> #set the working directory

> setwd("D:/Data Science project")

>

>

> #import the dataset in R

> data\_NFL <- read.csv("NFL\_SocialMedia\_sample\_data1.csv",head=T)

> head(data\_NFL)

content

1 NFL flexes Dallas Cowboys-Washington Redskins game http://t.co/Yim1aAzy

2 @special\_event32 redskins still suck

3 RG3 leads Redskins over Eagles 27-20 (The Associated Press) PHILADELPHIA (AP) -- With one http://t.co/U4TA4HRw

4 Correct me if I'm wrong, but #Giants can still get into playoffs if #Packers def #Vikings + #Redskins def #Cowboys? And NY wins of course.

5 RG3 leads Redskins over Eagles 27-20 http://t.co/UZqXBQoV

6 RT @\_2KnoMeIz2LuvMe: BREAKING NEWS - NFL - Cowboys-Redskins flexed to Sunday night game on Dec. 30; Packers-Vikings shifts to 4:25 pm ET

id tstamp

1 cbbbcf9395705611c3eeeffaa610a602 2012-12-24T09:51:43Z

2 9b50b8be10460eab6c0f6f3590067bd7 2012-12-24T09:52:19Z

3 77e1a37031884642b8d1bccad99516c6 2012-12-24T09:52:30Z

4 0d4f533e658b47eefecadee60b61278e 2012-12-24T09:52:37Z

5 a4a58402d1c33f85f3f38c0978255c7c 2012-12-24T09:53:16Z

6 83e6a52e1c43d23659d9916e1302daad 2012-12-24T09:53:48Z

profilelink

1 http://a0.twimg.com/profile\_images/415099904/ladainian-tomlinson\_normal.jpg

2 http://a0.twimg.com/profile\_images/2687360343/b74f0ade3eb5be4f92faf9e31d0a0a61\_normal.jpeg

3 http://a0.twimg.com/profile\_images/3000100432/4e63820c713a98befa84f7bb965154da\_normal.jpeg

4 http://a0.twimg.com/profile\_images/1103688472/jd\_normal.jpg

5 http://a0.twimg.com/profile\_images/2833068853/245284d72d32fffdb557f4ae5a854555\_normal.jpeg

6 http://a0.twimg.com/profile\_images/2992217644/28060208d1c3b2ad7eab9e1f513ee446\_normal.jpeg

screenname timezone

1 Fight4EveryYard Pacific Time (US & Canada)

2 \_jpappps Quito

3 CowboysPage Athens

4 jazadal London

5 lbgood122

6 Tone301 Central Time (US & Canada)

>

> #load the text mining package

> library(tm)

>

> # Build text corpus

> txt\_corpus <- Corpus(VectorSource(data\_NFL$content))

>

> #Inspect the corpus

> inspect(txt\_corpus[1:5])

A corpus with 5 text documents

The metadata consists of 2 tag-value pairs and a data frame

Available tags are:

create\_date creator

Available variables in the data frame are:

MetaID

[[1]]

NFL flexes Dallas Cowboys-Washington Redskins game http://t.co/Yim1aAzy

[[2]]

@special\_event32 redskins still suck

[[3]]

RG3 leads Redskins over Eagles 27-20 (The Associated Press) PHILADELPHIA (AP) -- With one http://t.co/U4TA4HRw

[[4]]

Correct me if I'm wrong, but #Giants can still get into playoffs if #Packers def #Vikings + #Redskins def #Cowboys? And NY wins of course.

[[5]]

RG3 leads Redskins over Eagles 27-20 http://t.co/UZqXBQoV

>

>

>

> #Perform Data Cleaning Operations

> clean\_corpus <- function(corpus)

+ {

+ corpus <- tm\_map(corpus, stripWhitespace)

+ #eliminate whitespace

+ corpus <- tm\_map(corpus, removePunctuation)

+ #remove punctuation

+ corpus <- tm\_map(corpus, removeNumbers)

+ #remove numbers

+ corpus <- tm\_map(corpus, tolower)

+ #Convert to lowercase

+ corpus <- tm\_map(corpus, removeWords, stopwords("english"))

+ #Remove Stopwords

+ return(corpus)

+ }

>

>

> mod\_corpus <- clean\_corpus(txt\_corpus)

> inspect(mod\_corpus[1:5])#Inspect refined corpus

A corpus with 5 text documents

The metadata consists of 2 tag-value pairs and a data frame

Available tags are:

create\_date creator

Available variables in the data frame are:

MetaID

[[1]]

nfl flexes dallas cowboyswashington redskins game httptcoyimaazy

[[2]]

specialevent redskins still suck

[[3]]

rg leads redskins eagles associated press philadelphia ap one httptcoutahrw

[[4]]

correct im wrong giants can still get playoffs packers def vikings redskins def cowboys ny wins course

[[5]]

rg leads redskins eagles httptcouzqxbqov

>

> #Create Document term matrix

> dtm <- DocumentTermMatrix(mod\_corpus)

> inspect(dtm[1:10,1000:1010])

A document-term matrix (10 documents, 11 terms)

Non-/sparse entries: 2/108

Sparsity : 98%

Maximal term length: 13

Weighting : term frequency (tf)

Terms

Docs ðÿž„ðÿž…ðÿž„ ðÿž…ðÿž„ðÿž\u0081 eagle eagles eaglesdaily eaglesnflfans earlier early earned

1 0 0 0 0 0 0 0 0 0

2 0 0 0 0 0 0 0 0 0

3 0 0 0 1 0 0 0 0 0

4 0 0 0 0 0 0 0 0 0

5 0 0 0 1 0 0 0 0 0

6 0 0 0 0 0 0 0 0 0

7 0 0 0 0 0 0 0 0 0

8 0 0 0 0 0 0 0 0 0

9 0 0 0 0 0 0 0 0 0

10 0 0 0 0 0 0 0 0 0

Terms

Docs earrings ears

1 0 0

2 0 0

3 0 0

4 0 0

5 0 0

6 0 0

7 0 0

8 0 0

9 0 0

10 0 0

>

> ##Determine the TF and tfxidf

> dtm\_tfxidf <- weightTfIdf(dtm)

> inspect(dtm\_tfxidf[1:10, 1000:1010])

A document-term matrix (10 documents, 11 terms)

Non-/sparse entries: 2/108

Sparsity : 98%

Maximal term length: 13

Weighting : term frequency - inverse document frequency (normalized) (tf-idf)

Terms

Docs ðÿž„ðÿž…ðÿž„ ðÿž…ðÿž„ðÿž\u0081 eagle eagles eaglesdaily eaglesnflfans earlier early earned

1 0 0 0 0.0000000 0 0 0 0 0

2 0 0 0 0.0000000 0 0 0 0 0

3 0 0 0 0.4659224 0 0 0 0 0

4 0 0 0 0.0000000 0 0 0 0 0

5 0 0 0 0.9318449 0 0 0 0 0

6 0 0 0 0.0000000 0 0 0 0 0

7 0 0 0 0.0000000 0 0 0 0 0

8 0 0 0 0.0000000 0 0 0 0 0

9 0 0 0 0.0000000 0 0 0 0 0

10 0 0 0 0.0000000 0 0 0 0 0

Terms

Docs earrings ears

1 0 0

2 0 0

3 0 0

4 0 0

5 0 0

6 0 0

7 0 0

8 0 0

9 0 0

10 0 0

>

> #use kmeans

> m <- as.matrix(dtm\_tfxidf)

> rownames(m) <- 1:nrow(m)

>

> #normalize the vectors

> norm\_eucl <- function(m) m/apply(m, MARGIN=1, FUN=function(x) sum(x^2)^.5)

> m\_norm <- norm\_eucl(m)

>

> ##Cluster the data into 10 clusters

> set.seed(100)

> km <- kmeans(m\_norm, 10)

> table(km$cluster)

1 2 3 4 5 6 7 8 9 10

44 47 28 52 182 41 39 191 57 1319

>

> ##Create Cluster\_out.csv

> x <- data.frame(Log = data\_NFL$content, Cluster = km$cluster)

> write.csv(x, file = "D:/Data Science project/ClusterOut.csv", row.names=TRUE)

>

> data1 <- read.csv("D:/Data Science project/ClusterOut.csv")

>

>

>

> #Find top 5 words in 10 clusters

>

> for(k in 1:length(km$withinss))

+

+ {

+ wordstodocument <- TermDocumentMatrix(mod\_corpus)

+ wordsmatrix <-as.matrix(wordstodocument)

+ word\_freq<- sort(rowSums(wordsmatrix[km$cluster == k, ]),decreasing = TRUE)

+ count <- as.vector(unlist(attributes(word\_freq)))

+ freqdisp <-data.frame(word\_freq, count, stringsAsFactors = FALSE)

+

+

+

+ if (k == 1)

+ {

+ x <- data.frame(k, length(which(data1$Cluster == k )), freqdisp$word\_freq[1:5],

+ freqdisp$count[1:5], as.numeric(rownames(x))[1:5])

+ colnames(x) = c("Loggroup", "Logcount", "word Count", "Top Words", "Counter")

+

+ }

+

+ else

+

+ {

+ y <- data.frame(k, length(which(data1$Cluster == k )), freqdisp$word\_freq[1:5],

+ freqdisp$count[1:5], as.numeric(rownames(x))[1:5])

+ colnames(y) = c("Loggroup", "Logcount", "word Count", "Top Words", "Counter")

+ x <- rbind(x, y)

+ }

+

+ }

>

> #output top 5 words in each cluster to Topwords.csv

> write.csv(x, file = "D:/Data Science project/Topwords.csv", row.names=FALSE)

>

>

>

> #load wordcloud package

> library(wordcloud)

>

>

> ##Perform visualization of clusters using wordcloud

>

> for (k in 1:10)

+ {

+ print(c("WordCount for Cluster", k), quote=FALSE)

+ capture.output( file = "D:/Data Science project/tests.doc", append = TRUE)

+ colors <- brewer.pal(8,"Dark2")

+ wordcloud(mod\_corpus[km$cluster==k], min.freq=5,

+ max.words=20,random.order=FALSE)

+

+ }

[1] WordCount for Cluster 1

[1] WordCount for Cluster 2

[1] WordCount for Cluster 3

[1] WordCount for Cluster 4

[1] WordCount for Cluster 5

[1] WordCount for Cluster 6

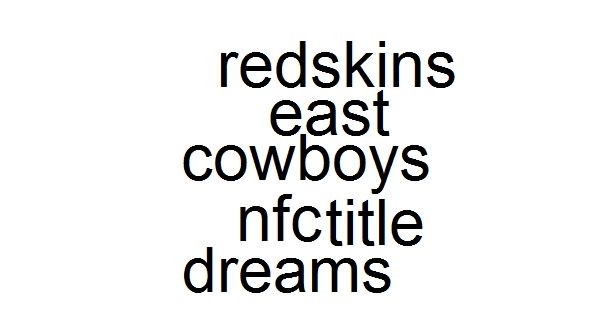
[1] WordCount for Cluster 7

[1] WordCount for Cluster 8

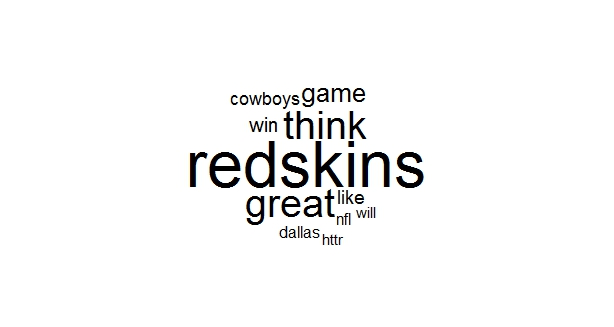
[1] WordCount for Cluster 9

[1] WordCount for Cluster 10

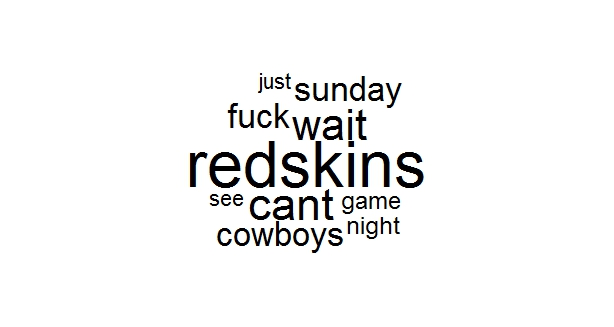
Word count for Cluster1



Word count for Cluster2



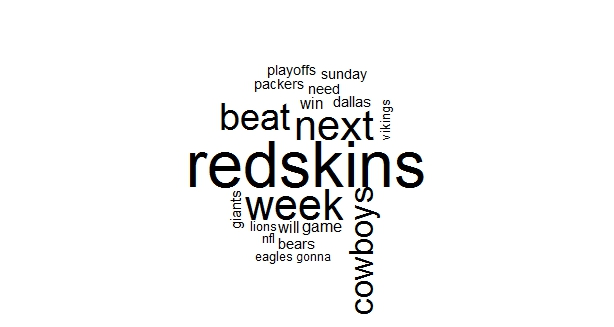
Word count for Cluster3



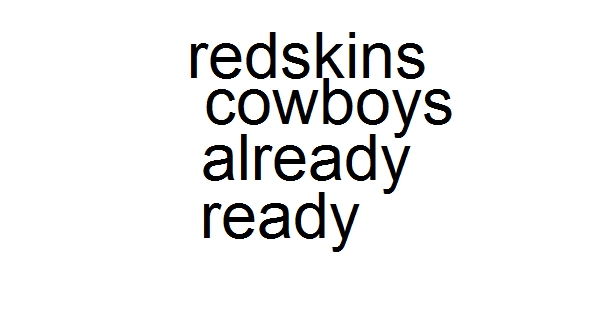
Word count for Cluster4



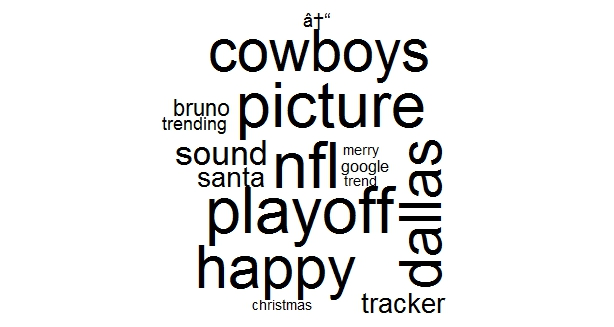
Word count for Cluster5



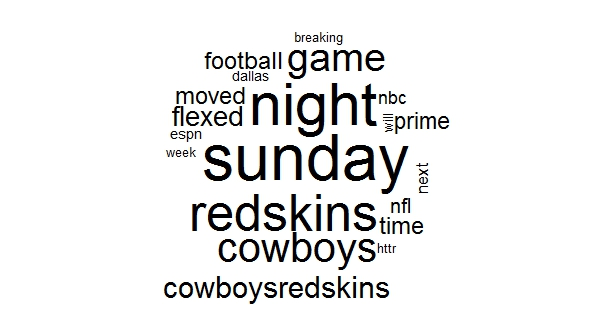
Word count for Cluster6



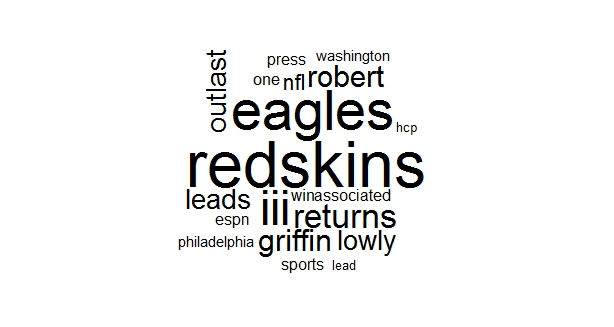
Word count for Cluster7



Word count for Cluster8



Word count for Cluster9



Word count for Cluster10

