ThesisCode

## Facebook Data Extraction

library(Rfacebook)  
  
# Fb Authorization  
fb\_oauth <- Rfacebook::fbOAuth(  
 app\_id="1380496555352781",  
 app\_secret="eb3abc42d1e00536e6f4e37e58fc0b5d",  
 extended\_permissions = TRUE)  
  
# Saving variable fb\_oauth in a file and loading it  
save(fb\_oauth, file="fb\_oauth")  
load("fb\_oauth")  
  
# Extract posts from turkish airlines page  
turkishairlines <- Rfacebook::getPage(page = "turkishairlines",   
 token = fb\_oauth, n = 2000)  
  
# Save the posts in R data file  
saveRDS(turkishairlines, "turkishairlines\_2000")

library(dplyr)

##   
## 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

# Read R data file and store in new variable  
data\_turkishairlines <- readRDS("turkishairlines\_2000")  
  
# Storing the post ids in a new variable  
data\_turkishairlines\_id <- data\_turkishairlines$id  
  
# Viewing limited variables and rows  
data\_turkishairlines %>%   
 dplyr::select(from\_id, likes\_count, type, comments\_count,   
 shares\_count, message) %>%   
 tibble::as\_tibble() %>%   
 head(5)

## # A tibble: 5 x 6  
## from\_id likes\_count type comments\_count shares\_count message   
## <chr> <dbl> <chr> <dbl> <dbl> <chr>   
## 1 90430042759 254 photo 12 18 Our flights t…  
## 2 90430042759 300 photo 132 92 "We are looki…  
## 3 90430042759 869 photo 39 43 The most plea…  
## 4 90430042759 261 photo 9 7 Famous actor …  
## 5 90430042759 227 photo 13 11 Within the sc…

data\_turkishairlines %>%   
 dplyr::select(message) %>%   
 tibble::as\_tibble() %>%   
 head(2)

## # A tibble: 2 x 1  
## message   
## <chr>   
## 1 Our flights to Samarkand start on March 16th, 2018!   
## 2 "We are looking for new pilots! Join us at Turkish Airlines Pilot Roads…

# Extracting comments and reactions from the extracted posts  
# The comments are extracted using post ids. Therefore, id variable  
# is used.   
all\_posts <- data\_turkishairlines\_id[1:300] %>%   
 lapply(getPost, n = 50000, token=fb\_oauth,   
 comments = TRUE, reactions = TRUE)  
  
# Save first 300 posts comments in a R data file.  
saveRDS(all\_posts, "300\_posts")

# Converting Facebook lists data into dataframe

# Reading the saved RDS file instead of applying getPost function repeatedly.  
# The resultant variable is a list which includes a post specific list.  
# Within each post specific list there are post comments and reactions.  
threehundred\_posts <- readRDS("300\_posts")   
  
# Saving comments for first post in new variable  
full\_comments <- threehundred\_posts[[1]]$comments$message %>%  
 as.data.frame() %>%   
 setNames("comments") %>%  
 dplyr::mutate(id = data\_turkishairlines\_id[1])  
  
# Expanding full\_comments by adding comments of remaining posts.   
for (i in 1:299) {  
 comment <- threehundred\_posts[[i+1]]$comments$message %>%   
 as.data.frame() %>%   
 setNames("comments") %>%  
 dplyr::mutate(id = data\_turkishairlines\_id[i+1])  
 full\_comments = rbind(full\_comments, comment)  
}  
  
# Assigning the post ids to its comments.   
# The posts ids are repeated when there are more than one comment.   
full\_comment\_post <- full\_comments %>%   
 dplyr::left\_join(data\_turkishairlines, by = "id")  
  
# Saving the comments of posts in R data file.   
saveRDS(full\_comment\_post, "300\_posts\_comments")

readRDS("300\_posts\_comments") %>%   
 subset(select = c(3,1,2,4,5,6,7,8,9,10,11)) %>%   
 tibble::as\_tibble() %>%   
 head(5)

## # A tibble: 5 x 11  
## from\_id comments id from\_name message created\_time type link story  
## <chr> <fct> <chr> <chr> <chr> <chr> <chr> <chr> <chr>  
## 1 9043004… Direct … 9043… Turkish … Our fl… 2017-10-05T… photo http… <NA>   
## 2 9043004… Just cu… 9043… Turkish … Our fl… 2017-10-05T… photo http… <NA>   
## 3 9043004… What ab… 9043… Turkish … Our fl… 2017-10-05T… photo http… <NA>   
## 4 9043004… What ab… 9043… Turkish … Our fl… 2017-10-05T… photo http… <NA>   
## 5 9043004… Very in… 9043… Turkish … Our fl… 2017-10-05T… photo http… <NA>   
## # ... with 2 more variables: likes\_count <dbl>, comments\_count <dbl>

readRDS("300\_posts\_comments") %>%   
 dplyr::select(comments) %>%   
 tibble::as\_tibble() %>%   
 head(2)

## # A tibble: 2 x 1  
## comments   
## <fct>   
## 1 Direct flight from Ireland to bodrum badly needed even twice a week in …  
## 2 Just curious you do flights to Tbilisi?

## Twitter Data Extraction

# Load Requried Packages  
library("SnowballC")  
library("tm")  
library("twitteR")  
library("syuzhet")  
  
# Authonitical keys  
consumer\_key <- 'tAyR9LyhATfD90aA7Ft1Zfj3I'  
consumer\_secret <- 'vX1RHqHHDpnmNOqrGPMVnmnQjQvG98X3xlB7T7zv4hKcvj7tVv'  
access\_token <- '2572842085-vExbB4HNvN57zmQhoQdbmutC16a4kdMdh1xVta5'  
access\_secret <- 'HtTHSeAOz1WPcUX8nfW5ddwZ1TbXZGFB4pSHU0IZ3agvA'  
  
twitteR::setup\_twitter\_oauth(consumer\_key, consumer\_secret,   
 access\_token, access\_secret)  
  
tweets <- userTimeline("turkishairlines", n=200)

## Trip Advisor Data Extraction

library("rvest")  
  
url <- "https://www.tripadvisor.com/Airline\_Review-d8729174-Reviews-Turkish-Airlines"  
url <- "https://www.tripadvisor.com/Airline\_Review-d8729174-Reviews-or20-Turkish-Airlines#REVIEWS"  
  
df\_total = data.frame()  
  
for (i in seq(0, 20050, 10))  
{  
 if (i == 0) {  
 url <- "https://www.tripadvisor.com/Airline\_Review-d8729069-Reviews-Emirates"  
 }  
   
 else {  
 url <- paste(  
 "https://www.tripadvisor.com/Airline\_Review-d8729069-Reviews-or",i,"-Emirates#REVIEWS",   
 sep = "")  
 }  
   
 reviews <- url %>%  
 read\_html() %>%  
 html\_nodes("#REVIEWS .innerBubble")  
   
 id <- reviews %>%  
 html\_node(".quote a") %>%  
 html\_attr("id")  
   
 quote <- reviews %>%  
 html\_node(".quote span") %>%  
 html\_text()  
   
 rating <- reviews %>%  
 html\_node(".rating .rating\_s\_fill") %>%  
 html\_attr("alt") %>%  
 gsub(" of 5 stars", "", .) %>%  
 as.integer()  
   
 date <- reviews %>%  
 html\_node(".rating .ratingDate") %>%  
 html\_attr("title") %>%  
 strptime("%b %d, %Y") %>%  
 as.POSIXct()  
   
 review <- reviews %>%  
 html\_node(".entry .partial\_entry") %>%  
 html\_text()  
   
 df <- data.frame(id, quote, rating, date, review, stringsAsFactors = FALSE)  
 df\_total <- rbind(df\_total, df)  
}  
  
# Save an object to a file  
saveRDS(df\_total, file = "tripadvisor\_turkishairlines6846.rds")

trip\_turkishairlines <- readRDS(file = "tripadvisor\_turkishairlines6846.rds")  
  
trip\_turkishairlines %>%  
 dplyr::select(id, date, quote, rating, review) %>%   
 tibble::as\_tibble() %>%   
 head(5)

## # A tibble: 5 x 5  
## id date quote rating review   
## <chr> <dttm> <chr> <int> <chr>   
## 1 rn575914601 2018-04-26 00:00:00 Never ag… NA "\nTo start with me an…  
## 2 rn575863809 2018-04-26 00:00:00 Istanbul… NA "\nTurkish airlines is…  
## 3 rn575859234 2018-04-26 00:00:00 Broke my… NA "\nTravelled from Atat…  
## 4 rn575854758 2018-04-26 00:00:00 comforta… NA "\nit is amazing trave…  
## 5 rn575817214 2018-04-26 00:00:00 Despite … NA "\nAlways like travell…

trip\_turkishairlines %>%   
 dplyr::select(review) %>%   
 tibble::as\_tibble() %>%   
 head(1)

## # A tibble: 1 x 1  
## review   
## <chr>   
## 1 "\nTo start with me and my sister had a connection in Istanbul where th…

# Tokenization

library(tidytext)  
data(stop\_words)  
  
# Custom stop words  
custom\_stop\_words <- data.frame(word = c("miss", "flight", "tukish",   
 "airlines", "flights",   
 "airline", "turkish", "de"),   
 lexicon = c("custom")) %>%  
 rbind(stop\_words)  
  
tokenize <- function(file, data\_type) {  
 data\_tibble <- readRDS(file = file) %>%   
 tibble::as\_tibble()  
   
 data\_vector <- data\_tibble %>%   
 dplyr::pull(data\_type) %>%   
 iconv(from = "UTF-8", to = "Latin1")  
   
 tokens <- tibble::as\_tibble(data\_vector) %>%   
 dplyr::filter(!is.na(value)) %>%   
 dplyr::mutate(response\_number = rownames(.)) %>%   
 dplyr::select(response\_number, value) %>%   
 tidytext::unnest\_tokens(word, value)  
   
 tokens  
}  
  
# Convert non\_base verbs into base verbs  
  
extract\_non\_base <- function(data) {  
 data %>%  
 dplyr::rename(non\_base = word) %>%  
 dplyr::left\_join(readRDS("sahban\_base\_lexicon"), by = "non\_base") %>%  
 dplyr::mutate(base = ifelse(is.na(base), non\_base, base)) %>%  
 dplyr::rename(word = base) %>%  
 dplyr::select(-one\_of("non\_base"))   
}  
  
facebook\_tokens <- tokenize("300\_posts\_comments", "comments") %>%   
 extract\_non\_base()  
tripadvisor\_tokens <- tokenize("tripadvisor\_turkishairlines6846.rds", "review") %>%   
 extract\_non\_base()

saveRDS(tokens\_count, "tokens\_count\_300")  
saveRDS(tokens\_count, "TA\_tokens\_count\_6846")

facebook\_tokens %>%   
 tibble::as\_tibble()

## # A tibble: 32,164 x 2  
## response\_number word   
## <chr> <chr>   
## 1 1 direct   
## 2 1 flight   
## 3 1 from   
## 4 1 ireland  
## 5 1 to   
## 6 1 bodrum   
## 7 1 badly   
## 8 1 need   
## 9 1 even   
## 10 1 twice   
## # ... with 32,154 more rows

tripadvisor\_tokens %>%   
 tibble::as\_tibble()

## # A tibble: 325,228 x 2  
## response\_number word   
## <chr> <chr>   
## 1 1 to   
## 2 1 start   
## 3 1 with   
## 4 1 me   
## 5 1 and   
## 6 1 my   
## 7 1 sister   
## 8 1 had   
## 9 1 a   
## 10 1 connection  
## # ... with 325,218 more rows

# Word Count

word\_count <- function(data) {  
 readRDS(data) %>%   
 dplyr::anti\_join(custom\_stop\_words, by = "word") %>%   
 dplyr::count(word, sort = TRUE) %>%   
 tibble::as\_tibble()  
}  
  
facebook\_word\_count <- word\_count("facebook\_tokens")

## Warning: Column `word` joining character vector and factor, coercing into  
## character vector

tripadvisor\_word\_count <- word\_count("tripadvisor\_tokens")

## Warning: Column `word` joining character vector and factor, coercing into  
## character vector

facebook\_word\_count

## # A tibble: 7,715 x 2  
## word n  
## <chr> <int>  
## 1 fly 236  
## 2 love 224  
## 3 istanbul 185  
## 4 nice 141  
## 5 service 96  
## 6 3 82  
## 7 time 79  
## 8 travel 76  
## 9 day 67  
## 10 world 66  
## # ... with 7,705 more rows

tripadvisor\_word\_count

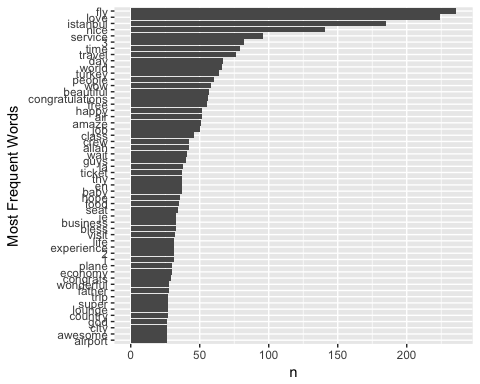
## # A tibble: 9,198 x 2  
## word n  
## <chr> <int>  
## 1 service 3067  
## 2 food 2994  
## 3 istanbul 2531  
## 4 fly 2431  
## 5 time 2310  
## 6 seat 1991  
## 7 staff 1419  
## 8 class 1149  
## 9 comfortable 1002  
## 10 airport 949  
## # ... with 9,188 more rows

# Word Count Plot

word\_count\_plot <- function(data, min\_count) {  
 tokens\_count <- readRDS(data)  
   
 tokens\_count %>%   
 dplyr::anti\_join(custom\_stop\_words) %>%   
 dplyr::filter(n > min\_count) %>%  
 dplyr::mutate(word = reorder(word, n)) %>%  
 ggplot2::ggplot(mapping = ggplot2::aes(word, n)) +  
 ggplot2::geom\_col() +  
 ggplot2::xlab("Most Frequent Words") +   
 ggplot2::coord\_flip()  
}  
  
word\_count\_plot("tokens\_count\_300", 25)

## Joining, by = "word"

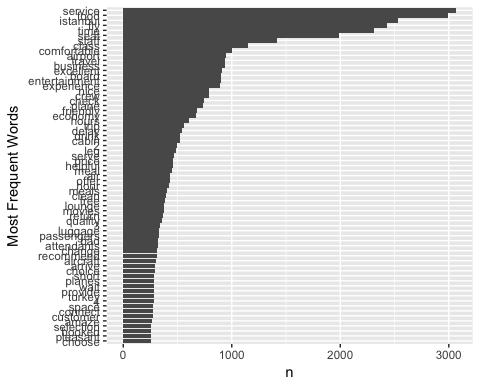
## Warning: Column `word` joining character vector and factor, coercing into  
## character vector



word\_count\_plot("TA\_tokens\_count\_6846", 250)

## Joining, by = "word"

## Warning: Column `word` joining character vector and factor, coercing into  
## character vector



# Adding words manually to the lexicon  
sahban\_base\_lexicon <- readRDS("sahban\_base\_lexicon") %>%   
 dplyr::bind\_rows(data.frame(base = c("travel", "travel"),   
 non\_base = c("travelled", "travelling")))  
saveRDS(sahban\_base\_lexicon, "sahban\_base\_lexicon")

# Indexed Response Function

numbered\_response\_tokens <- function(file, response\_type) {  
   
 dataset <- readRDS(file = file)  
   
 dataset %>%   
 tibble::as\_tibble() %>%   
 dplyr::mutate(response\_type =   
 iconv(pull(., response\_type),   
 from = "UTF-8", to = "Latin1")) %>%   
 dplyr::mutate(post\_number = as.numeric(factor(id))) %>%   
 dplyr::group\_by(id) %>%   
 dplyr::mutate\_if(is.factor, as.character)  
}

# Convert non\_base verbs into base verbs  
  
extract\_non\_base <- function(data) {  
 data %>%  
 dplyr::rename(non\_base = word) %>%  
 dplyr::left\_join(readRDS("sahban\_base\_lexicon"), by = "non\_base") %>%  
 dplyr::mutate(base = ifelse(is.na(base), non\_base, base)) %>%  
 dplyr::rename(word = base) %>%  
 dplyr::select(-one\_of("non\_base"))   
}

# Indexed Tokenization

The numbererd\_response\_function() is used to find out numbered tokenization.

tidy\_response\_facebook <-   
 numbered\_response\_tokens("300\_posts\_comments", "comments") %>%   
 dplyr::mutate(comment\_number = row\_number()) %>%   
 dplyr::select(id, post\_number, response\_number = comment\_number,   
 comments, created\_time, type, likes\_count, comments\_count,   
 shares\_count) %>%   
 tidytext::unnest\_tokens(word, comments) %>%   
 dplyr::anti\_join(custom\_stop\_words) %>%   
 extract\_non\_base()

## Joining, by = "word"

## Warning: Column `word` joining character vector and factor, coercing into  
## character vector

tidy\_response\_tripadvisor <-   
 numbered\_response\_tokens("tripadvisor\_turkishairlines6846.rds", "review") %>%   
 dplyr::select(id, response\_number = post\_number, review, quote, rating, date) %>%   
 tidytext::unnest\_tokens(word, review) %>%   
 dplyr::anti\_join(custom\_stop\_words) %>%   
 extract\_non\_base()

## Joining, by = "word"

## Warning: Column `word` joining character vector and factor, coercing into  
## character vector

tidy\_response\_facebook %>%   
 tibble::as\_tibble() %>%   
 dplyr::select(post\_number, response\_number, word, type) %>%   
 subset(select = c(2,3,4,5))

## Adding missing grouping variables: `id`

## # A tibble: 23,912 x 4  
## post\_number response\_number word type   
## <dbl> <int> <chr> <chr>  
## 1 293 1 direct photo  
## 2 293 1 ireland photo  
## 3 293 1 bodrum photo  
## 4 293 1 badly photo  
## 5 293 1 week photo  
## 6 293 1 summer photo  
## 7 293 1 fantastic photo  
## 8 293 2 curious photo  
## 9 293 2 tbilisi photo  
## 10 293 3 fligths photo  
## # ... with 23,902 more rows

tidy\_response\_tripadvisor %>%   
 tibble::as\_tibble() %>%   
 dplyr::select(id, response\_number, word, quote)

## # A tibble: 124,181 x 4  
## # Groups: id [6,846]  
## id response\_number word quote   
## <chr> <dbl> <chr> <chr>   
## 1 rn575914601 6846 start Never again..!!  
## 2 rn575914601 6846 sister Never again..!!  
## 3 rn575914601 6846 connection Never again..!!  
## 4 rn575914601 6846 istanbul Never again..!!  
## 5 rn575914601 6846 staff Never again..!!  
## 6 rn575914601 6846 sister Never again..!!  
## 7 rn575914601 6846 15 Never again..!!  
## 8 rn575914601 6846 minutes Never again..!!  
## 9 rn575914601 6846 explanation Never again..!!  
## 10 rn575914601 6846 talk Never again..!!  
## # ... with 124,171 more rows

# Sentence Tokenization

untidy\_response\_facebook <-   
 numbered\_response\_tokens("300\_posts\_comments", "comments") %>%   
 dplyr::mutate(comment\_number = row\_number()) %>%   
 dplyr::select(id, post\_number, comment\_number, comments,   
 created\_time, type, likes\_count, comments\_count,   
 shares\_count)  
  
untidy\_response\_tripadvisor <-   
 numbered\_response\_tokens("tripadvisor\_turkishairlines6846.rds", "review") %>%   
 dplyr::select(id, post\_number, review, quote, rating, date)  
  
  
sentence\_tokens <- function(data = untidy\_response\_facebook,   
 response\_column = "comments",   
 group\_by = "comment\_number") {  
 # English Dictionary  
 qdapDictionaries::DICTIONARY[,1]  
   
 en\_word\_comments <- data %>%   
 dplyr::ungroup() %>%   
 tidytext::unnest\_tokens\_("word", response\_column) %>%   
 dplyr::filter(word %in% qdapDictionaries::DICTIONARY[,1])  
   
 en\_word\_sentence\_comments <- en\_word\_comments %>%   
 dplyr::group\_by\_("id", group\_by) %>%   
 dplyr::mutate(sentence = paste(word, collapse = " ")) %>%  
 dplyr::distinct(sentence, .keep\_all = TRUE) %>%   
 dplyr::as\_data\_frame() %>%   
 dplyr::mutate(sentence = iconv(sentence, to = 'latin1')) %>%   
 dplyr::ungroup()  
   
 # Sentence as tokens with post number and comment number  
   
 en\_word\_sentence\_comments %>%   
 dplyr::select\_("id", group\_by, "sentence") %>%  
 dplyr::ungroup() %>%   
 tidytext::unnest\_tokens(sentences, sentence, token = "sentences")  
}  
  
facebook\_sentence\_tokens <-   
 sentence\_tokens(data = untidy\_response\_facebook,  
 response\_column = "comments",  
 group\_by = "comment\_number")  
  
tripadvisor\_sentence\_tokens <-   
 sentence\_tokens(data = untidy\_response\_tripadvisor,  
 response\_column = "review",  
 group\_by = "post\_number")  
  
facebook\_sentence\_tokens   
  
tripadvisor\_sentence\_tokens  
  
tripadvisor\_sentence\_tokens  
 dplyr::select(sentences)

# Response sentiments

response\_sentiments <- function(data, lexicon, group\_by = sentiment) {  
 data %>%   
 dplyr::inner\_join(get\_sentiments(lexicon), by = "word") %>%   
 dplyr::count(response\_number, sentiment) %>%   
 tidyr::spread(sentiment, n, fill = 0) %>%   
 dplyr::mutate(sentiment = positive - negative) %>%   
 dplyr::ungroup()  
}  
  
# bing  
facebook\_sentiments\_bing <- response\_sentiments(tidy\_response\_facebook, "bing")  
tripadvisor\_sentiments\_bing <- response\_sentiments(tidy\_response\_tripadvisor, "bing")  
  
# nrc  
facebook\_sentiments\_nrc <- response\_sentiments(tidy\_response\_facebook, "nrc")  
tripadvisor\_sentiments\_nrc <- response\_sentiments(tidy\_response\_tripadvisor, "nrc")  
  
#afinn  
sentiments\_afinn <- function(data) {  
 data %>%   
 dplyr::ungroup() %>%   
 dplyr::inner\_join(tidytext::get\_sentiments("afinn"), by = "word") %>%   
 dplyr::group\_by(id, response\_number) %>%   
 dplyr::summarise(score = sum(score))   
}  
  
facebook\_sentiments\_afinn <- sentiments\_afinn(tidy\_response\_facebook)  
tripadvisor\_sentiments\_afinn <- sentiments\_afinn(tidy\_response\_tripadvisor)  
  
facebook\_sentiments\_bing %>%   
 dplyr::select(response\_number, negative, positive, sentiment)

## # A tibble: 1,586 x 4  
## response\_number negative positive sentiment  
## <int> <dbl> <dbl> <dbl>  
## 1 1 0 1 1  
## 2 2 0 1 1  
## 3 3 0 1 1  
## 4 3 1 0 -1  
## 5 4 0 1 1  
## 6 2 0 1 1  
## 7 3 3 1 -2  
## 8 18 0 1 1  
## 9 21 0 2 2  
## 10 23 0 1 1  
## # ... with 1,576 more rows

tripadvisor\_sentiments\_bing

## # A tibble: 6,423 x 5  
## id response\_number negative positive sentiment  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 rn342674723 1 1 0 -1  
## 2 rn342740773 2 1 3 2  
## 3 rn342772345 3 4 1 -3  
## 4 rn342937701 4 0 3 3  
## 5 rn343011985 5 1 1 0  
## 6 rn343117108 6 0 1 1  
## 7 rn343117743 7 0 2 2  
## 8 rn343141565 8 4 2 -2  
## 9 rn343168849 9 0 4 4  
## 10 rn343189478 10 0 3 3  
## # ... with 6,413 more rows

facebook\_sentiments\_nrc

## # A tibble: 1,867 x 13  
## id response\_number anger anticipation disgust fear joy negative  
## <chr> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 904300… 1 0 0 0 0 2 0  
## 2 904300… 2 0 0 0 0 1 0  
## 3 904300… 3 0 0 0 0 1 0  
## 4 904300… 4 0 0 0 0 0 0  
## 5 904300… 1 0 1 0 0 0 1  
## 6 904300… 4 0 0 0 0 0 0  
## 7 904300… 2 1 2 0 1 1 0  
## 8 904300… 3 0 0 0 0 0 3  
## 9 904300… 4 0 0 0 1 0 0  
## 10 904300… 5 0 1 0 0 0 0  
## # ... with 1,857 more rows, and 5 more variables: positive <dbl>,  
## # sadness <dbl>, surprise <dbl>, trust <dbl>, sentiment <dbl>

tripadvisor\_sentiments\_nrc

## # A tibble: 6,788 x 13  
## id response\_number anger anticipation disgust fear joy negative  
## <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 rn3426… 1 0 2 0 0 0 2  
## 2 rn3427… 2 1 2 1 2 3 0  
## 3 rn3427… 3 2 0 2 1 0 3  
## 4 rn3429… 4 0 2 0 0 3 0  
## 5 rn3430… 5 1 2 1 1 1 1  
## 6 rn3431… 6 0 2 0 0 1 0  
## 7 rn3431… 7 0 3 0 0 4 0  
## 8 rn3431… 8 1 1 1 2 2 4  
## 9 rn3431… 9 0 3 0 0 3 0  
## 10 rn3431… 10 1 2 0 1 2 0  
## # ... with 6,778 more rows, and 5 more variables: positive <dbl>,  
## # sadness <dbl>, surprise <dbl>, trust <dbl>, sentiment <dbl>

facebook\_sentiments\_afinn

## # A tibble: 1,645 x 3  
## # Groups: id [?]  
## id response\_number score  
## <chr> <int> <int>  
## 1 90430042759\_10155125620582760 1 5  
## 2 90430042759\_10155125620582760 2 3  
## 3 90430042759\_10155125620582760 3 3  
## 4 90430042759\_10155125620582760 7 2  
## 5 90430042759\_10155126945012760 3 2  
## 6 90430042759\_10155126945012760 4 2  
## 7 90430042759\_10155129495572760 2 3  
## 8 90430042759\_10155129495572760 3 -3  
## 9 90430042759\_10155129495572760 11 -1  
## 10 90430042759\_10155129495572760 18 1  
## # ... with 1,635 more rows

tripadvisor\_sentiments\_afinn

## # A tibble: 6,131 x 3  
## # Groups: id [?]  
## id response\_number score  
## <chr> <dbl> <int>  
## 1 rn342674723 1 0  
## 2 rn342740773 2 7  
## 3 rn342772345 3 -2  
## 4 rn342937701 4 5  
## 5 rn343011985 5 -3  
## 6 rn343117108 6 2  
## 7 rn343117743 7 5  
## 8 rn343141565 8 -5  
## 9 rn343168849 9 6  
## 10 rn343189478 10 7  
## # ... with 6,121 more rows

# Sentiments to All Words Ratio

sentiment\_token\_ratio <- function(data, sentiment\_type = "negative") {  
   
 negative\_sentiment <- get\_sentiments("bing") %>%   
 dplyr::filter(sentiment == sentiment\_type)  
   
 wordcounts <- data %>%   
 dplyr::group\_by(response\_number) %>%  
 dplyr::summarize(word = n())  
   
 data %>%  
 dplyr::semi\_join(negative\_sentiment) %>%  
 dplyr::group\_by(id, response\_number) %>%  
 dplyr::summarize(negativewords = n()) %>%  
 dplyr::left\_join(wordcounts, by = c("response\_number")) %>%  
 dplyr::mutate(ratio = negativewords/word) %>%  
 dplyr::top\_n(10) %>%  
 dplyr::ungroup() %>%   
 dplyr::arrange(desc(ratio))  
   
}  
  
facebook\_negative\_sentiment\_ratio <-   
 sentiment\_token\_ratio(tidy\_response\_facebook, "negative")

## Joining, by = "word"

## Selecting by ratio

tripadvisor\_negative\_sentiment\_ratio <-   
 sentiment\_token\_ratio(tidy\_response\_tripadvisor, "negative")

## Joining, by = "word"  
## Selecting by ratio

facebook\_positive\_sentiment\_ratio <-   
 sentiment\_token\_ratio(tidy\_response\_facebook, "positive")

## Joining, by = "word"  
## Selecting by ratio

tripadvisor\_positive\_sentiment\_ratio <-   
 sentiment\_token\_ratio(tidy\_response\_tripadvisor, "positive")

## Joining, by = "word"  
## Selecting by ratio

facebook\_negative\_sentiment\_ratio

## # A tibble: 192 x 5  
## id response\_number negativewords word ratio  
## <chr> <int> <int> <int> <dbl>  
## 1 90430042759\_101551815520227… 281 1 2 0.5   
## 2 90430042759\_101551815520227… 805 1 2 0.5   
## 3 90430042759\_101551815520227… 661 1 3 0.333   
## 4 90430042759\_101551815520227… 687 1 3 0.333   
## 5 90430042759\_101551815520227… 503 2 11 0.182   
## 6 90430042759\_101551815520227… 358 1 6 0.167   
## 7 90430042759\_101551815520227… 314 1 7 0.143   
## 8 90430042759\_101551815520227… 157 1 8 0.125   
## 9 90430042759\_101551815520227… 475 1 10 0.1   
## 10 90430042759\_101551815520227… 69 2 30 0.0667  
## # ... with 182 more rows

tripadvisor\_negative\_sentiment\_ratio

## # A tibble: 3,384 x 5  
## id response\_number negativewords word ratio  
## <chr> <dbl> <int> <int> <dbl>  
## 1 rn541387435 4976 5 7 0.714  
## 2 rn465760178 3008 4 8 0.5   
## 3 rn525881971 4705 4 8 0.5   
## 4 rn518927507 4384 5 11 0.455  
## 5 rn513737468 3993 4 9 0.444  
## 6 rn425622278 2284 9 21 0.429  
## 7 rn557983424 5965 3 7 0.429  
## 8 rn569818446 6477 3 7 0.429  
## 9 rn530267095 4794 4 10 0.4   
## 10 rn547414313 5547 4 10 0.4   
## # ... with 3,374 more rows

numbered\_response\_tokens(  
 "tripadvisor\_turkishairlines6846.rds", "review") %>%   
 dplyr::select(id, response\_number = post\_number, review,   
 quote, rating, date) %>%   
 dplyr::filter(response\_number == 4976) %>%   
 dplyr::select(review, id)

## # A tibble: 1 x 2  
## # Groups: id [1]  
## review id   
## <chr> <chr>   
## 1 "\nthe flight was very bad and the food very bad it so bad and … rn5413…

# Most Frequent Sentiments

frequent\_sentiments <- function(data) {  
 data %>%   
 dplyr::inner\_join(tidytext::get\_sentiments("bing")) %>%   
 dplyr::ungroup() %>%   
 dplyr::count(word, sentiment, sort = TRUE)  
}  
  
facebook\_frequent\_sentiments <- frequent\_sentiments(tidy\_response\_facebook)

## Joining, by = "word"

tripadvisor\_frequent\_sentiments <- frequent\_sentiments(tidy\_response\_tripadvisor)

## Joining, by = "word"

facebook\_frequent\_sentiments

## # A tibble: 384 x 3  
## word sentiment n  
## <chr> <chr> <int>  
## 1 love positive 328  
## 2 nice positive 155  
## 3 beautiful positive 87  
## 4 congratulations positive 83  
## 5 wow positive 80  
## 6 amaze positive 73  
## 7 free positive 72  
## 8 happy positive 69  
## 9 bless positive 45  
## 10 super positive 37  
## # ... with 374 more rows

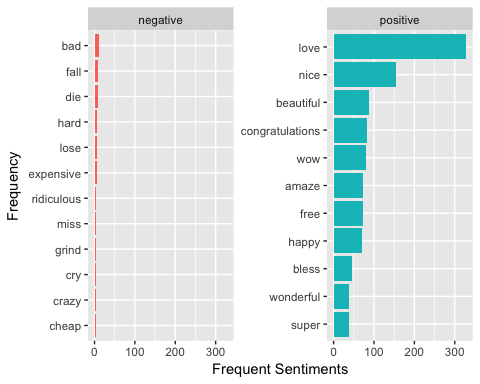
tripadvisor\_frequent\_sentiments

## # A tibble: 1,317 x 3  
## word sentiment n  
## <chr> <chr> <int>  
## 1 comfortable positive 1026  
## 2 excellent positive 927  
## 3 nice positive 816  
## 4 friendly positive 700  
## 5 delay negative 552  
## 6 helpful positive 472  
## 7 clean positive 406  
## 8 free positive 399  
## 9 bad negative 324  
## 10 recommend positive 318  
## # ... with 1,307 more rows

# Plot Frequent Sentiments Counts

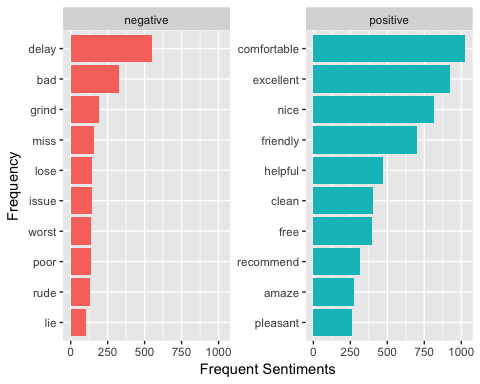
plot\_sentiment\_count <- function(data) {  
 data %>%   
 dplyr::group\_by(sentiment) %>%   
 dplyr::top\_n(10) %>%   
 dplyr::ungroup() %>%   
 dplyr::mutate(word = reorder(word, n)) %>%   
 ggplot2::ggplot(ggplot2::aes(word, n, fill = sentiment)) +  
 ggplot2::geom\_col(show.legend = FALSE) +  
 ggplot2::facet\_wrap(~sentiment, scales = "free\_y") +  
 ggplot2::labs(y = "Frequent Sentiments",  
 x = "Frequency") +  
 ggplot2::coord\_flip()  
}  
  
plot\_sentiment\_count(facebook\_frequent\_sentiments)

## Selecting by n



plot\_sentiment\_count(tripadvisor\_frequent\_sentiments)

## Selecting by n



# Word Cloud

library(wordcloud)

## Loading required package: RColorBrewer

word\_cloud <- function(data, max\_words) {  
 data %>%  
 dplyr::ungroup() %>%   
 dplyr::count(word) %>%  
 with(wordcloud::wordcloud(word, n, max.words = max\_words))  
}  
  
word\_cloud(tidy\_response\_facebook, max\_words = 50)



word\_cloud(tidy\_response\_tripadvisor, max\_words = 75)

## Warning in wordcloud::wordcloud(word, n, max.words = max\_words): istanbul  
## could not be fit on page. It will not be plotted.

## Warning in wordcloud::wordcloud(word, n, max.words = max\_words): experience  
## could not be fit on page. It will not be plotted.



# Sentiment Cloud

library(reshape2)  
  
sentiment\_cloud <- function(dataset) {  
 dataset %>%  
 dplyr::inner\_join(tidytext::get\_sentiments("bing")) %>%  
 dplyr::count(word, sentiment, sort = TRUE) %>%  
 reshape2::acast(word ~ sentiment, value.var = "n", fill = 0) %>%  
 wordcloud::comparison.cloud(colors = c("#F8766D", "#00BFC4"),  
 max.words = 100)  
}  
  
sentiment\_cloud(tidy\_response\_facebook)

## Joining, by = "word"

## Aggregation function missing: defaulting to length



sentiment\_cloud(tidy\_response\_tripadvisor)

## Joining, by = "word"  
## Aggregation function missing: defaulting to length

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : pleasant could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : enjoy could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : perfect could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : smile could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : smooth could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : efficient could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : impress could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : decent could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : pretty could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : wonderful could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : modern could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : fantastic could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : variety could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : courteous could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : happy could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : reasonable could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : spacious could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : warm could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : fresh could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : worth could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : outstanding could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : delight could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : pleasantly could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : prefer could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : lovely could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : superb could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : unfriendly could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : enjoyable could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : comfy could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : convenient could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : pleasure could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : disappointing could not be fit on page. It will not be  
## plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : downside could not be fit on page. It will not be plotted.

## Warning in wordcloud::comparison.cloud(., colors = c("#F8766D",  
## "#00BFC4"), : awesome could not be fit on page. It will not be plotted.

