Sentiment Analysis

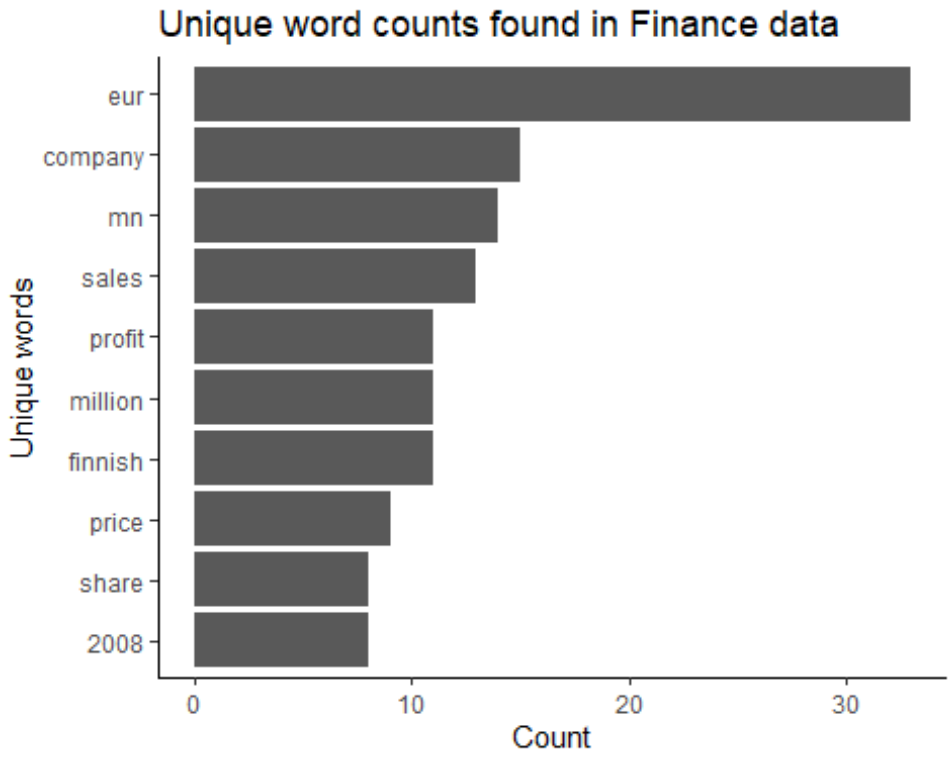
Objective: Sentiment Analysis of financial sentences as part of Lab Migration Project.

Methods:

(i) Import and load the dataset.
(ii) Remove stop words and find most common words in dataset.
(iii) Find the common positive and negative words and visualize it using graph.
(iv) Apply sentiment analysis on the data and find score of positive, negative and neutral sentences, and plot it.
(v) Conclusion

```
#To clear the environment
rm(list=ls())

#Load packages
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

library(tidyr)
library(tidytext)
library(textdata)
library(ggplot2)
library(purrr)

#Import and load the dataset
data <- read.csv("finance.csv")
data <- head(data,100)

data$striptxt <- gsub("$S+","",data$Sentence)

data_stem <- data %>%
```

```
  select(striptxt) %>%
  unnest_tokens(word,striptxt)

head(data_stem,10)

##            word
## 1           the
## 2  geosolutions
## 3    technology
## 4          will
## 5      leverage
## 6       benefon
## 7             s
## 8           gps
## 9     solutions
## 10           by

#Remove stop words using anti-join
cleaned_data.Sentence <- data_stem %>%
  anti_join(stop_words)

## Joining, by = "word"

head(cleaned_data.Sentence,10)

##            word
## 1  geosolutions
## 2    technology
## 3      leverage
## 4       benefon
## 5           gps
## 6     solutions
## 7     providing
## 8      location
## 9         based
## 10       search

cleaned_data.Sentence %>%
  count(word,sort=TRUE) %>%
  top_n(10) %>%
  mutate(word=reorder(word, n)) %>%
  ggplot(aes(x = word, y = n)) + geom_col() + xlab(NULL) + coord_flip() +
theme_classic() + labs(x = "Unique words", y = "Count", title = "Unique word
counts found in Finance data")

## Selecting by n
```



Inference: After preprocessing the data and cleaning it, the given are the top 10 words used in the finance dataset.

```
get_sentiments("bing") %>%
  filter(sentiment == "positive")
```

```
## # A tibble: 2,005 x 2
##    word        sentiment
##    <chr>       <chr>
##  1 abound      positive
##  2 abounds     positive
##  3 abundance   positive
##  4 abundant    positive
##  5 accessable  positive
##  6 accessible  positive
##  7 acclaim     positive
##  8 acclaimed   positive
##  9 acclamation positive
## 10 accolade    positive
## # ... with 1,995 more rows
## # i Use `print(n = ...)` to see more rows
```

```
get_sentiments("bing") %>%
  filter(sentiment == "negative")
```

```
## # A tibble: 4,781 x 2
##    word        sentiment
```

```
##    <chr>       <chr>
##  1 2-faces     negative
##  2 abnormal    negative
##  3 abolish     negative
##  4 abominable  negative
##  5 abominably  negative
##  6 abominate   negative
##  7 abomination negative
##  8 abort       negative
##  9 aborted     negative
## 10 aborts      negative
## # ... with 4,771 more rows
## # i Use `print(n = ...)` to see more rows
```

```
#AFINN lexicon model scores the words in a range from -5 to 5
get_sentiments("afinn") %>%
  filter(value == "3")
```

```
## # A tibble: 172 x 2
##    word         value
##    <chr>        <dbl>
##  1 admire           3
##  2 admired          3
##  3 admires          3
##  4 admiring         3
##  5 adorable         3
##  6 adore            3
##  7 adored           3
##  8 adores           3
##  9 affection        3
## 10 affectionate     3
## # ... with 162 more rows
## # i Use `print(n = ...)` to see more rows
```

```
get_sentiments("afinn") %>%
  filter(value == "5")
```

```
## # A tibble: 5 x 2
##   word         value
##   <chr>        <dbl>
## 1 breathtaking     5
## 2 hurrah           5
## 3 outstanding      5
## 4 superb           5
## 5 thrilled         5
```

```
get_sentiments("afinn") %>%
  filter(value == "-3")
```

```
## # A tibble: 264 x 2
##    word        value
```

```
##    <chr>       <dbl>
##  1 abhor          -3
##  2 abhorred       -3
##  3 abhorrent      -3
##  4 abhors         -3
##  5 abuse          -3
##  6 abused         -3
##  7 abuses         -3
##  8 abusive        -3
##  9 acrimonious    -3
## 10 agonise        -3
## # ... with 254 more rows
## # i Use `print(n = ...)` to see more rows
```

```
#Use "bing" lexicon and implement filter() over words that correspond to
postive sentiment
positive_senti <- get_sentiments("bing") %>%
  filter(sentiment == "positive")
cleaned_data.Sentence %>%
  semi_join(positive_senti) %>%
  count(word, sort = TRUE)
```

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

```
##             word n
## 1         bullish 2
## 2           fresh 2
## 3             top 2
## 4          awards 1
## 5      commitment 1
## 6     competitive 1
## 7      complement 1
## 8         delight 1
## 9       efficient 1
## 10           fair 1
## 11      favorable 1
## 12           fine 1
## 13           gold 1
## 14       improved 1
## 15           lead 1
## 16            led 1
## 17       leverage 1
## 18        pleased 1
## 19       powerful 1
## 20        premier 1
## 21         pretty 1
## 22          ready 1
## 23 recommendation 1
## 24  restructuring 1
## 25         secure 1
```

```
## 26           soft 1
## 27         tender 1
## 28        tougher 1
```

```
#Most common positive and negative words
bing_data = cleaned_data.Sentence %>%
  inner_join(get_sentiments("bing")) %>%
  count(word, sentiment, sort = TRUE) %>%
  ungroup()
```

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

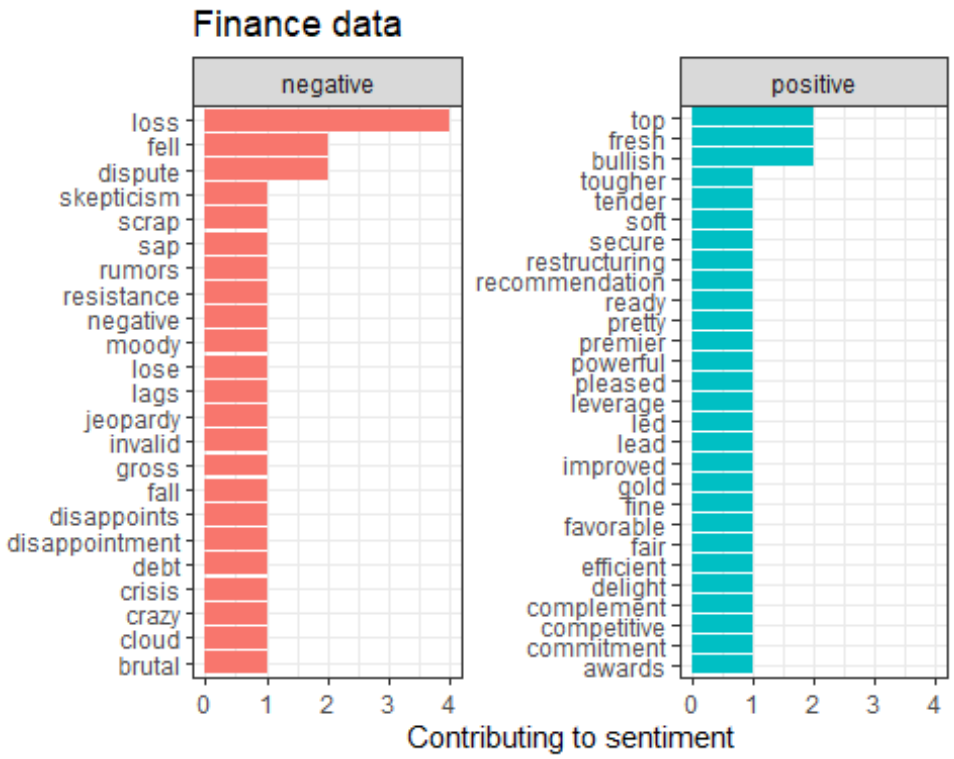
```
bing_data
```

```
##              word sentiment n
## 1            loss  negative 4
## 2         bullish  positive 2
## 3         dispute  negative 2
## 4            fell  negative 2
## 5           fresh  positive 2
## 6             top  positive 2
## 7          awards  positive 1
## 8          brutal  negative 1
## 9           cloud  negative 1
## 10     commitment  positive 1
## 11    competitive  positive 1
## 12     complement  positive 1
## 13          crazy  negative 1
## 14         crisis  negative 1
## 15           debt  negative 1
## 16        delight  positive 1
## 17 disappointment  negative 1
## 18    disappoints  negative 1
## 19      efficient  positive 1
## 20           fair  positive 1
## 21           fall  negative 1
## 22      favorable  positive 1
## 23           fine  positive 1
## 24           gold  positive 1
## 25          gross  negative 1
## 26       improved  positive 1
## 27        invalid  negative 1
## 28       jeopardy  negative 1
## 29           lags  negative 1
## 30           lead  positive 1
## 31            led  positive 1
## 32       leverage  positive 1
## 33           lose  negative 1
## 34          moody  negative 1
## 35       negative  negative 1
## 36        pleased  positive 1
```

```
## 37       powerful  positive 1
## 38        premier  positive 1
## 39         pretty  positive 1
## 40          ready  positive 1
## 41 recommendation  positive 1
## 42     resistance  negative 1
## 43  restructuring  positive 1
## 44         rumors  negative 1
## 45            sap  negative 1
## 46          scrap  negative 1
## 47         secure  positive 1
## 48     skepticism  negative 1
## 49           soft  positive 1
## 50         tender  positive 1
## 51        tougher  positive 1
```

Inference: After applying the lexicons on the sentences, we find the positive and negative words in the dataset.

```
bing_data %>%
  group_by(sentiment) %>%
  top_n(10) %>%
  ungroup() %>%
  mutate(word = reorder(word, n)) %>%
  ggplot(aes(word, n, fill = sentiment)) + geom_col(show.legend = FALSE) +
facet_wrap(~sentiment, scales = "free_y") + labs(title = "Finance data", y =
"Contributing to sentiment", x = NULL) + coord_flip() + theme_bw()

## Selecting by n
```



Inference: The given plot shows all the top 10 positive and negative words used in the dataset.

Applying sentiment analysis on the data, and splitting on whether the score of the sentence is zero or a non-zero value:

```
sentiment_bing = function(txt){
  #Perform basic cleaning
  txt_tbl = tibble(text = txt) %>%
    mutate(
      #Remove $ elements
      stripped_text = gsub("$S+","",text)
    ) %>%
    unnest_tokens(word,stripped_text) %>%
    anti_join(stop_words) %>% #Remove stop words
    inner_join(get_sentiments("bing")) %>% #Merge with bing sentiment
    count(word, sentiment, sort = TRUE) %>%
    ungroup() %>%
    #Create a column "score", that assigns -1 to all negative words, and +1
to all positive words
    mutate(
      score = case_when(
        sentiment == 'negative'~n*(-1),
        sentiment == 'positive'~n*1)
    )
  #Calculate total score
  sent.score = case_when(
```

```
    nrow(txt_tbl) == 0~0, #if there are no words, then the score is 0
    nrow(txt_tbl) > 0~sum(txt_tbl$score)
  )
  #To keep track of which sentence contained no words at all from the bing
list
  zero.type = case_when(
    nrow(txt_tbl) == 0~"Type 1", #Type 1: no words at all, zero = no
    nrow(txt_tbl) > 0~"Type 2" #Type 2: zero means sum of words = 0
  )
  list(score = sent.score, type = zero.type, txt_tbl = txt_tbl)
}

#lapply function
data_sent = lapply(data$Sentence,function(x){sentiment_bing(x)})
```

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

```
data_sent
```

```
## [[1]]
## [[1]]$score
## [1] 2
##
## [[1]]$type
## [1] "Type 2"
##
## [[1]]$txt_tbl
## # A tibble: 2 x 4
##   word     sentiment     n score
##   <chr>    <chr>     <int> <dbl>
## 1 leverage positive      1     1
## 2 powerful positive      1     1
```

```
##
##
## [[2]]
## [[2]]$score
## [1] 0
##
## [[2]]$type
## [1] "Type 1"
##
## [[2]]$txt_tbl
## # A tibble: 0 x 4
## # ... with 4 variables: word <chr>, sentiment <chr>, n <int>, score <dbl>
## # i Use `colnames()` to see all variable names
##
##
## [[3]]
## [[3]]$score
## [1] -1
##
## [[3]]$type
## [1] "Type 2"
##
## [[3]]$txt_tbl
## # A tibble: 1 x 4
##   word  sentiment     n score
##   <chr> <chr>     <int> <dbl>
## 1 loss  negative      1    -1
##
##
## [[4]]
## [[4]]$score
## [1] 0
##
## [[4]]$type
## [1] "Type 1"
##
## [[4]]$txt_tbl
## # A tibble: 0 x 4
## # ... with 4 variables: word <chr>, sentiment <chr>, n <int>, score <dbl>
## # i Use `colnames()` to see all variable names
##
##
## [[5]]
## [[5]]$score
## [1] 0
##
## [[5]]$type
## [1] "Type 1"
##
## [[5]]$txt_tbl
```

```
## # A tibble: 0 x 4
## # ... with 4 variables: word <chr>, sentiment <chr>, n <int>, score <dbl>
## # i Use `colnames()` to see all variable names
##
##
## [[6]]
## [[6]]$score
## [1] 0
##
## [[6]]$type
## [1] "Type 1"
##
## [[6]]$txt_tbl
## # A tibble: 0 x 4
## # ... with 4 variables: word <chr>, sentiment <chr>, n <int>, score <dbl>
## # i Use `colnames()` to see all variable names
##
##
## [[7]]
## [[7]]$score
## [1] -1
##
## [[7]]$type
## [1] "Type 2"
##
## [[7]]$txt_tbl
## # A tibble: 1 x 4
##   word       sentiment     n score
##   <chr>      <chr>     <int> <dbl>
## 1 skepticism negative      1    -1
##
##
## [[8]]
## [[8]]$score
## [1] -1
##
## [[8]]$type
## [1] "Type 2"
##
## [[8]]$txt_tbl
## # A tibble: 1 x 4
##   word  sentiment     n score
##   <chr> <chr>     <int> <dbl>
## 1 loss  negative      1    -1
##
...
...
## [[95]]
## [[95]]$score
## [1] 0
```

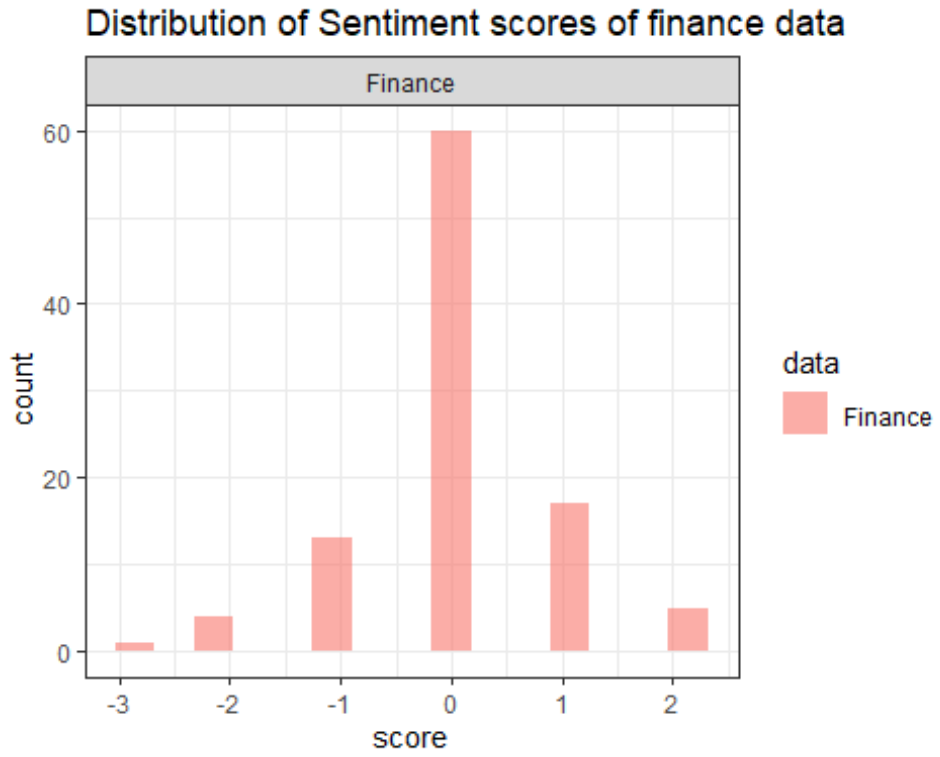
```
##
## [[95]]$type
## [1] "Type 1"
##
## [[95]]$txt_tbl
## # A tibble: 0 x 4
## # ... with 4 variables: word <chr>, sentiment <chr>, n <int>, score <dbl>
## # i Use `colnames()` to see all variable names
##
##
## [[96]]
## [[96]]$score
## [1] -1
##
## [[96]]$type
## [1] "Type 2"
##
## [[96]]$txt_tbl
## # A tibble: 1 x 4
##   word   sentiment     n score
##   <chr>  <chr>     <int> <dbl>
## 1 rumors negative      1    -1
##
##
## [[97]]
## [[97]]$score
## [1] 0
##
## [[97]]$type
## [1] "Type 2"
##
## [[97]]$txt_tbl
## # A tibble: 2 x 4
##   word      sentiment     n score
##   <chr>     <chr>     <int> <dbl>
## 1 favorable positive      1     1
## 2 scrap     negative      1    -1
##
##
## [[98]]
## [[98]]$score
## [1] -2
##
## [[98]]$type
## [1] "Type 2"
##
## [[98]]$txt_tbl
## # A tibble: 2 x 4
##   word  sentiment     n score
##   <chr> <chr>     <int> <dbl>
```

```
## 1 debt  negative      1    -1
## 2 moody negative      1    -1
##
##
## [[99]]
## [[99]]$score
## [1] 0
##
## [[99]]$type
## [1] "Type 1"
##
## [[99]]$txt_tbl
## # A tibble: 0 x 4
## # ... with 4 variables: word <chr>, sentiment <chr>, n <int>, score <dbl>
## # i Use `colnames()` to see all variable names
##
##
## [[100]]
## [[100]]$score
## [1] 0
##
## [[100]]$type
## [1] "Type 1"
##
## [[100]]$txt_tbl
## # A tibble: 0 x 4
## # ... with 4 variables: word <chr>, sentiment <chr>, n <int>, score <dbl>
## # i Use `colnames()` to see all variable names
```

```
data_sentiment = bind_rows(
  tibble(
    data = 'Finance',
    score = unlist(map(data_sent,'score')),
    type = unlist(map(data_sent,'type'))
  )
)
data_sentiment
```

```
## # A tibble: 100 x 3
##    data    score type
##    <chr>   <dbl> <chr>
##  1 Finance     2 Type 2
##  2 Finance     0 Type 1
##  3 Finance    -1 Type 2
##  4 Finance     0 Type 1
##  5 Finance     0 Type 1
##  6 Finance     0 Type 1
##  7 Finance    -1 Type 2
##  8 Finance    -1 Type 2
##  9 Finance     0 Type 1
```

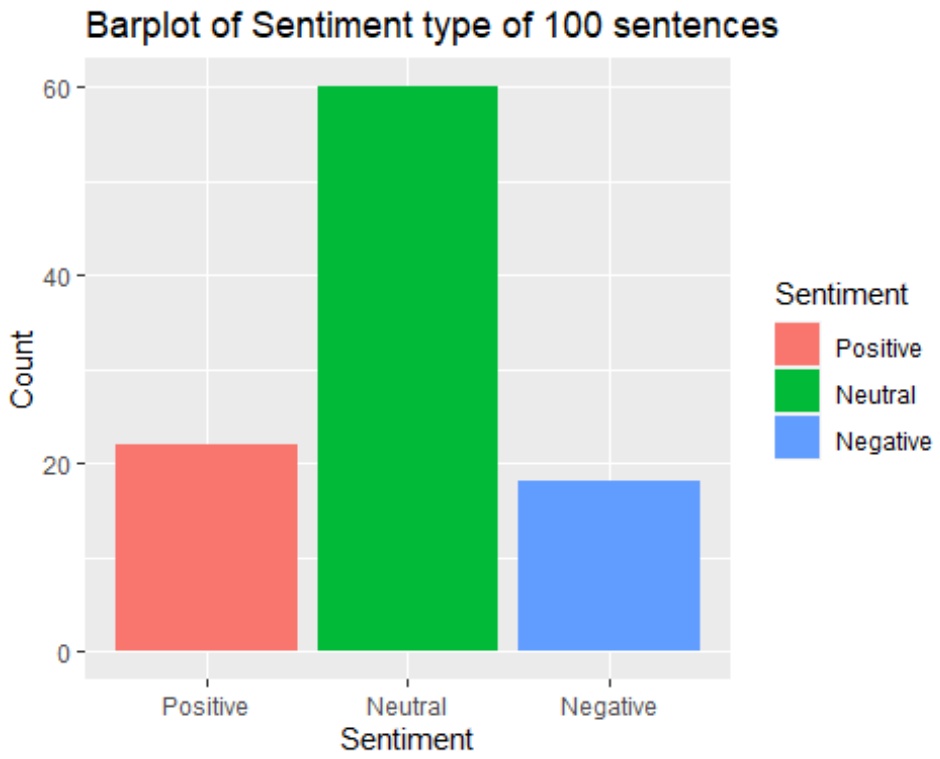
```
## 10 Finance     0 Type 1
## # ... with 90 more rows
## # i Use `print(n = ...)` to see more rows

head(data_sentiment)

## # A tibble: 6 x 3
##   data    score type
##   <chr>   <dbl> <chr>
## 1 Finance     2 Type 2
## 2 Finance     0 Type 1
## 3 Finance    -1 Type 2
## 4 Finance     0 Type 1
## 5 Finance     0 Type 1
## 6 Finance     0 Type 1

ggplot(data_sentiment,aes(x = score, fill = data)) + geom_histogram(bins =
15, alpha = 0.6) + facet_grid(~data) + ggtitle("Distribution of Sentiment
scores of finance data") + theme_bw()
```



Inference: This is a histogram showing the distribution of different sentiment scores. We can see that neutral sentence with sentiment score of 0 are more frequent, then comes positive sentences with score>0 and negative sentences are the least frequent with score<0.

```
#Barplot of sentiment type
neutral <- length(which(data_sentiment$score == 0))
```

```
positive <- length(which(data_sentiment$score > 0))
negative <- length(which(data_sentiment$score < 0))
Sentiment <- c("Positive","Neutral","Negative")
Count <- c(positive,neutral,negative)
output <- data.frame(Sentiment,Count)
output$Sentiment<-factor(output$Sentiment,levels=Sentiment)
ggplot(output, aes(x=Sentiment,y=Count))+
  geom_bar(stat = "identity", aes(fill = Sentiment))+
  ggtitle("Barplot of Sentiment type of 100 sentences")
```



Inference: Here is a bar plot showing the number of sentences that are positive, negative, and neutral.

Conclusion:

The dataset that contains financial sentences is pre-processed and cleaned by removing unwanted words. The most frequent words and positive and negative words are found. Then on applying sentiment analysis, using the AFINN lexicon, the scores are computed. These are then plotted graphically and show that most sentences are neutral, followed by positive sentences. Comparatively a smaller number of negative sentiments are found in the given dataset.