Hotel Reviews

Torrey Capobianco

5/8/2021

Libraries

```
library(dplyr)
```

```
## Registered S3 methods overwritten by 'tibble':
##   method     from
##   format.tbl pillar
##   print.tbl  pillar
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(tidytext)
library(stringr)
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

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

```
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
```

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

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

```
library(ggraph)
```

```
## Loading required package: ggplot2
```

```r
library(plotly)
```

```
##
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':
##
##     last_plot

## The following object is masked from 'package:igraph':
##
##     groups

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

Data

```r
df <- read.csv('data/hotel_text_cln.csv')

head(df)
```

```
##   hotel_city latitude longitude               name hotel_state
## 1   Mableton 45.42161  12.37619 Hotel Russo Palace          GA
## 2   Mableton 45.42161  12.37619 Hotel Russo Palace          GA
## 3   Mableton 45.42161  12.37619 Hotel Russo Palace          GA
## 4   Mableton 45.42161  12.37619 Hotel Russo Palace          GA
## 5   Mableton 45.42161  12.37619 Hotel Russo Palace          GA
## 6   Mableton 45.42161  12.37619 Hotel Russo Palace          GA
##                         date rating
## 1 2013-09-22 00:00:00+00:00      4
## 2 2015-04-03 00:00:00+00:00      5
## 3 2013-10-27 00:00:00+00:00      5
## 4 2015-04-05 00:00:00+00:00      5
## 5 2014-06-10 00:00:00+00:00      4
## 6 2015-08-02 00:00:00+00:00      3
##
## 1
## 2
## 3 We stayed here for four nights in October. The hotel staff were welcoming, friendly and helpful. A
## 4                             We loved staying on the island of Lido! You need to take a water is fr
## 5
## 6
##   user_city user_state                    language    score
## 1                       ('en', -390.9012541770935) positive
## 2                       ('en', -535.3024659156799) positive
## 3                       ('en', -713.2087678909302) positive
## 4                       ('en', -797.3543155193329) positive
## 5                      ('en', -125.38225984573364) positive
## 6                      ('en', -280.76494693756104)  neutral
##
```

```
## 1                                                                       pleasant min walk along sea
## 2                                                                really lovely hotel stay top floor
## 3 stay four night october hotel staff welcome friendly helpful assist book ticket opera room clean c
## 4                      love stay island lido need take water venice get train station boat ride take
## 5
## 6
##
## 1                                                                       pleasant min walk along sea
## 2                                                                really lovely hotel stay top floor
## 3 stay four night october hotel staff welcome friendly helpful assist book ticket opera room clean c
## 4                      love stay island lido need take water venice get train station boat ride take
## 5
## 6
```

```
# separate positive, negative scores and all scores

positive <- df$cln_full_sentence[df$score == "positive"]
negative <- df$cln_full_sentence[df$score == "negative"]
all <- df$cln_full_sentence


# convert positive/negative to tibble
pos <- tibble(line = 1:20737, text = positive)

neg <- tibble(line = 1:2849, text = negative)

all <- tibble(line= 1:31670, text = all)
```

Positive Reviews

Frequent Single Words

```
pos_tokens <- pos %>%
  unnest_tokens(word, text)


data(stop_words)

pos_tokens <- pos_tokens %>%
  anti_join(stop_words, by =c("word" = "word"))

pos_tokens %>%
  count(word, sort=TRUE)
```

```
## # A tibble: 13,656 x 2
##    word            n
##    <chr>       <int>
##  1 hotel       11634
##  2 stay        11324
##  3 staff        7313
##  4 clean        7149
##  5 nice         6118
##  6 breakfast    5726
##  7 friendly     4334
##  8 comfortable  3753
##  9 night        3083
## 10 bed          3073
## # ... with 13,646 more rows
```

BiGrams

```
# get 2 word combo

pos_bigram <- pos %>%
  unnest_tokens(bigram, text, token = "ngrams", n=2)

pos_bigram
```

```
## # A tibble: 442,796 x 2
##     line bigram
##    <int> <chr>
##  1     1 pleasant min
##  2     1 min walk
##  3     1 walk along
##  4     1 along sea
##  5     1 sea front
##  6     1 front water
##  7     1 water bus
##  8     1 bus restaurant
##  9     1 restaurant etc
## 10     1 etc hotel
## # ... with 442,786 more rows
```

Most common n-grams

```
pos_bigram %>%
  count(bigram, sort = TRUE)
```

```
## # A tibble: 195,280 x 2
##    bigram                n
##    <chr>             <int>
##  1 room clean         1786
##  2 staff friendly     1417
##  3 front desk         1298
##  4 friendly helpful    850
##  5 clean comfortable   837
##  6 place stay          787
##  7 great location      697
##  8 bed comfortable     695
##  9 friendly staff      695
## 10 stay hotel          680
## # ... with 195,270 more rows
```

```
# separate into two columns
pos_bigrams_separated <- pos_bigram %>%
  separate(bigram, c("word1", "word2"), sep=" ")

# new bigram counts:
pos_bigram_counts <- pos_bigrams_separated %>%
  count(word1, word2, sort = TRUE)
```

```
pos_bigram_counts
```

```
## # A tibble: 195,280 x 3
##    word1    word2           n
##    <chr>    <chr>       <int>
##  1 room     clean        1786
##  2 staff    friendly     1417
##  3 front    desk         1298
##  4 friendly helpful       850
##  5 clean    comfortable   837
##  6 place    stay          787
##  7 great    location      697
##  8 bed      comfortable   695
##  9 friendly staff         695
## 10 stay     hotel         680
## # ... with 195,270 more rows
```

```
pos_bigrams_separated %>%
  filter(word1 == "nice") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 886 x 3
##    word1 word2         n
##    <chr> <chr>     <int>
##  1 nice  hotel       470
##  2 nice  clean       417
##  3 nice  room        340
##  4 nice  place       251
##  5 nice  staff       183
##  6 nice  breakfast   173
##  7 nice  stay        167
##  8 nice  helpful     108
##  9 nice  pool        108
## 10 nice  touch        85
## # ... with 876 more rows
```

```
pos_bigrams_separated %>%
  filter(word1 == "need") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 600 x 3
##    word1 word2        n
##    <chr> <chr>    <int>
##  1 need  place       70
##  2 need  room        45
##  3 need  update      44
##  4 need  hotel       38
##  5 need  stay        37
##  6 need  clean       36
##  7 need  anything    30
##  8 need  get         23
##  9 need  would       23
## 10 need  go          21
## # ... with 590 more rows
```

```
pos_bigrams_separated %>%
  filter(word1 == "area") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 800 x 3
##    word1 word2         n
##    <chr> <chr>     <int>
##  1 area  great        67
##  2 area  room         62
##  3 area  nice         61
##  4 area  hotel        56
##  5 area  clean        55
##  6 area  would        42
##  7 area  stay         40
##  8 area  close        38
##  9 area  good         30
## 10 area  breakfast    28
## # ... with 790 more rows
```

```
pos_bigrams_separated %>%
  filter(word1 == "room") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 1,653 x 3
##    word1 word2           n
##    <chr> <chr>       <int>
##  1 room  clean        1786
##  2 room  nice          554
##  3 room  comfortable   383
##  4 room  great         356
##  5 room  spacious      289
##  6 room  good          215
##  7 room  large         193
##  8 room  service       177
##  9 room  small         125
## 10 room  hotel         115
## # ... with 1,643 more rows
```

```
pos_bigrams_separated %>%
  filter(word1 == "love") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 515 x 3
##    word1 word2          n
##    <chr> <chr>      <int>
##  1 love  hotel        127
##  2 love  stay         123
##  3 love  pool          56
##  4 love  place         49
##  5 love  room          47
##  6 love  location      39
##  7 love  everything    31
##  8 love  fact          21
##  9 love  every         18
## 10 love  view          18
## # ... with 505 more rows
```

TriGrams

```
pos_trigram <- pos %>%
  unnest_tokens(bigram, text, token = "ngrams", n=3)

pos_trigram
```

```
## # A tibble: 422,654 x 2
##     line bigram
##    <int> <chr>
##  1     1 pleasant min walk
##  2     1 min walk along
##  3     1 walk along sea
##  4     1 along sea front
##  5     1 sea front water
##  6     1 front water bus
##  7     1 water bus restaurant
##  8     1 bus restaurant etc
##  9     1 restaurant etc hotel
## 10     1 etc hotel comfortable
## # ... with 422,644 more rows
```

Most common tri-grams

```
pos_trigram %>%
  count(bigram, sort = TRUE)
```

```
## # A tibble: 353,837 x 2
##    bigram                     n
##    <chr>                  <int>
##  1 <NA>                     595
##  2 staff friendly helpful   432
##  3 would definitely stay    344
##  4 room clean comfortable   281
##  5 within walk distance     256
##  6 front desk staff         245
##  7 stay one night           235
##  8 great place stay         213
##  9 room clean bed           169
## 10 clean bed comfortable    168
## # ... with 353,827 more rows
```

```
# separate into two columns
pos_trigrams_separated <- pos_trigram %>%
  separate(bigram, c("word1", "word2", "word3"), sep=" ")

# new trigram counts:
pos_trigram_counts <- pos_trigrams_separated %>%
  count(word1, word2, word3, sort = TRUE)

pos_trigram_counts
```

```
## # A tibble: 353,837 x 4
##    word1  word2      word3          n
##    <chr>  <chr>      <chr>      <int>
##  1 <NA>   <NA>       <NA>         595
##  2 staff  friendly   helpful      432
##  3 would  definitely stay         344
##  4 room   clean      comfortable  281
##  5 within walk       distance     256
##  6 front  desk       staff        245
##  7 stay   one        night        235
##  8 great  place      stay         213
##  9 room   clean      bed          169
## 10 clean  bed        comfortable  168
## # ... with 353,827 more rows
```

```
pos_trigrams_separated %>%
  filter(word1 == "need") %>%
  count(word1, word2, word3, sort=TRUE)
```

```
## # A tibble: 1,452 x 4
##    word1 word2       word3          n
##    <chr> <chr>       <chr>      <int>
##  1 need  place       stay          38
##  2 need  place       sleep         13
##  3 need  one         night         10
##  4 need  comfortable bed            7
##  5 need  comfortable stay           7
##  6 need  would       definitely     7
##  7 need  bed         comfortable    5
##  8 need  hotel       area           5
##  9 need  overnight   stay           5
## 10 need  room        night          5
## # ... with 1,442 more rows
```

Graph Network

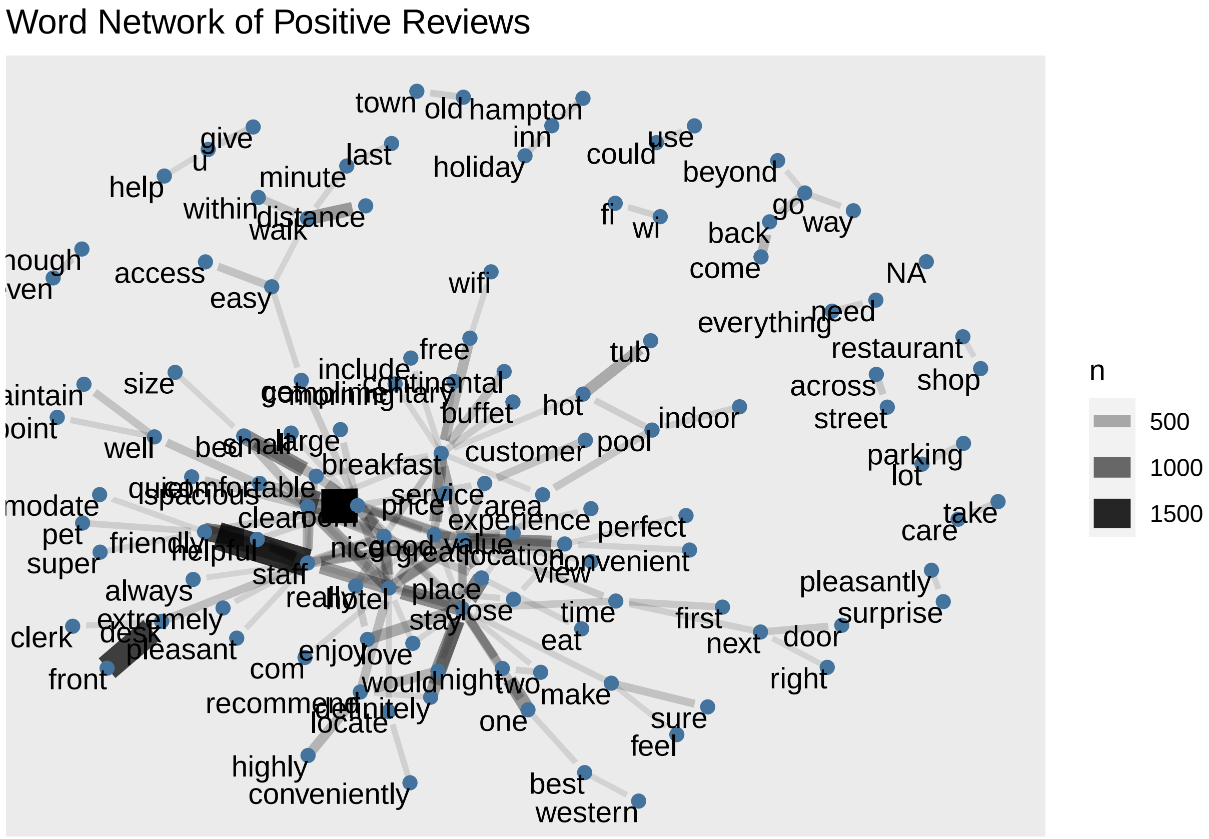
```
# filter for only relatively common combinations
pos_bigram_graph <- pos_bigram_counts %>%
  filter(n > 115) %>%
  graph_from_data_frame()
```

```
## Warning in graph_from_data_frame(.): In `d' `NA' elements were replaced with
## string "NA"
```

```
set.seed(2017)

ggsave("graphs/positive_network.png", width = 15, height = 10, dpi = 300)

ggraph(pos_bigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha=n, edge_width=n),
                 end_cap=circle(.07, 'inches')) +
  geom_node_point(color='#44749D', size=2) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1, size=4) +
 labs(title="Word Network of Positive Reviews")
```



Word Network of Positive Reviews

Negative Reviews

Frequent Single Words

```
neg_tokens <- neg %>%
  unnest_tokens(word, text)


data(stop_words)

neg_tokens <- neg_tokens %>%
  anti_join(stop_words, by =c("word" = "word"))

neg_tokens %>%
  count(word, sort=TRUE)
```

```
## # A tibble: 5,902 x 2
##    word       n
##    <chr> <int>
##  1 hotel   1930
##  2 stay    1555
##  3 bed      789
##  4 night    765
##  5 check    645
##  6 bad      600
##  7 dirty    598
##  8 desk    563
##  9 front   527
## 10 smell   513
## # ... with 5,892 more rows
```

BiGrams

```
neg_bigram <- neg %>%
  unnest_tokens(bigram, text, token = "ngrams", n=2)

neg_bigram
```

```
## # A tibble: 85,953 x 2
##     line bigram
##    <int> <chr>
##  1     1 stay unless
##  2     1 unless less
##  3     1 less foot
##  4     1 foot tall
##  5     1 tall like
##  6     1 like sleep
##  7     1 sleep centipede
##  8     1 centipede th
##  9     1 th floor
## 10     1 floor room
## # ... with 85,943 more rows
```

Most common n-grams

```
neg_bigram %>%
  count(bigram, sort = TRUE)
```

```
## # A tibble: 54,561 x 2
##    bigram           n
##    <chr>        <int>
##  1 front desk     426
##  2 room smell     143
##  3 never stay     140
##  4 stay hotel     124
##  5 smoking room   122
##  6 hotel com      121
##  7 non smoking    113
##  8 get room       106
##  9 smell like     106
## 10 look like      104
## # ... with 54,551 more rows
```

```
# separate into two columns
neg_bigrams_separated <- neg_bigram %>%
  separate(bigram, c("word1", "word2"), sep=" ")

# new bigram counts:
neg_bigram_counts <- neg_bigrams_separated %>%
  count(word1, word2, sort = TRUE)
```

```
neg_bigram_counts
```

```
## # A tibble: 54,561 x 3
##    word1   word2       n
##    <chr>   <chr>   <int>
##  1 front   desk      426
##  2 room    smell     143
##  3 never   stay      140
##  4 stay    hotel     124
##  5 smoking room      122
##  6 hotel   com       121
##  7 non     smoking   113
##  8 get     room      106
##  9 smell   like      106
## 10 look    like      104
## # ... with 54,551 more rows
```

```
neg_bigrams_separated %>%
  filter(word1 == "check") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 275 x 3
##    word1 word2     n
##    <chr> <chr> <int>
##  1 check room     46
##  2 check u        24
##  3 check hotel    22
##  4 check day      15
##  5 check early    15
##  6 check go       13
##  7 check ask      12
##  8 check pm       12
##  9 check time     12
## 10 check tell     11
## # ... with 265 more rows
```

```
neg_bigrams_separated %>%
  filter(word1 == "go") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 286 x 3
##    word1 word2         n
##    <chr> <chr>     <int>
##  1 go    back         69
##  2 go    room         36
##  3 go    front        30
##  4 go    check        27
##  5 go    get          24
##  6 go    another      16
##  7 go    somewhere    13
##  8 go    stay         11
##  9 go    bed          10
## 10 go    find         10
## # ... with 276 more rows
```

```
neg_bigrams_separated %>%
  filter(word1 == "one") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 259 x 3
##    word1 word2      n
##    <chr> <chr>  <int>
##  1 one   night     73
##  2 one   bad       26
##  3 one   bed       17
##  4 one   room      15
##  5 one   person    12
##  6 one   towel     10
##  7 one   time       9
##  8 one   two        9
##  9 one   front      8
## 10 one   star       8
## # ... with 249 more rows
```

```
neg_bigrams_separated %>%
  filter(word1 == "room") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 921 x 3
##    word1 word2         n
##    <chr> <chr>     <int>
##  1 room  smell       143
##  2 room  clean        96
##  3 room  dirty        85
##  4 room  room         50
##  5 room  available    42
##  6 room  hotel        41
##  7 room  filthy       38
##  8 room  service      34
##  9 room  one          32
## 10 room  give         30
## # ... with 911 more rows
```

```
pos_bigrams_separated %>%
  filter(word1 == "bad") %>%
  count(word1, word2, sort=TRUE)
```

```
## # A tibble: 170 x 3
##    word1 word2          n
##    <chr> <chr>      <int>
##  1 bad   thing         35
##  2 bad   experience    20
##  3 bad   hotel         16
##  4 bad   review        16
##  5 bad   room          14
##  6 bad   say           12
##  7 bad   place         11
##  8 bad   stay          10
##  9 bad   weather        9
## 10 bad   price          6
## # ... with 160 more rows
```

TriGrams

```
neg_trigram <- neg %>%
  unnest_tokens(bigram, text, token = "ngrams", n=3)

neg_trigram
```

```
## # A tibble: 83,176 x 2
##     line bigram
##    <int> <chr>
##  1     1 stay unless less
##  2     1 unless less foot
##  3     1 less foot tall
##  4     1 foot tall like
##  5     1 tall like sleep
##  6     1 like sleep centipede
##  7     1 sleep centipede th
##  8     1 centipede th floor
##  9     1 th floor room
## 10     1 floor room attic
## # ... with 83,166 more rows
```

Most common tri-grams

```
neg_trigram %>%
  count(bigram, sort = TRUE)
```

```
## # A tibble: 76,557 x 2
##    bigram                    n
##    <chr>                 <int>
##  1 non smoking room         89
##  2 <NA>                     72
##  3 call front desk          55
##  4 bad hotel ever           50
##  5 room smell like          47
##  6 hotel ever stay          30
##  7 go front desk            29
##  8 would recommend hotel    28
##  9 front desk staff         25
## 10 smell like smoke         25
## # ... with 76,547 more rows
```

```
# separate into two columns
neg_trigrams_separated <- neg_trigram %>%
  separate(bigram, c("word1", "word2", "word3"), sep=" ")

# new bigram counts:
neg_trigram_counts <- neg_trigrams_separated %>%
  count(word1, word2, word3, sort = TRUE)

neg_trigram_counts
```

```
## # A tibble: 76,557 x 4
##    word1 word2     word3     n
##    <chr> <chr>     <chr> <int>
##  1 non   smoking   room     89
##  2 <NA>  <NA>      <NA>     72
##  3 call  front     desk     55
##  4 bad   hotel     ever     50
##  5 room  smell     like     47
##  6 hotel ever      stay     30
##  7 go    front     desk     29
##  8 would recommend hotel    28
##  9 front desk      staff    25
## 10 smell like      smoke    25
## # ... with 76,547 more rows
```

```
neg_trigrams_separated %>%
  filter(word1 == "go") %>%
  count(word1, word2, word3, sort=TRUE)
```

```
## # A tibble: 571 x 4
##    word1 word2     word3      n
##    <chr> <chr>     <chr>  <int>
##  1 go    front     desk      29
##  2 go    somewhere else      13
##  3 go    check     room       9
##  4 go    another   hotel      8
##  5 go    back      desk       7
##  6 go    back      room       7
##  7 go    back      hotel      6
##  8 go    anywhere  else       4
##  9 go    back      front      4
## 10 go    take      shower     4
## # ... with 561 more rows
```

```
neg_trigrams_separated %>%
  filter(word1 == "room") %>%
  count(word1, word2, word3, sort=TRUE)
```

```
## # A tibble: 2,817 x 4
##    word1 word2 word3        n
##    <chr> <chr> <chr>    <int>
##  1 room  smell like        47
##  2 room  give  u           12
##  3 room  look  like         9
##  4 room  smell bad          9
##  5 room  smell musty        8
##  6 room  front desk         7
##  7 room  hotel com          7
##  8 room  non   smoking      7
##  9 room  smell old          7
## 10 room  smell terrible     7
## # ... with 2,807 more rows
```

```
neg_trigrams_separated %>%
  filter(word1 == "check", word2 == "room") %>%
  count(word1, word2, word3, sort=TRUE)
```

```
## # A tibble: 34 x 4
##    word1 word2 word3          n
##    <chr> <chr> <chr>      <int>
##  1 check room  ready          3
##  2 check room  see            3
##  3 check room  check          2
##  4 check room  dirty          2
##  5 check room  go             2
##  6 check room  literally      2
##  7 check room  look           2
##  8 check room  make           2
##  9 check room  manager        2
## 10 check room  call           1
## # ... with 24 more rows
```

```
neg_trigrams_separated %>%
  filter(word1 == "room", word2 == "key") %>%
  count(word1, word2, word3, sort=TRUE)
```

```
## # A tibble: 14 x 4
##    word1 word2 word3             n
##    <chr> <chr> <chr>         <int>
##  1 room  key   check             4
##  2 room  key   go                2
##  3 room  key   room              2
##  4 room  key   call              1
##  5 room  key   card              1
##  6 room  key   disappoint        1
##  7 room  key   fee               1
##  8 room  key   hand              1
##  9 room  key   open              1
## 10 room  key   reprogrammed      1
## 11 room  key   since             1
## 12 room  key   walk              1
## 13 room  key   without           1
## 14 room  key   work              1
```

Graph Network

```
# filter for only relatively common combinations
neg_bigram_graph <- neg_bigram_counts %>%
  filter(n > 30) %>%
  graph_from_data_frame()

neg_bigram_graph
```

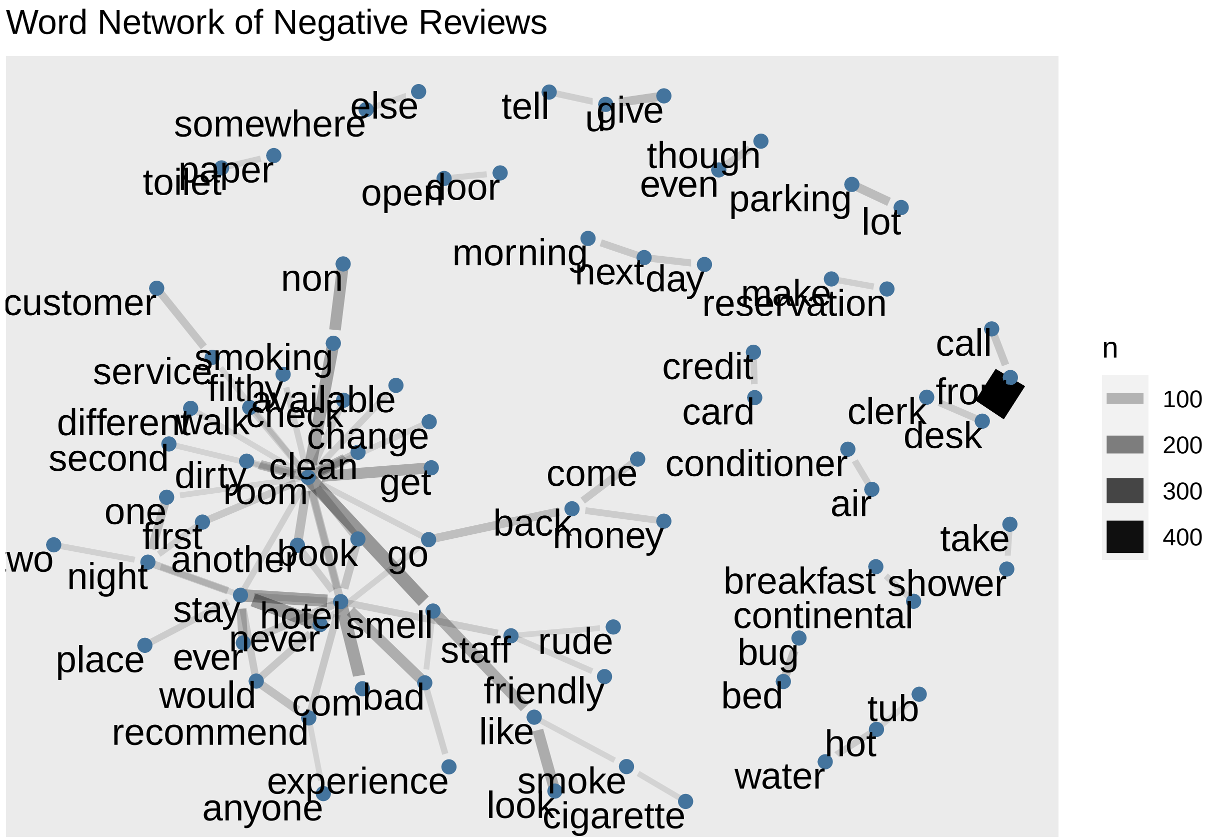
```
## IGRAPH 5c2f6b5 DN-- 79 79 --
## + attr: name (v/c), n (e/n)
## + edges from 5c2f6b5 (vertex names):
##  [1] front   ->desk      room    ->smell     never   ->stay
##  [4] stay    ->hotel     smoking ->room      hotel   ->com
##  [7] non     ->smoking   get     ->room      smell   ->like
## [10] look    ->like      bad     ->hotel     room    ->clean
## [13] room    ->dirty     ever    ->stay      another ->room
## [16] give    ->u         parking ->lot       one     ->night
## [19] bed     ->bug       go      ->back      book    ->room
## [22] would   ->recommend hotel   ->room      book    ->hotel
## + ... omitted several edges
```

```
set.seed(2017)

a <- grid::arrow(type = "closed", length = unit(.15, "inches"))

ggsave("graphs/negative_network.png", width = 15, height = 10, dpi = 300)

ggraph(neg_bigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha=n, edge_width=n),
               end_cap=circle(.07, 'inches')) +
  geom_node_point(color='#44749D', size=2) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1, size=5) +
 labs(title="Word Network of Negative Reviews")
```



All Reviews

```
all_tokens <- all %>%
  unnest_tokens(word, text)


data(stop_words)

all_tokens <- all_tokens %>%
  anti_join(stop_words, by =c("word" = "word"))
```

BiGrams

```
all_bigram <- all %>%
  unnest_tokens(bigram, text, token = "ngrams", n=2)

all_bigram
```

```
## # A tibble: 737,433 x 2
##     line bigram
##    <int> <chr>
##  1     1 pleasant min
##  2     1 min walk
##  3     1 walk along
##  4     1 along sea
##  5     1 sea front
##  6     1 front water
##  7     1 water bus
##  8     1 bus restaurant
##  9     1 restaurant etc
## 10     1 etc hotel
## # ... with 737,423 more rows
```

Most common n-grams

```
all_bigram %>%
  count(bigram, sort = TRUE)
```

```
## # A tibble: 305,149 x 2
##    bigram                n
##    <chr>             <int>
##  1 front desk         2631
##  2 room clean         2533
##  3 staff friendly     1808
##  4 stay hotel         1019
##  5 place stay          999
##  6 one night           982
##  7 friendly helpful    977
##  8 bed comfortable     964
##  9 clean comfortable   963
## 10 would stay          919
## # ... with 305,139 more rows
```

```
# separate into two columns
all_bigrams_separated <- all_bigram %>%
  separate(bigram, c("word1", "word2"), sep=" ")

# new bigram counts:
all_bigram_counts <- all_bigrams_separated %>%
  count(word1, word2, sort = TRUE)

all_bigram_counts
```

```
## # A tibble: 305,149 x 3
##    word1    word2           n
##    <chr>    <chr>       <int>
##  1 front    desk         2631
##  2 room     clean        2533
##  3 staff    friendly     1808
##  4 stay     hotel        1019
##  5 place    stay          999
##  6 one      night         982
##  7 friendly helpful       977
##  8 bed      comfortable   964
##  9 clean    comfortable   963
## 10 would    stay          919
## # ... with 305,139 more rows
```

```r
# filter for only relatively common combinations
all_bigram_graph <- all_bigram_counts %>%
  filter(n > 250) %>%
  graph_from_data_frame()
```

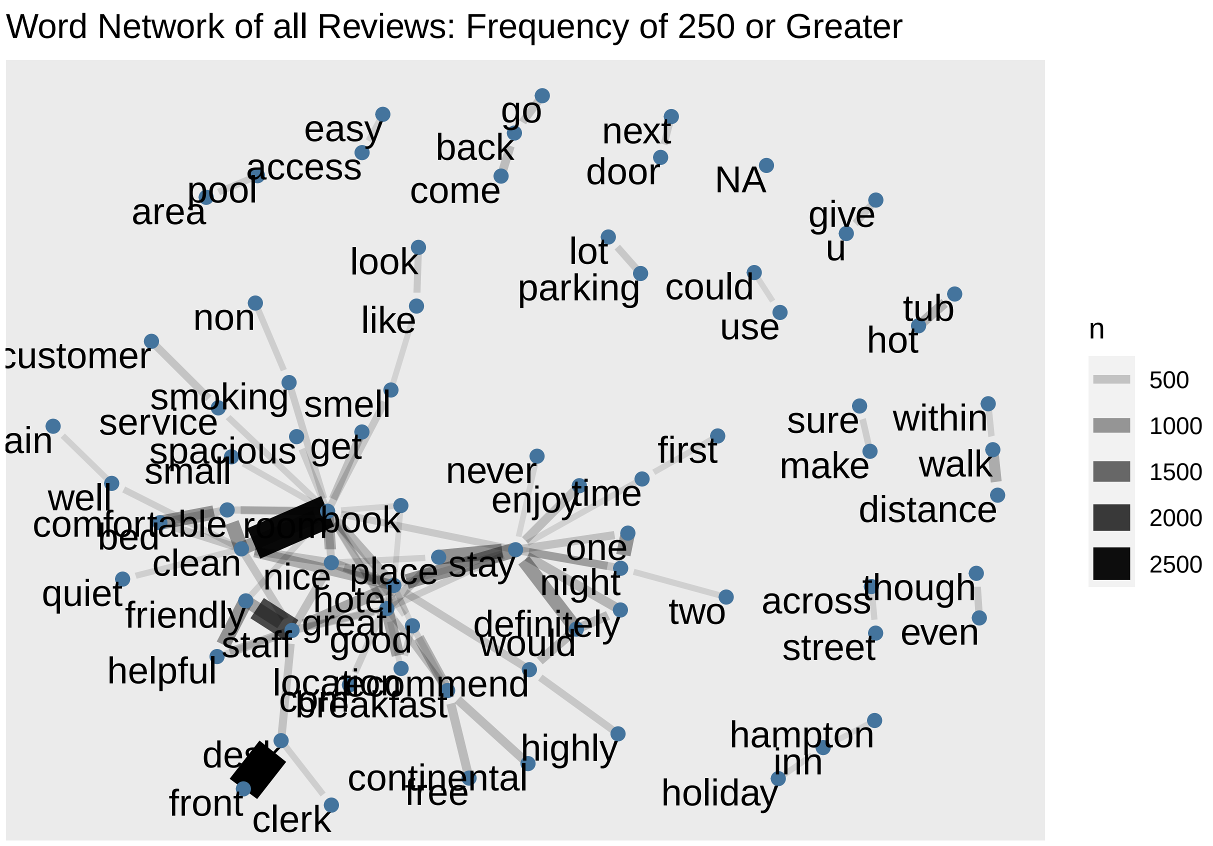
```
## Warning in graph_from_data_frame(.): In `d' `NA' elements were replaced with
## string "NA"
```

```r
set.seed(2017)

a <- grid::arrow(type = "closed", length = unit(.15, "inches"))

ggsave("graphs/network_all_250.png", width = 15, height = 10, dpi = 300)

ggraph(all_bigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha=n, edge_width=n),
                 end_cap=circle(.07, 'inches')) +
  geom_node_point(color='#44749D', size=2) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1, size=5) +
 labs(title="Word Network of all Reviews: Frequency of 250 or Greater")
```



```
# filter for only relatively common combinations
all_bigram_graph <- all_bigram_counts %>%
  filter(n > 150) %>%
  graph_from_data_frame()
```

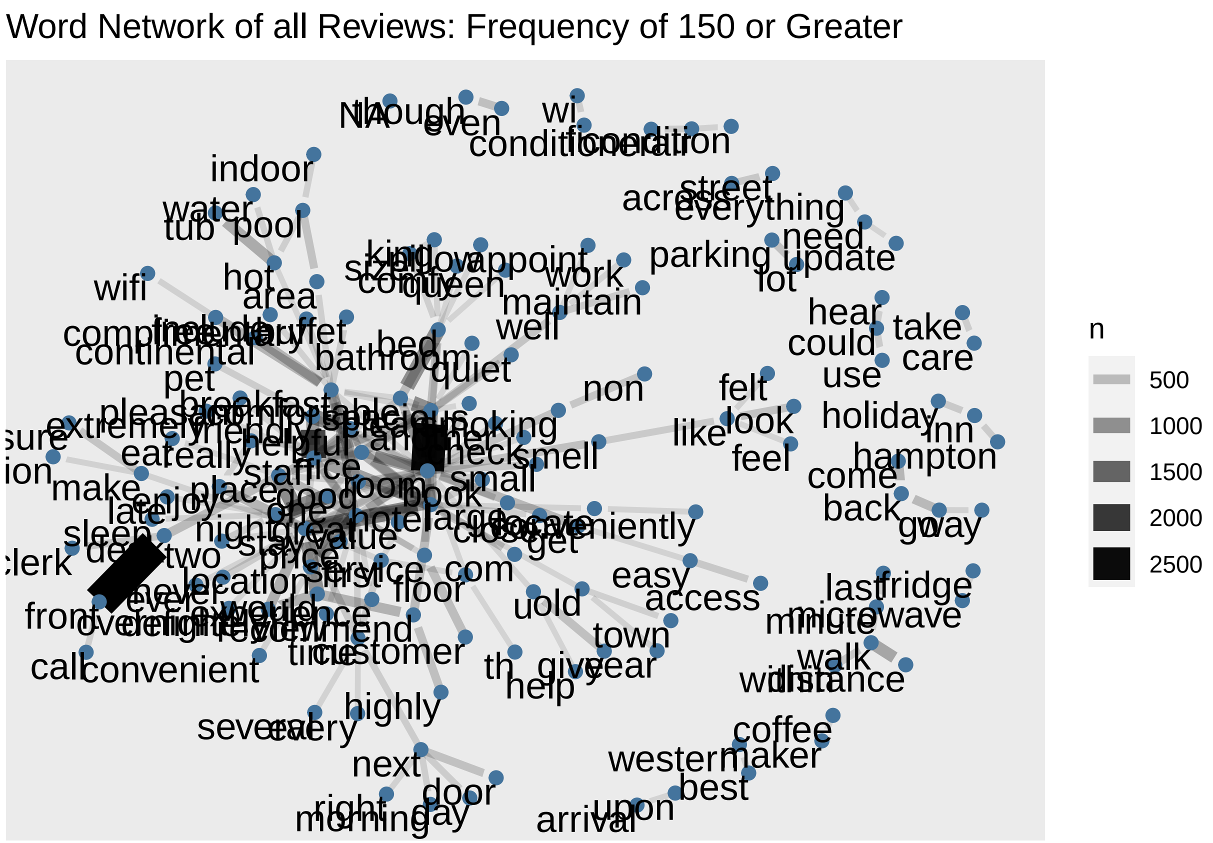
```
## Warning in graph_from_data_frame(.): In `d' `NA' elements were replaced with
## string "NA"
```

```
set.seed(2017)

a <- grid::arrow(type = "closed", length = unit(.15, "inches"))

ggsave("graphs/network_all_150.png", width = 15, height = 10, dpi = 300)

ggraph(all_bigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha=n, edge_width=n),
                 end_cap=circle(.07, 'inches')) +
  geom_node_point(color='#44749D', size=2) +
  geom_node_text(aes(label = name), vjust = 1, hjust = 1, size=5) +
 labs(title="Word Network of all Reviews: Frequency of 150 or Greater")
```



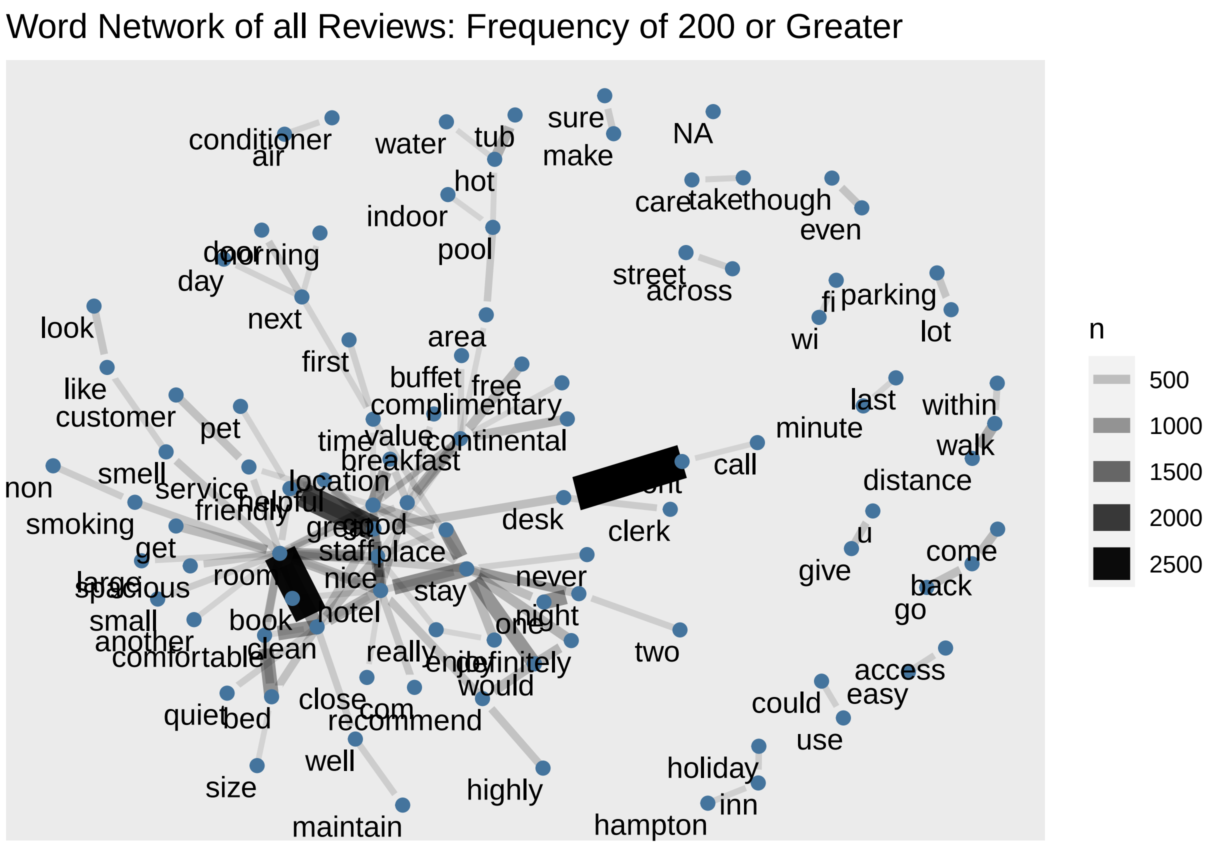
```
# filter for only relatively common combinations
all_bigram_graph <- all_bigram_counts %>%
  filter(n > 200) %>%
  graph_from_data_frame()
```

```
## Warning in graph_from_data_frame(.): In `d' `NA' elements were replaced with
## string "NA"
```

```
set.seed(2017)


ggsave("graphs/network_all_200.png", width = 15, height = 10, dpi = 300)

ggraph(all_bigram_graph, layout = "fr") +
  geom_edge_link(aes(edge_alpha=n, edge_width=n),
                 end_cap=circle(.07, 'inches')) +
  geom_node_point(color='#44749D', size=2) +
  geom_node_text(aes(label = name), vjust = 1.5, hjust = 1, size=4) +
 labs(title="Word Network of all Reviews: Frequency of 200 or Greater")
```


Word Network of all Reviews: Frequency of 200 or Greater