Amazon Review Analysis

```
# install.packages("stringi")
# install.packages("stringr")
# install.packages("qdap")
# install.packages("rJava")
# install.packages("ggthemes")
# install.packages("gutenbergr")
# install.packages("janeaustenr")
# install.packages("tm")
# install.packages("tidyr")
# install.packages("ggplot2")
# install.packages("scales")
# install.packages("tidytext")
# install.packages("SnowballC")
# install.packages("hunspell")
# install.packages("tokenizers")
# install.packages("dplyr")
#install.packages("wordcloud2")
```

load pakcages

```
## Loading required package: qdapDictionaries

## Loading required package: qdapRegex

## Loading required package: qdapTools

## Loading required package: RColorBrewer

##
## Attaching package: 'qdap'

## The following objects are masked from 'package:base':
##
##     Filter, proportions

## Loading required package: NLP

##
## Attaching package: 'NLP'

## The following object is masked from 'package:qdap':
##
##     ngrams
```

```
##
## Attaching package: 'tm'

## The following objects are masked from 'package:qdap':
##
##     as.DocumentTermMatrix, as.TermDocumentMatrix

##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:NLP':
##
##     annotate

## The following object is masked from 'package:qdapRegex':
##
##     %+%

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:qdapTools':
##
##     id

## The following object is masked from 'package:qdapRegex':
##
##     explain

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

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

##
## Attaching package: 'reshape2'

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

##
## Attaching package: 'igraph'

## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
## The following object is masked from 'package:tidyr':
## 
##     crossing

## The following object is masked from 'package:qdap':
## 
##     diversity

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

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

Explore dataset

```
amazon <- read.csv("Amazon_UnlockedMobile.csv")

# Check for missing values in features of interest

anyNA(amazon$Reviews)

## [1] FALSE

anyNA(amazon$Rating)

## [1] FALSE

head(amazon)

## 
Product.Name
## 1 Acer Liquid Jade Z Andoid KitKat Unlocked Quad-Core 5" IPS Smartphone - 
Retail Packaging - Charcoal Gray
## 2 Acer Liquid Jade Z Andoid KitKat Unlocked Quad-Core 5" IPS Smartphone - 
Retail Packaging - Charcoal Gray
## 3 Acer Liquid Jade Z Andoid KitKat Unlocked Quad-Core 5" IPS Smartphone - 
Retail Packaging - Charcoal Gray
## 4 Acer Liquid Jade Z Andoid KitKat Unlocked Quad-Core 5" IPS Smartphone - 
Retail Packaging - Charcoal Gray
## 5                                                          Acer Liquid M22
0 Windows 8.1 Smartphone - Black
## 6                                                          Acer Liquid M22
0 Windows 8.1 Smartphone - Black
##   Brand.Name  Price Rating
## 1       Acer 129.99      1
## 2       Acer 129.99      2
## 3       Acer 129.99      1
## 4       Acer 129.99      2
## 5       Acer  34.95      3
## 6       Acer  34.95      5
##
Reviews
## 1
```

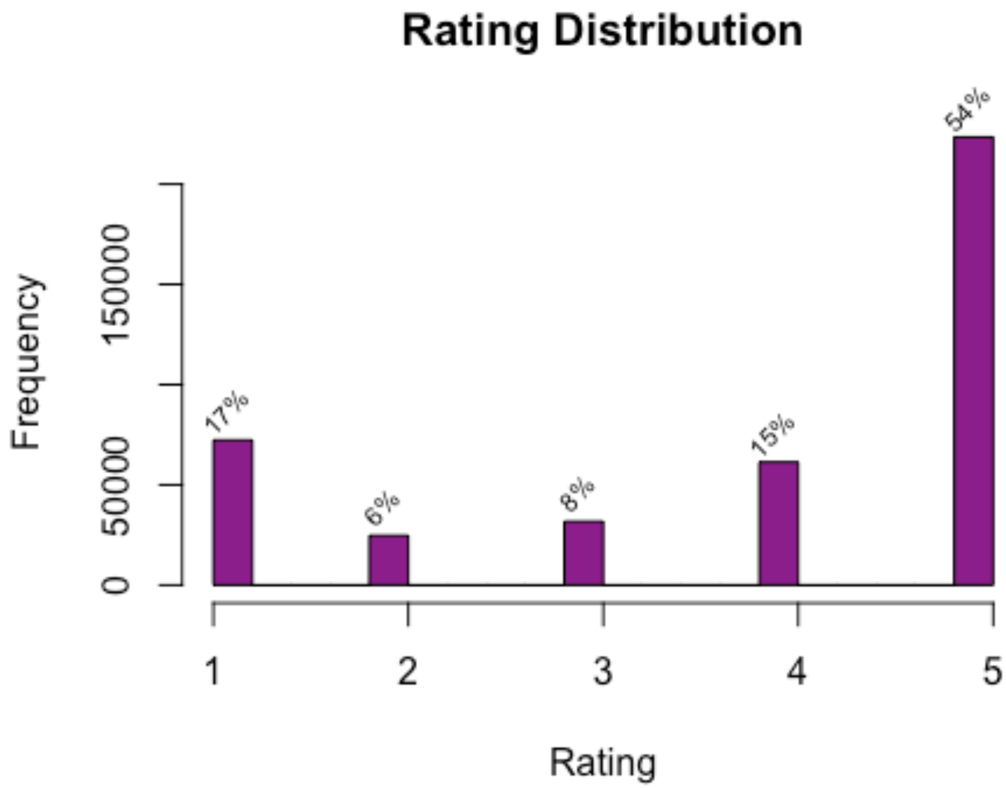
```
categorical_col<- c("Brand.Name", "Reviews","Product.Name")


summary(amazon[, !(colnames(amazon) %in% categorical_col), drop = FALSE])
```

```
##      Price             Rating       Review.Votes
##  Min.   :   1.73   Min.   :1.000   Min.   :  0.000
##  1st Qu.:  79.99   1st Qu.:3.000   1st Qu.:  0.000
##  Median : 144.71   Median :5.000   Median :  0.000
##  Mean   : 226.88   Mean   :3.819   Mean   :  1.508
##  3rd Qu.: 269.99   3rd Qu.:5.000   3rd Qu.:  1.000
##  Max.   :2598.00   Max.   :5.000   Max.   :645.000
##  NA's   :5926                      NA's   :12291
```

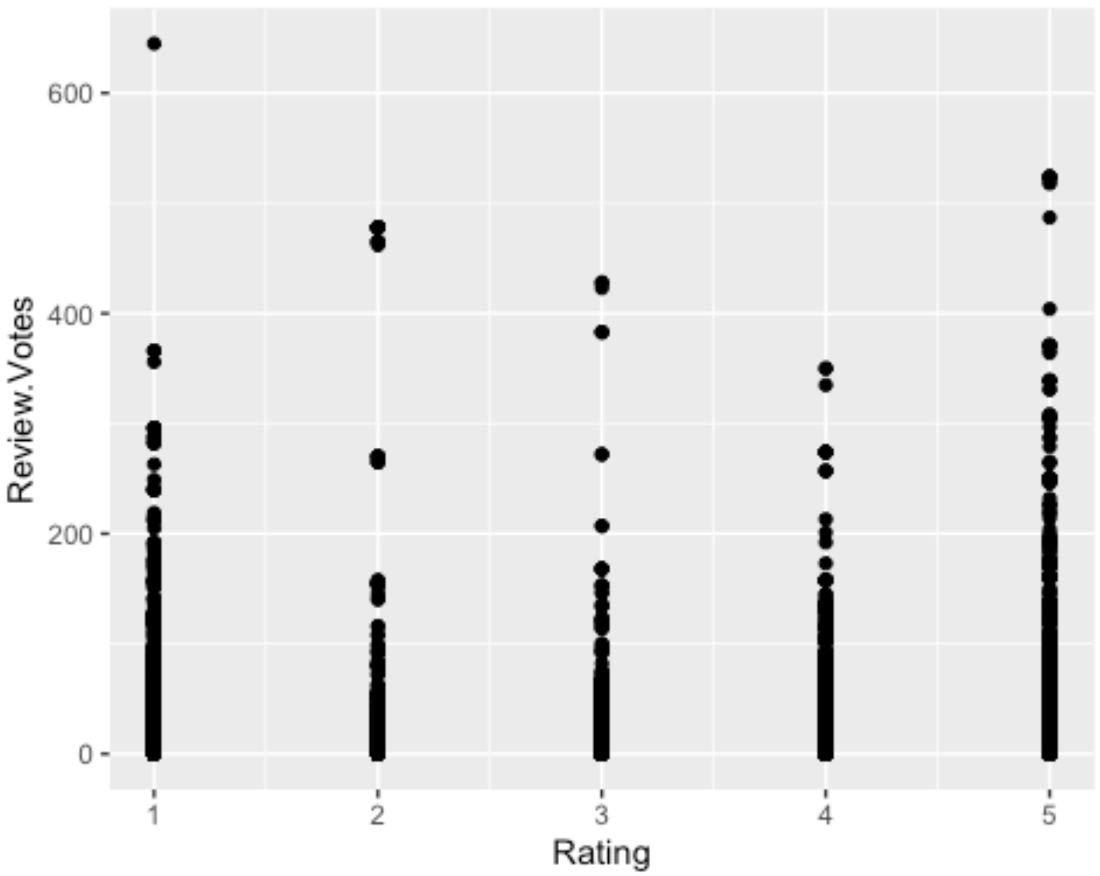
```
# explore summary statistics for rating for each brands


amazon %>%
  group_by(Brand.Name) %>%
  summarize(
    Mean_Rating = mean(Rating, na.rm = TRUE),
    Median_Rating = median(Rating, na.rm = TRUE),
    Std_Dev_Rating = sd(Rating, na.rm = TRUE)
  )
```

```
## # A tibble: 375 × 4
##    Brand.Name                                          Mean_Rating Median_Ra…¹ S
td_D…²
##    <chr>                                                     <dbl>       <dbl>
<dbl>
##  1 ""                                                         3.84         5
1.56
##  2 "Acer"                                                     3.09         3.5
1.72
##  3 "Aeku"                                                     5            5
0
##  4 "AeroAntenna"                                              5            5
NA
##  5 "AKUA"                                                     5            5
0
##  6 "Alcatel"                                                  4.05         5
1.36
```

```
##  7 "amar"                                               2.94         2
1.64
##  8 "Amazon"                                             4.2          5
1.23
##  9 "Amazon.com, LLC *** KEEP PORules ACTIVE ***"        4.05         5
1.61
## 10 "AMM Global Enterprises"                             4.89         5
0.333
## # … with 365 more rows, and abbreviated variable names ¹Median_Rating,
## #   ²Std_Dev_Rating

## Distribution of the rating in the dataset

h <- hist(amazon$Rating,

          plot = FALSE)

h$density <- with(h, 100 * density* diff(breaks)[1])
labs <- paste(round(h$density), "%", sep="")

plot(h,main="Rating Distribution",
     xlab="Rating",
     ylab = "Frequency",
     col="darkmagenta",
     )
text(h$mids, h$counts + 1,pos = 3,cex = 0.7, srt=45, xpd=TRUE, ifelse(h$count
s == 0, "",labs),
)
```



```
#Scatterplot of "rating" vs. "votes"


ggplot(amazon, aes(x = Rating, y = Review.Votes)) +
  geom_point()

## Warning: Removed 12291 rows containing missing values (`geom_point()`).
```



Cleaning & transforming Dataset

```
# Creating dictionary for stop-words

my_stopwords <- tibble(word = c(as.character(1:10),
                                "received","blu","lot","cell","mobile","bough
t","phones" ,"seller","buy","unlocked","iphone","device","product","amazon","
phone","purchase","day","time","
things.the" ,"the"))

custom_stop_words <- bind_rows(tibble(word = c("received","blu","lot","cell",
"mobile","bought","phones" ,"seller","buy","unlocked","iphone","device","prod
uct","amazon","phone","purchase","day","time", "wifi","straight","talk","4g",
"lte","
things.the" ,"thingsthe","the")
                                      , lexicon = c("custom")), stop_words)


# Using the following code chunk, create a dataframe using review text varaib
le

review <- data.frame(ID=seq(1:nrow(amazon)),text=amazon$Reviews)
```

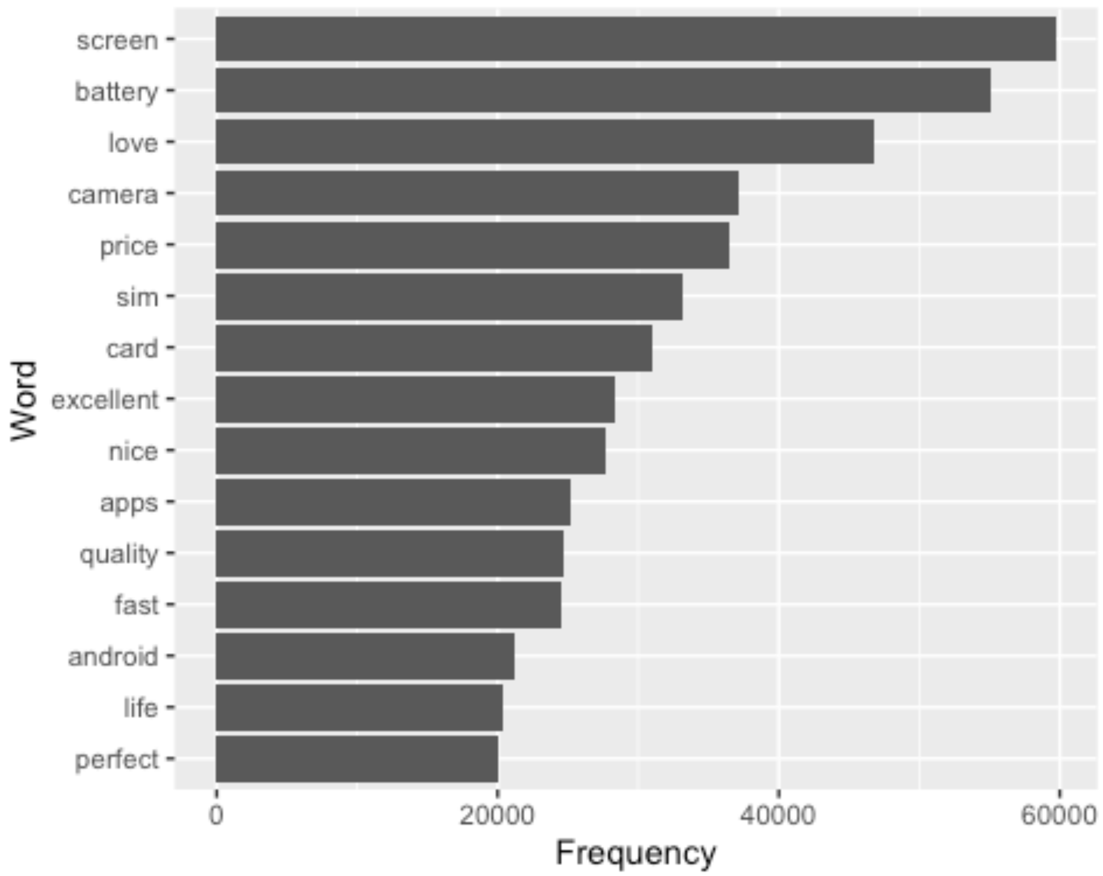
#analyzing reviews

Q1. What are the most frequent words in the reviews?

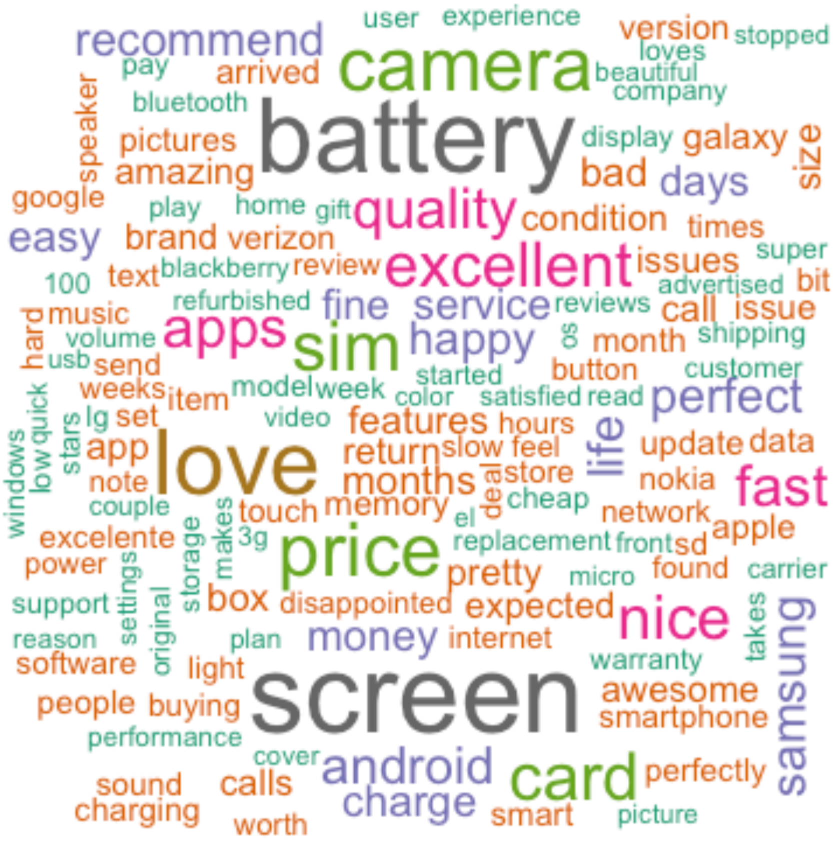
```
count_words <- review %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)%>%
  count(word,sort= TRUE)
```

```
## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"
```

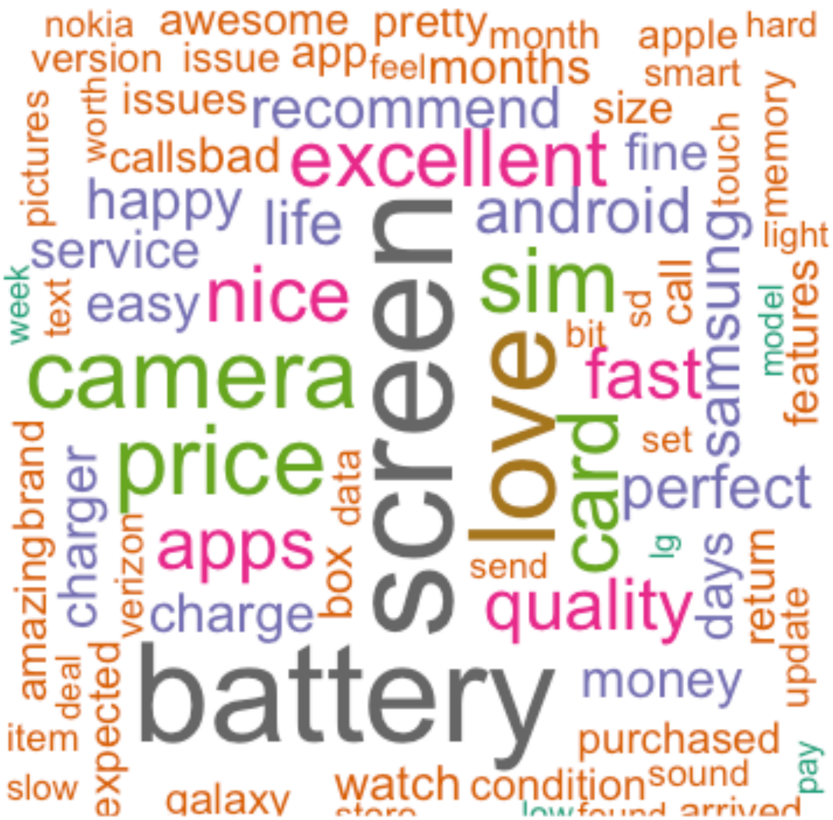
```
View(count_words[1:10,])

## Visualize the top frequent words
count_words %>%
filter(n>19900) %>%
  mutate(word=reorder(word,n)) %>%
  ggplot(aes(word,n)) +
  geom_bar(stat="identity") +
ylab ("Frequency" ) +
  xlab("Word")+
  scale_fill_grey(start = 0.10, end = 0.75) +
  coord_flip()
```



```
######### TF-IDF#################

par(mar = c(1, 1, 1, 1))
wordcloud(words = count_words$word, freq = count_words$n, scale=c(3,.5), max.
words=150, colors=brewer.pal(8, "Dark2"))
```

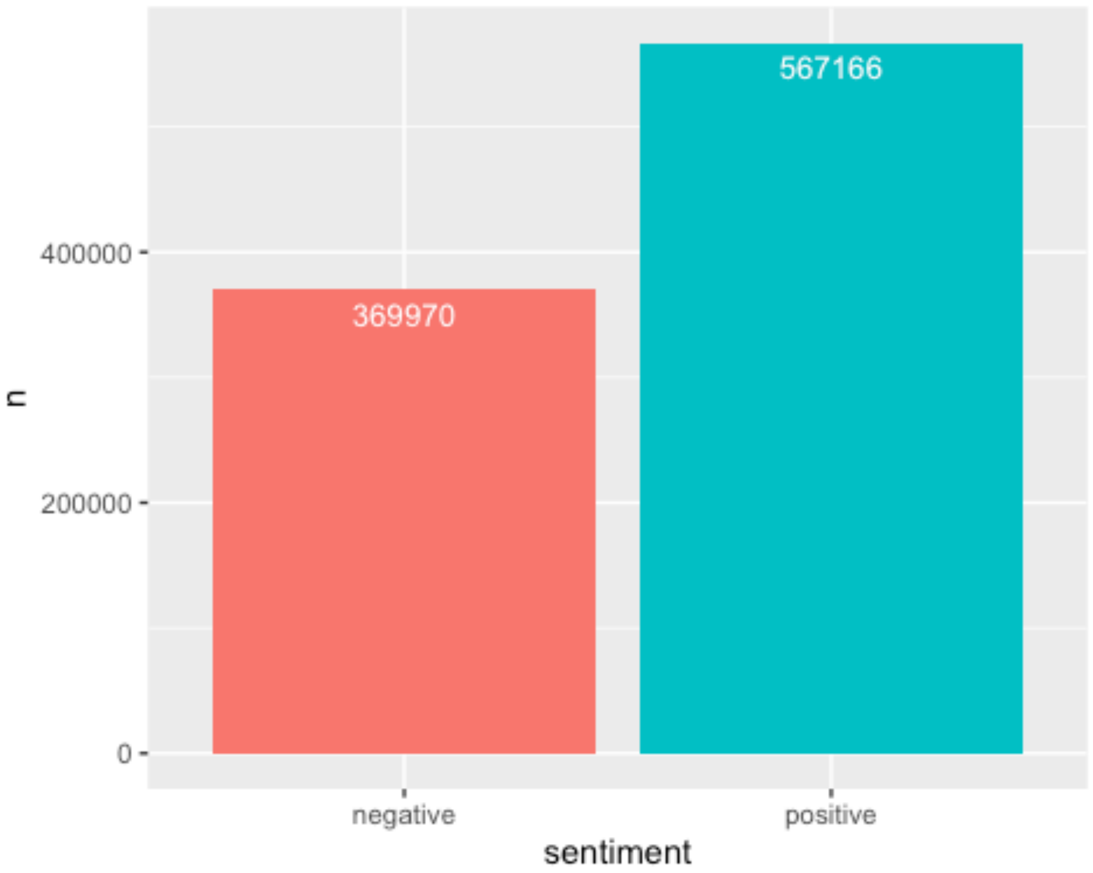


```
set.seed(1234)
wordcloud(words = count_words$word, freq = count_words$n, min.freq = 1, max.words=100, random.order=FALSE, rot.per=0.35,            colors=brewer.pal(8, "Dark2"))
```



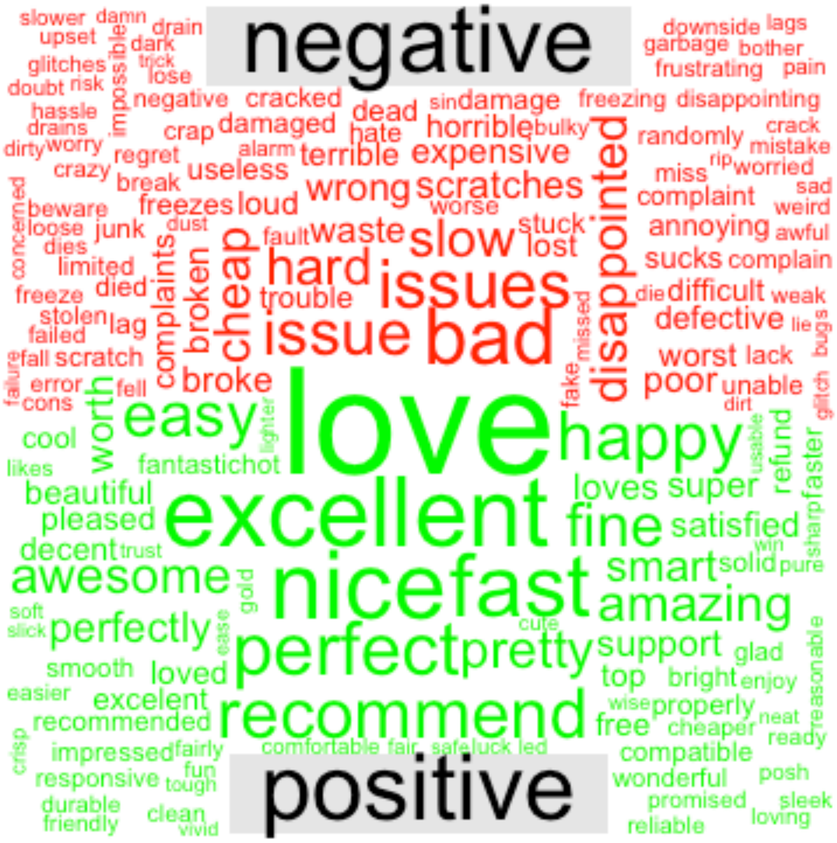
Q2. Does customer reviews have more positive and negative sentiment in general? Can we quantify this sentiment with a positive or negative value?

```
reviews <- review %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)

## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"

bing <- get_sentiments("bing")

pos_ne <- reviews%>%
  inner_join(bing) %>%
  group_by(sentiment)%>%
  count(sentiment, sort = TRUE) %>%
  ungroup()

## Joining, by = "word"

options(scipen = 999)
ggplot(pos_ne , aes(x = sentiment , y = n , fill = sentiment))+
  geom_col(show.legend = FALSE)+
  geom_text(aes(label= n), vjust=1.6, color="white", size=3.5)
```

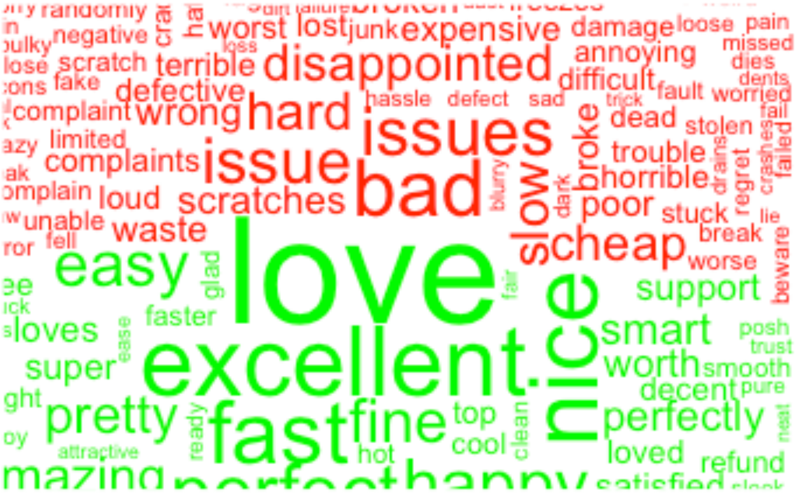


Q2.1 & Q2.2 “What are the top n words that contribute to positive and negative sentiment and their ratios? What is word polarity using opinion lexicons such Bing?

```
par(mfrow=c(1,1))
reviews %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  acast(word ~ sentiment, value.var = "n", fill = 0) %>%
  comparison.cloud(colors = c("red","green"),random.order = FALSE , title.siz
e=2.5, max.words=400)

## Joining, by = "word"
```



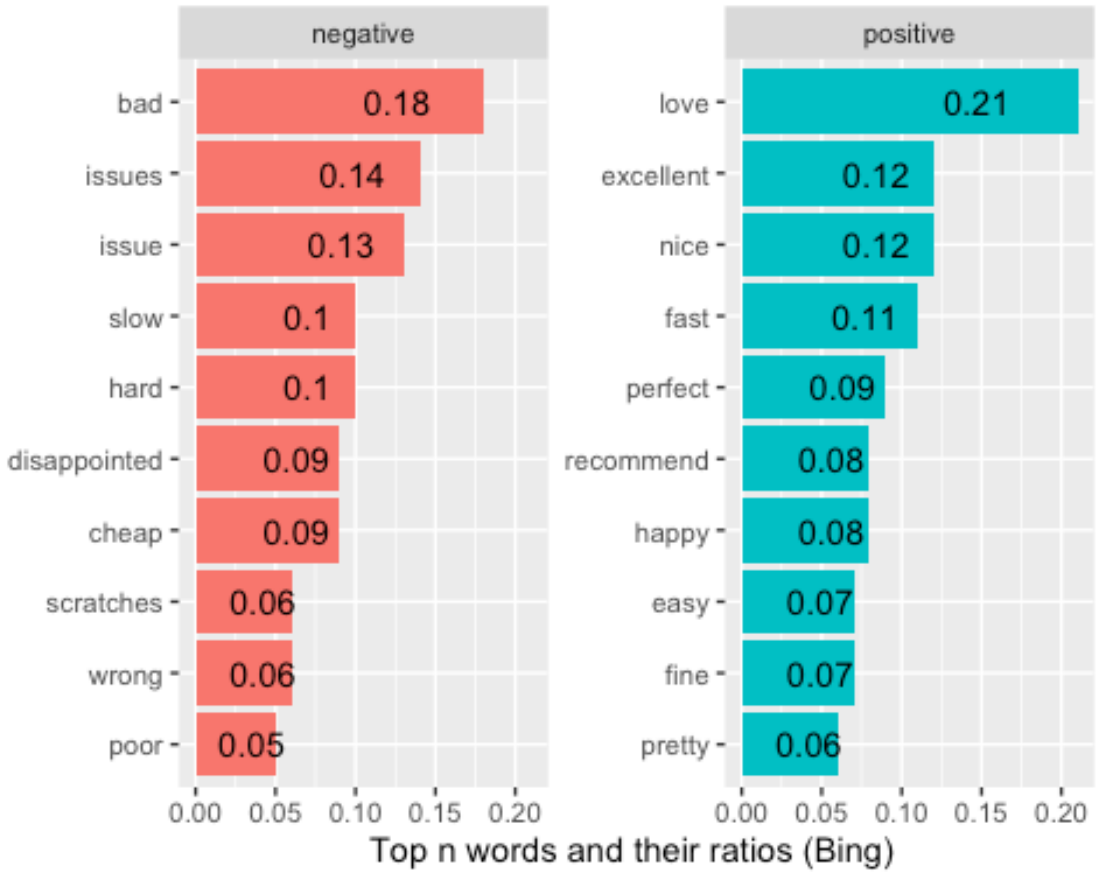
```
par(mfrow=c(1,1))
reviews %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  reshape2::acast(word ~ sentiment, value.var = "n", fill = 0)%>%
  comparison.cloud(colors = c("red","green"),
                   max.words = 400)

## Joining, by = "word"
```



Q3. What are the top n words that contribute to positive and negative sentiment and their ratios using sentiment lexicons like Bing?

```
top_pos_ne_ratio <- reviews%>%
  inner_join(bing) %>%
  count(sentiment,word, sort = TRUE) %>%
  ungroup()
```

```
## Joining, by = "word"

 top_pos_ne_ratio %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  mutate(word = reorder(word,n)) %>%
  mutate(percent = round(n/sum(n),2)) %>%
  ggplot(aes(x = word, y = percent, fill = sentiment, label = percent))%>%
  + geom_col(show.legend = FALSE) + facet_wrap(~sentiment, scales = "free_y")
+
  geom_text(aes(y = 0.7*percent))+
  labs(y = "Top n words and their ratios (Bing) ", x = NULL) + coord_flip()

## Selecting by n
```



Q4.Does the rating accurately reflect customer reviews, and is there a difference in sentiment across different rating scores?

```
# creating dataframe for all products fall under rating score 1
a1 <- amazon %>%
  group_by(Rating)%>%
  filter(Rating == 1)

review_1 <- data.frame(ID=seq(1:nrow(a1)),text=a1$Reviews)

countwords1 <- review_1 %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)
```

```
## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"
```

```r
s1 <- countwords1%>%
  inner_join(bing) %>%
  count(sentiment, sort = TRUE) %>%
  ungroup()

## Joining, by = "word"

s1 <- s1 %>%
  mutate(rating = 1 )


# creating dataframe for all products fall under rating score 2
a2 <- amazon %>%
  group_by(Rating)%>%
  filter(Rating == 2)

review_2 <- data.frame(ID=seq(1:nrow(a2)),text=a2$Reviews)

countwords2 <- review_2 %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)

## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"

s2 <- countwords2%>%
  inner_join(bing) %>%
  count(sentiment, sort = TRUE) %>%
  ungroup()

## Joining, by = "word"

s2 <- s2 %>%
  mutate(rating = 2 )

# creating dataframe for all products fall under rating score 3

a3 <- amazon %>%
  group_by(Rating)%>%
  filter(Rating == 3)

review_3 <- data.frame(ID=seq(1:nrow(a3)),text=a3$Reviews)

countwords3 <- review_3 %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)
```

```
## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"
```

```r
s3 <- countwords3%>%
  inner_join(bing) %>%
  count(sentiment, sort = TRUE) %>%
  ungroup()
```

```
## Joining, by = "word"
```

```r
s3 <- s3 %>%
  mutate(rating = 3 )

# creating dataframe for all products fall under rating score 4
a4 <- amazon %>%
  group_by(Rating)%>%
  filter(Rating == 4)

review_4 <- data.frame(ID=seq(1:nrow(a4)),text=a4$Reviews)

countwords4 <- review_4 %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)
```

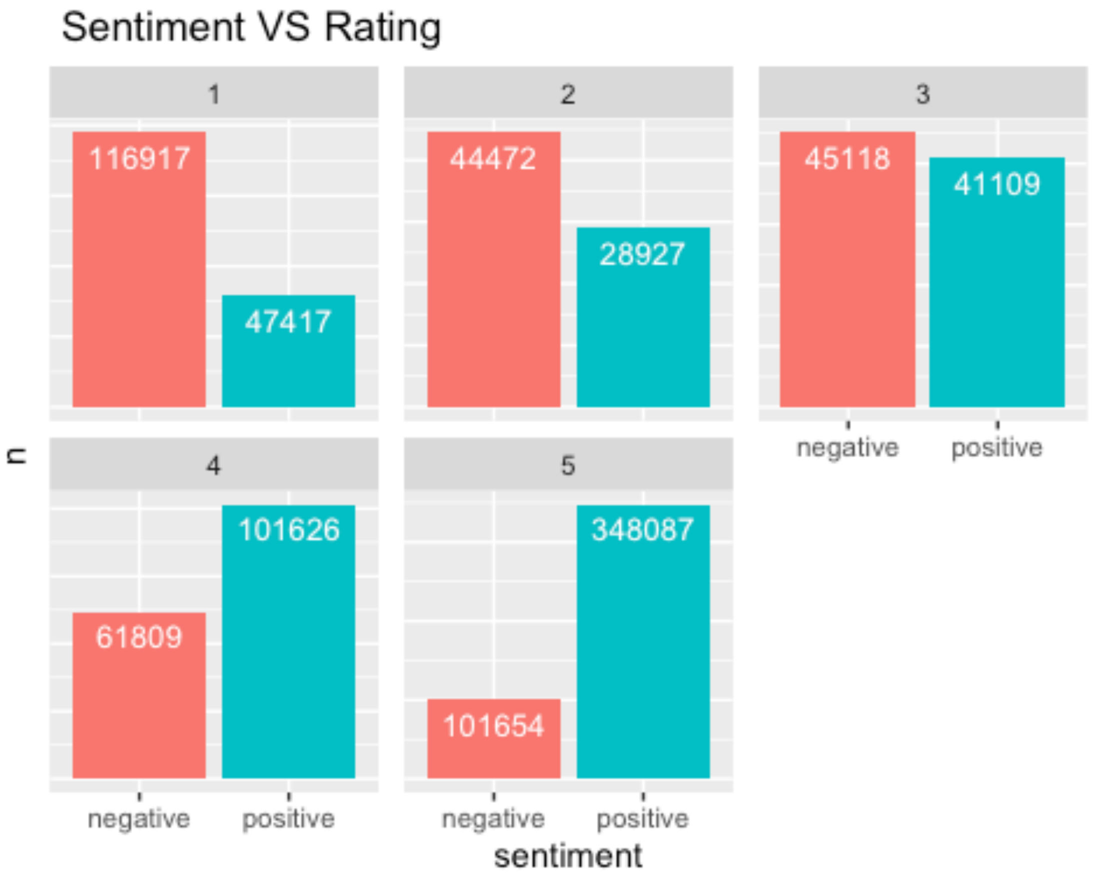
```
## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"
```

```r
s4 <- countwords4%>%
  inner_join(bing) %>%
  count(sentiment, sort = TRUE) %>%
  ungroup()
```

```
## Joining, by = "word"
```

```r
s4 <- s4 %>%
  mutate(rating = 4 )

# creating dataframe for all products fall under rating score 5

a5 <- amazon %>%
  group_by(Rating)%>%
  filter(Rating == 5)
```

```
review_5 <- data.frame(ID=seq(1:nrow(a5)),text=a5$Reviews)

countwords5 <- review_5 %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)

## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"

s5 <- countwords5%>%
  inner_join(bing) %>%
  count(sentiment, sort = TRUE) %>%
  ungroup()

## Joining, by = "word"

s5 <- s5 %>%
  mutate(rating = 5 )

##########
rating_score <- rbind(s1,s2,s3,s4, s5)

rating_score %>%
  ggplot(aes(x = sentiment, y = n , fill = sentiment))+
   geom_col(show.legend = FALSE) +
  facet_wrap(~ rating , scales = "free_y")+
   geom_text(aes(label= n), vjust=1.6, color="white", size=3.5)+
   theme(
        axis.text.y=element_blank(),
        axis.ticks.y=element_blank()
        )+
labs(title = " Sentiment & Rating ")
```



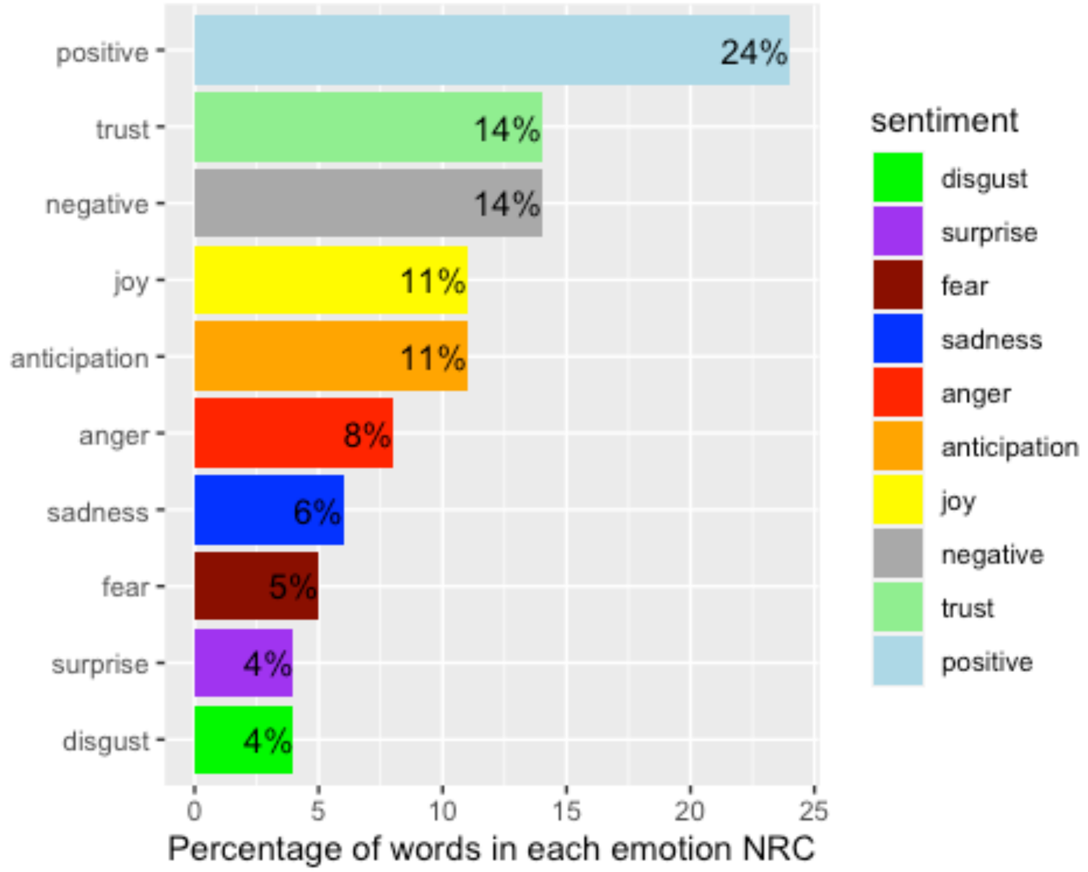
Q5. What is the frequency of words associated with each emotion set in the NRC lexicon?

```
nrc <- reviews%>%
  inner_join(get_sentiments("nrc")) %>%
  count( sentiment, sort = TRUE) %>%
  ungroup()
```

```
## Joining, by = "word"
```

```
# add new column percent
nrc_e <- nrc %>%
  mutate(percent=round(n/sum(n)*100))



my_colors <- c("anger" = "red", "anticipation" = "orange", "disgust" = "green
", "fear" = "darkred",
               "joy" = "yellow", "negative" = "darkgray", "positive" = "light
blue", "sadness" = "blue",
               "surprise" = "purple", "trust" = "lightgreen")
```

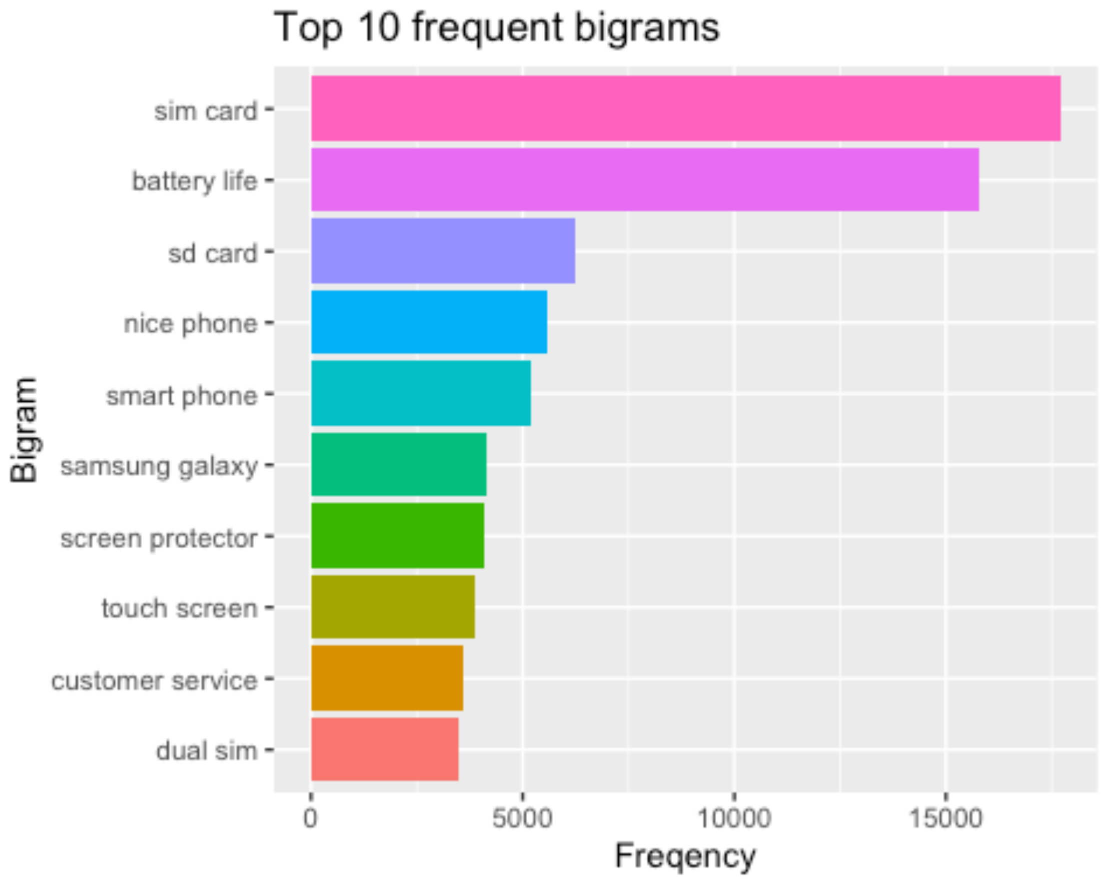
```
# Visualize it with custom colors


nrc_e %>%
  mutate(sentiment = reorder(sentiment, n)) %>%
  ggplot(aes(sentiment, percent, fill = sentiment)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = my_colors) +  # Apply the custom color palette
  labs(y = "Percentage of words in each emotion NRC", x = NULL) +
  geom_text(aes(label = paste0(percent, "%")), hjust = 1, vjust = 0.5, color
= "black") +  # Add percentage labels
  coord_flip()
```



Q6.What are the most top n phrases in the reviews?

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

bigrams_s <- bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")
bigrams_c1 <- bigrams_s%>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

custom_stop_2 <- bind_rows(tibble(word = c("internet.the" , " things.the", "b
ased","pack.all" , "cell phone"), lexicon = c("custom")), stop_words)

bigrams_cl <- bigrams_c1%>%
  filter(!word1 %in% custom_stop_2$word) %>%
  filter(!word2 %in% custom_stop_2$word)

bigrams_united <- bigrams_cl %>%
  unite(bigram, word1, word2, sep = " ")

bigram_count <- bigrams_united %>%
  count(bigram , sort = TRUE)

bigram_count2 <- bigram_count %>%
  filter(!grepl('NA NA', bigram))

bigram_f <- bigram_count2 %>%
  filter(!grepl('cell phone', bigram))


bigram_f %>%
  top_n(10)%>%
    mutate(bigram = reorder(bigram, n)) %>%
           ggplot(aes(x = bigram, y = n , fill = bigram)) +
           geom_col(show.legend = FALSE) +
           labs(title = "Top 10 frequent bigrams  ") +
           coord_flip() +
           xlab("Bigram")+
           ylab("Freqency")
```

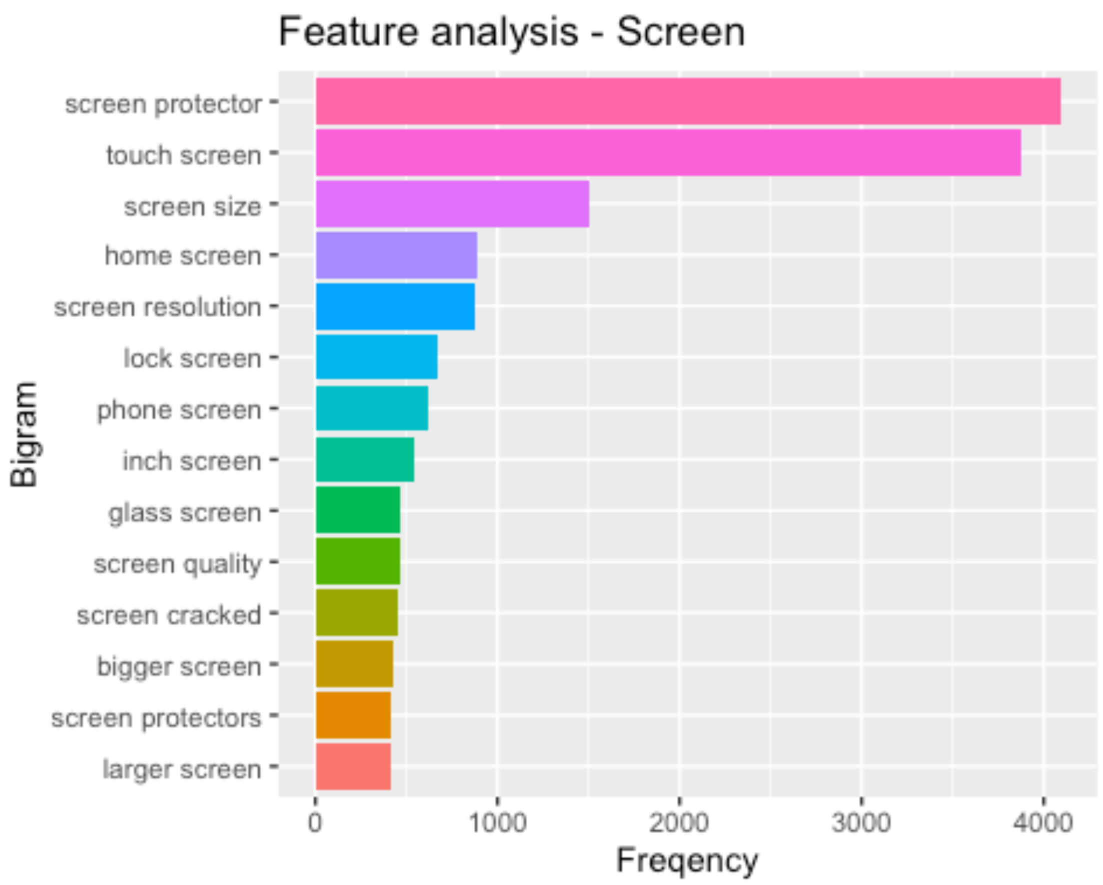
```
## Selecting by n
```



```
           theme(plot.title = element_text(hjust = 0.5))

## List of 1
##  $ plot.title:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

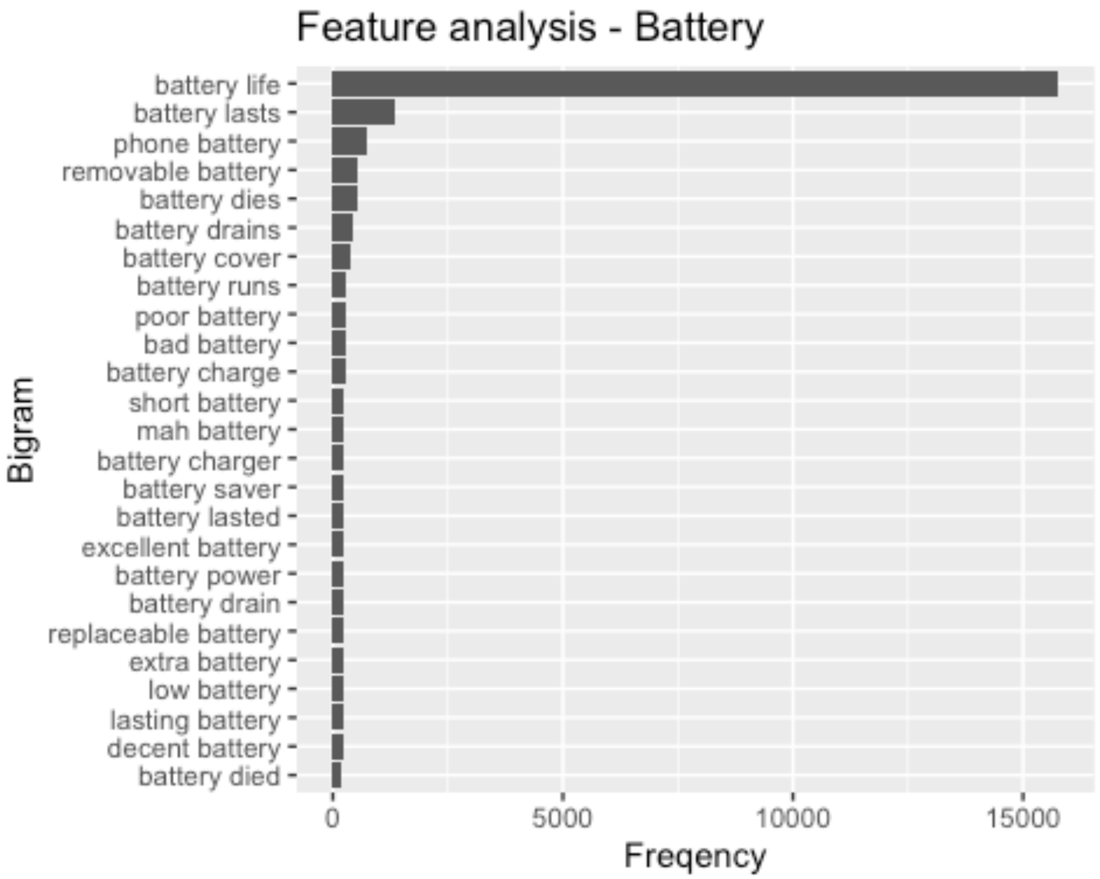
Q7. What are the common phrases associated with the most frequently mentioned product features or services, and what are the common issues?

```
 ### screen analysis###########

# extract bi-gram that has screen words
   bigram_screen <- bigram_f %>%
  filter(grepl('screen', bigram))

  bigram_screen %>%
    filter(n >= 400)%>%
    mutate(bigram = reorder(bigram, n)) %>%
           ggplot(aes(x = bigram, y = n , fill = bigram)) +
           geom_col(show.legend = FALSE) +
           coord_flip() +
          labs(title = "Feature analysis - Screen") +
           xlab("Bigram")+
           ylab("Freqency")
```



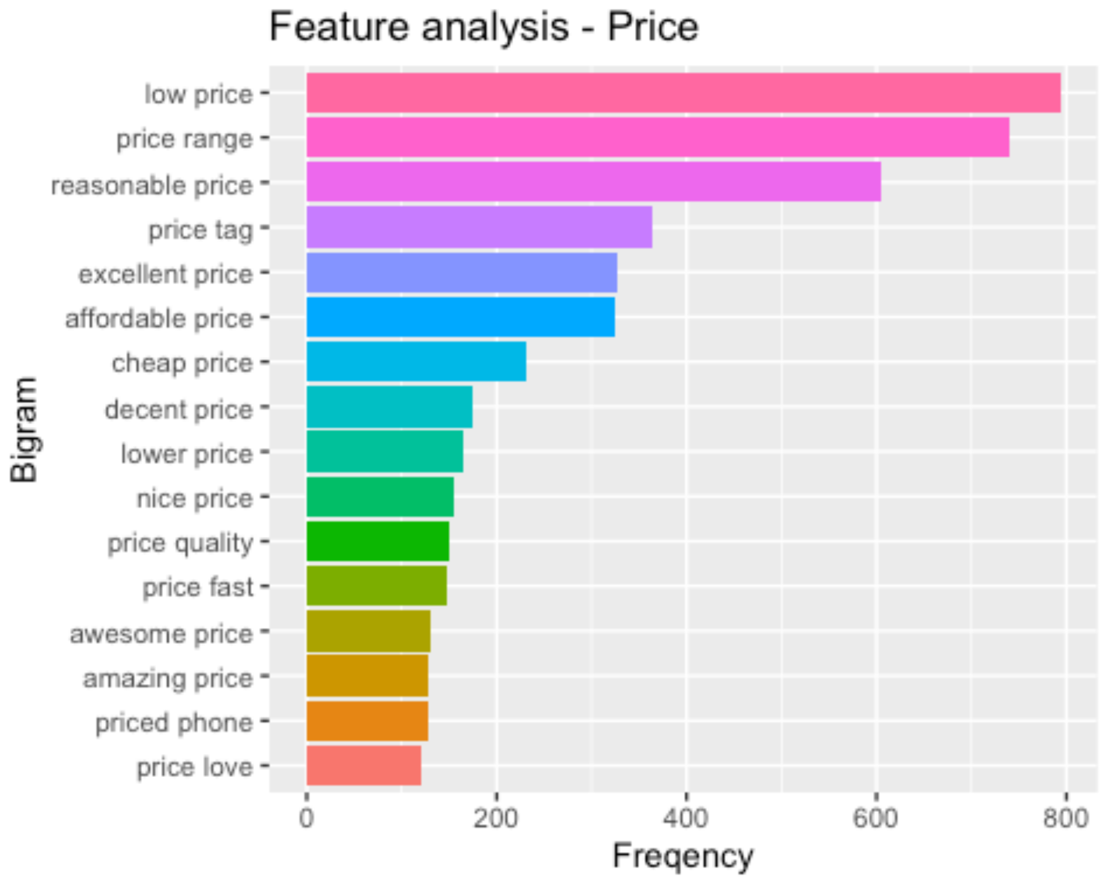
```
           theme(plot.title = element_text(hjust = 0.5))
```

```
## List of 1
##  $ plot.title:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE

### battery analysis

# extract the bi-grams that has battery words

  bigram_battery <- bigram_f %>%
  filter(grepl('battery', bigram))

     bigram_battery %>%
    filter(n >= 200)%>%
    mutate(bigram = reorder(bigram, n)) %>%
           ggplot(aes(x = bigram, y = n )) +
           geom_col(show.legend = FALSE) +
           coord_flip() +
          labs(title = "Feature analysis - Battery") +
           xlab("Bigram")+
           ylab("Freqency")
```



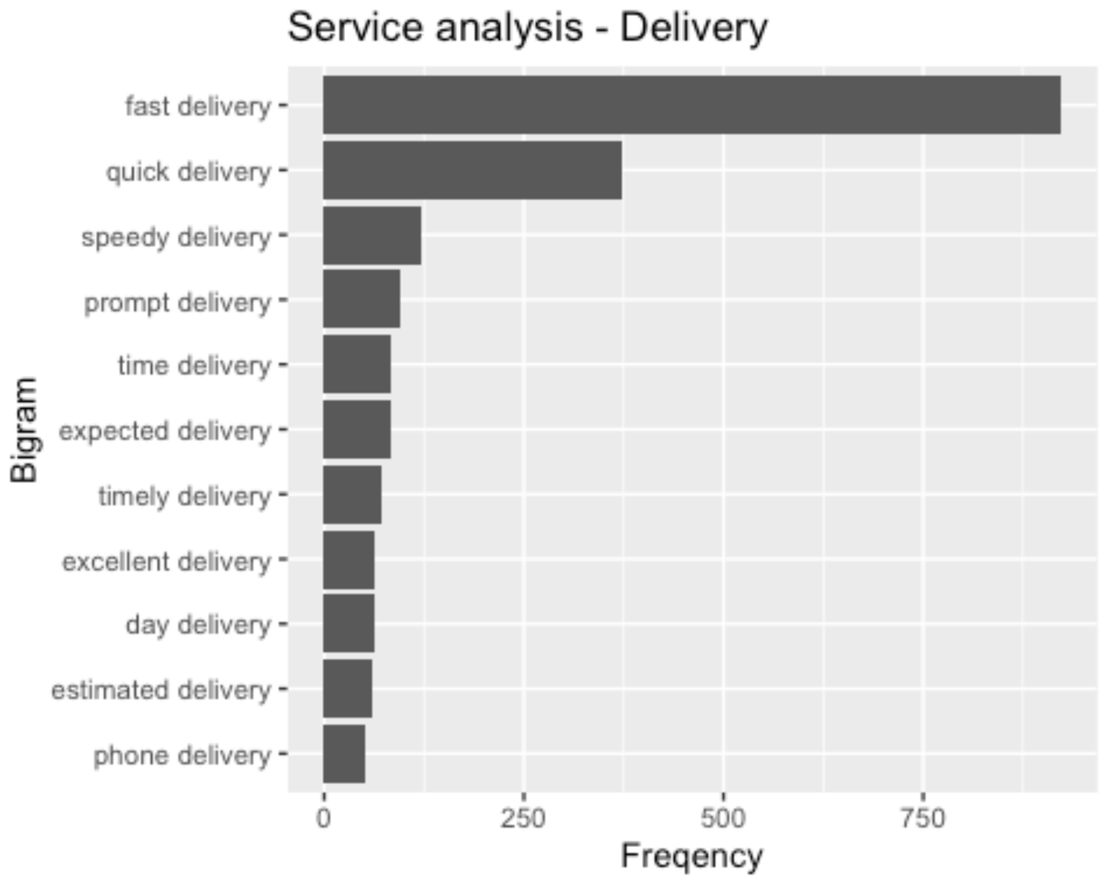
```
           theme(plot.title = element_text(hjust = 0.5))

## List of 1
##  $ plot.title:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE

 ### price analysis

# extract the bi-grams that has price words
```

```
  bigram_price <- bigram_f %>%
  filter(grepl('price', bigram))

  # present top 15

bigram_price %>%
    filter(n >= 121)%>%
    mutate(bigram = reorder(bigram, n)) %>%
           ggplot(aes(x = bigram, y = n , fill = bigram )) +
           geom_col(show.legend = FALSE) +
           coord_flip() +
          labs(title = "Feature analysis - Price") +
           xlab("Bigram")+
           ylab("Freqency")
```

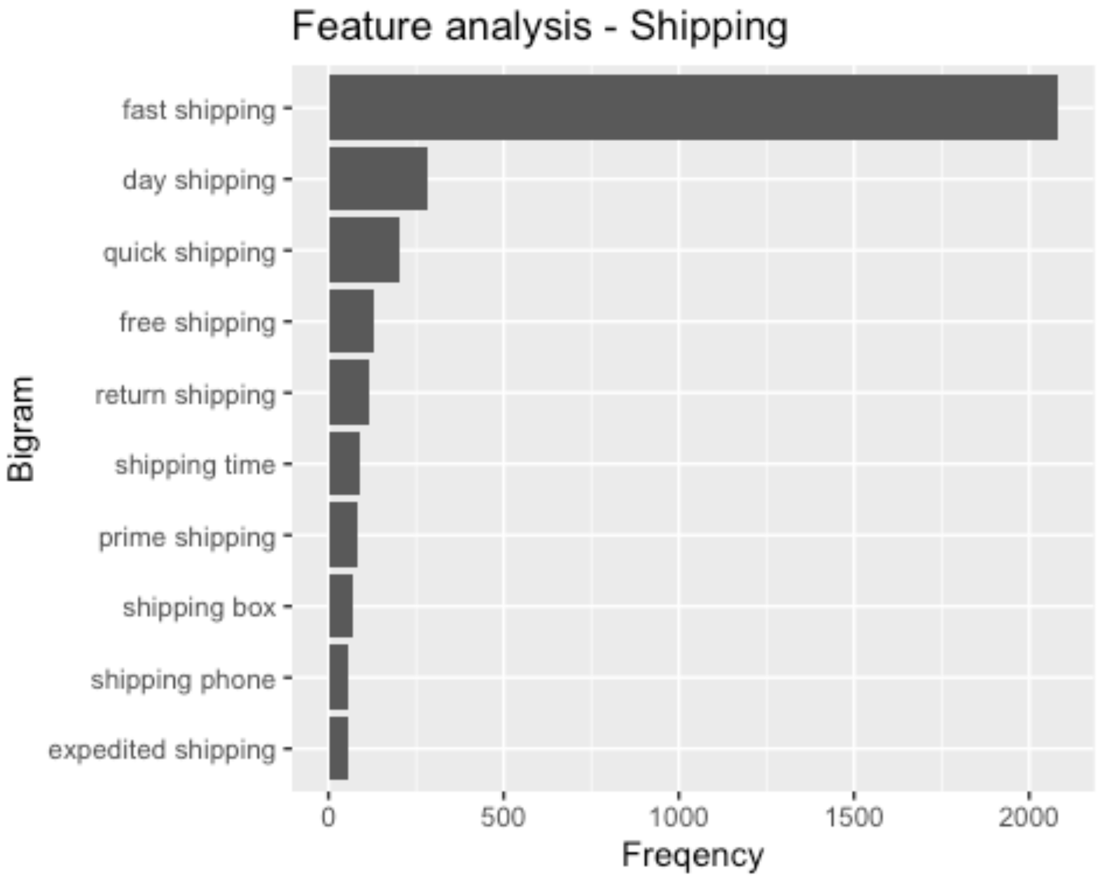


```
           theme(plot.title = element_text(hjust = 0.5))

## List of 1
##  $ plot.title:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

```
 # extract the bi-grams that has delivery words

      bigram_delivery <- bigram_f %>%
  filter(grepl(' delivery', bigram))

   bigram_delivery %>%
    filter(n >= 50)%>%
    mutate(bigram = reorder(bigram, n)) %>%
           ggplot(aes(x = bigram, y = n )) +
           geom_col(show.legend = FALSE) +
           coord_flip() +
          labs(title = "Service analysis - Delivery") +
           xlab("Bigram")+
           ylab("Freqency")
```



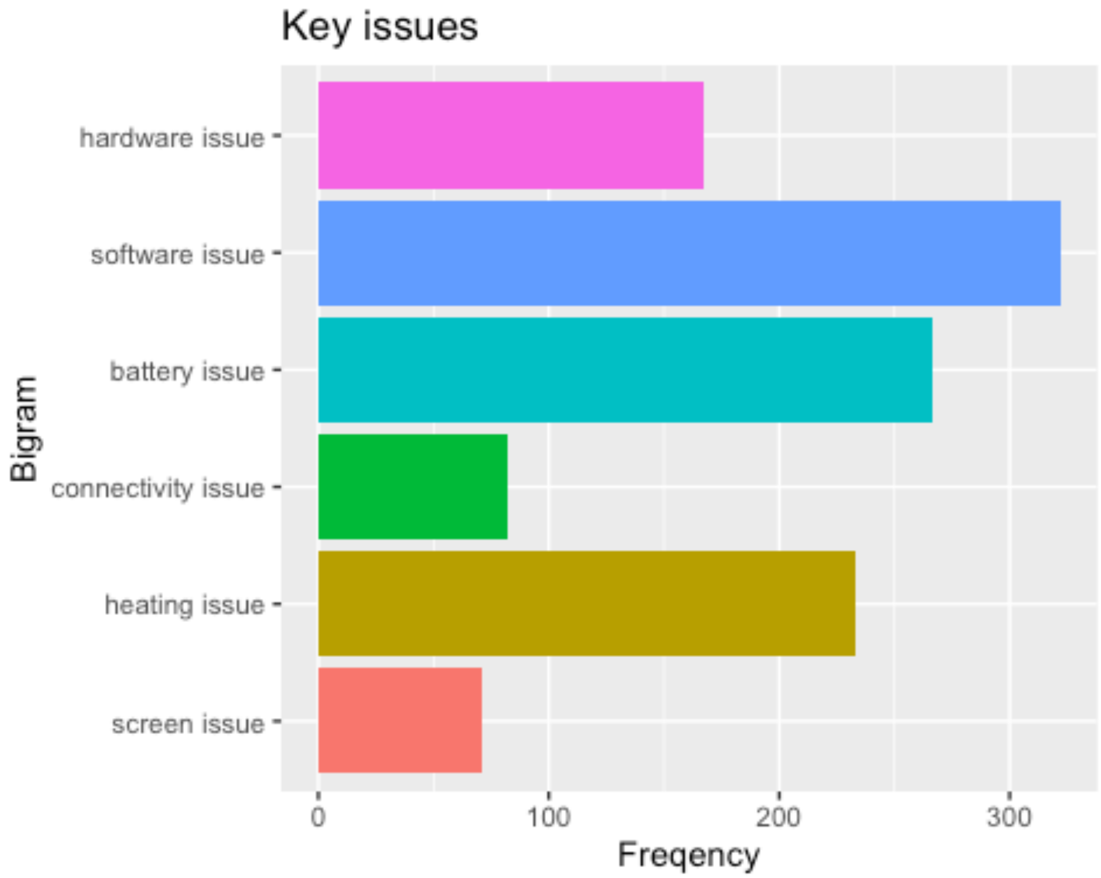
```
           theme(plot.title = element_text(hjust = 0.5))

## List of 1
##  $ plot.title:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE

# extract the bi-grams that has shipping words

bigram_shipping <- bigram_f %>%
  filter(grepl('shipping', bigram))

     bigram_shipping %>%
   top_n(10)%>%
    mutate(bigram = reorder(bigram, n)) %>%
           ggplot(aes(x = bigram, y = n )) +
           geom_col(show.legend = FALSE) +
           coord_flip() +
          labs(title = "Feature analysis - Shipping") +
           xlab("Bigram")+
           ylab("Freqency")

## Selecting by n
```



```
           theme(plot.title = element_text(hjust = 0.5))

## List of 1
##  $ plot.title:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

Q6.2

```
# extract the bi-grams that has issue words

bigram_issues <- bigram_f %>%
  filter(grepl('issue', bigram))

bigram_issues$bigram <- str_replace_all(bigram_issues$bigram, 'issues', 'issu
e')

bigram_issues <- bigram_issues %>%
  filter(!grepl('major', bigram))

bigram_issues <- bigram_issues %>%
  filter(!grepl('minor ', bigram))

bigram_issues <- bigram_issues %>%
  filter(!grepl('biggest ', bigram))

bigram_issues <- bigram_issues %>%
  filter(!grepl('whatsoever ', bigram))

bigram_issues <- bigram_issues %>%
  filter(!grepl('main ', bigram))

bigram_issues <- bigram_issues %>%
  filter(!grepl('common ', bigram))

bigram_issues2 <- bigram_issues %>%
  filter(!grepl('issue whatsoever ', bigram))

bigram_issues3 <- bigram_issues %>%
  filter(bigram !='issue whatsoever')

bigram_issues3$bigram <- str_replace_all(bigram_issues3$bigram, 'overheating'
,'heating')

     bigram_issues3 %>%
   top_n(10)%>%
    mutate(bigram = reorder(bigram, n)) %>%
           ggplot(aes(bigram,n , fill = bigram)) +
           geom_col(show.legend = FALSE) +
           coord_flip() +
          labs(title = "Key issues") +
           xlab("Bigram")+
           ylab("Freqency")

## Selecting by n
```



```
           theme(plot.title = element_text(hjust = 0.5))

## List of 1
##  $ plot.title:List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi FALSE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi FALSE
##  - attr(*, "validate")= logi TRUE
```

RQ6.3.

```
review_corp <- review %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)
```

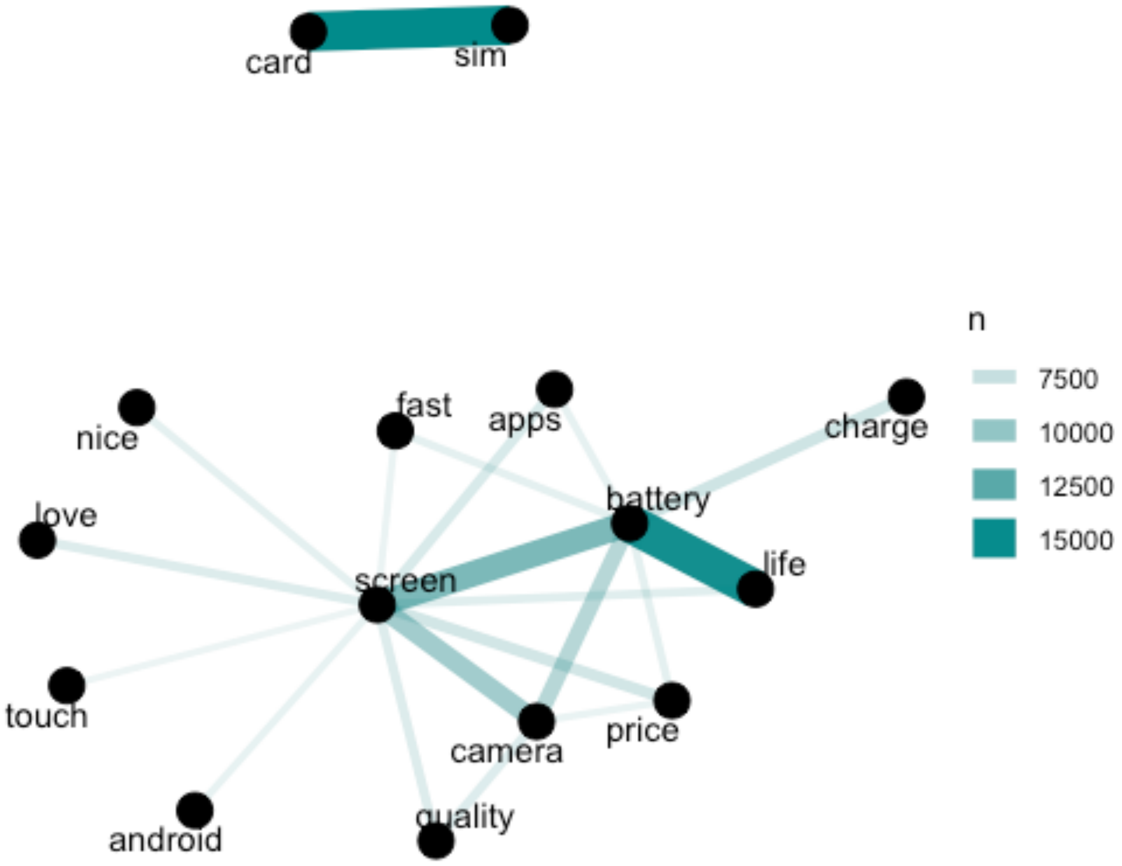
```
## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"
```

```
title_word_pairs <- review_corp %>%
 pairwise_count(word, ID, sort = TRUE, upper = FALSE)
title_word_pairs
```

```
## # A tibble: 19,542,310 × 3
##    item1   item2       n
##    <chr>   <chr>   <dbl>
##  1 sim     card    15122
##  2 battery life    14767
##  3 screen  battery 11040
##  4 screen  camera   9256
##  5 battery camera   8236
##  6 battery charge   7025
##  7 screen  price    6861
##  8 screen  apps     6466
##  9 screen  quality  6253
## 10 screen  love     6248
## # … with 19,542,300 more rows
```

```
# pairs of words that occur together

set.seed(1234)
title_word_pairs %>%
  filter(n >= 5422) %>%
  graph_from_data_frame() %>%
  ggraph(layout = "fr") +
  geom_edge_link(aes(edge_alpha = n, edge_width = n), edge_colour = "cyan4")
+
  geom_node_point(size = 5) +
  geom_node_text(aes(label = name), repel = TRUE,
                 point.padding = unit(0.2, "lines")) +
  theme_void()
```

```
## Warning: Using the `size` aesthetic in this geom was deprecated in ggplot2
3.4.0.
## ℹ Please use `linewidth` in the `default_aes` field and elsewhere instead
.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



```
count_words <- review %>%
  unnest_tokens(word,text)%>%
  mutate(word = tolower(word))%>%
  anti_join(stop_words)%>%
  anti_join(my_stopwords)%>%
  anti_join(custom_stop_words)

## Joining, by = "word"
## Joining, by = "word"
## Joining, by = "word"

word_pairs <- count_words%>%
  pairwise_count(word, ID, sort = TRUE)

# screen correlation
```

```
 screen_corr <- word_pairs %>%
  filter(item1 == "screen")
# battery correlation
 battery_corr <- word_pairs %>%
   filter(item1 == "battery")
```