ThuDo(Sofia)
Text Analytics and Natural Language Processing (NLP)

BUSINESS INSIGHT REPORT

Airbnb is an application for vacation rental online marketplace that provides the platform for hosts to accommodate guests with short or long term visits. ("Airbnb," 2021). On the Airbnb platform, after using the services, hosts and guests have the ability to leave reviews about the experience.

By understanding the importance of reviews to the business insights, this report will perform the text analysis of Airbnb reviews from different city datasets to analyze the quality of Airbnb around the world. The purposes of this report are to analyze the comments text from guests when they visited one of 3 cities- Berlin, Melbourne, Toronto compared to the cities from the United States (Boston and Seattle). Five different datasets represented for 5 cities will be imported and extracted by the text column from the guests' reviews.

Before jumping into the text mining process, each dataset will be tokenized and counted the frequency of words to get ready for the analysis.

A. FRAMEWORK 1- CORRELATION TEST

The first framework of correlation tests was used to quantify how similar and different sets of word frequencies. Boston and Seattle will represent for the East and West Coast cities in the US which will be used to compare with 3 other cities (Berlin, Melbourne, and Toronto)
The result from the code showed that:

1. With Boston baseline:
    - Boston vs Berlin: 0.816
    - Boston vs Melbourne: 0.888
    - Boston vs Toronto: 0.926

Based on the correlation output, we can see that the most common word frequency from the Toronto dataset was pretty similar to the Boston one with 92%. It is easy to understand that the location, the culture, the weather between Toronto and Boston are more mutual features than other cities which leads to the most similarity for the tokenized words from guests' reviews. We can take a deeper analysis by visualizing correlogram (keywords segmentation) in the next framework. The second rank with 88.8% similar word frequency set is the Melbourne dataset and the last one will be Berlin with 81.6%.
Overall, it can be said that the Boston data has a word frequency set that is very similar to all 3 targeted city datasets with more than 80% of similarity. It could be explained by the good comment words, the location words, or other factors.

2. With Seattle baseline:
    - Seattle vs Berlin: 0.644
    - Seattle vs Melbourne: 0.751
    - Seattle vs Toronto: 0.857

Besides that, let take a look at the Seattle baseline:

- The first rank is also Toronto data when compared with Seattle baseline, we can say that the 2 cities are both on the same continent. However, we can see that the gap between the East Coast (Boston baseline) and the West Coast baseline will be slightly different because the location or weather of Boston will be more similar to Toronto city.
- Second and the third rank has the same pattern as the Boston baseline with the larger mutual words on the Melbourne dataset.
- Overall, the similarity of the 3 targeted city datasets compare with the Seattle baseline is more different than the Boston one.

B. FRAMEWORK 2- CORRELOGRAM

The second framework will show the visualization of word three sets of texts for each baseline and could drive some good insights for the business.

1. Boston baseline

Figure: Correlogram with Boston baseline

The graph can be explained by the words that are close to the line in plots that have similar frequencies in these sets of texts (*1 The Tidy Text Format | Text Mining with R*, n.d.). We can see that "comfortable", "apartment", "close" will be the similar words that appeared in these datasets. However, the words that are far from the line are words that are found more in one set of texts than in another (*1 The Tidy Text Format | Text Mining with R*, n.d.). As we can clearly see that the words such as "berlin", "melbourne", "toronto" will appear a lot in their matched datasets but not in the baseline one. Similarly, "boston", "beacon" has appeared only on the Boston set

2. Seattle baseline

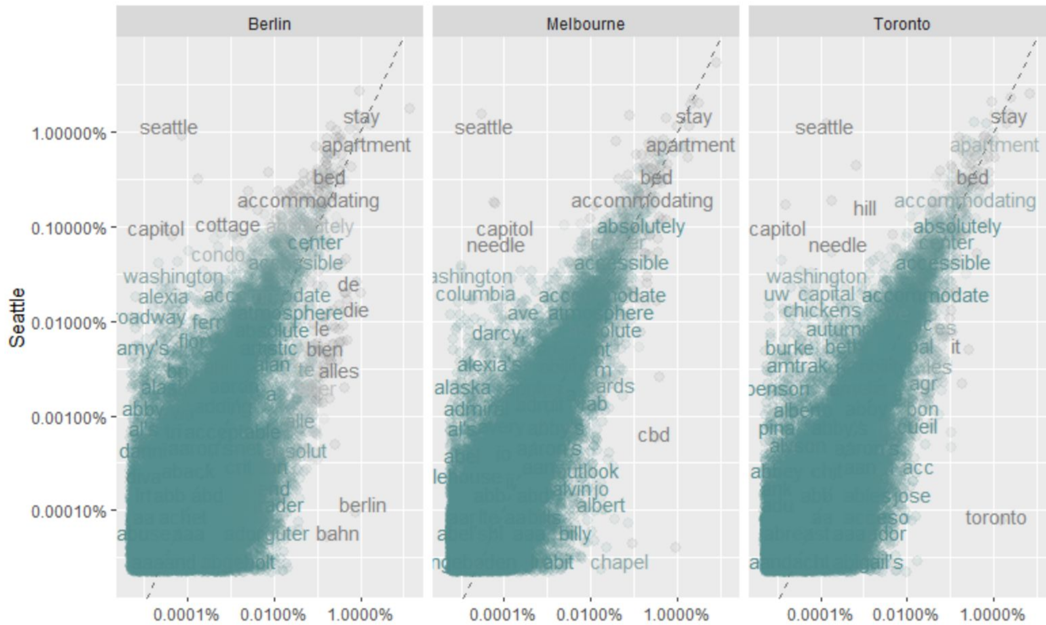


Figure: Correlogram with Seattle baseline

The result from this graph can be explained as the same on the Boston baseline. The common words that are mutual from 4 different cities are “apartment”, “accommodating”, “stay”, and “bed”. Unlike “bahn” which is the currency of Germany so it only appeared in the Berlin dataset, “chapel”, “albert” was just shown with Melbourne data and the same “toronto” word for the Toronto dataset.

Overall of both 2 baselines, we can say that the words in all panels are close to the zero-slope line and notice that the words extend to lower frequencies in all panels. These points indicate that the comments from guests among these use many similar words. It can be explained by the common characteristics of reviews is the description of their feelings, of the locations, of the amenities and convenience during their vacation in these cities. All the cities in this analysis are the famous spot for tourism that’s why the common words to describe guests' comments are pretty much similar to each other.

C. FRAMEWORK 3- SENTIMENTS

Using Boston dataset from reviews of guests as an example:

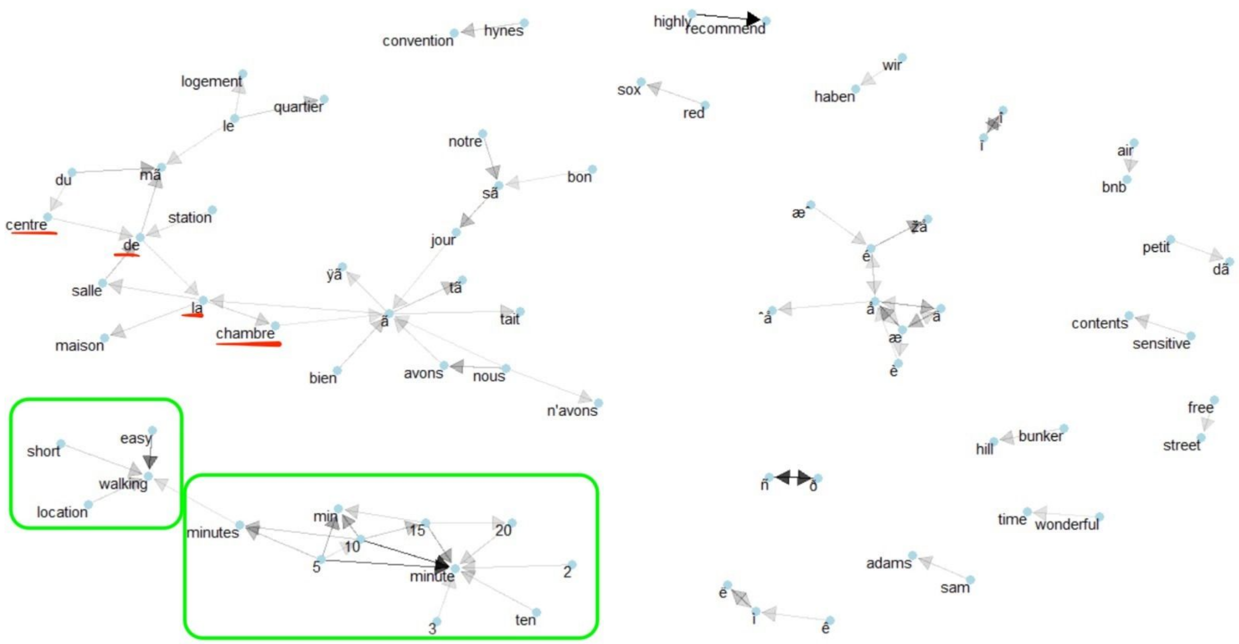
	word	n
1	clean	23177
2	perfect	10815
3	helpful	8669
4	friendly	6766
5	wonderful	6763
6	beautiful	6392
7	lovely	5650
8	excellent	4242
9	safe	3695
10	food	2372
11	found	2052
12	pleasant	2050
13	pretty	2011
14	love	1842
15	happy	1363
16	enjoy	1253
17	friend	1234
18	welcomed	1198
19	hope	1158
20	fun	1111

	word	n
1	hot	1302
2	noisy	704
3	bad	703
4	money	489
5	feeling	391
6	smell	314
7	disappointed	286
8	treat	239
9	hanging	223
10	broken	216
11	complaint	201
12	limited	188
13	rail	181
14	hit	145
15	rob	141
16	grab	139
17	fee	137
18	tree	135
19	challenge	134
20	honest	124

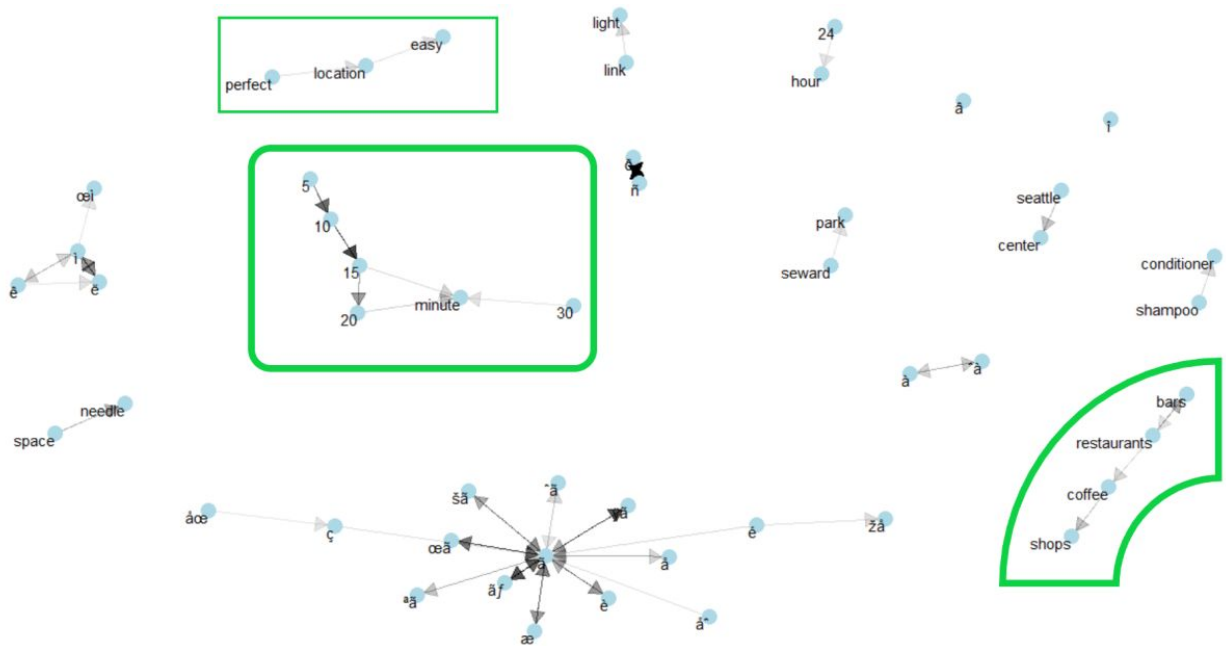
tidy_boston data with "Joy" sentiments and "Anger" sentiments

In this framework, the "nrc" dictionary which categories positive, negative sentiments such as anger, joy, sadness, surprise, and trust, etc. will be used to determine 20 frequent words that appeared from the comments of guests. As we can see from the result, the number of "joy" sentiment is much more than the number of "anger" ones which can indicate the good experience that the guests are enjoying during their stay during the vacation in Boston. Based on the sentiments, the host can improve the quality of the apartments/rooms to match with the guests' experience that can be beneficial in the future

D. FRAMEWORK 4- N-GRAMS



The trigram described that the minutes are linked with walking distance based on the location of the Airbnb that guests emphasized in their comments. The most location of Airbnb in Boston is near the center of the town so people can easily walk to the restaurant or downtown spot with a short walk. It can be beneficial for the host in the future to put these features in the description to attract more guests to book their apartments. Besides that, we can see there are a lot of comments by French from guests, it can be explained Boston would be the good location that attracts European visitors.



Different from the Boston dataset, there are more comments by German in Seattle data but the same target customer segment from Europe and be applied for both cities in the US. In Seattle, the same pattern of minutes appeared indicating the walking distances of the location are always important in any city. The closer the accommodation to the downtown the better review received. Another feature that the comments mentioned a lot about Seattle Airbnb is the restaurants, bars, shops, coffee, etc. which can be used to put into the description for future references of the hosts to attract more guests.

CONCLUSION

Text analysis can drive better business decisions when we apply the appropriate frameworks to it. We can use text mining to understand and generate insights into what the customers are thinking (*What Is Text Mining and How Can It Be Used to Create Value for Business?*, 2017).

With sentiment analysis, the texts will be analyzed to understand the attitudes and opinions of customers through their comments/ reviews. When data in a tidy structure, sentiment analysis can be implemented to understand how a narrative arc changes or the important emotional content for particular text (*1 The Tidy Text Format | Text Mining with R*, n.d.). More than that, we can use n-gram to explore the relationships and connections among words which helps us to analyze the connection of the words, co-occurrences, and correlations when they appeared after others (*1 The Tidy Text Format | Text Mining with R*, n.d.).

In conclusion, text mining will help analysts determine crucial insights across a wider range of documents and sources especially when we need to combine with external data sources. Text mining will be a powerful technique to put a variety of internal and external data sources that can improve both the speed and competency of decision making (*What Is Text Mining and How Can It Be Used to Create Value for Business?*, 2017).

REFERENCES

1 The tidy text format | Text Mining with R. (n.d.). Retrieved February 10, 2021, from https://www.tidytextmining.com/tidytext.html

Airbnb. (2021). In *Wikipedia*. https://en.wikipedia.org/w/index.php?title=Airbnb&oldid=1005443011

What is text mining and how can it be used to create value for business? (2017, April 11). Mastodon C. https://www.mastodonc.com/2017/04/12/what-is-text-mining-and-how-can-it-be-used-to-create-value-for-business/

APPENDIX

A. CODE

```
#LOAD ALL LIBRARIES
library(tidytext)
library(tidyverse)
library(dplyr)
library(stringr)
library(scales)
library(ggplot2)
library(tm)

#PREPARE DATA SET
data("stop_words")
setwd("C:/Users/asus/Desktop/MsBA/Spring semester/Text Analysis/Individual assignment/Dataset")

##########################
##BERLIN REVIEW
##########################
review_1 =read.csv("Berlin.csv", stringsAsFactors = FALSE)

comments_1 <- review_1$comments

berlin <- data.frame(line=1:401963, text=comments_1)


tidy_berlin <- berlin %>%
          unnest_tokens(word, text) %>%
          anti_join(stop_words)
tidy_berlin%>%
  count(word, sort=TRUE)

##########################
##TORONTO
##########################
review_2 =read.csv("Toronto.csv", stringsAsFactors = FALSE)
comments_2 <- review_2$comments
toronto <- data.frame(line=1:576806, text=comments_2)
tidy_toronto <- toronto %>%
            unnest_tokens(word, text) %>%
            anti_join(stop_words)
tidy_toronto%>%
  count(word, sort=TRUE)
```

```
##########################
#BOSTON
##########################
review_3 =read.csv("Boston.csv", stringsAsFactors = FALSE)
comments_3 <- review_3$comments
boston <- data.frame(line=1:68275, text=comments_3)
tidy_boston <- boston %>%
                    unnest_tokens(word, text) %>%
                    anti_join(stop_words)

tidy_boston%>%
  count(word, sort=TRUE)

##########################
#MELBOURNE
#########################
review_4 =read.csv("Melbourne.csv", stringsAsFactors = FALSE)
comments_4 <- review_4$comments
melbourne <- data.frame(line=1:486920, text=comments_4)

tidy_melbourne <- melbourne %>%
                            unnest_tokens(word, text) %>%
                            anti_join(stop_words)

tidy_melbourne%>%
  count(word, sort=TRUE)

##########################
#SEATTLE
##########################
review_5 =read.csv("Seattle.csv", stringsAsFactors = FALSE)
comments_5 <- review_5$comments
seattle <- data.frame(line=1:84849, text=comments_5)

tidy_seattle <- seattle %>%
                    unnest_tokens(word, text) %>%
                    anti_join(stop_words)

tidy_seattle%>%
  count(word, sort=TRUE)

########################################
#BOSTON IS THE BASELINE
```

```
###################################################
## FRAMEWORK TO COMPARE DIFFERENT TEXTS ##
###################################################

#prepare data by combining all the datasets and do frequencies
library(tidyr)
frequency <- bind_rows(mutate(tidy_berlin, city="Berlin"),
                       mutate(tidy_melbourne, city= "Melbourne"),
                       mutate(tidy_toronto, city= "Toronto"),
                       mutate(tidy_boston, city="Boston")
)%>% #closing bind_rows
  mutate(word=str_extract(word, "[a-z']+")) %>%
  count(city, word) %>%
  group_by(city) %>%
  mutate(proportion = n /sum(n))%>%
  select(-n) %>%
  spread(city, proportion) %>%
  gather(city, proportion,`Toronto`, `Melbourne`, `Berlin`)


#### FRAMEWORK 1: .CORR() TEST --> find correlation coefficients
###########################################

cor.test(data=frequency[frequency$city == "Melbourne",],
         ~proportion + `Boston`)
cor.test(data=frequency[frequency$city == "Berlin",],
         ~proportion + `Boston`)
cor.test(data=frequency[frequency$city == "Toronto",],
         ~proportion + `Boston`)

#### FRAMEWORK 2: CORRELOGRAMS (keywords segmentation)
###########################################
library(scales)
ggplot(frequency, aes(x=proportion, y=`Boston`,
                      color = abs(`Boston`- proportion)))+
  geom_abline(color="grey40", lty=2)+
  geom_jitter(alpha=.1, size=2.5, width=0.3, height=0.3)+
  geom_text(aes(label=word), check_overlap = TRUE, vjust=1.5) +
  scale_x_log10(labels = percent_format())+
  scale_y_log10(labels= percent_format())+
  scale_color_gradient(limits = c(0,0.001), low = "darkslategray4", high = "gray75")+
  facet_wrap(~city, ncol=3)+
  theme(legend.position = "none")+
  labs(y= "Boston", x=NULL)
```

```
##########################################
#SEATTLE IS THE BASELINE

##################################################
## FRAMEWORK TO COMPARE DIFFERENT TEXTS ##
##################################################

#prepare data by combining all the datasets and do frequencies
library(tidyr)
frequency <- bind_rows(mutate(tidy_berlin, city="Berlin"),
                       mutate(tidy_melbourne, city= "Melbourne"),
                       mutate(tidy_toronto, city= "Toronto"),
                       mutate(tidy_seattle, city="Seattle")
)%>%#closing bind_rows
  mutate(word=str_extract(word, "[a-z']+")) %>%
  count(city, word) %>%
  group_by(city) %>%
  mutate(proportion = n /sum(n))%>%
  select(-n) %>%
  spread(city, proportion) %>%
  gather(city, proportion,`Toronto`, `Melbourne`, `Berlin`)

#FRAMEWORK 1- CORRELATION
cor.test(data=frequency[frequency$city == "Melbourne",],
         ~proportion + `Seattle`)

cor.test(data=frequency[frequency$city == "Berlin",],
         ~proportion + `Seattle`)

cor.test(data=frequency[frequency$city == "Toronto",],
         ~proportion + `Seattle`)

# FRAMEWORK 2: CORRELOGRAMS (keywords segmentation)
library(scales)
ggplot(frequency, aes(x=proportion, y=`Seattle`,
                      color = abs(`Seattle`- proportion)))+
  geom_abline(color="grey40", lty=2)+
  geom_jitter(alpha=.1, size=2.5, width=0.3, height=0.3)+
  geom_text(aes(label=word), check_overlap = TRUE, vjust=1.5) +
  scale_x_log10(labels = percent_format())+
  scale_y_log10(labels= percent_format())+
  scale_color_gradient(limits = c(0,0.001), low = "darkslategray4", high = "gray75")+
```

```
  facet_wrap(~city, ncol=3)+
  theme(legend.position = "none")+
  labs(y= "Seattle", x=NULL)


#############################################################
#FRAMEWORK 3: SENTIMENTS
#############################################################
library(textdata)
library(dplyr)
library(stringr)
library(tidyverse)
library(tidytext)

afinn <- get_sentiments("afinn") #Negative vs positive sentiment
nrc <- get_sentiments("nrc")     #emotions
bing <- get_sentiments("bing")   #binary

sentiments <- bind_rows(mutate(afinn, lexicon="afinn"),
                        mutate(nrc, lexicon= "nrc"),
                        mutate(bing, lexicon="bing")
)

###########################
#BOSTON CITY
############################
#1- SENTIMENT FRAMEWORK with JOY
nrcsurprise <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

#inner joining the emma book and the surprise sentiments
tidy_boston %>%
  inner_join(nrcsurprise) %>%
  count(word, sort=T)%>%
  filter(n>1100)


#2- SENTIMENT FRAMEWORK with ANGER
nrcsurprise <- get_sentiments("nrc") %>%
  filter(sentiment == "anger")

#inner joining the emma book and the surprise sentiments
tidy_boston %>%
  inner_join(nrcsurprise) %>%
```

```
  count(word, sort=T)%>%
  filter(n>120)

###########################
#SEATTLE CITY
############################

#1- SENTIMENT FRAMEWORK with JOY
nrcsurprise <- get_sentiments("nrc") %>%
  filter(sentiment == "joy")

#inner joining the emma book and the surprise sentiments
tidy_seattle %>%
  inner_join(nrcsurprise) %>%
  count(word, sort=T)%>%
  filter(n>2000)

#2- SENTIMENT FRAMEWORK with ANGER
nrcsurprise <- get_sentiments("nrc") %>%
  filter(sentiment == "anger")

#inner joining the emma book and the surprise sentiments
tidy_seattle %>%
  inner_join(nrcsurprise) %>%
  count(word, sort=T)%>%
  filter(n>165)

#####################################################
#FRAMEWORK 4: n-gram
#####################################################
library(igraph)
library(ggraph)

#1-QUADROGRAM- SEATTLE
quadrogram <- seattle%>%
        unnest_tokens(quadrogram, text, token = "ngrams", n=4) %>%
        filter(!is.na(quadrogram))%>%
        separate(quadrogram, c("word1", "word2", "word3", "word4"), sep=" ") %>%
        filter(!word1 %in% stop_words$word) %>%
        filter(!word2 %in% stop_words$word) %>%
        filter(!word3 %in% stop_words$word) %>%
        filter(!word4 %in% stop_words$word)
```

```
quadrogram_counts <- quadrogram %>%
                       count(word1, word2, word3, word4, sort = TRUE)

#create matrix to draw trigram network
graph <- quadrogram_counts %>%
  filter(n>12) %>%
  graph_from_data_frame()

#visualize trigram network
a <- grid::arrow(type = "closed", length = unit(.15, "inches"))
ggraph(graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha = n), show.legend = FALSE,
                 arrow = a, end_cap = circle(.07, 'inches')) +
  geom_node_point(color = "lightblue", size = 5) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1) +
  theme_void()

#2- TRIGRAM-BOSTON
trigram <- boston%>%
  unnest_tokens(trigram, text, token = "ngrams", n=3) %>%
  filter(!is.na(trigram))%>%
  separate(trigram, c("word1", "word2", "word3"), sep=" ") %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word) %>%
  filter(!word3 %in% stop_words$word)

trigram_counts <- trigram %>%
  count(word1, word2, word3, sort = TRUE)

#create matrix to draw trigram network
trigram_graph <- trigram_counts %>%
  filter(n>60) %>%
  graph_from_data_frame()

#visualize trigram network
a <- grid::arrow(type = "closed", length = unit(.15, "inches"))
ggraph(trigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha = n), show.legend = FALSE,
                 arrow = a, end_cap = circle(.07, 'inches')) +
  geom_node_point(color = "lightblue", size = 3) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1) +
  theme_void()
```

B. OUTPUT

CORRELATION TEST

```
data:  proportion and Boston
t = 253.03, df = 32021, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8127767 0.8200804
sample estimates:
      cor
0.8164612
```

Figure 1: Boston vs Berlin

```
data:  proportion and Boston
t = 323.48, df = 28010, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8856695 0.8906156
sample estimates:
      cor
0.8881683
```

Figure 2: Boston vs Melbourne

```
data:  proportion and Boston
t = 436.09, df = 31482, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.9246798 0.9278179
sample estimates:
      cor
0.9262649
```

Figure 3: Boston vs Toronto

```
data:  proportion and Seattle
t = 136.22, df = 26205, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.6367151 0.6508921
sample estimates:
      cor
0.6438588
```

Figure 4: Seattle vs Berlin

```
data:  proportion and Seattle
t = 183.84, df = 26025, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.7463031 0.7568746
sample estimates:
      cor
0.7516371
```

Figure 5: Seattle vs Melbourne

```
data:  proportion and Seattle
t = 274.82, df = 27427, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.8533201 0.8596261
sample estimates:
      cor
0.8565051
```

Figure 6: Seattle vs Toronto

CORRELOGRAM

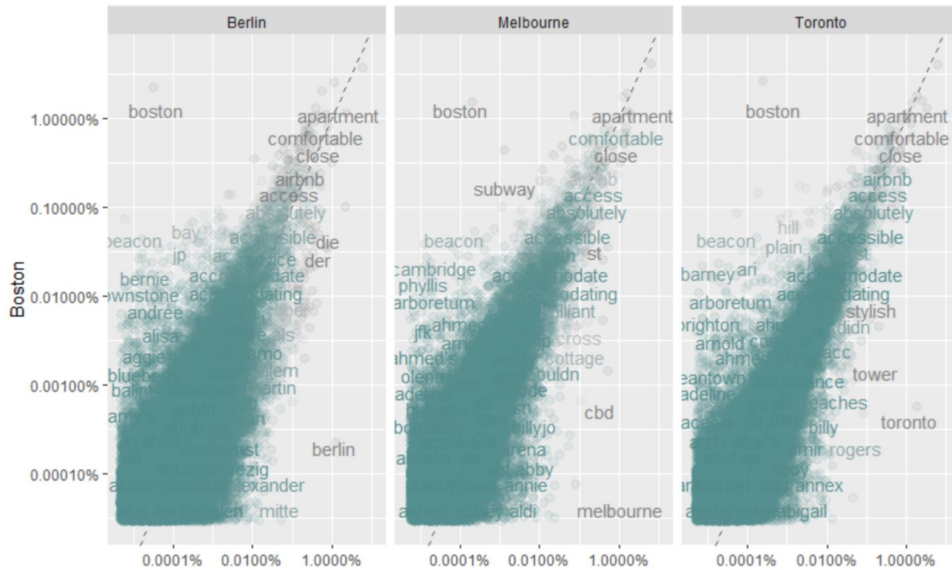


Figure 7: Correlogram with Boston baseline

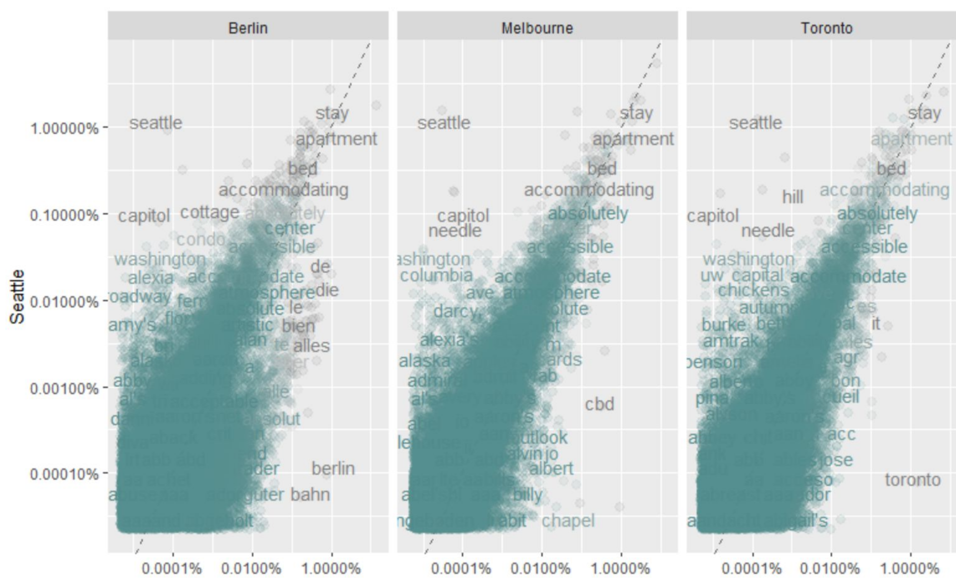


Figure 8: Correlogram with Seattle baseline

SENTIMENTS

1. BOSTON

	word	n
1	clean	23177
2	perfect	10815
3	helpful	8669
4	friendly	6766
5	wonderful	6763
6	beautiful	6392
7	lovely	5650
8	excellent	4242
9	safe	3695
10	food	2372
11	found	2052
12	pleasant	2050
13	pretty	2011
14	love	1842
15	happy	1363
16	enjoy	1253
17	friend	1234
18	welcomed	1198
19	hope	1158
20	fun	1111

	word	n
1	hot	1302
2	noisy	704
3	bad	703
4	money	489
5	feeling	391
6	smell	314
7	disappointed	286
8	treat	239
9	hanging	223
10	broken	216
11	complaint	201
12	limited	188
13	rail	181
14	hit	145
15	rob	141
16	grab	139
17	fee	137
18	tree	135
19	challenge	134
20	honest	124

Figure 9: Joy sentiment and Anger sentiment with Boston data

2. SEATTLE

	word	n
1	clean	29332
2	perfect	16280
3	wonderful	12276
4	helpful	10232
5	beautiful	9717
6	friendly	9610
7	lovely	8990
8	excellent	5414
9	safe	3520
10	love	3412
11	fun	3100
12	food	3033
13	found	2929
14	pretty	2836
15	pleasant	2723
16	garden	2495
17	enjoy	2225
18	sweet	2163
19	happy	2151
20	hope	2076

	word	n
1	hot	2035
2	rail	1859
3	bad	649
4	feeling	641
5	noisy	613
6	treat	572
7	money	426
8	rob	410
9	hanging	371
10	disappointed	350
11	tree	321
12	smell	292
13	intrusive	275
14	grab	250
15	complaint	237
16	hit	227
17	limited	208
18	burke	185
19	cross	177
20	blast	169

Figure 10: Joy sentiment and Anger sentiment with Seattle data

N-GRAMS

1. Quadrogram for Seattle data

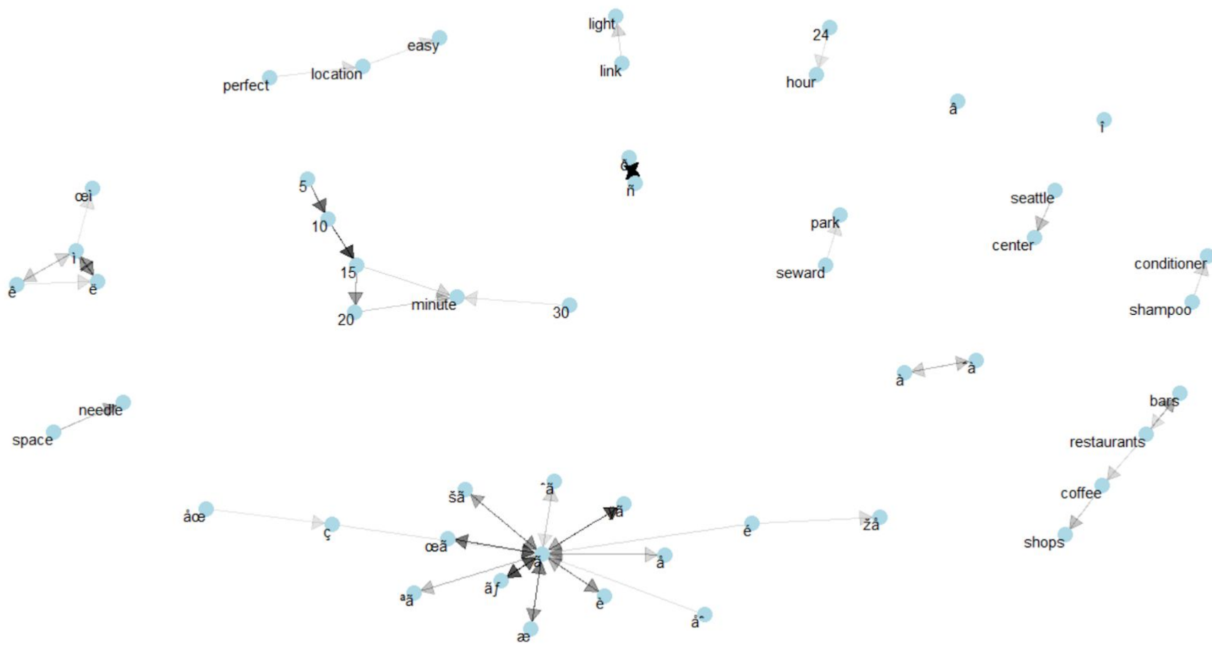


Figure 11: Quadrogram for Seattle comments data

2. Trigram for Boston data

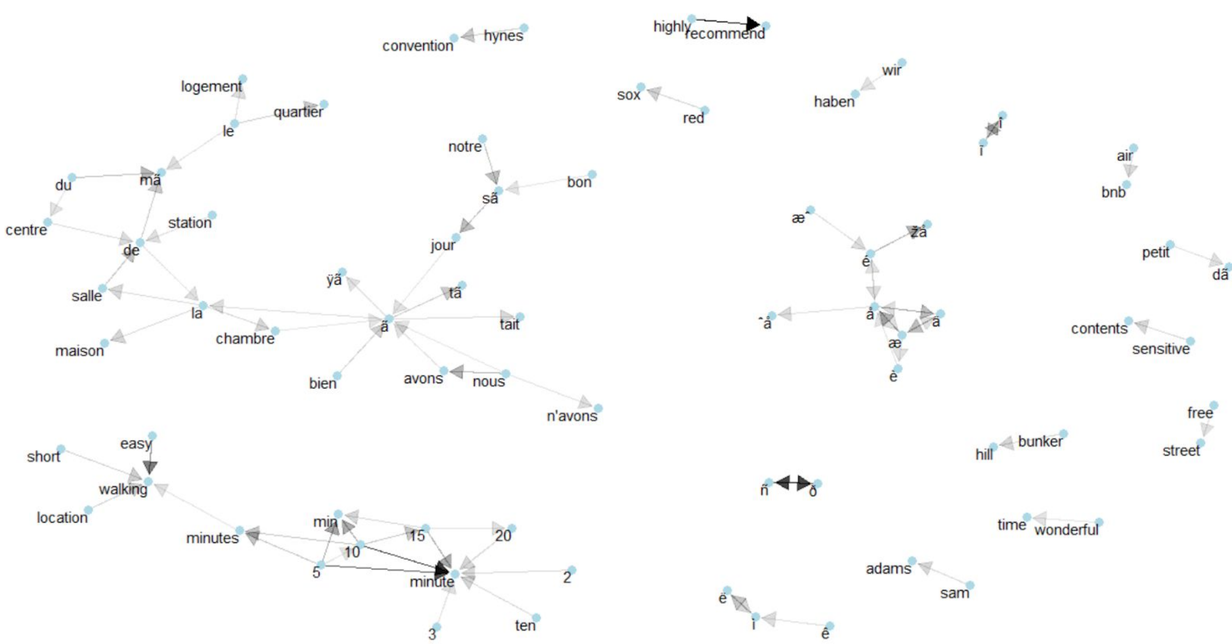


Figure 12: Trigram of Boston comments data