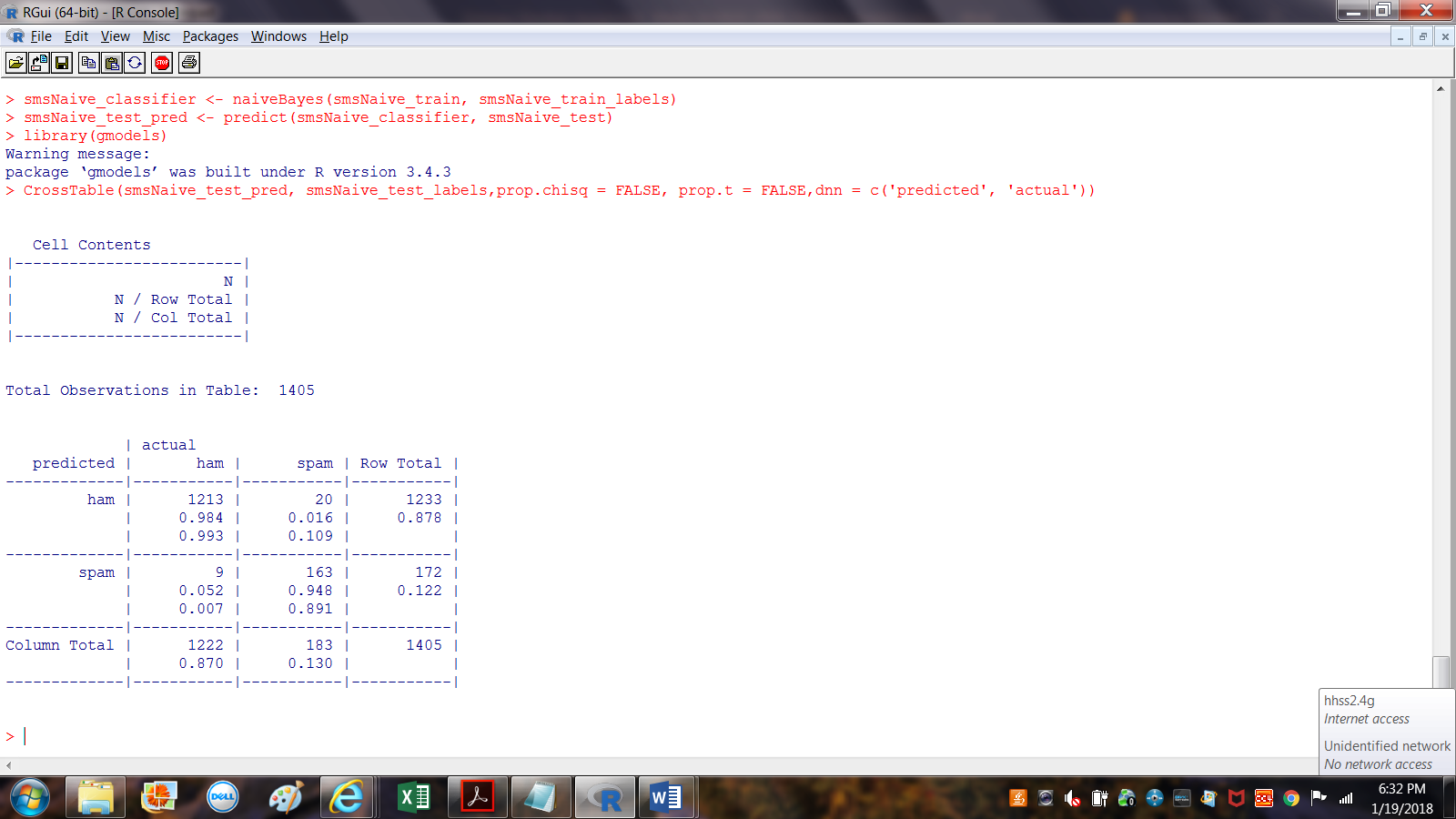
**’**

**Data preparation – creating indicator features for frequent words**

**Step 3 – training a model on the data**

**Step 4 – evaluating model performance**

1. install.packages("e1071")
2. smsNaive\_classifier <- naiveBayes(smsNaive\_train, smsNaive\_train\_labels)
3. smsNaive\_test\_pred <- predict(smsNaive\_classifier, smsNaive\_test)
4. library(gmodels)CrossTable(smsNaive\_test\_pred, smsNaive\_test\_labels,prop.chisq = FALSE, prop.t = FALSE,dnn = c('predicted', 'actual'))

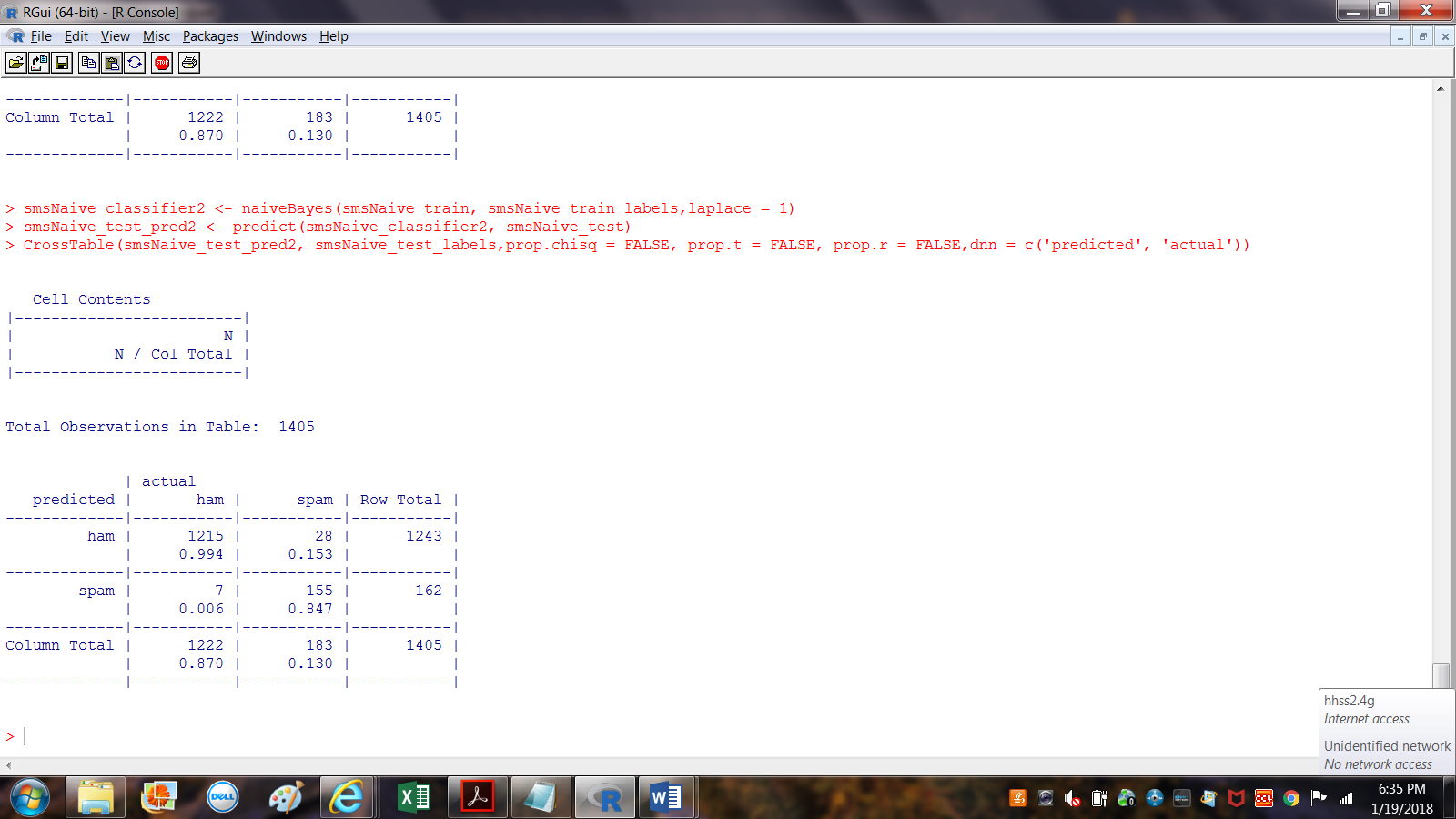


In the current date we observed 1405 SMS messages. Looking the above result 9+20 messages are wrongly classified. 9 out of 1222 messages were notified as spam even though they are ham.

In the same way 20 of spam messages are predicted as ham

**Step 5 – improving model performance**

1. smsNaive\_classifier2 <- naiveBayes(smsNaive\_train, smsNaive\_train\_labels,laplace = 1)
2. smsNaive\_test\_pred2 <- predict(smsNaive\_classifier2, smsNaive\_test)
3. CrossTable(smsNaive\_test\_pred2, smsNaive\_test\_labels,prop.chisq = FALSE, prop.t = FALSE, prop.r = FALSE,dnn = c('predicted', 'actual'))



In the current date we observed 1405 SMS messages. Looking the above result 7 +28 messages are wrongly classified. 7 out of 1222 messages were notified as spam even though they are ham. In the same way 28 of spam messages are predicted as ham. Adding the Laplace estimator reduced the number of false positives (ham messages erroneously classified as spam) from nine to seven and the number of false negatives from 20 to 28.

**PART B (Awati, K.)**:

HouseVotes84 dataset, which contains US congressional voting records for 1984. The HouseVotes84 dataset describes how 435 representatives voted – yes (y), no (n) or unknown (NA) – on 16 key issues presented to Congress. The dataset also provides the party affiliation of each representative – democrat or republican.

#load mlbench library

library(mlbench)

#set working directory if needed (modify path as needed)

setwd("C:\\narahariTransactions\AddmissionsInfo\\ALY6020 20941 Predictive Analytics SEC 01\\Week2\\Wdir")

#load HouseVotes84 dataset

data(HouseVotes84)

**Imputation of Data:**

Naïve Bayes algorithms typically handle NA values either by ignoring records that contain any NA values or by ignoring just the NA values. These choices are indicated by the value of the variable na.action in the naïve Bayes algorithm, which is set to na.omit (to ignore the record) or na.pass (to ignore the value).

**Function for Imputation:**

p\_y\_col\_class <- function(col,cls){sum\_y<-sum(HouseVotes84[,col]=='y' & HouseVotes84$Class==cls,na.rm = TRUE)

sum\_n<-sum(HouseVotes84[,col]=='n' & HouseVotes84$Class==cls,na.rm = TRUE)

return(sum\_y/(sum\_y+sum\_n))

}

**To impute the NA values following code randomly assigning values (y or n) to NAs, based on the proportion of members of a party who have voted y or n. In practice, this can be achieved by invoking** the uniform distribution and setting an NA value to y if the random number returned is less than the probability of a yes vote and to n otherwise.

nb\_multiple\_runs <- function(train\_fraction,n)

{

fraction\_correct <- rep(NA,n)

for(i in 1:n){HouseVotes84[,"train"] <- ifelse(runif(nrow(HouseVotes84))<train\_fraction,1,0)

trainColNum <- grep("train",names(HouseVotes84))

trainHouseVotes84 <- HouseVotes84[HouseVotes84$train == 1,-trainColNum]

testHouseVotes84 <- HouseVotes84[HouseVotes84$train ==0,-trainColNum]

nb\_model <- naiveBayes(Class~.,data = trainHouseVotes84)

nb\_test\_predict <- predict(nb\_model,testHouseVotes84[,-1])

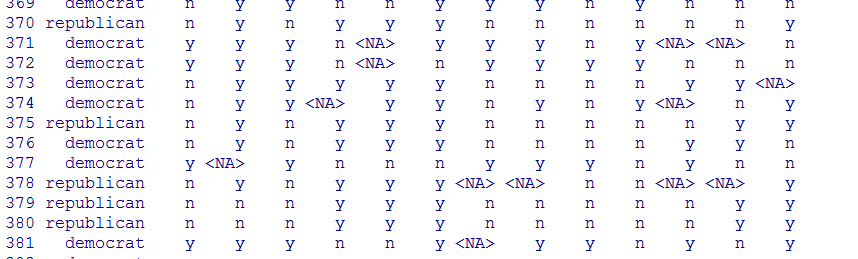
fraction\_correct[i] <- mean(nb\_test\_predict == testHouseVotes84$Class)

}

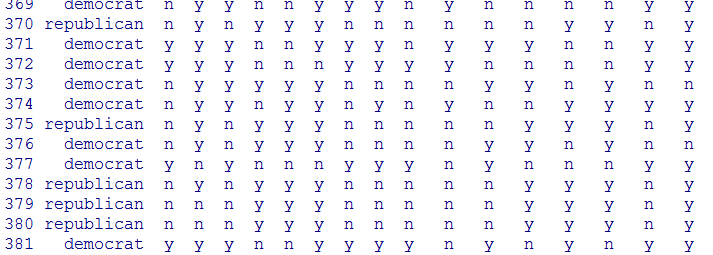
return(fraction\_correct)

}

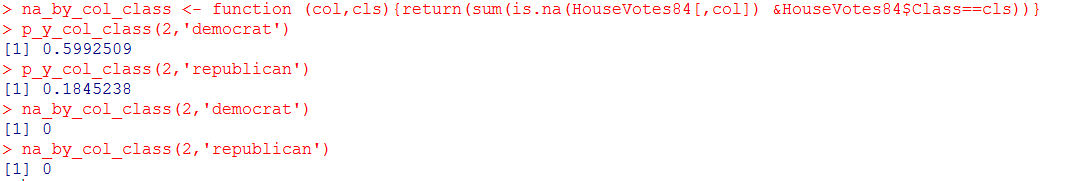
**Before Impute Data records from 370 to 381:**



**After Imputing Data records from 370 to 381:**



**The following image shows that there are zero NA records:**



**Dived data to training and test data sets, the training data set will be used to train the algorithm and produce a predictive model. The effectiveness of the model will then be tested using the test dataset.**

HouseVotes84[,"train"] <- ifelse(runif(nrow(HouseVotes84))<0.80,1,0)

trainColNum <- grep("train",names(HouseVotes84))

trainHouseVotes84 <- HouseVotes84[HouseVotes84$train==1,-trainColNum]

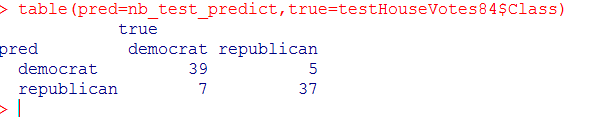
testHouseVotes84 <- HouseVotes84[HouseVotes84$train==0,-trainColNum]

**Invoke naïve Bayes method**:

nb\_model <- naiveBayes(Class~.,data = trainHouseVotes84)

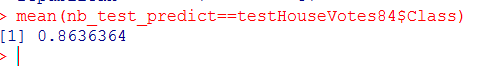
nb\_test\_predict <- predict(nb\_model,testHouseVotes84[,-1])

table(pred=nb\_test\_predict,true=testHouseVotes84$Class)



**The true values are in columns and the predicted values in rows**.

**The algorithm has classified 39 out of 46(i.e. 39+7) Democrats and 37 out of 42 Republicans (i.e. 37 +5).**



**The above mean in the image represents fraction of correct effectiveness. With value 0.8636364 To calculate effectiveness of prediction, we need to run. The following function returns vector of containing proportions of the correct predictions.**

nb\_multiple\_runs <- function(train\_fraction,n)

{

fraction\_correct <- rep(NA,n)

for(i in 1:n){HouseVotes84[,"train"] <- ifelse(runif(nrow(HouseVotes84))<train\_fraction,1,0)

trainColNum <- grep("train",names(HouseVotes84))

trainHouseVotes84 <- HouseVotes84[HouseVotes84$train == 1,-trainColNum]

testHouseVotes84 <- HouseVotes84[HouseVotes84$train ==0,-trainColNum]

nb\_model <- naiveBayes(Class~.,data = trainHouseVotes84)

nb\_test\_predict <- predict(nb\_model,testHouseVotes84[,-1])

fraction\_correct[i] <- mean(nb\_test\_predict == testHouseVotes84$Class)

}

return(fraction\_correct)

}

**By following code will perform 20 runs with the same training fraction (0.8).**

#20 runs, 80% of data randomly selected for training set in each run

fraction\_correct\_predictions <- nb\_multiple\_runs(0.8,20)

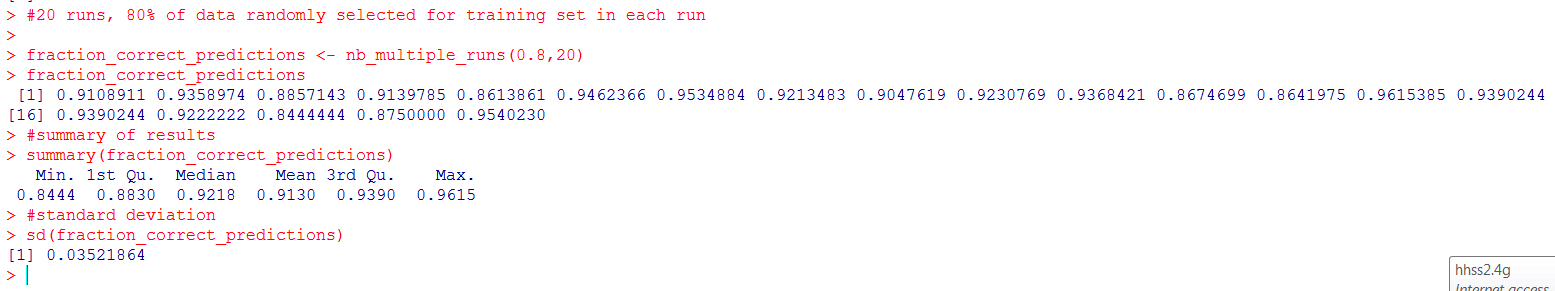
fraction\_correct\_predictions

#summary of results

summary(fraction\_correct\_predictions)

#standard deviation

sd(fraction\_correct\_predictions)



**The above result show that the outcome of the runs is quite close together, in the 0.85 to 0.95 range with a standard deviation of 0.025. The effective ness is close to 100% and SD is low we can say that** **Naive Bayes performing better.**

**Conclusion:** This assignment helped us to learn about fundamentals of the Naïve Bayes theorem**.** By working on this assignment learned about the implementation and usage of the Naïve Bayes theorem. Understood about the fundamentals of probability and different types of probability like conditional probability and Joint probability. By implementing Naïve Bayes theorem for predicting the spam and ham SMS messages learned different packages and functions present in R studio. By working on the data set for predicting election results gained more exposure about R. By working on data learned about text processing and plotting methods. Gained exposure for writing functions in R. Learned about imputation of data and preparing training and dataset. Did measured the effectiveness of the Naïve Bayes algorithm. We got exposure in interpreting the out of the result. Due this exposure regarding R, using Naïve Bayes theorem in R and working on data sets, we could able to work on production problems by using Machine Learning.

**References:**

Lantz, B. (n.d.). *Machine Learning with R Second Edition* (second ed.). Copyright © 2015 Packt Publishing. Retrieved from <http://books.tarsoit.com/Machine%20Learning%20with%20R%20-%20Second%20Edition.pdf>

Awati, K. (n.d.). A gentle introduction to Naïve Bayes classification using R. Retrieved from https://eight2late.wordpress.com/2015/11/06/a-gentle-introduction-to-naive-bayes-classification-using-r/