Module 6 Assignment 2

## R Markdown

1.Load the packages tidytext and tidyverse

library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.3 v purrr 0.3.4  
## v tibble 3.1.0 v dplyr 1.0.5  
## v tidyr 1.1.3 v stringr 1.4.0  
## v readr 1.4.0 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(tidytext)

1. Read-in the review dataset for Airbnb listings in Boston in R (the dataset is available on Canvas). Change the name of the variable id to comment\_id.

(20 points)

#read in the data  
reviews <- read\_csv("reviews-boston.csv")

##   
## -- Column specification --------------------------------------------------------  
## cols(  
## listing\_id = col\_double(),  
## id = col\_double(),  
## date = col\_character(),  
## reviewer\_id = col\_double(),  
## reviewer\_name = col\_character(),  
## comments = col\_character()  
## )

reviews<-reviews %>%  
select(listing\_id, comment\_id=id,date,reviewer\_id,reviewer\_name, comments)

1. Create a tidytext dataset of guests’ reviews - Tokenize by one-word-per-row and filter stop-words.

Pre-process the data (e.g. filter out stop words, words with more than 3 characters, numbers, whitespaces, etc.)

(20 points)

tidy <- reviews %>%  
 unnest\_tokens("word", comments) %>%  
 anti\_join(stop\_words)%>% #remove the stop words   
 # filter (!word %in% undesirable\_words) %>%   
 distinct() %>% #get rid of any duplicate records  
 filter(nchar(word) > 3) %>% #remove words with less than 3 characters  
 filter(   
 !str\_detect(word, "^\\b\\d+\\b"), #remove numbers  
 !str\_detect(word, "\\s+"), #remove white spaces  
 !str\_detect(word, "[^a-zA-Z]")) #remove special characters

## Joining, by = "word"

1. Load the package widyr.

library(widyr)

1. Use pairwise\_count() from the widyr package to count how many times each pair of words occurs together in the comments field. Explain the results.{it may take a few minutes to run this chunck of code}

(20 points)

word\_pairs <- tidy %>%  
 pairwise\_count(word, listing\_id, sort = TRUE)

## Warning: `distinct\_()` was deprecated in dplyr 0.7.0.  
## Please use `distinct()` instead.  
## See vignette('programming') for more help

## Warning: `tbl\_df()` was deprecated in dplyr 1.0.0.  
## Please use `tibble::as\_tibble()` instead.

word\_pairs

## # A tibble: 139,679,626 x 3  
## item1 item2 n  
## <chr> <chr> <dbl>  
## 1 clean stay 1772  
## 2 stay clean 1772  
## 3 boston stay 1770  
## 4 stay boston 1770  
## 5 location stay 1763  
## 6 stay location 1763  
## 7 nice stay 1729  
## 8 stay nice 1729  
## 9 host stay 1721  
## 10 stay host 1721  
## # ... with 139,679,616 more rows

“pairwise\_count” determines the number of times two words appear in the same document. In this case, a “document” is an individual comment. The words “clean” and “stay” appeared in the same comment 1772 times. The words “Boston” and “stay” appeared in the same comment 1770 times. The first bunch of rows seems to all the different combinations of “clean”, “stay”, “boston”, “location”, “nice”, and “host”.

1. Use pairwise\_cor() from the widyr package to calculate the correlation among words in the comments field (filter for at least relatively common words first n()>=2000).Explain the results.{it may take a few minutes to run this chunck of code}

(20 points)

word\_cors <- tidy %>%  
 group\_by(word) %>%  
 filter(n() >= 2000) %>%  
 pairwise\_cor(word, comment\_id, sort = TRUE)  
  
word\_cors

## # A tibble: 26,406 x 3  
## item1 item2 correlation  
## <chr> <chr> <dbl>  
## 1 distance walking 0.807  
## 2 walking distance 0.807  
## 3 transportation public 0.582  
## 4 public transportation 0.582  
## 5 highly recommend 0.482  
## 6 recommend highly 0.482  
## 7 respond quick 0.460  
## 8 quick respond 0.460  
## 9 walk minute 0.366  
## 10 minute walk 0.366  
## # ... with 26,396 more rows

Whenever “distance” or “walking” are mentioned, 80.7% of the time they occur together. Whenever “public” or “transportation” are mentioned, 58.2% of the time they occur together. Whenever “respond” or “quick” are mentioned, 48.2% of the time they occur together. “pairwise\_cor” goes through our data and determines the number of times each word occurs with every other word. The higher the number, the higher the proportion that they occur together.

1. load the packages igraph and ggraph

library (igraph)

## Warning: package 'igraph' was built under R version 4.0.5

##   
## 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

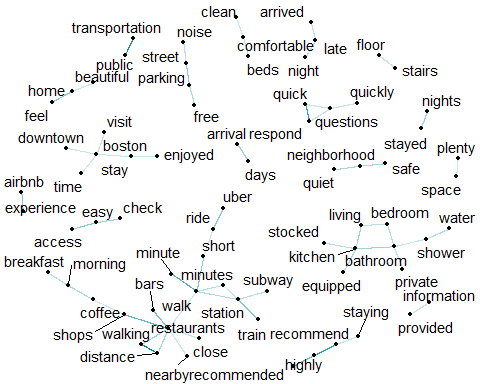
## The following object is masked from 'package:base':  
##   
## union

library (ggraph)

## Warning: package 'ggraph' was built under R version 4.0.5

1. Plot networks of these correlations among words (use the ggraph package and the layout=”fr”)).Explain the results.

set.seed(2016)  
  
word\_cors %>%  
 filter(correlation > .15) %>%  
 graph\_from\_data\_frame() %>%  
 ggraph(layout = "fr") +  
 geom\_edge\_link(aes(edge\_alpha = correlation), edge\_colour = "cyan4", show.legend = FALSE) +  
 geom\_node\_point( size = 1) +  
 geom\_node\_text(aes(label = name), repel = TRUE) +  
 theme\_void()



Some of the big clusters have the following themes: There’s one that has to do with transportation and neighborhood spots, with words like restaurant, walk, bars, coffee, morning, nearby, and subway. One refers to the living quarters, with words like bathroom, kitchen, shower, and bedroom. Some of the smaller clusters have words that always seem to go together, like “arrival days”, “public transportation”, and “clean comfortable beds”.

1. play with the visualization setting to create an alternative graph for the word correlations.

(20 points)

set.seed(2016)  
  
word\_cors %>%  
 filter(correlation > .15) %>%  
 graph\_from\_data\_frame() %>%  
 ggraph(layout = "fr") +  
 geom\_edge\_link(aes(edge\_alpha = abs(correlation), edge\_width = abs(correlation), color = correlation)) +  
 scale\_edge\_colour\_gradientn(limits = c(0, 1), colors = c("firebrick2", "dodgerblue2")) +  
 geom\_node\_point( size = 1) +  
 geom\_node\_text(aes(label = name), size = 4, repel = TRUE) +  
 theme\_void()

