Word Cloud and Cogs

#Ref : Code chunks taken from tidytext class excercises

if (!require(tidytext)) {install.packages("tidytext")}

## Loading required package: tidytext

if (!require(tibble)) {install.packages("tibble")}

## Loading required package: tibble

if (!require(tidyverse)) {install.packages("tidyverse")}

## Loading required package: tidyverse

## ── Attaching packages ────────────────────────────────────────────────────────────────────────────────────────── tidyverse 1.2.1 ──

## ✔ ggplot2 3.1.1 ✔ purrr 0.3.2  
## ✔ tidyr 0.8.3 ✔ dplyr 0.8.1  
## ✔ readr 1.3.1 ✔ stringr 1.4.0  
## ✔ ggplot2 3.1.1 ✔ forcats 0.4.0

## ── Conflicts ───────────────────────────────────────────────────────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()

if (!require(wordcloud)) {install.packages("wordcloud")}

## Loading required package: wordcloud

## Loading required package: RColorBrewer

if (!require(ggplot2)) {install.packages("ggplot2")}  
if (!require(igraph)) {install.packages("igraph")}

## Loading required package: igraph

##   
## Attaching package: 'igraph'

## The following objects are masked from 'package:dplyr':  
##   
## as\_data\_frame, groups, union

## The following objects are masked from 'package:purrr':  
##   
## compose, simplify

## The following object is masked from 'package:tidyr':  
##   
## crossing

## The following object is masked from 'package:tibble':  
##   
## as\_data\_frame

## The following objects are masked from 'package:stats':  
##   
## decompose, spectrum

## The following object is masked from 'package:base':  
##   
## union

Read swiggy and zomato files

swiggy <- readLines('data/swiggy.txt')  
zomato <- readLines('data/zomato.txt')

Build document term matrix

build\_dtm <- function(corpus) {  
 df = data\_frame(text = corpus) #create dataframe  
 df\_tokens = df %>%   
 mutate(doc = row\_number()) %>%   
 unnest\_tokens(word, text) %>%   
 anti\_join(stop\_words) %>%   
 group\_by(doc) %>%   
 count(word, sort=TRUE)  
   
 df\_counts = df\_tokens %>% rename(value = n)  
 dtm = df\_counts %>% cast\_sparse(doc, word, value)  
   
 # order rows and colms putting max mass on the top-left corner of the DTM  
 colsum = apply(dtm, 2, sum)   
 col.order = order(colsum, decreasing=TRUE)  
 row.order = order(rownames(dtm) %>% as.numeric())  
 dtm1 = dtm[row.order, col.order]  
 return(dtm1)   
}  
  
swiggy\_dtm = build\_dtm(swiggy)

## Warning: `data\_frame()` is deprecated, use `tibble()`.  
## This warning is displayed once per session.

## Joining, by = "word"

zomato\_dtm = build\_dtm(zomato)

## Joining, by = "word"

Plot Word Clouds

build\_wordcloud <- function(dtm) {  
 if (ncol(dtm) > 20000) {  
 chunk = round(ncol(dtm)/100)  
 a = rep(chunk,99)  
 b = cumsum(a)  
 rm(a)  
 b = c(0,b,ncol(dtm))  
   
 ss.col = c(NULL)  
 for (i in 1:(length(b)-1)) {  
 tempdtm = dtm[,(b[i]+1):(b[i+1])]  
 s = colSums(as.matrix(tempdtm))  
 ss.col = c(ss.col,s)  
 }  
  
 tsum = ss.col  
 }  
 else {  
 tsum = apply(dtm, 2, sum)  
 }  
  
 tsum = tsum[order(tsum, decreasing = T)]  
 return (tsum)  
}  
  
tsum <- build\_wordcloud(swiggy\_dtm)  
wordcloud(names(tsum), tsum, #List of words and frequencies  
 scale = c(2.5, 0.5), #define scale  
 5, # min.freq of words to consider  
 max.words = 150, # max no of words to consider in word cloud  
 colors = brewer.pal(8, "Dark2"))

## Warning in wordcloud(names(tsum), tsum, scale = c(2.5, 0.5), 5, max.words =  
## 150, : pathetic could not be fit on page. It will not be plotted.

## Warning in wordcloud(names(tsum), tsum, scale = c(2.5, 0.5), 5, max.words =  
## 150, : money could not be fit on page. It will not be plotted.

title(sub = "Swiggy Tweets Word Cloud")

tsum <- build\_wordcloud(zomato\_dtm)  
wordcloud(names(tsum), tsum, #List of words and frequencies  
 scale = c(2.2, 0.5), #define scale  
 5, # min.freq of words to consider  
 max.words = 150, # max no of words to consider in word cloud  
 colors = brewer.pal(8, "Dark2"))   
 title(sub = "Zomato Tweets Word Cloud")

plot.barchart <- function(dtm) {  
 a0 = apply(dtm, 2, sum)  
 a1 = order(a0, decreasing = TRUE)  
 tsum = a0[a1]  
 return (tsum)  
}  
 # plot barchart for top tokens  
tsum <- plot.barchart(swiggy\_dtm)  
test = as.data.frame(round(tsum[1:15],0)) #max words to plot  
p = ggplot(test, aes(x = rownames(test), y = test[,])) +   
 geom\_bar(stat = "identity", fill = "Brown") +  
 geom\_text(aes(label = test[,]), vjust= -0.20) +   
 theme(axis.text.x = element\_text(angle = 90, hjust = 1)) +  
 xlab("Words") +  
 ylab("No of occurances") +  
 ggtitle("Top 15 words for Swiggy")  
plot(p)

tsum <- plot.barchart(zomato\_dtm)  
test = as.data.frame(round(tsum[1:15],0)) #max words to plot  
p = ggplot(test, aes(x = rownames(test), y = test[,])) +   
 geom\_bar(stat = "identity", fill = "Brown") +  
 geom\_text(aes(label = test[,]), vjust= -0.20) +   
 theme(axis.text.x = element\_text(angle = 90, hjust = 1)) +  
 xlab("Words") +  
 ylab("No of occurances") +  
 ggtitle("Top 15 words for Zomato")  
plot(p)

```{r create cog}

max\_center\_nodes <- 4

max\_vertices <- 5

distill.cog = function(dtm) {

dtm\_matrix = as.matrix(dtm) #convert dtm to matrix

adj.mat = t(dtm\_matrix) %\*% dtm\_matrix #Transpose the matrix

diag(adj.mat) = 0 #remove self word references from matrix

col\_sum = order(apply(adj.mat, 2, sum), decreasing = T) #order by sum

sum\_matrix = as.matrix(adj.mat[col\_sum[1:50], col\_sum[1:50]])

a = colSums(sum\_matrix) # get sum in vector

b = order(-a) #arrange in descending order

row\_col\_matrix = sum\_matrix[b, b] #create matrix with rows and columns

diag(row\_col\_matrix) = 0

word\_count = NULL

for (i in 1:max\_center\_nodes) {

thresh1 = row\_col\_matrix[i,][order(-row\_col\_matrix[i, ])[max\_vertices]]

row\_col\_matrix[i, row\_col\_matrix[i,] < thresh1] = 0

row\_col\_matrix[i, row\_col\_matrix[i,] > 0 ] = 1

word = names(row\_col\_matrix[i, row\_col\_matrix[i,] > 0])

row\_col\_matrix[(i+1):nrow(row\_col\_matrix), match(word,colnames(row\_col\_matrix))] = 0

word\_count = c(word\_count, word)

}

row\_col\_matrix1 = row\_col\_matrix[match(word\_count, colnames(row\_col\_matrix)), match(word\_count, colnames(row\_col\_matrix))]

order = colnames(row\_col\_matrix)[which(!is.na(match(colnames(row\_col\_matrix), colnames(row\_col\_matrix1))))] #remove NA rows

row\_col\_matrix2 = row\_col\_matrix1[match(order, colnames(row\_col\_matrix1)), match(order, colnames(row\_col\_matrix1))]

return (row\_col\_matrix2)

}

#plot swiggy cogs

dtm\_plot <- distill.cog(swiggy\_dtm)

graph <- graph.adjacency(dtm\_plot, mode = "undirected", weighted=T) #Create Network object

graph <- simplify(graph)

V(graph)$color[1:max\_center\_nodes] = "green" #central node color

V(graph)$color[max\_center\_nodes+1:length(V(graph))] = "pink" #vertex colors

graph = delete.vertices(graph, V(graph)[ degree(graph) == 0 ]) #delete empty vertices

plot(graph, layout = layout.kamada.kawai, main = "Swiggy words graph")

```

```{r zomato cogs}

dtm\_plot <- distill.cog(zomato\_dtm)

graph <- graph.adjacency(dtm\_plot, mode = "undirected", weighted=T) #Create Network object

graph <- simplify(graph)

V(graph)$color[1:max\_center\_nodes] = "green" #central node color

V(graph)$color[max\_center\_nodes+1:length(V(graph))] = "pink" #vertex colors

graph = delete.vertices(graph, V(graph)[ degree(graph) == 0 ]) #delete empty vertices

plot(graph, layout = layout.kamada.kawai, main = "Zomato words graph")

```