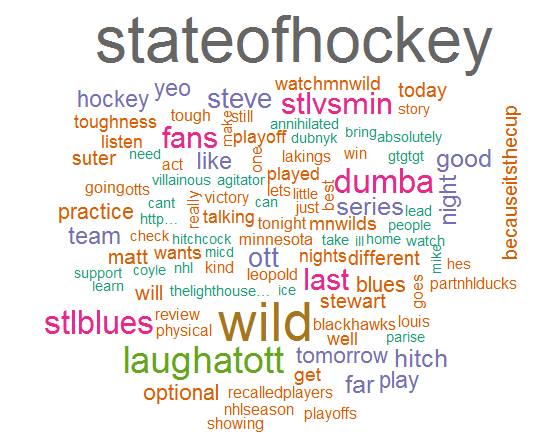
**Text Mining in R using Twitter ®**

Mining text data involves looking for frequent words and finding associations between words in text documents. A tweet is very simple text document created by a Twitter user. Using several R packages we can search for tweets that meet a specified criterion and download them into R. We can then use text cleaning capabilities from several packages in R to “clean” the tweets so that we can look for frequent words and look for associations between words. One of common displays used to represent the frequent words found text documents is a word cloud. Below is a word cloud created from tweets about the Minnesota Wild following their dominant performance in Game 3 vs. the St. Louis Blues.



Packages in R for Twitter mining and text mining in general:

* tm, TwitteR, stringr, wordcloud, RCurl, RJSONIO, igraph

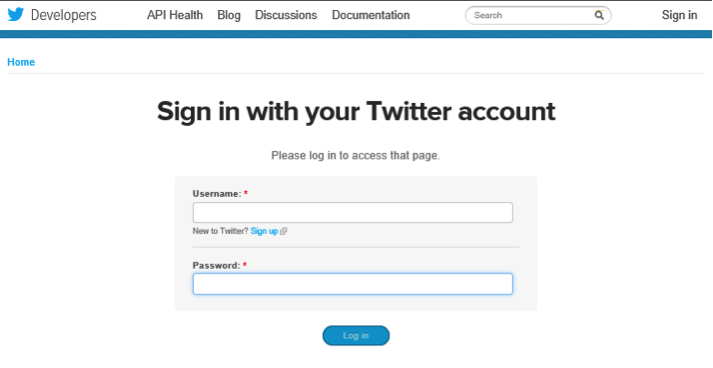
The first four packages are the ones I used directly to create the word cloud shown above. The package TwitteR allows you to search for tweets and download them into R. The tm and stringr packages contain functions to clean tweets, stripping off things like web addresses, re-tweets (RT) and hashtags (#) and at (@) nomenclature. The package wordcloud obviously is used to create word clouds like the one shown above.

In order to be search for and download tweets from Twitter, first need to get key codes from Twitter. The process for gaining access to tweets follow the procedure outlined below.

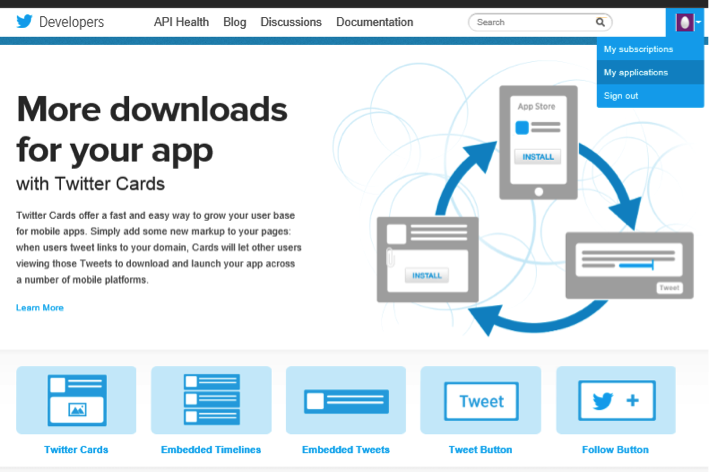
1. Download and load the package twitteR.
2. In order to extract tweets, you will need a Twitter application and hence a Twitter account. If you don’t have a Twitter account you will need to create one.
3. Use your Twitter login ID and password to sign in at Twitter Developers (<https://apps.twitter.com/> ).

**1.  Steps to Create a Twitter Application**

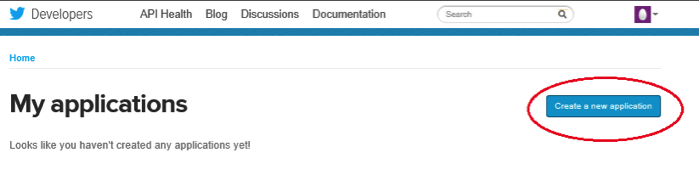
a.  Navigate to My Applications in the upper right hand corner.

[](http://blog.credera.com/wp-content/uploads/2014/05/Twitter1.png)

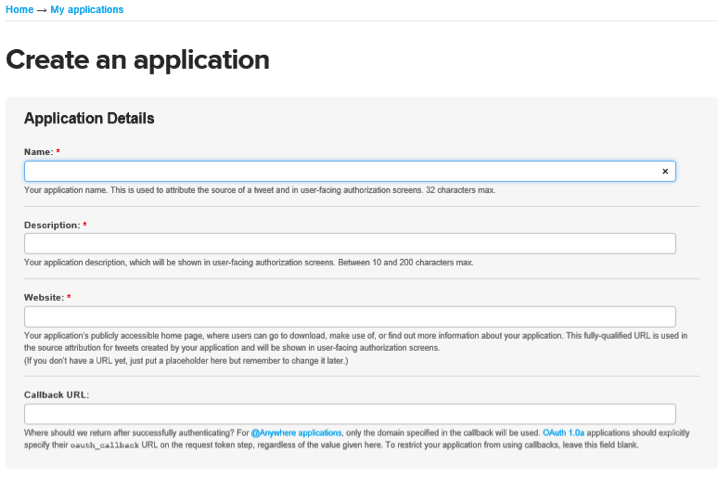
b.  Navigate to My Applications in the upper right hand corner.

[](http://blog.credera.com/wp-content/uploads/2014/05/Twitter2.png)

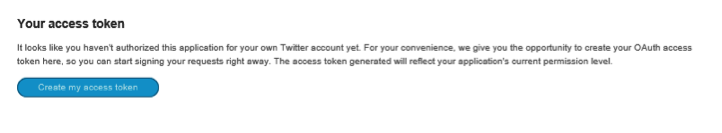
c.  Create a new application.

[](http://blog.credera.com/wp-content/uploads/2014/05/TwitterC.png)

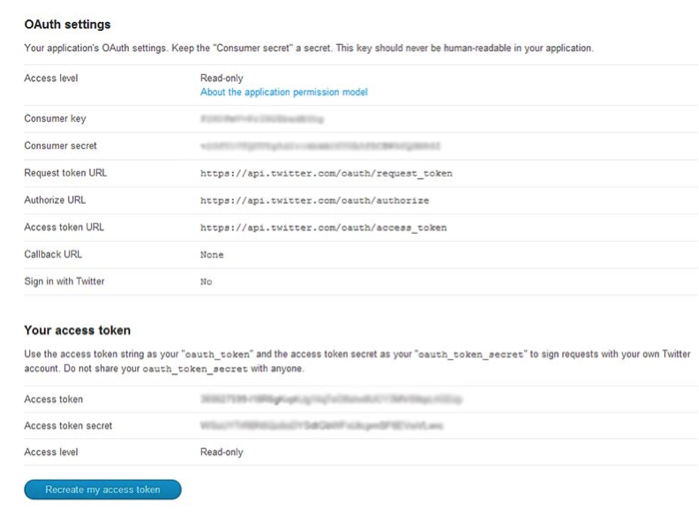
d.  Fill out the new app form. Names should be unique, i.e., no one else should have used this name for their Twitter app. Givea brief description of the app. You can change this later on if needed. Enteryour website or blog address, I just entered my http://course1.winona.edu/bdeppa website and it worked. Callback URL can be left blank. Once you’ve done this, make sure you’ve read the “Developer Rules Of The Road” blurb, check the “Yes, I agree” box, fill in the CAPTCHA and click the “Create Your Twitter Application” button.



e.  Scroll down and click on “Create my access token” button.

[](http://blog.credera.com/wp-content/uploads/2014/05/Twitter4.png)

f.  Note the values of consumer key and consumer secret and keep them handy for future use. You should keep these secret. If anyone was to get these keys, they could effectively access your Twitter account.



Once you have the four keys ghosted out in the screenshot above you can enter them into a function in twitteR package to gain access to Twitter tweets. In R do the following:

> api.key = “3478239q64789uifghdfahdgfgsd” 🡨 Consumer key above.

> api.secret = “dfhjaghsfjkg4jk35ry8234789326893” 🡨 Consumer secret above.

> access.token = “dayeu6237467tysduiaty783467821394” 🡨 Access token above.

> token.secret = “324238yauisghgfhagdhahsdg” 🡨 Access token secret above.

Also change the Access level to Read, Write and Access by changing Permissions under that tab for application. After completing all of these steps, run the following line of code in R.

> setup\_twitter\_oauth(api.key,api.secret,access.token,token.secret)

You can now use different tweet search capabilities in the twitteR package to download tweets.

> mnwild.tweets = searchTwitter("mnwild",n=1000,lang="en")

> mnwild.tweets[1:5]

[[1]]

[1] "BITWLFC33: RT @Russostrib: Devan Dubnyk, #mnwild vow to redeem itself from Game 4 debacle vs. #stlblues. My ADVANCE on tonight's game&gt;&gt;&gt; http://t.co/D…"

[[2]]

[1] "joleeo613: RT @mnwild: Game 5 is tonight! #STLvsMIN http://t.co/3HBGbY2RQX"

[[3]]

[1] "mnjaime: @mnwild Let's Go Wild!!!! Those of us that are true fans know you've got this. #GoGetThatCup \xed��\xed�\u0080"

[[4]]

[1] "gingervitous1: RT @STLBluesSociety: Who wins Game 5? \nRT for the @StLouisBlues \nFAV for the @mnwild http://t.co/vCaWsw09dF"

[[5]]

[1] "DMaetzMedia: How serious is Minnesota about it's hockey? St. Paul mayor tweets the @MNWild \nLove the passion. http://t.co/1A3xwZg2zZ #hockey"

Obviously there is lots of garbage in the tweets that would not contribute to a meaningful representation of the sentiment in these tweets.

An optional, but potentially useful, step in this process is to convert the tweets into a data frame that will not only contain the content of the tweet, but also other information about it.

> mnwild.df = twListToDF(mnwild.tweets)

> names(mnwild.df)

[1] "text" "favorited" "favoriteCount" "replyToSN" "created"

[6] "truncated" "replyToSID" "id" "replyToUID" "statusSource"

[11] "screenName" "retweetCount" "isRetweet" "retweeted" "longitude"

[16] "latitude"

> mnwild.text = mnwild.df$text  
> mnwild.text[1:5]

[1] "RT @Russostrib: Devan Dubnyk, #mnwild vow to redeem itself from Game 4 debacle vs. #stlblues. My ADVANCE on tonight's game&gt;&gt;&gt; http://t.co/D…"

[2] "RT @mnwild: Game 5 is tonight! #STLvsMIN http://t.co/3HBGbY2RQX"

[3] "@mnwild Let's Go Wild!!!! Those of us that are true fans know you've got this. #GoGetThatCup \xed��\xed�\u0080"

[4] "RT @STLBluesSociety: Who wins Game 5? \nRT for the @StLouisBlues \nFAV for the @mnwild http://t.co/vCaWsw09dF"

[5] "How serious is Minnesota about it's hockey? St. Paul mayor tweets the @MNWild \nLove the passion. http://t.co/1A3xwZg2zZ #hockey"

Here are some functions that make use of base R, tm, and stringr functions to help clean up tweets. The CleanTweets function is probably the easiest to tweak in order the clean up the tweets you are working with. You can also do some additional clean up at the end of running of these functions using the str\_replace\_all function.

The basic form of the call is shown below:

> tweets.clean2 = str\_replace\_all(tweets.clean1,”string to replace”,””)

You can also add lines like this to the CleanTweets function below to improve it’s performance for a given set of tweets. You might also want to modify it by removing some of the lines in the function below.  
  
Also you will get some tweeters who spam the same tweet multiple times, you remove these annoyances by using the unique function in R.

> tweets.clean3 = unique(tweets.clean2)

**Tweet cleaners:** (in my opinion the CleanTweets function tends to work better and is definitely easier to play around with.)

clean.text = function(some\_txt){

some\_txt = gsub("(RT|via)((?:\\b\\W\*@\\w+)+)", "", some\_txt)

some\_txt = gsub("@\\w+", "", some\_txt)

some\_txt = gsub("[[:punct:]]", "", some\_txt)

some\_txt = gsub("[[:digit:]]", "", some\_txt)

some\_txt = gsub("http\\w+", "", some\_txt)

some\_txt = gsub("[ \t]{2,}", "", some\_txt)

some\_txt = gsub("^\\s+|\\s+$", "", some\_txt)

some\_txt = gsub("amp", "", some\_txt)

# define "tolower error handling" function

try.tolower = function(x)

{

y = NA

try\_error = tryCatch(tolower(x), error=function(e) e)

if (!inherits(try\_error, "error"))

y = tolower(x)

return(y)

}

some\_txt = sapply(some\_txt, try.tolower)

some\_txt = some\_txt[some\_txt != ""]

names(some\_txt) = NULL

return(some\_txt)

}

CleanTweets = function(tweets) {

tweets = str\_replace\_all(tweets,"http://t.co/[a-z,A-Z,0-9]{8}","")

tweets = str\_replace\_all(tweets,"http://t.co/[A-Z,a-z,0-9]","")

tweets = str\_replace\_all(tweets,"https://t.co/[a-z,A-Z,0-9]{8}","")

tweets = str\_replace\_all(tweets,"\\n","")

tweets = str\_replace\_all(tweets,"RT @[a-z,A-Z,0-9]\*: ","")

tweets = str\_replace\_all(tweets,"RT [a-z,A-Z,0-9]\*: ","")

tweets = str\_replace\_all(tweets,"#[a-z,A-Z]\*","")

tweets = str\_replace\_all(tweets,"@[a-z,A-Z]\*","")

tweets = str\_replace\_all(tweets,"&gt","")

tweets = str\_replace\_all(tweets,":","")

return(tweets)

}

In order to tabulate the popular words found in the cleaned tweets we need to create a Term Document Matrix (TDM). A TDM has unique “words” as rows and documents (tweets) as columns. The columns will contain counts of the number of times each “word” occurs in the document. In order to create the TDM you need to create a Corpus which a data object the tm package uses in order to create the TDM. The function does one last text clean up and creates the Corpus.

CleanCorpus = function(tweets) {

tweetCorpus = Corpus(VectorSource(tweets))

tweetCorpus = tm\_map(tweetCorpus,content\_transformer(tolower))

tweetCorpus = tm\_map(tweetCorpus,removePunctuation)

tweetCorpus = tm\_map(tweetCorpus,removeNumbers)

tweetCorpus = tm\_map(tweetCorpus,removeWords,stopwords("english"))

return(tweetCorpus)

}

We are now ready clean up the tweets collected, create the Corpus & TDM, then tabulate word counts and display them graphically.

> mnwild.tweets = searchTwitter("mnwild",n=1500,lang="en")

> mnwild.tweets[1:5]

[[1]]

[1] "BITWLFC33: RT @Russostrib: Devan Dubnyk, #mnwild vow to redeem itself from Game 4 debacle vs. #stlblues. My ADVANCE on tonight's game&gt;&gt;&gt; http://t.co/D…"

[[2]]

[1] "joleeo613: RT @mnwild: Game 5 is tonight! #STLvsMIN http://t.co/3HBGbY2RQX"

[[3]]

[1] "mnjaime: @mnwild Let's Go Wild!!!! Those of us that are true fans know you've got this. #GoGetThatCup \xed��\xed�\u0080"

[[4]]

[1] "gingervitous1: RT @STLBluesSociety: Who wins Game 5? \nRT for the @StLouisBlues \nFAV for the @mnwild http://t.co/vCaWsw09dF"

[[5]]

[1] "DMaetzMedia: How serious is Minnesota about it's hockey? St. Paul mayor tweets the @MNWild \nLove the passion. http://t.co/1A3xwZg2zZ #hockey"

The body of tweet is contained in the variable mnwild.df$text within the data frame mnwild.df created above.

Inspect the last 11 tweets…

> mnwild.df$text[990:1000]

[1] "RT @mnwild: Dubnyk: \"That's a forgettable one. Games can go like that sometimes. We're in the same situation if we had lost 1-0 in overtime…"

[2] "...and it shall rise tomorrow. #roadiceadvantage #gowild! https://t.co/YdSZpNp6wH"

[3] "Calm and cool despite a loss. Such an awesome dude! https://t.co/2mNkZHqb2L"

[4] "At the start of this series I think 90% of us would have been happy with a 2-2 after 4. If not... your lying to yourself. #mnwild"

[5] "RT @mnwild: Dubnyk: \"The sun came up this morning.\" #mnwild"

[6] "RT @Russostrib: Yeo said safe to assume there will be lineup changes for Game 5 #mnwild"

[7] "@mnwild Can Dubnyk write that on a giant piece of paper for me and frame it? I need to remember this! #thedubabides #cominbackswinging"

[8] "RT @mnwild: Dubnyk: \"We know what we're capable of. We didn't do that last night...We'll bounce back.\" #mnwild"

[9] "RT @mnwild: Dubnyk: \"We know what we're capable of. We didn't do that last night...We'll bounce back.\" #mnwild"

[10] "RT @Russostrib: Dubnyk: \"it's a beautiful day here in Minnesota and the sun came up this morning.\" He has no doubt #mnwild responds. \"Zero …"

[11] "RT @mnwild: Dubnyk: \"The sun came up this morning.\" #mnwild"

> wild.clean1 = clean.text(mnwild.df$text)

> wild.clean1[1:10]

[1] "devan dubnyk mnwild vow to redeem itself from gamedebacle vs stlblues my advance on tonights gamegtgtgt …"

[2] "gameis tonight stlvsmin"

[3] "lets go wild those of us that are true fans know youve got this gogetthatcup"

[4] "who wins game\nrt for the\nfav for the"

[5] "how serious is minnesota about its hockey st paul mayor tweets the\nlove the passionhockey"

[6] "its rally cap time for thetonight"

[7] "awesome coworkersthat know i love mnwild im keeping the mug"

[8] "gameis tonight stlvsmin"

[9] "the pressure remains on the blues mnwild coach mike yeo says story by"

[10] "its rally cap time for thetonight"

Try CleanTweets instead

> wild.clean2 = CleanTweets(mnwild.df$text)

> wild.clean2[1:10]

[1] "Devan Dubnyk, vow to redeem itself from Game 4 debacle vs. . My ADVANCE on tonight's game;;; …"

[2] "Game 5 is tonight! QX"

[3] " Let's Go Wild!!!! Those of us that are true fans know you've got this.

[4] "Who wins Game 5? RT for the FAV for the dF"

[5] "How serious is Minnesota about it's hockey? St. Paul mayor tweets the Love the passion. zZ "

[6] "It's rally cap time for the tonight! 0G af"

[7] "Awesome co-workers 411 that know I love . I'm keeping the mug. tg"

[8] "Game 5 is tonight! QX"

[9] "The pressure remains on the Blues, coach Mike Yeo says. Story by 6g"

[10] "It's rally cap time for the tonight! 0G af"

Try running CleanTweets on the results from clean.text.

> wild.clean3 = CleanTweets(wild.clean1)

> wild.clean3[1:10]

[1] "devan dubnyk mnwild vow to redeem itself from gamedebacle vs stlblues my advance on tonights gamegtgtgt …"

[2] "gameis tonight stlvsmin"

[3] "lets go wild those of us that are true fans know youve got this gogetthatcup"

[4] "who wins gamert for thefav for the"

[5] "how serious is minnesota about its hockey st paul mayor tweets thelove the passionhockey"

[6] "its rally cap time for thetonight"

[7] "awesome coworkersthat know i love mnwild im keeping the mug"

[8] "gameis tonight stlvsmin"

[9] "the pressure remains on the blues mnwild coach mike yeo says story by"

[10] "its rally cap time for thetonight"

Replace mnwild reference in the tweets and we could do the same for stlblues and stlvsmin also.

> wild.clean5 = str\_replace\_all(wild.clean3,"mnwild","")

> wild.clean5[1:10]

[1] "devan dubnyk vow to redeem itself from gamedebacle vs stlblues my advance on tonights gamegtgtgt …"

[2] "gameis tonight stlvsmin"

[3] "lets go wild those of us that are true fans know youve got this gogetthatcup 攼㹤愼㸰戼㹤攼㹤戼㸸㠼㸰"

[4] "who wins gamert for thefav for the"

[5] "how serious is minnesota about its hockey st paul mayor tweets thelove the passionhockey"

[6] "its rally cap time for thetonight"

[7] "awesome coworkersthat know i love im keeping the mug"

[8] "gameis tonight stlvsmin"

[9] "the pressure remains on the blues coach mike yeo says story by"

[10] "its rally cap time for thetonight"

> wild.clean5 = str\_replace\_all(wild.clean5,"stlblues","")

> wild.clean5 = str\_replace\_all(wild.clean5,"stlvsmin","")

Though the tweets are still a bit messy and some of the relevant words might have gotten truncated/eliminated we will proceed to the next steps. Tweaking some of the text cleaning options with in the clean.text and CleanTweets functions may produce a better result for these particular tweets. To eliminate spammed tweets we can extract only the unique ones (this is optional).

> wild.unique = unique(wild.clean5)  
> length(wild.unique)

[1] 496

To create a word cloud we first need to create a “matrix” that contains all the unique “words” in our cleaned tweets as rows and columns indicating tweets (i.e. documents). The entries in the matrix will provide a count of the number times each “word” appears in the document. This is called a Term Document Matrix (TDM).

> wildCorpus = CleanCorpus(wild.unique)

> wildTDM = TermDocumentMatrix(wildCorpus)

> wildTDM

<<TermDocumentMatrix (terms: 1642, documents: 496)>>

Non-/sparse entries: 3521/810911

Sparsity : 100%

Maximal term length: 72

Weighting : term frequency (tf)

> wild.mat = as.matrix(wildTDM)

> wild.mat[301:320,1:10]

Docs

Terms 1 2 3 4 5 6 7 8 9 10

darling 0 0 0 0 0 0 0 0 0 0

day 0 0 0 0 0 0 0 0 0 0

days 0 0 0 0 0 0 0 0 0 0

dead 0 0 0 0 0 0 0 0 0 0

deal 0 0 0 0 0 0 0 0 0 0

dear 0 0 0 0 0 0 0 0 0 0

death 0 0 0 0 0 0 0 0 0 0

debacle 0 0 0 0 0 0 0 0 0 0

deciding 0 0 0 0 0 0 0 0 0 0

decision 0 0 0 0 0 0 0 0 0 0

decisive 0 0 0 0 0 0 0 0 0 0

declutter 0 0 0 0 0 0 0 0 0 0

dedication 0 0 0 0 0 0 0 0 0 0

deep 0 0 0 0 0 0 0 0 0 0

defend 0 0 0 0 0 0 0 0 0 0

demolished 0 0 0wild 0 0 0 0 0 0 0

despite 0 0 0 0 0 0 0 0 0 0

detwat 0 0 0 0 0 0 0 0 0 0

devan 1 0 0 0 0 0 0 0 0 0

dew 0 0 0 0 0 0 0 0 0 0

> word.freq = sort(rowSums(wild.mat),decreasing=T)

> word.freq[1:20]

game wild night get tonight

73 70 41 34 34

series will last win tomorrow

33 31 29 29 25

yeo blues dubnyk becauseitsthecup can

24 22 22 21 21

lets time just day know

20 20 19 18 18

> findAssocs(wildTDM,"dubnyk",.2)

dubnyk

sun 0.47

devan 0.42

came 0.37

dmyers 0.31

advance 0.30

beautiful 0.30

capable 0.30

gamegtgtgt 0.30

nightwell 0.30

redeem 0.30

vow 0.30

debacle 0.27

yet 0.27

know 0.26

morning 0.26

remember 0.24

afrom 0.21

begin 0.21

bench 0.21

came… 0.21

cominbackswinging 0.21

count 0.21

couple 0.21

darling 0.21

dear 0.21

doubt 0.21

dream 0.21

etal 0.21

fire 0.21

forgettable 0.21

frame 0.21

giant 0.21

gobacktoyourpreciousvikingsplease 0.21

heytwitter 0.21

overtime… 0.21

paper 0.21

perspective 0.21

ppl 0.21

pulling 0.21

reading 0.21

realized 0.21

responds 0.21

saying 0.21

situation 0.21

sometimes 0.21

spirits 0.21

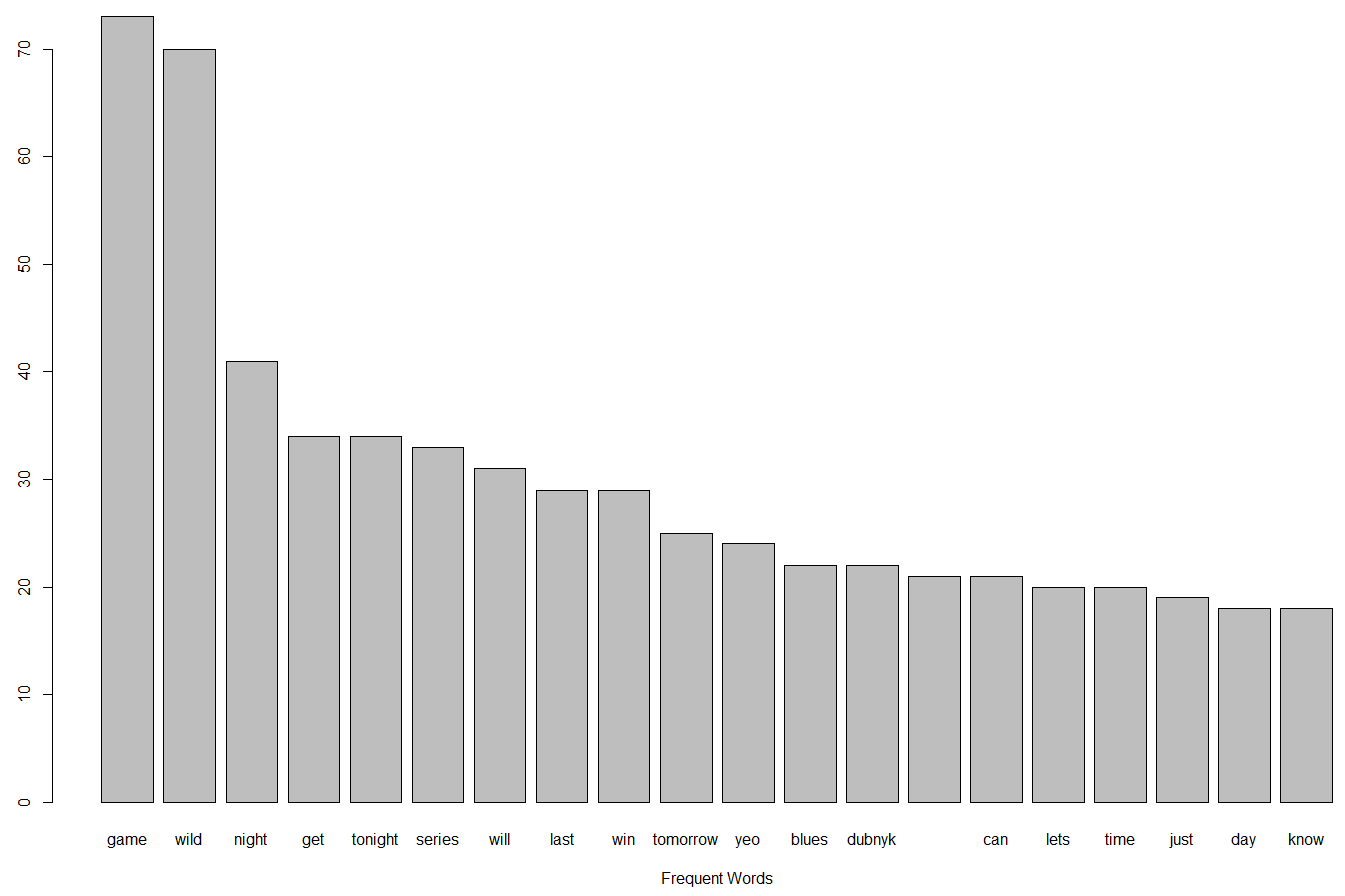
sprinkles 0.21

starts 0.21

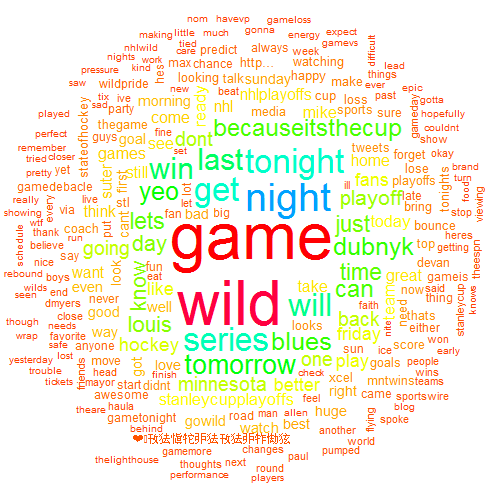
stops 0.21

stunk 0.21

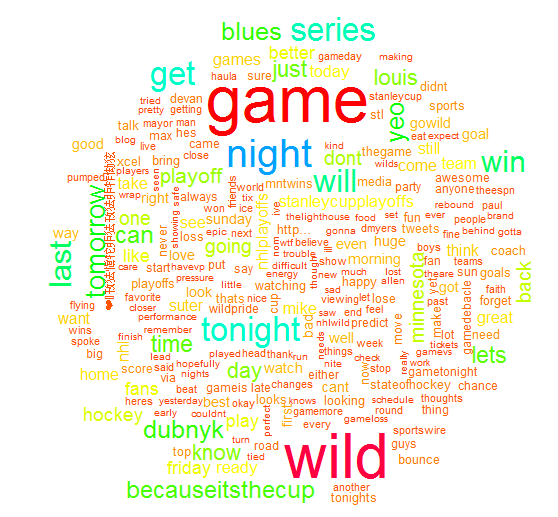
> barplot(word.freq[1:20],xlab=”Frequent Words”)



> wordcloud(words=names(word.freq),freq=word.freq,random.order=F,col=rainbow(1000))



> wordcloud(words=names(word.freq),freq=word.freq,random.order=T,col=rainbow(1000))



> word.freq.trimmed = word.freq[-c(1:6)]

> wordcloud(words=names(word.freq.trimmed),freq=word.freq.trimmed,  
+ random.order=T,col=rainbow(1000))

