Appendix 1.1

Salman Saleem Virani

2023-03-20

Contents

Introduction

This work will try to compare the sentiment scores of Vader and Textblob on Sentiment140 tweets data set and US Airlines Tweets data set. First, we will try to test for any statistical significance in the scoring of these two popular sentiment analysis technique.

Libraries and Seed Setting

Some important packages that will be used in the analysis are: 1. ‘readr’ that will be used for reading and writing the csv files they we will be working on in this project. 2. ‘dplyr’ which is one the most popular packages of the tidy verse framework will be used for data wrangling tasks. 3. ‘textclean’, ‘tidytext’ and ‘stringr’ will be used for text manipulation work. 4. ‘caret’ will be used for machine learning and extracting model measurement works. 5. ‘purrr’ will be used specifically for the map_dbl function which will apply the word count function over clean_tweet column of the data set. 6. ‘broom’ will be used for cleaning the results of the models and bringing it in a presentable manner.

```r
library(readr)
library(dplyr)
library(textclean)
library(tidytext)
library(stringr)
library(caret)
library(purrr)
library(broom)
library(ggplot2)
library(tm)
library(wordcloud)
library(topicmodels)

set.seed(12345)
```

Data

Sentiment140

Sentiment140 data will be directly downloaded from the web and necessary adjustments will be made before the data is ready to use.

```r
file_path1 <- "./data/training.zip"

if(!file.exists(file_path1)){
  dir.create("./data")
  url <- "http://cs.stanford.edu/people/alecmgo/trainingandtestdata.zip"
  download.file(url, file_path1)
}

data_sentiment140 <- read_csv(unz("data/training.zip",
                                  "training.1600000.processed.noemoticon.csv"),
                              col_names= F,
                              locale = locale(encoding = "Latin1"))
```

```
data_sentiment140 <- data_sentiment140 %>%
  select(X1,X6)

colnames(data_sentiment140) <- c("target", "tweet")

head(data_sentiment140)
```

```
## # A tibble: 6 x 2
##   target tweet
##    <dbl> <chr>
## 1      0 @switchfoot http://twitpic.com/2y1zl - Awww, that's a bummer.  You sho~
## 2      0 is upset that he can't update his Facebook by texting it... and might ~
## 3      0 @Kenichan I dived many times for the ball. Managed to save 50%  The re~
## 4      0 my whole body feels itchy and like its on fire
## 5      0 @nationwideclass no, it's not behaving at all. i'm mad. why am i here?~
## 6      0 @Kwesidei not the whole crew
```

US Airlines Data

Data set will be loaded from the project repository.

```
data_usairline <- read_csv("data/Tweets.csv", locale = locale(encoding =
                                                                "UTF-8"))


data_usairline <- data_usairline %>%
  select(airline_sentiment, text)
```

Apple Twitter Sentiment Dataset

```
data_apple <- read_csv("data/apple.csv", locale = locale(encoding = "Latin1"))

data_apple <- data_apple %>%
  select(sentiment, text) %>%
  filter(sentiment != "not_relevant")
```

Text Preprocessing and Sentiment Scoring

Text Pre processing steps are explained in the Research Methodology section of the paper.

```
text_cleaning <- function(x) {
  x %>%
    replace_non_ascii() %>%
    str_replace_all(pattern = "\\@.*? |\\@.*?[:punct:]", replacement = " ") %>%
    replace_url() %>%
    replace_hash() %>%
    replace_contraction() %>%
    str_replace_all("[:digit:]", " ") %>%
    str_trim() %>%
    str_squish()
}
```

```
data_sentiment140 <- data_sentiment140 %>%
  mutate(
    clean_tweet = text_cleaning(tweet)
  )
```

```
data_usairline <- data_usairline %>%
  mutate(
    clean_tweet = text_cleaning(text)
  )
```

```
data_apple <- data_apple %>%
  mutate(
    clean_tweet = text_cleaning(text)
  )
```

After cleaning the tweets, the rows of the data where the clean_tweet column had less than three words were removed.

```
word_count_sentiment140 <- map_dbl(data_sentiment140$clean_tweet,
                      function(x) str_split(x, " ") %>%
                        unlist() %>%
                        length()
                      )
```

```
word_count_usairline <- map_dbl(data_usairline$clean_tweet,
                      function(x) str_split(x, " ") %>%
                        unlist() %>%
                        length()
                      )
```

```
word_count_apple <- map_dbl(data_apple$clean_tweet,
                      function(x) str_split(x, " ") %>%
                        unlist() %>%
                        length()
                      )
```

```
data_sentiment140 <- data_sentiment140 %>%
  filter(word_count_sentiment140 > 3)
```

```
data_usairline <- data_usairline %>%
  filter(word_count_usairline > 3)
```

```
data_apple <- data_apple %>%
  filter(word_count_apple > 3)
```

Distribution of Sentiment Categories in the Sentiment140 Dataset

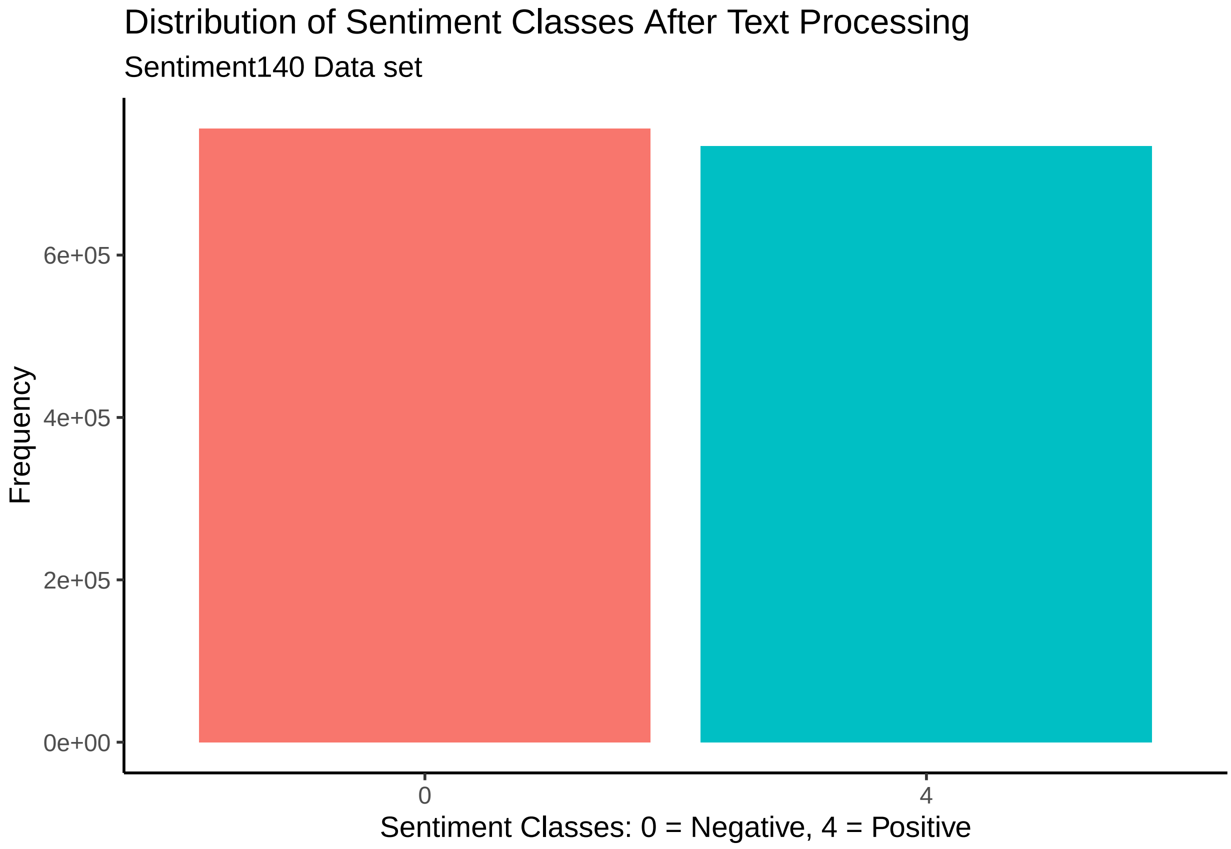
```
table(data_sentiment140$target)
```

```
##
##      0      4
## 755853 734371
```

```
prop.table(table(data_sentiment140$target))
```

```
##
##         0         4
## 0.5072076 0.4927924
```

```
data_sentiment140$target <- as.factor(data_sentiment140$target)
ggplot(data_sentiment140, aes(target, fill = target)) +
  geom_bar() +
  labs(
    x = "Sentiment Classes: 0 = Negative, 4 = Positive",
    y = "Frequency",
    title = "Distribution of Sentiment Classes After Text Processing",
    subtitle = "Sentiment140 Data set",
  ) +
  theme_classic() +
  theme(legend.position="none")
```



Distribution of Sentiment Categories in the US Airline Dataset Dataset

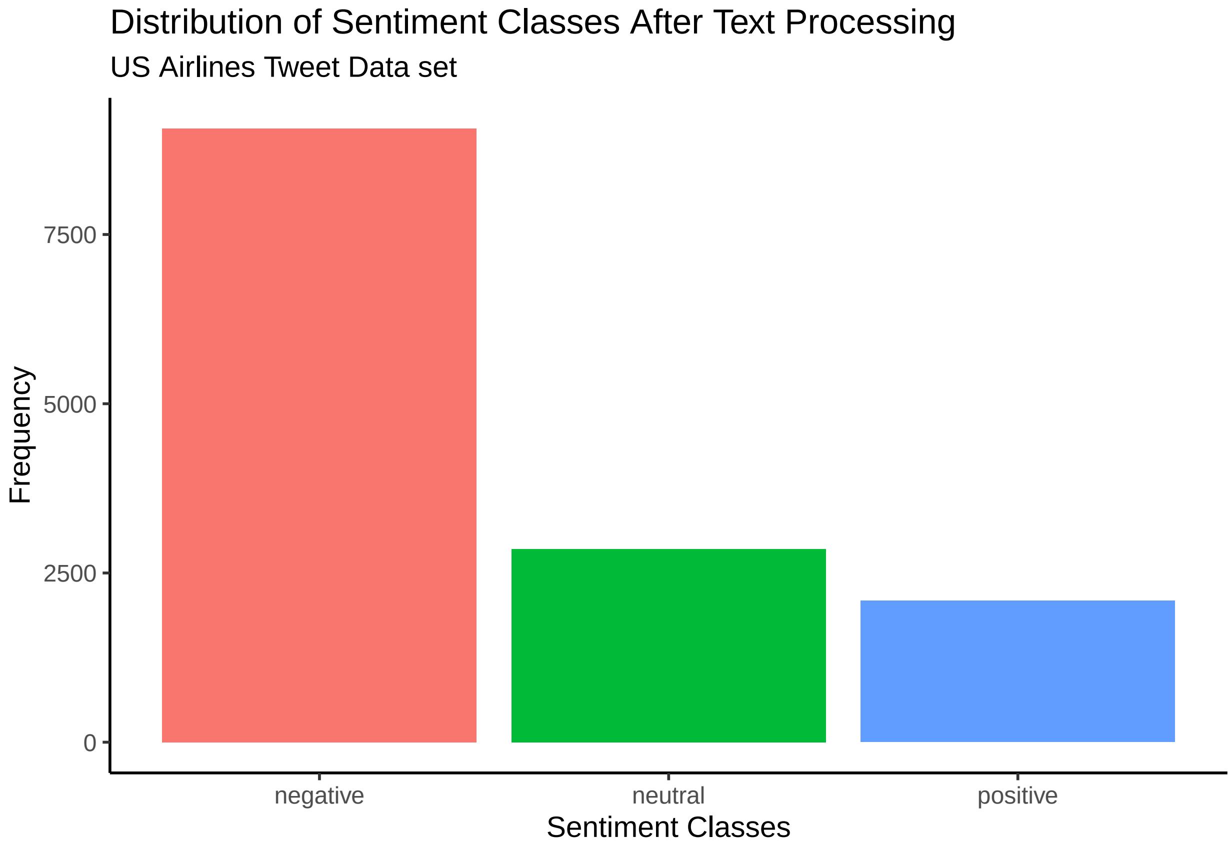
```
table(data_usairline$airline_sentiment)
```

```
##
## negative  neutral positive
##     9065     2854     2090
```

```
prop.table(table(data_usairline$airline_sentiment))
```

```
##
##  negative   neutral  positive
## 0.6470840 0.2037262 0.1491898
```

```
ggplot(data_usairline, aes(airline_sentiment, fill = airline_sentiment)) +
  geom_bar() +
  labs(
    x = "Sentiment Classes",
    y = "Frequency",
    title = "Distribution of Sentiment Classes After Text Processing",
    subtitle = "US Airlines Tweet Data set",
  ) +
  theme_classic() +
  theme(legend.position="none")
```



Distribution of Sentiment Categories in the Apple Dataset

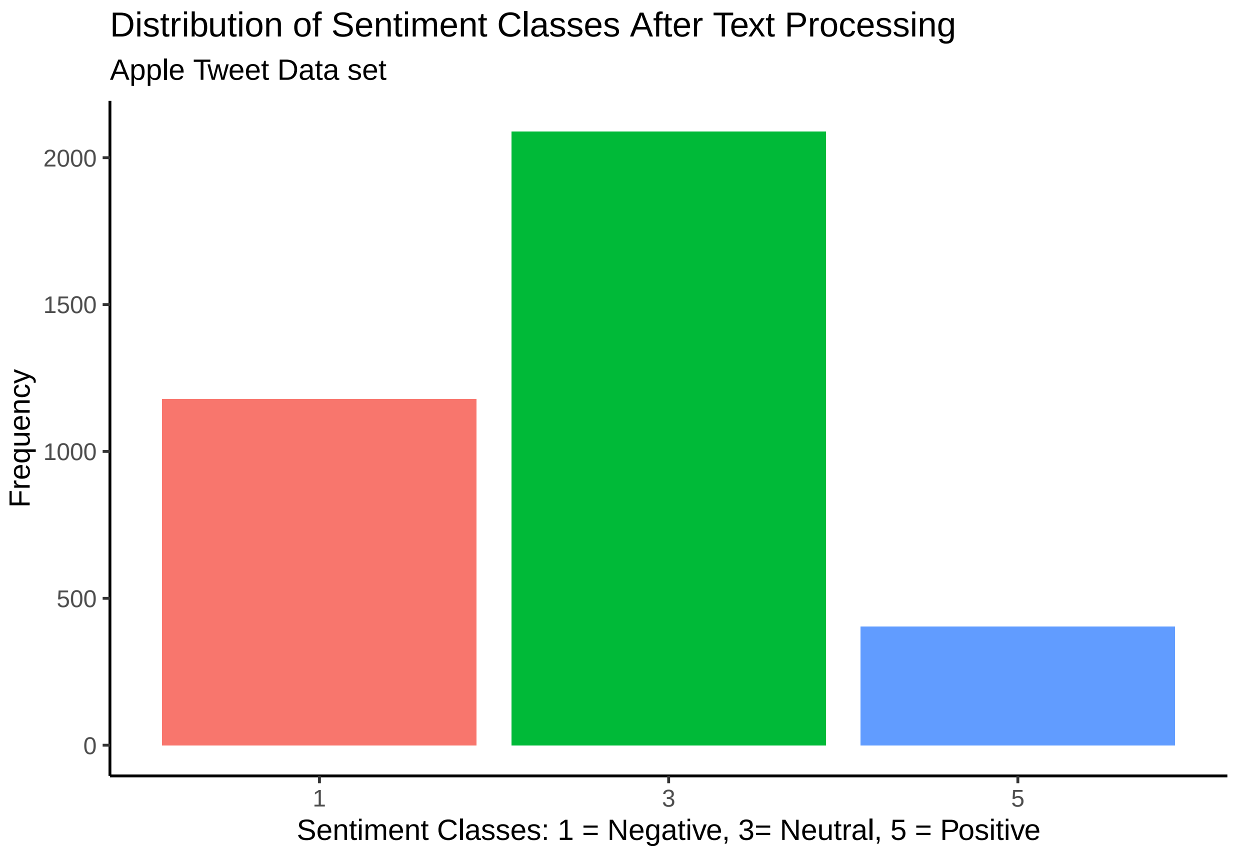
```
table(data_apple$sentiment)
```

```
##
##    1    3    5
## 1178 2089  404
```

```
prop.table(table(data_apple$sentiment))
```

```
##
##         1         3         5
## 0.3208935 0.5690548 0.1100518
```

```
ggplot(data_apple, aes(sentiment, fill = sentiment)) +
  geom_bar() +
  labs(
    x = "Sentiment Classes: 1 = Negative, 3= Neutral, 5 = Positive",
    y = "Frequency",
    title = "Distribution of Sentiment Classes After Text Processing",
    subtitle = "Apple Tweet Data set",
  ) +
  theme_classic() +
  theme(legend.position="none")
```



```
rm(word_count_sentiment140)
rm(word_count_usairline)
rm(word_count_apple)
```

Exporting the cleaned text data

After the text cleaning task is completed, data are exported as .csv files and the files are loaded in Python for sentiment scoring through TEXTBLOB and VADER techniques.

```
write_csv(data_sentiment140, "data/sentiment140_clean_tweet.csv")
write_csv(data_usairline, "data/usairline_clean_tweet.csv")
```

```
write_csv(data_apple, "data/apple_clean_tweet.csv")
```

Importing data with Vader and Textblob Scores

The dataset with the sentiment scores are then loaded back into R.

```
data_final_sentiment140 <- read_csv("data/sentiment140_scores.csv",
                                    locale = locale(encoding = "Latin1"))

data_final_usairline <- read_csv("data/usairline_scores.csv",
                                 locale = locale(encoding = "Latin1"))
```

```
data_final_apple <- read_csv("data/apple_scores.csv",
                             locale = locale(encoding = "Latin1"))
```

```
summary(data_final_sentiment140$textblob)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
## -1.0000  0.0000  0.0000  0.1031  0.3000  1.0000 
```

```
summary(data_final_sentiment140$vader)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
## -0.9985 -0.1280  0.0000  0.1487  0.5574  0.9987 
```

```
summary(data_final_usairline$textblob)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
## -1.00000 -0.01389  0.00000  0.05009  0.20000  1.00000 
```

```
summary(data_final_usairline$vader)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
## -0.96680 -0.29600  0.00000  0.04862  0.43290  0.97600 
```

```
summary(data_final_apple$textblob)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max. 
## -1.00000  0.00000  0.00000  0.03892  0.13523  1.00000
```

```
summary(data_final_apple$vader)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
## -0.978700 -0.226300  0.000000  0.003365  0.273200  0.939300
```

```
rm(data_sentiment140)
rm(data_usairline)
```

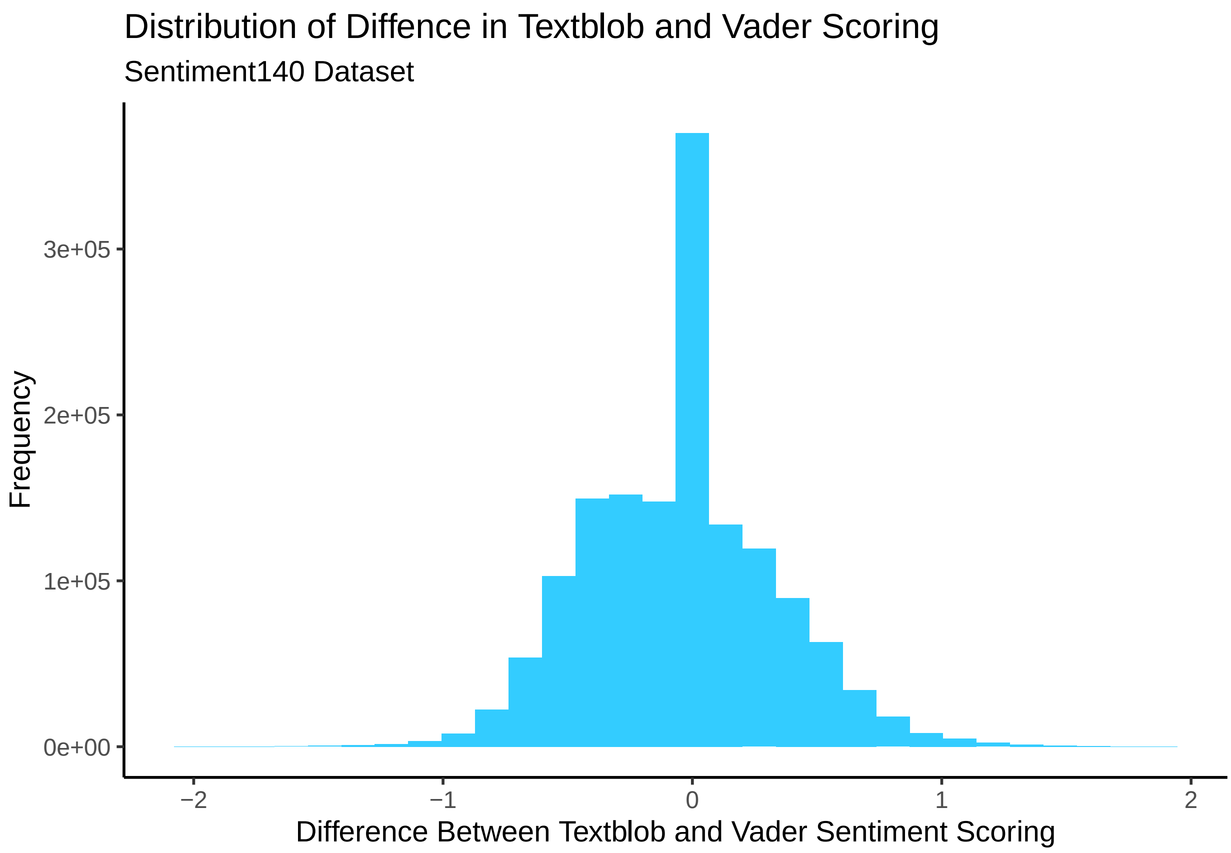
Statistical Testing

Analysis of Distribution of Differences in Sentiment Scoring by Textblob

and Vader

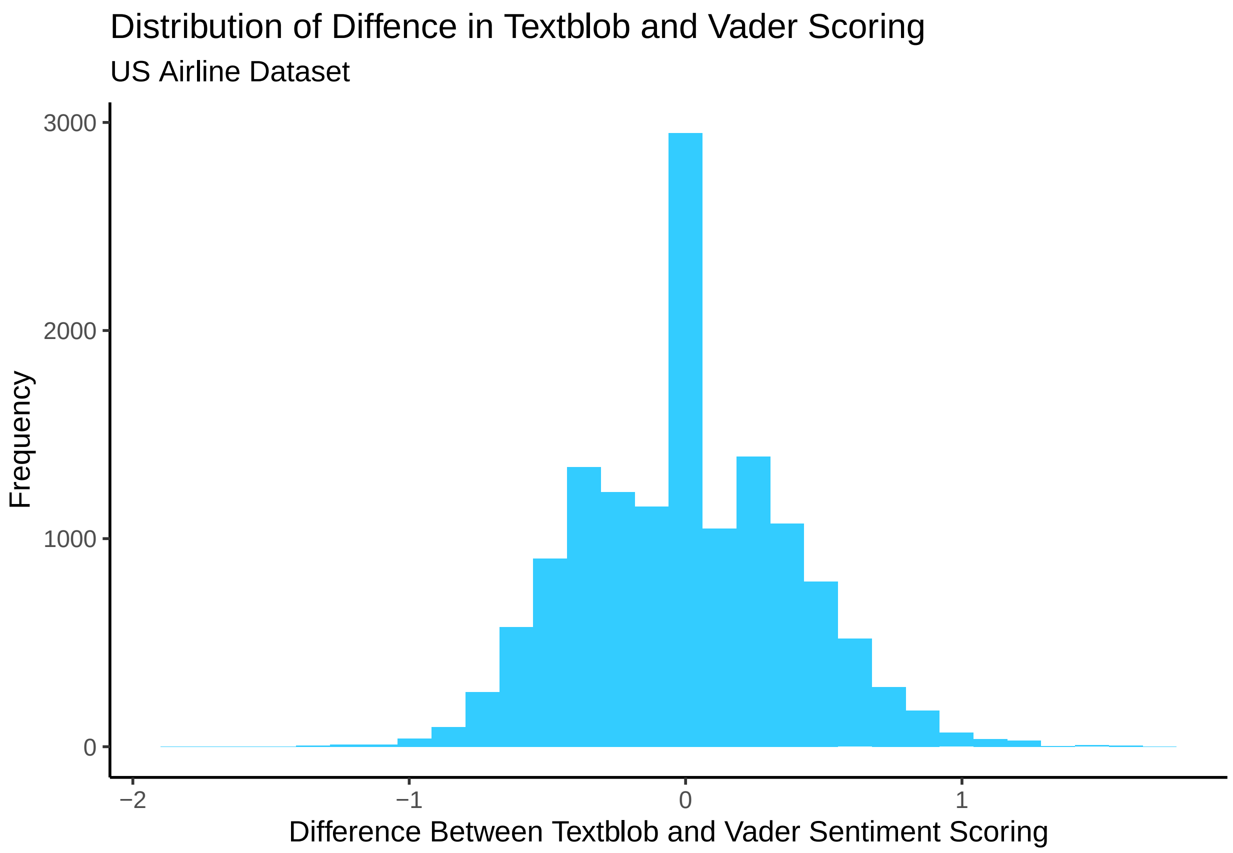
Sentiment140 dataset

```
data_final_sentiment140 <- data_final_sentiment140 %>%
  mutate(
    diff = textblob - vader
  )

ggplot(data_final_sentiment140,aes(diff)) +
  geom_histogram(fill = "#33ccff") +
  labs(y = "Frequency",
       x = "Difference Between Textblob and Vader Sentiment Scoring",
      title = "Distribution of Diffence in Textblob and Vader Scoring",
      subtitle = "Sentiment140 Dataset") +
  theme_classic()
```



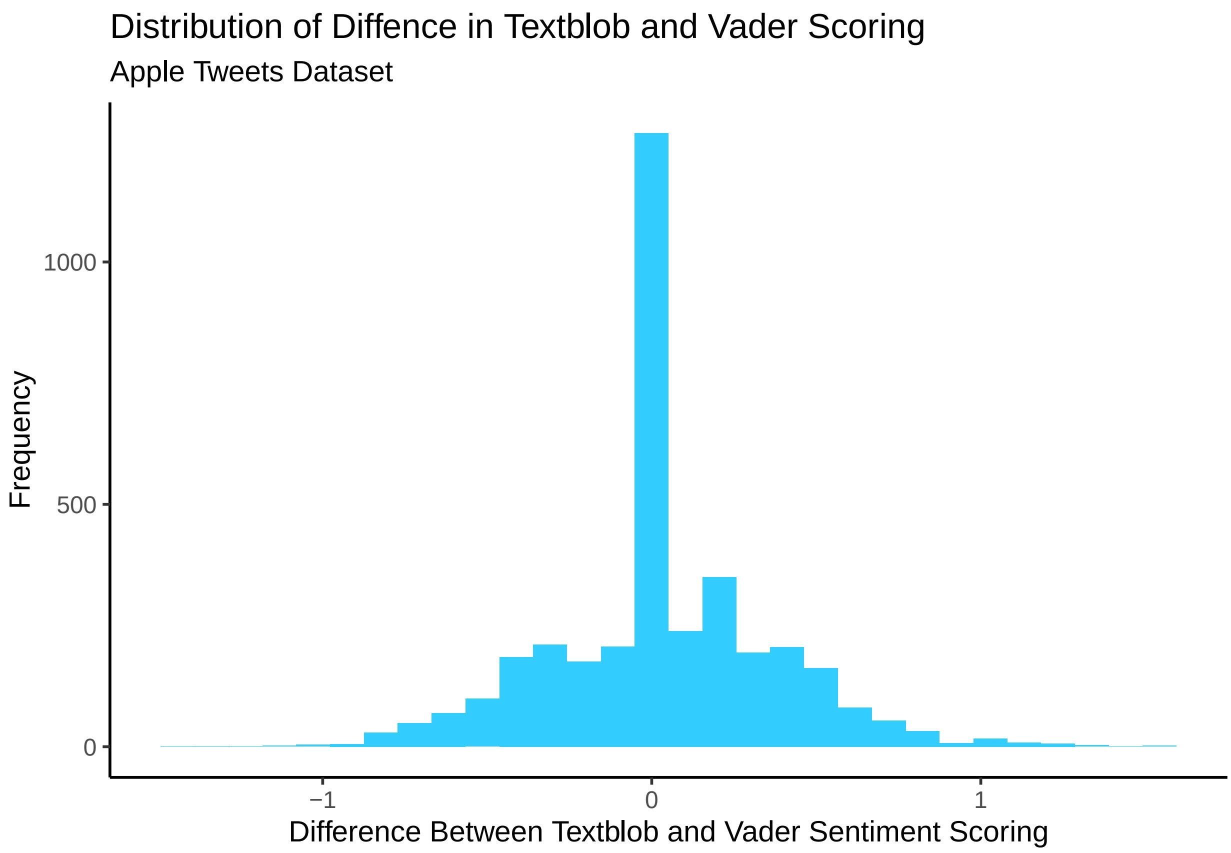
US Airline Data set

```
data_final_usairline <- data_final_usairline %>%
  mutate(
    diff = textblob - vader
  )

ggplot(data_final_usairline,aes(diff)) +
  geom_histogram(fill = "#33ccff") +
  labs(y = "Frequency",
       x = "Difference Between Textblob and Vader Sentiment Scoring",
      title = "Distribution of Diffence in Textblob and Vader Scoring",
      subtitle = "US Airline Dataset") +
  theme_classic()
```



Apple Tweets Data set

```
data_final_apple <- data_final_apple %>%
  mutate(
    diff = textblob - vader
  )

ggplot(data_final_apple,aes(diff)) +
  geom_histogram(fill = "#33ccff") +
  labs(y = "Frequency",
        x = "Difference Between Textblob and Vader Sentiment Scoring",
       title = "Distribution of Diffence in Textblob and Vader Scoring",
       subtitle = "Apple Tweets Dataset") +
  theme_classic()
```



Testing for any Statistical difference

Sentiment140 Dataset

```
tidy(t.test(data_final_sentiment140$textblob, data_final_sentiment140$vader,
            paired = T, alternative = "two.sided"))
```

```
## # A tibble: 1 x 8
##   estimate statistic p.value parameter conf.low conf.high method     alternative
##      <dbl>     <dbl>   <dbl>     <dbl>    <dbl>     <dbl> <chr>      <chr>
## 1  -0.0456     -148.       0   1490223  -0.0462   -0.0450 Paired t-~ two.sided
```

US Airline Dataset

```
tidy(t.test(data_final_usairline$textblob, data_final_usairline$vader,
            paired = T, alternative = "two.sided"))
```

```
## # A tibble: 1 x 8
##   estimate statistic p.value parameter conf.low conf.high method     alternative
##      <dbl>     <dbl>   <dbl>     <dbl>    <dbl>     <dbl> <chr>      <chr>
## 1  0.00147     0.449   0.654     14016 -0.00495   0.00788 Paired t-~ two.sided
```

Apple Tweets Dataset

```
tidy(t.test(data_final_apple$textblob, data_final_apple$vader, paired = T,
            alternative = "two.sided"))
```

```
## # A tibble: 1 x 8
##   estimate statistic  p.value parameter conf.low conf.high method    alternative
##      <dbl>     <dbl>    <dbl>     <dbl>    <dbl>     <dbl> <chr>     <chr>
## 1   0.0356      6.32 2.86e-10      3670   0.0245    0.0466 Paired t~ two.sided
```

Tweets classification based on Sentiment Scores and Accuracy Measures

Sentiment140 Data set

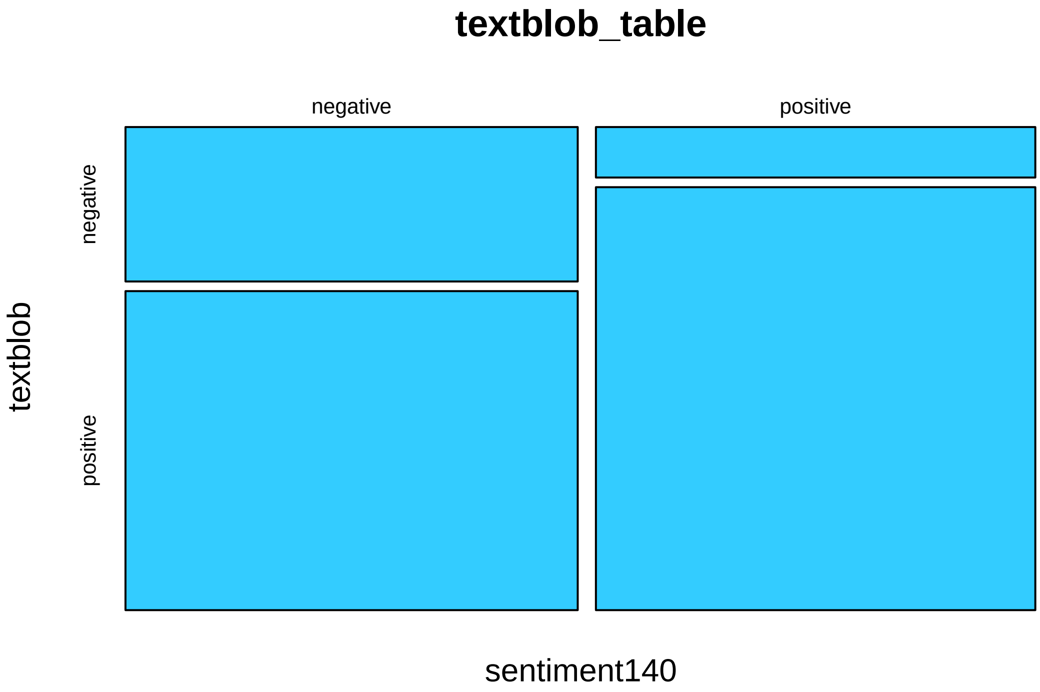
```
data_final_sentiment140 <- data_final_sentiment140 %>%
  mutate(
    senti140_class = case_when(
      target == 0 ~ "negative",
      target == 4 ~ "positive"
    ),
    textblob_class = case_when(
      textblob >= 0 ~ "positive",
      textblob < 0 ~ "negative"
    ),
    vader_class = case_when(
      vader >= 0 ~ "positive",
      vader < 0 ~ "negative"
  )
  )
```

```
textblob_table <- table(sentiment140 = data_final_sentiment140$senti140_class,
                        textblob = data_final_sentiment140$textblob_class)

(conf_mat_textblob <- confusionMatrix(textblob_table))
```

```
## Confusion Matrix and Statistics
##
##             textblob
## sentiment140 negative positive
##     negative   246538   509315
##     positive    78298   656073
##
##                Accuracy : 0.6057
##                  95% CI : (0.6049, 0.6065)
##     No Information Rate : 0.782
##     P-Value [Acc > NIR] : 1
##
```

```
##                   Kappa : 0.2177
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7590
##             Specificity : 0.5630
##          Pos Pred Value : 0.3262
##          Neg Pred Value : 0.8934
##              Prevalence : 0.2180
##          Detection Rate : 0.1654
##    Detection Prevalence : 0.5072
##       Balanced Accuracy : 0.6610
##
##        'Positive' Class : negative
##
```

```
plot(textblob_table, color = "#33ccff")
```



```
textblob_sent140_results <- rbind(as.matrix(confusionMatrix(textblob_table),
                                            what = "overall"),
                                  as.matrix(confusionMatrix(textblob_table),
                                            what = "classes"))

Measurements <- rownames(textblob_sent140_results)
Textblob <- textblob_sent140_results[1:18]
```

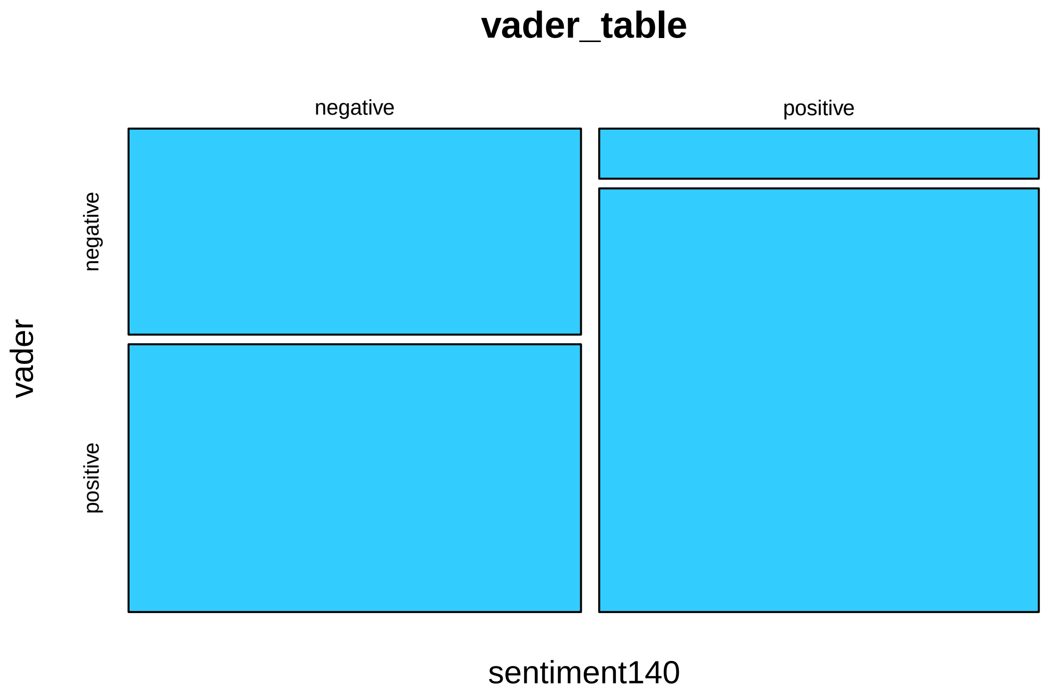
```r
textblob_sent140_results_df <- data.frame(Measurements,Textblob)
str(textblob_sent140_results_df)
```

```
## ’data.frame’:    18 obs. of  2 variables:
##  $ Measurements: chr  "Accuracy" "Kappa" "AccuracyLower" "AccuracyUpper" ...
##  $ Textblob    : num  0.606 0.218 0.605 0.606 0.782 ...
```

```r
vader_table <- table(sentiment140 = data_final_sentiment140$senti140_class,
                     vader = data_final_sentiment140$vader_class)

(conf_mat_vader <- confusionMatrix(vader_table))
```

```
## Confusion Matrix and Statistics
##
##             vader
## sentiment140 negative positive
##     negative   328681   427172
##     positive    77670   656701
##
##                Accuracy : 0.6612
##                  95% CI : (0.6605, 0.662)
##     No Information Rate : 0.7273
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.3269
##
##  Mcnemar’s Test P-Value : <2e-16
##
##             Sensitivity : 0.8089
##             Specificity : 0.6059
##          Pos Pred Value : 0.4348
##          Neg Pred Value : 0.8942
##              Prevalence : 0.2727
##          Detection Rate : 0.2206
##    Detection Prevalence : 0.5072
##       Balanced Accuracy : 0.7074
##
##        ’Positive’ Class : negative
##
```

```r
plot(vader_table, color = "#33ccff")
```



```
vader_sent140_results <- rbind(as.matrix(confusionMatrix(vader_table),
                                         what = "overall"),
                               as.matrix(confusionMatrix(vader_table),
                                         what = "classes"))

vader_sent140_results_df <- data.frame(vader_sent140_results[1:18])
colnames(vader_sent140_results_df) <- "Vader"
str(vader_sent140_results_df)
```

```
## ’data.frame’:    18 obs. of  1 variable:
##  $ Vader: num  0.661 0.327 0.66 0.662 0.727 ...
```

```
results_sent140 <- cbind(textblob_sent140_results_df, vader_sent140_results_df)
results_sent140
```

```
##            Measurements  Textblob     Vader
## 1              Accuracy 0.6056881 0.6612308
## 2                 Kappa 0.2177367 0.3268731
## 3         AccuracyLower 0.6049030 0.6604703
## 4         AccuracyUpper 0.6064729 0.6619907
## 5          AccuracyNull 0.7820220 0.7273222
## 6        AccuracyPValue 1.0000000 1.0000000
## 7         McnemarPValue 0.0000000 0.0000000
## 8           Sensitivity 0.7589614 0.8088598
## 9           Specificity 0.5629653 0.6058837
## 10       Pos Pred Value 0.3261719 0.4348478
## 11       Neg Pred Value 0.8933809 0.8942360
## 12            Precision 0.3261719 0.4348478
## 13               Recall 0.7589614 0.8088598
## 14                   F1 0.4562608 0.5656167
## 15           Prevalence 0.2179780 0.2726778
## 16       Detection Rate 0.1654369 0.2205581
## 17 Detection Prevalence 0.5072076 0.5072076
## 18    Balanced Accuracy 0.6609634 0.7073718
```

```
rm(conf_mat_textblob)
rm(conf_mat_vader)
rm(textblob_sent140_results)
rm(textblob_sent140_results_df)
rm(vader_sent140_results)
rm(vader_sent140_results_df)
rm(Measurements)
rm(Textblob)
rm(textblob_table)
rm(vader_table)
```

US Airlines Dataset

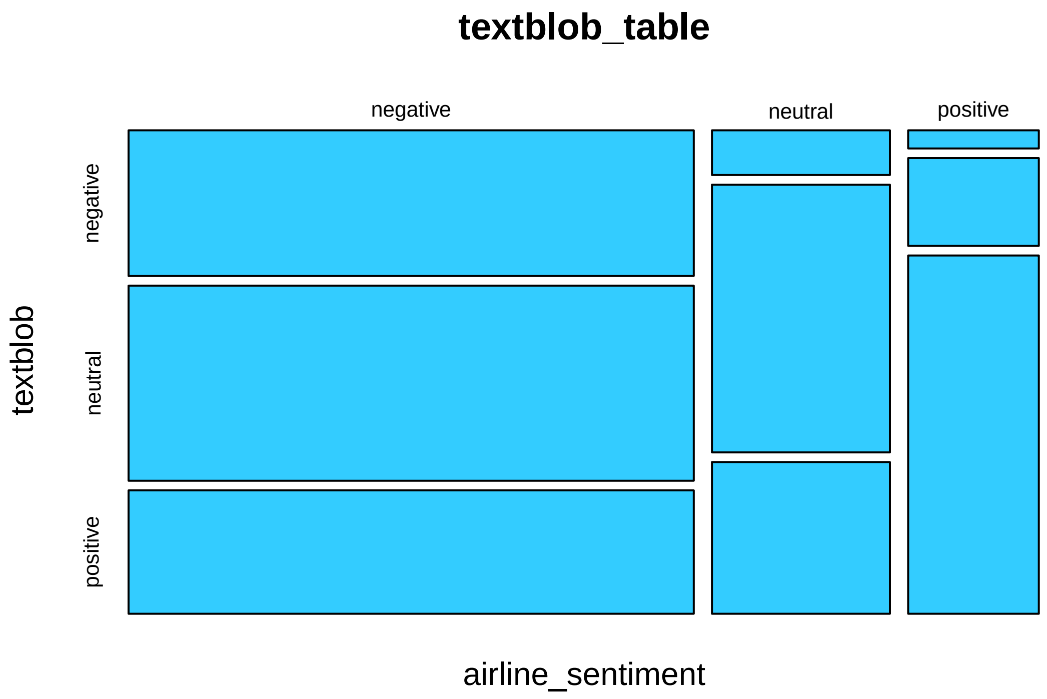
```
data_final_usairline <- data_final_usairline %>%
  mutate(
    textblob_class = case_when(
      textblob > 0.05 ~ "positive",
      textblob < -0.05 ~ "negative",
      textblob >= -0.05 & textblob <= 0.05 ~ "neutral"
    ),
    vader_class = case_when(
      vader > 0.05 ~ "positive",
      vader < -0.05 ~ "negative",
      vader >= -0.05 & vader <= 0.05 ~ "neutral"
  )
  )
```

```
textblob_table <- table(airline_sentiment =
                          data_final_usairline$airline_sentiment,
                        textblob = data_final_usairline$textblob_class)

(conf_mat_textblob <- confusionMatrix(textblob_table))
```

```
## Confusion Matrix and Statistics
##
##                   textblob
## airline_sentiment negative neutral positive
##          negative     2845    3811     2410
##          neutral       275    1647      933
##          positive       82     396     1618
##
```

```
## Overall Statistics
##
##                Accuracy : 0.4359
##                  95% CI : (0.4277, 0.4442)
##     No Information Rate : 0.4176
##     P-Value [Acc > NIR] : 6.273e-06
##
##                   Kappa : 0.2102
##
##  Mcnemar’s Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: negative Class: neutral Class: positive
## Sensitivity                   0.8885         0.2813          0.3261
## Specificity                   0.4248         0.8520          0.9472
## Pos Pred Value                0.3138         0.5769          0.7719
## Neg Pred Value                0.9279         0.6231          0.7196
## Prevalence                    0.2284         0.4176          0.3539
## Detection Rate                0.2030         0.1175          0.1154
## Detection Prevalence          0.6468         0.2037          0.1495
## Balanced Accuracy             0.6566         0.5667          0.6367
```

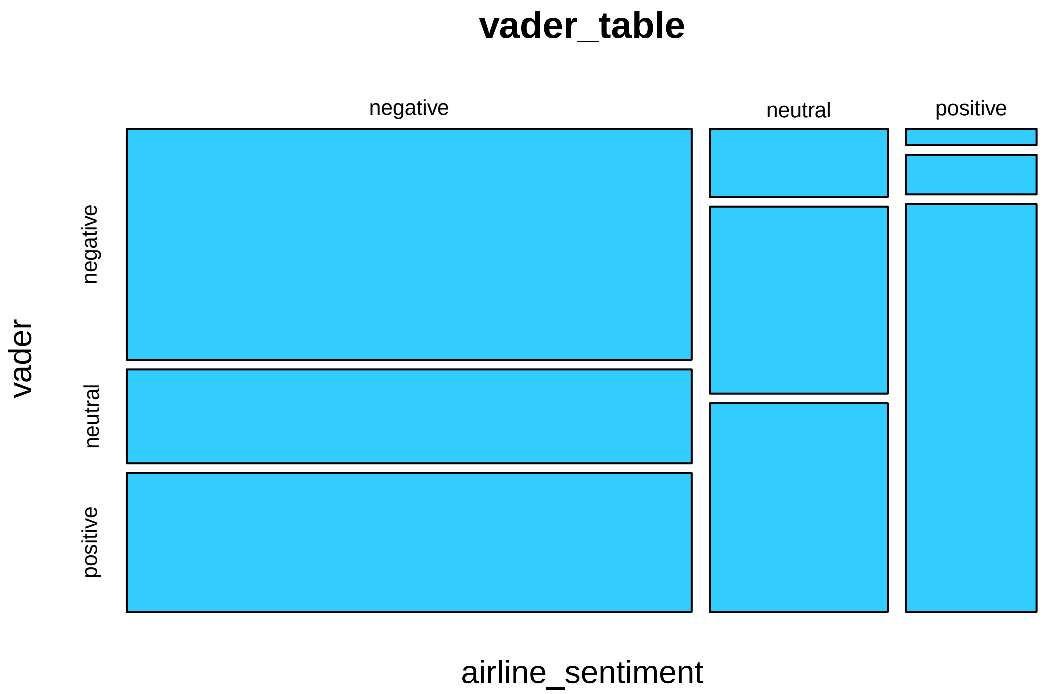
```
plot(textblob_table, color = "#33ccff")
```



```
vader_table <- table(airline_sentiment = data_final_usairline$airline_sentiment,
                     vader = data_final_usairline$vader_class)

(conf_mat_vader <- confusionMatrix(vader_table))
```

```
## Confusion Matrix and Statistics
##
##                  vader
## airline_sentiment negative neutral positive
##          negative     4516    1836     2714
##          neutral       420    1151     1284
##          positive       74     179     1843
##
## Overall Statistics
##
##                Accuracy : 0.5358
##                  95% CI : (0.5275, 0.5441)
##     No Information Rate : 0.4167
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2972
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: negative Class: neutral Class: positive
## Sensitivity                   0.9014        0.36355          0.3155
## Specificity                   0.4948        0.84296          0.9691
## Pos Pred Value                0.4981        0.40315          0.8793
## Neg Pred Value                0.9002        0.81948          0.6646
## Prevalence                    0.3574        0.22587          0.4167
## Detection Rate                0.3222        0.08211          0.1315
## Detection Prevalence          0.6468        0.20368          0.1495
## Balanced Accuracy             0.6981        0.60326          0.6423
```

```
plot(vader_table, color = "#33ccff")
```



```
textblob_usairline_overall <- as.matrix(conf_mat_textblob, what = "overall")
Measurements <- rownames(textblob_usairline_overall)
Textblob <- textblob_usairline_overall[1:7]
vader_usairline_overall <- as.matrix(conf_mat_vader, what = "overall")
Vader <-  vader_usairline_overall[1:7]

usairline_overall_df <- data.frame(cbind(Measurements, Textblob, Vader))
usairline_overall_df
```

```
##     Measurements             Textblob                 Vader
## 1       Accuracy    0.435899265177998     0.535777983876721
## 2          Kappa    0.210232660180692     0.297173201068076
## 3  AccuracyLower    0.427668582558055      0.52747948477537
## 4  AccuracyUpper    0.444156569115973     0.544061624707042
## 5   AccuracyNull    0.417635728044517     0.416708282799458
## 6 AccuracyPValue 6.2730455959362e-06 5.57152816029222e-177
## 7  McnemarPValue                    0                     0
```

```
textblob_usairline_byclass <- as.matrix(conf_mat_textblob, what = "classes")
Measurements <- data.frame(rownames(textblob_usairline_byclass))
colnames(Measurements) <- "Measurements"
negative <- data.frame(textblob_usairline_byclass[1:11])
colnames(negative) <- "Negative"
neutral <- data.frame(textblob_usairline_byclass[12:22])
colnames(neutral) <- "Neutral"
```

```
positive <- data.frame(textblob_usairline_byclass[23:33])
colnames(positive) <- "Positive"

textblob_usairline_byclass_df <- cbind(Measurements,negative,neutral,positive)
textblob_usairline_byclass_df
```

```
##            Measurements  Negative   Neutral  Positive
## 1           Sensitivity 0.8885072 0.2813461 0.3261439
## 2           Specificity 0.4247804 0.8520152 0.9472173
## 3        Pos Pred Value 0.3138098 0.5768827 0.7719466
## 4        Neg Pred Value 0.9278934 0.6230962 0.7195705
## 5             Precision 0.3138098 0.5768827 0.7719466
## 6                Recall 0.8885072 0.2813461 0.3261439
## 7                    F1 0.4638083 0.3782294 0.4585518
## 8            Prevalence 0.2284369 0.4176357 0.3539274
## 9        Detection Rate 0.2029678 0.1175002 0.1154313
## 10 Detection Prevalence 0.6467860 0.2036812 0.1495327
## 11    Balanced Accuracy 0.6566438 0.5666806 0.6366806
```

```
vader_usairline_byclass <- as.matrix(conf_mat_vader, what = "classes")
Measurements <- data.frame(rownames(vader_usairline_byclass))
colnames(Measurements) <- "Measurements"
Classes <- colnames(vader_usairline_byclass)
negative <- data.frame(vader_usairline_byclass[1:11])
colnames(negative) <- "Negative"
neutral <- data.frame(vader_usairline_byclass[12:22])
colnames(neutral) <- "Neutral"
positive <- data.frame(vader_usairline_byclass[23:33])
colnames(positive) <- "Positive"

vader_usairline_byclass_df <- cbind(Measurements,negative,neutral,positive)
vader_usairline_byclass_df
```

```
##            Measurements  Negative    Neutral  Positive
## 1           Sensitivity 0.9013972 0.36355022 0.3155282
## 2           Specificity 0.4948373 0.84296378 0.9690558
## 3        Pos Pred Value 0.4981249 0.40315236 0.8792939
## 4        Neg Pred Value 0.9002222 0.81947680 0.6646255
## 5             Precision 0.4981249 0.40315236 0.8792939
## 6                Recall 0.9013972 0.36355022 0.3155282
## 7                    F1 0.6416596 0.38232852 0.4644072
## 8            Prevalence 0.3574231 0.22586859 0.4167083
## 9        Detection Rate 0.3221802 0.08211458 0.1314832
## 10 Detection Prevalence 0.6467860 0.20368124 0.1495327
## 11    Balanced Accuracy 0.6981173 0.60325700 0.6422920
```

```
rm(conf_mat_textblob)
rm(conf_mat_vader)
rm(Measurements)
rm(negative)
rm(neutral)
rm(positive)
```

```r
rm(textblob_usairline_byclass)
rm(textblob_usairline_overall)
rm(vader_usairline_byclass)
rm(vader_usairline_overall)
rm(Classes)
rm(Textblob)
rm(textblob_table)
rm(Vader)
rm(vader_table)
```

Apple Tweets Dataset

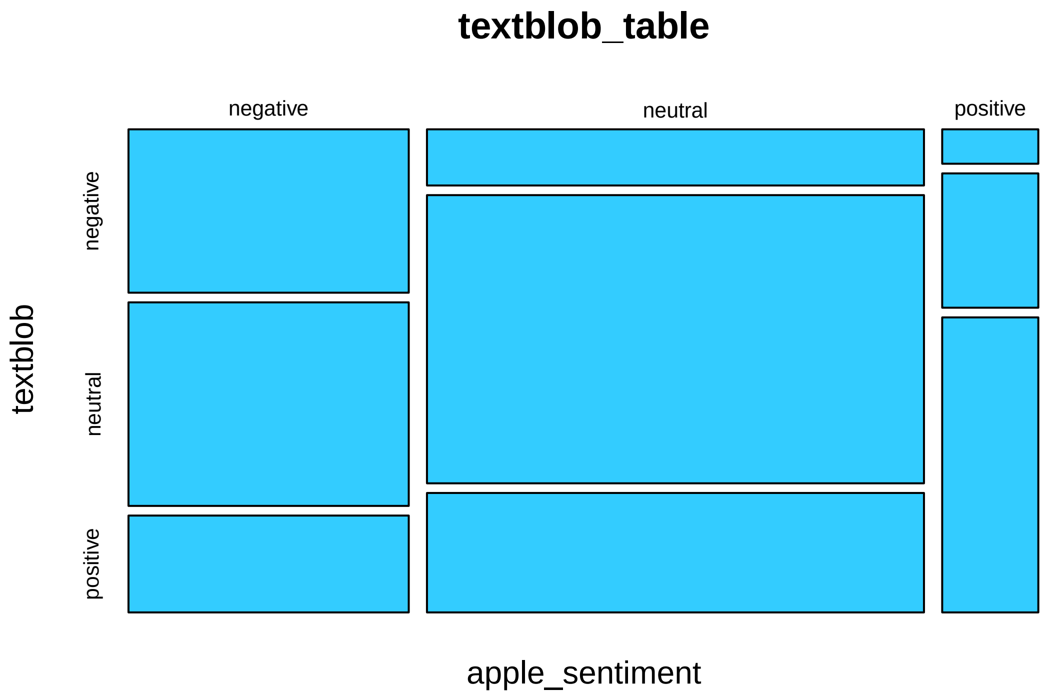
```r
data_final_apple <- data_final_apple %>%
  mutate(
    apple_class = case_when(
      sentiment == 1 ~ "negative",
      sentiment == 3 ~ "neutral",
      sentiment == 5 ~ "positive"
    ),
    textblob_class = case_when(
      textblob > 0.05 ~ "positive",
      textblob < -0.05 ~ "negative",
      textblob >= -0.05 & textblob <= 0.05 ~ "neutral"
    ),
    vader_class = case_when(
      vader > 0.05 ~ "positive",
      vader < -0.05 ~ "negative",
      vader >= -0.05 & vader <= 0.05 ~ "neutral"
  )
  )
```

```r
textblob_table <- table(apple_sentiment = data_final_apple$apple_class,
                        textblob = data_final_apple$textblob_class)

(conf_mat_textblob <- confusionMatrix(textblob_table))
```

```
## Confusion Matrix and Statistics
##
##                  textblob
## apple_sentiment negative neutral positive
##        negative      415     517      246
##        neutral       253    1298      538
##        positive       30     117      257
##
## Overall Statistics
##
##                Accuracy : 0.5366
##                  95% CI : (0.5203, 0.5529)
##     No Information Rate : 0.5263
##     P-Value [Acc > NIR] : 0.1075
##
```

```
##                    Kappa : 0.2383
##
##  Mcnemar's Test P-Value : <2e-16
##
## Statistics by Class:
##
##                      Class: negative Class: neutral Class: positive
## Sensitivity                   0.5946         0.6718         0.24688
## Specificity                   0.7434         0.5451         0.94411
## Pos Pred Value                0.3523         0.6213         0.63614
## Neg Pred Value                0.8865         0.5992         0.76002
## Prevalence                    0.1901         0.5263         0.28357
## Detection Rate                0.1130         0.3536         0.07001
## Detection Prevalence          0.3209         0.5691         0.11005
## Balanced Accuracy             0.6690         0.6085         0.59549
```

```
plot(textblob_table, color = "#33ccff")
```

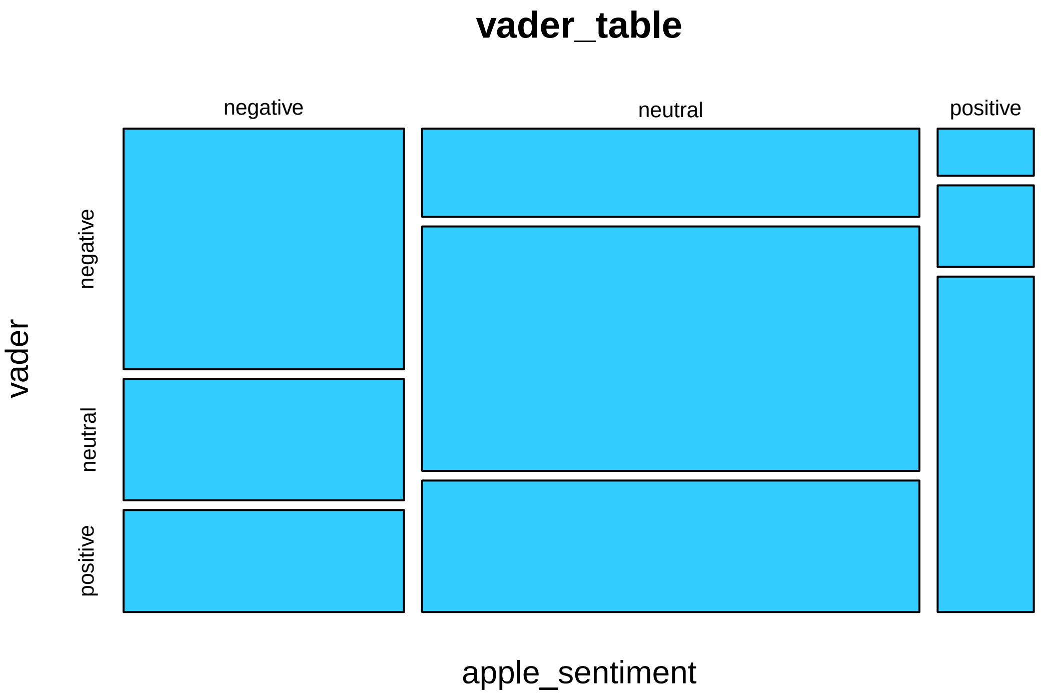


```
vader_table <- table(apple_sentiment =
                       data_final_apple$apple_class,
                     vader = data_final_apple$vader_class)

(conf_mat_vader <- confusionMatrix(vader_table))
```

```
## Confusion Matrix and Statistics
```

```
##
##                vader
## apple_sentiment negative neutral positive
##        negative      611     308      259
##        neutral       397    1100      592
##        positive       41      71      292
##
## Overall Statistics
##
##                Accuracy : 0.5456
##                  95% CI : (0.5294, 0.5618)
##     No Information Rate : 0.4029
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2953
##
##  Mcnemar’s Test P-Value : < 2.2e-16
##
## Statistics by Class:
##
##                      Class: negative Class: neutral Class: positive
## Sensitivity                   0.5825         0.7437         0.25547
## Specificity                   0.7838         0.5488         0.95570
## Pos Pred Value                0.5187         0.5266         0.72277
## Neg Pred Value                0.8243         0.7604         0.73952
## Prevalence                    0.2858         0.4029         0.31136
## Detection Rate                0.1664         0.2996         0.07954
## Detection Prevalence          0.3209         0.5691         0.11005
## Balanced Accuracy             0.6831         0.6463         0.60558
```

```
plot(vader_table, color = "#33ccff")
```



```
textblob_apple_overall <- as.matrix(conf_mat_textblob, what = "overall")
Measurements <- rownames(textblob_apple_overall)
Textblob <- textblob_apple_overall[1:7]
vader_apple_overall <- as.matrix(conf_mat_vader, what = "overall")
Vader <-  vader_apple_overall[1:7]

apple_overall_df <- data.frame(cbind(Measurements, Textblob, Vader))
apple_overall_df
```

```
##     Measurements             Textblob                 Vader
## 1       Accuracy     0.536638518114955     0.545627894306728
## 2          Kappa     0.238257709529203     0.295299084509863
## 3  AccuracyLower     0.520347694685346     0.529353895089821
## 4  AccuracyUpper     0.552870946574713     0.561829169974033
## 5   AccuracyNull     0.526287115227458     0.402887496594933
## 6 AccuracyPValue     0.107536138915015  3.60602634786736e-68
## 7  McnemarPValue 1.39099504088157e-114 3.48594413585111e-125
```

```
textblob_apple_byclass <- as.matrix(conf_mat_textblob, what = "classes")
Measurements <- data.frame(rownames(textblob_apple_byclass))
colnames(Measurements) <- "Measurements"
negative <- data.frame(textblob_apple_byclass[1:11])
colnames(negative) <- "Negative"
neutral <- data.frame(textblob_apple_byclass[12:22])
colnames(neutral) <- "Neutral"
```

```
positive <- data.frame(textblob_apple_byclass[23:33])
colnames(positive) <- "Positive"

textblob_apple_byclass_df <- cbind(Measurements,negative,neutral,positive)
textblob_apple_byclass_df
```

```
##            Measurements  Negative   Neutral   Positive
## 1           Sensitivity 0.5945559 0.6718427 0.24687800
## 2           Specificity 0.7433569 0.5451409 0.94410646
## 3        Pos Pred Value 0.3522920 0.6213499 0.63613861
## 4        Neg Pred Value 0.8864822 0.5992415 0.76002449
## 5             Precision 0.3522920 0.6213499 0.63613861
## 6                Recall 0.5945559 0.6718427 0.24687800
## 7                    F1 0.4424307 0.6456105 0.35570934
## 8            Prevalence 0.1901389 0.5262871 0.28357396
## 9        Detection Rate 0.1130482 0.3535821 0.07000817
## 10 Detection Prevalence 0.3208935 0.5690548 0.11005176
## 11    Balanced Accuracy 0.6689564 0.6084918 0.59549223
```

```
vader_apple_byclass <- as.matrix(conf_mat_vader, what = "classes")
Measurements <- data.frame(rownames(vader_apple_byclass))
colnames(Measurements) <- "Measurements"
Classes <- colnames(vader_apple_byclass)
negative <- data.frame(vader_apple_byclass[1:11])
colnames(negative) <- "Negative"
neutral <- data.frame(vader_apple_byclass[12:22])
colnames(neutral) <- "Neutral"
positive <- data.frame(vader_apple_byclass[23:33])
colnames(positive) <- "Positive"

vader_apple_byclass_df <- cbind(Measurements,negative,neutral,positive)
vader_apple_byclass_df
```

```
##            Measurements  Negative   Neutral   Positive
## 1           Sensitivity 0.5824595 0.7437458 0.25546807
## 2           Specificity 0.7837529 0.5488139 0.95569620
## 3        Pos Pred Value 0.5186757 0.5265677 0.72277228
## 4        Neg Pred Value 0.8243081 0.7604298 0.73951638
## 5             Precision 0.5186757 0.5265677 0.72277228
## 6                Recall 0.5824595 0.7437458 0.25546807
## 7                    F1 0.5487203 0.6165919 0.37750485
## 8            Prevalence 0.2857532 0.4028875 0.31135930
## 9        Detection Rate 0.1664397 0.2996459 0.07954236
## 10 Detection Prevalence 0.3208935 0.5690548 0.11005176
## 11    Balanced Accuracy 0.6831062 0.6462798 0.60558213
```

```
rm(conf_mat_textblob)
rm(conf_mat_vader)
rm(Measurements)
rm(negative)
rm(neutral)
rm(positive)
rm(textblob_apple_byclass)
rm(textblob_apple_overall)
rm(vader_apple_byclass)
rm(vader_apple_overall)
rm(Classes)
rm(Textblob)
rm(textblob_table)
rm(Vader)
rm(vader_table)
```

Exporting the Results as CSV files

```
write_csv(results_sent140, "data/sentiment140_classification_results.csv")
write_csv(usairline_overall_df, "data/usairline_overall_results.csv")
write_csv(textblob_usairline_byclass_df, "data/usairline_textblob_results.csv")
write_csv(vader_usairline_byclass_df, "data/usairline_vader_results.csv")

write_csv(apple_overall_df, "data/apple_overall_results.csv")
write_csv(textblob_apple_byclass_df, "data/apple_textblob_results.csv")
write_csv(vader_apple_byclass_df, "data/apple_vader_results.csv")
```

Climate Change Sentiment Analysis anf Topic Modelling

```
climate <- read_csv("climate.csv")

climate <- climate %>%
  select(Embedded_text, Timestamp)

text_cleaning <- function(x) {
  x %>%
    replace_non_ascii() %>%
    str_replace_all(pattern = "\\@.*? |\\@.*?[:punct:]", replacement = " ") %>%
    replace_url() %>%
    replace_hash() %>%
    replace_contraction() %>%
    str_replace_all("[:digit:]", " ") %>%
    str_trim() %>%
    str_squish()
}

climate <- climate %>%
  mutate(clean_tweet = text_cleaning(Embedded_text))

word_count_climate <- map_dbl(climate$clean_tweet,
                              function(x) str_split(x, " ") %>%
                                unlist() %>%
                                length()
)
```

```r
climate <- climate %>%
  filter(word_count_climate > 3)

climate <- climate %>%
  select(clean_tweet, Timestamp)

write_csv(climate, "data/climate_clean_tweet.csv")

climate <- read_csv("data/climate_vader_scores.csv")

climate <- climate %>%
  mutate(
    sentiment = case_when(
      vader > 0.05 ~ "positive",
      vader < -0.05 ~ "negative",
      vader >= -0.05 & vader <= 0.05 ~ "neutral"
    )
  )
```

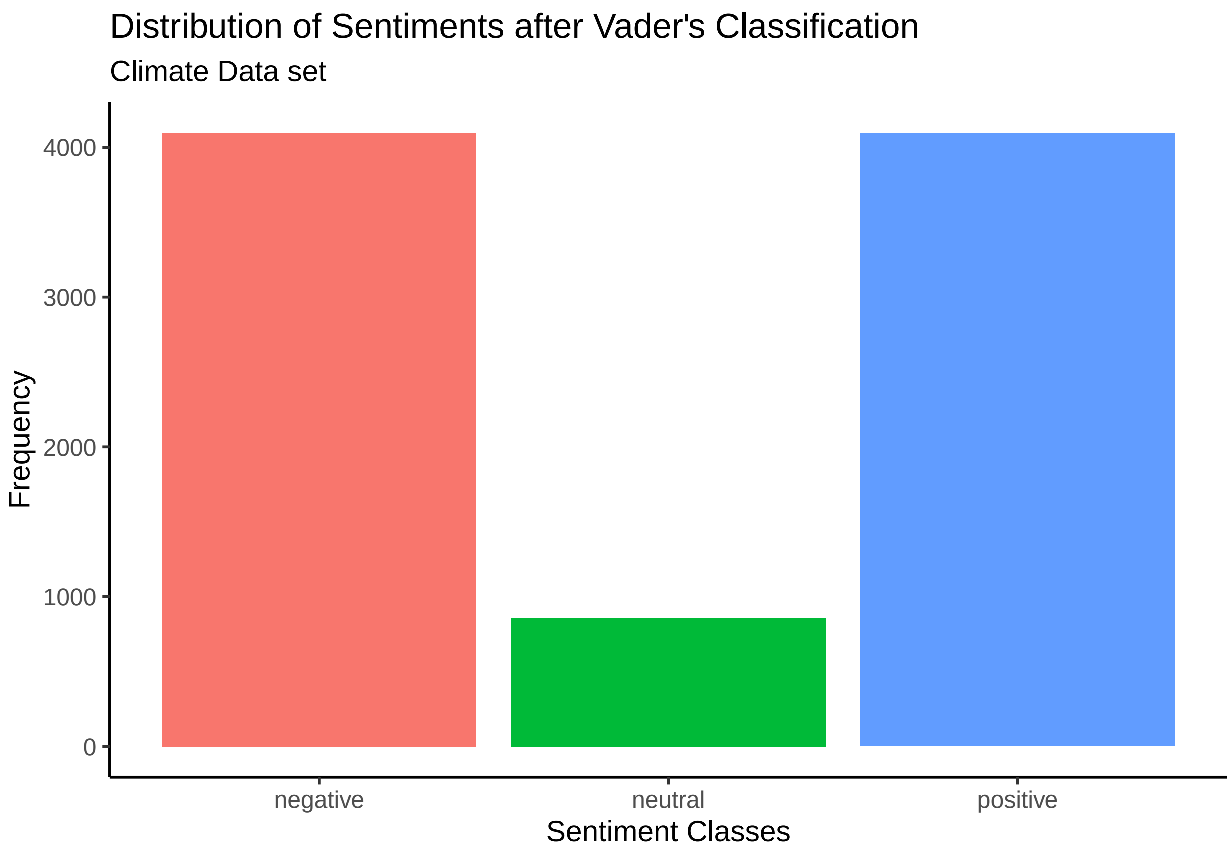
```r
table(climate$sentiment)
```

```
##
## negative  neutral positive
##     4097      859     4092
```

```r
prop.table(table(climate$sentiment))
```

```
##
##   negative    neutral   positive
## 0.45280725 0.09493811 0.45225464
```

```r
climate$sentiment <- as.factor(climate$sentiment)
ggplot(climate, aes(sentiment, fill = sentiment)) +
  geom_bar() +
  labs(
    x = "Sentiment Classes",
    y = "Frequency",
    title = "Distribution of Sentiments after Vader's Classification",
    subtitle = "Climate Data set",
  ) +
  theme_classic() +
  theme(legend.position="none")
```



```
# Overall Dataset

tweets_corpus <- Corpus(VectorSource(climate$clean_tweet))
tweets_corpus <- tm_map(tweets_corpus, content_transformer(tolower))
tweets_corpus <- tm_map(tweets_corpus, removeNumbers)
tweets_corpus <- tm_map(tweets_corpus, removePunctuation, preserve_intra_word_dashes = T)
tweets_corpus <- tm_map(tweets_corpus, removeWords, stopwords("english"))
tweets_corpus <- tm_map(tweets_corpus, stripWhitespace)

tdm <- TermDocumentMatrix(tweets_corpus)
freq <- sort(rowSums(as.matrix(tdm)), decreasing = TRUE)
freq <- freq[!names(freq) %in% c("climate", "change", "replying", "tweet",
                                 "will", "can", "one", "quote")]

png("graph/wordcloud_overall.png", width = 700, height = 700, res = 72)

wordcloud(words = names(freq), freq = freq, scale = c(5, 0.5),
          min.freq = 1, max.words = 200, random.order = FALSE,
          rot.per = 0.35, colors = brewer.pal(8, "Dark2"))

dev.off()
```

```
## pdf
##   2
```

```r
top_words_overall <- head(names(freq), 10)
top_words_overall
```

```
##  [1] "new"    "people" "now"    "world"  "just"   "action" "global" "like"  
##  [9] "need"   "years"
```

```r
# Positive Sentiments

positive <- climate %>%
  filter(sentiment == "positive")

tweets_corpus <- Corpus(VectorSource(positive$clean_tweet))
tweets_corpus <- tm_map(tweets_corpus, content_transformer(tolower))
tweets_corpus <- tm_map(tweets_corpus, removePunctuation)
tweets_corpus <- tm_map(tweets_corpus, removeWords, stopwords("english"))

tdm <- TermDocumentMatrix(tweets_corpus)
freq <- sort(rowSums(as.matrix(tdm)), decreasing = TRUE)
freq <- freq[!names(freq) %in% c("climate", "change","replying", "tweet",
                                 "will", "can", "one", "quote")]

png("graph/wordcloud_positive.png", width = 700, height = 700, res = 72)

wordcloud(words = names(freq), freq = freq, scale = c(5, 0.5),
          min.freq = 1, max.words = 200, random.order = FALSE,
          rot.per = 0.35, colors = brewer.pal(8, "Dark2"))

dev.off()
```

```
## pdf 
##   2
```

```r
top_words_positive <- head(names(freq), 10)
top_words_positive
```

```
##  [1] "new"    "people" "energy" "like"   "action" "just"   "help"   "now"   
##  [9] "global" "world"
```

```r
# Negative Sentiments

negative <- climate %>%
  filter(sentiment == "negative")

tweets_corpus <- Corpus(VectorSource(negative$clean_tweet))
tweets_corpus <- tm_map(tweets_corpus, content_transformer(tolower))
tweets_corpus <- tm_map(tweets_corpus, removePunctuation)
tweets_corpus <- tm_map(tweets_corpus, removeWords, stopwords("english"))

tdm <- TermDocumentMatrix(tweets_corpus)
freq <- sort(rowSums(as.matrix(tdm)), decreasing = TRUE)
freq <- freq[!names(freq) %in% c("climate", "change", "replying", "tweet",
                                 "will", "can", "one", "quote")]

png("graph/wordcloud_negative.png", width = 700, height = 700, res = 72)

wordcloud(words = names(freq), freq = freq, scale = c(5, 0.5),
          min.freq = 1, max.words = 200, random.order = FALSE,
          rot.per = 0.35, colors = brewer.pal(8, "Dark2"))

dev.off()
```

```
## pdf
##   2
```

```
top_words_negative <- head(names(freq), 10)
top_words_negative
```

```
##  [1] "people" "new"    "now"    "world"  "crisis" "global" "just"   "years"
##  [9] "need"   "action"
```

```
# Neutral Sentiments

neutral <- climate %>%
  filter(sentiment == "neutral")

tweets_corpus <- Corpus(VectorSource(neutral$clean_tweet))
tweets_corpus <- tm_map(tweets_corpus, content_transformer(tolower))
tweets_corpus <- tm_map(tweets_corpus, removePunctuation)
tweets_corpus <- tm_map(tweets_corpus, removeWords, stopwords("english"))

tdm <- TermDocumentMatrix(tweets_corpus)
freq <- sort(rowSums(as.matrix(tdm)), decreasing = TRUE)
freq <- freq[!names(freq) %in% c("climate", "change", "replying", "tweet",
                                 "will", "can", "one", "quote")]

png("graph/wordcloud_neutral.png", width = 700, height = 700, res = 72)

wordcloud(words = names(freq), freq = freq, scale = c(5, 0.5),
          min.freq = 1, max.words = 200, random.order = FALSE,
          rot.per = 0.35, colors = brewer.pal(8, "Dark2"))

dev.off()
```

```
## pdf
##   2
```

```
top_words_neutral <- head(names(freq), 10)
top_words_neutral
```

```
##  [1] "new"    "world"  "people" "now"    "years"  "action" "just"   "water"
##  [9] "global" "time"
```

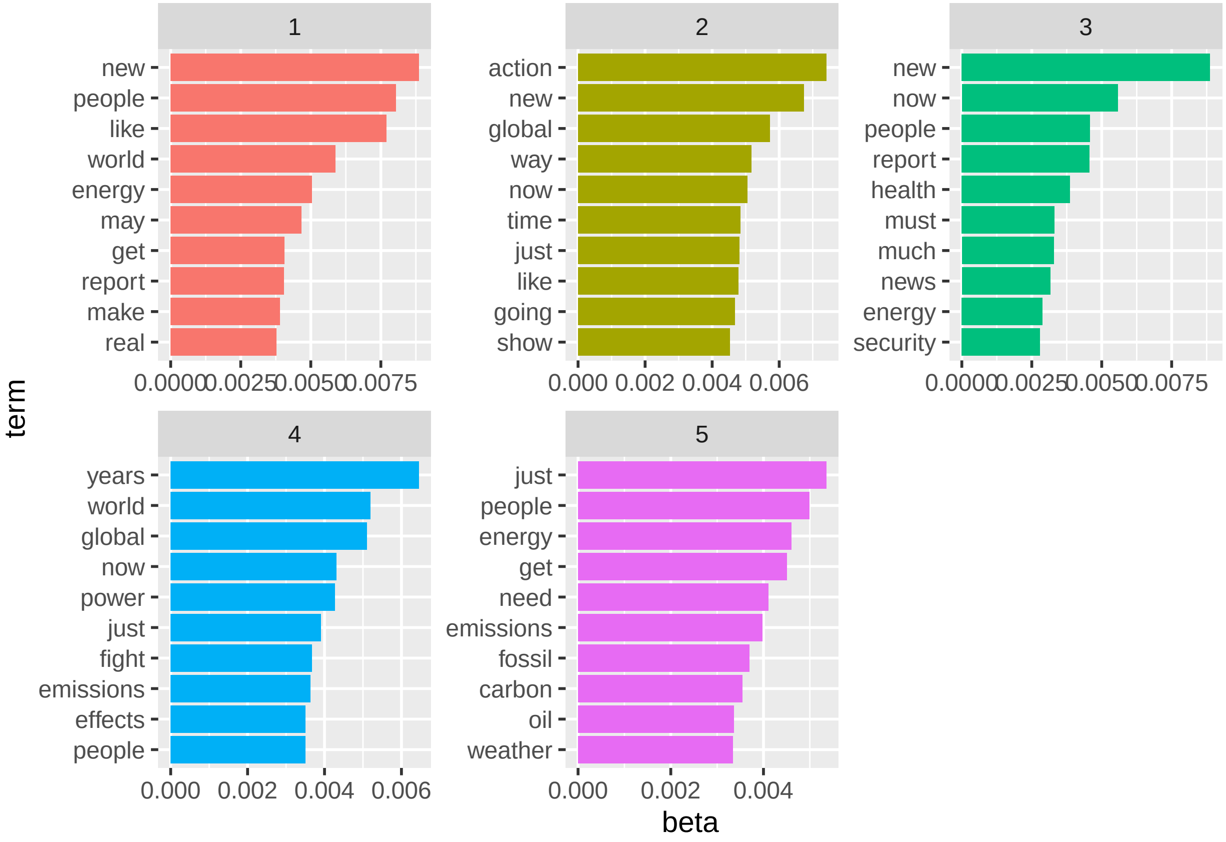
```
bad_words <- c("climate", "change", "replying", "tweet", "will", "can",
               "quote", "one")

text_cleaning <- function(x) {
  x %>%
    str_to_lower() %>%
    replace_non_ascii() %>%
    str_replace_all(pattern = "\\@.*? |\\@.*?[:punct:]", replacement = " ") %>%
    replace_url() %>%
    replace_hash() %>%
    replace_contraction() %>%
    str_replace_all("[:digit:]", " ") %>%
    str_replace_all(paste(bad_words, collapse = "|"), " ") %>%
    str_trim() %>%
    str_squish()
}

climate <- climate %>%
  mutate(
    clean_tweet = text_cleaning(clean_tweet)
  )

word_count_climate <- map_dbl(climate$clean_tweet,
                              function(x) str_split(x, " ") %>%
                                unlist() %>%
                                length()
)

climate <- climate %>%
  filter(word_count_climate > 10)

tweets_corpus <- Corpus(VectorSource(climate$clean_tweet))
tweets_corpus <- tm_map(tweets_corpus, content_transformer(tolower))
tweets_corpus <- tm_map(tweets_corpus, removeNumbers)
tweets_corpus <- tm_map(tweets_corpus, removePunctuation,
                        preserve_intra_word_dashes = T)
tweets_corpus <- tm_map(tweets_corpus, removeWords, stopwords("english"))
tweets_corpus <- tm_map(tweets_corpus, stripWhitespace)

dtm <- DocumentTermMatrix(tweets_corpus)


model_lda <- LDA(dtm, k = 5, control = list(seed = 1234))
model_lda
```

```
## A LDA_VEM topic model with 5 topics.
```

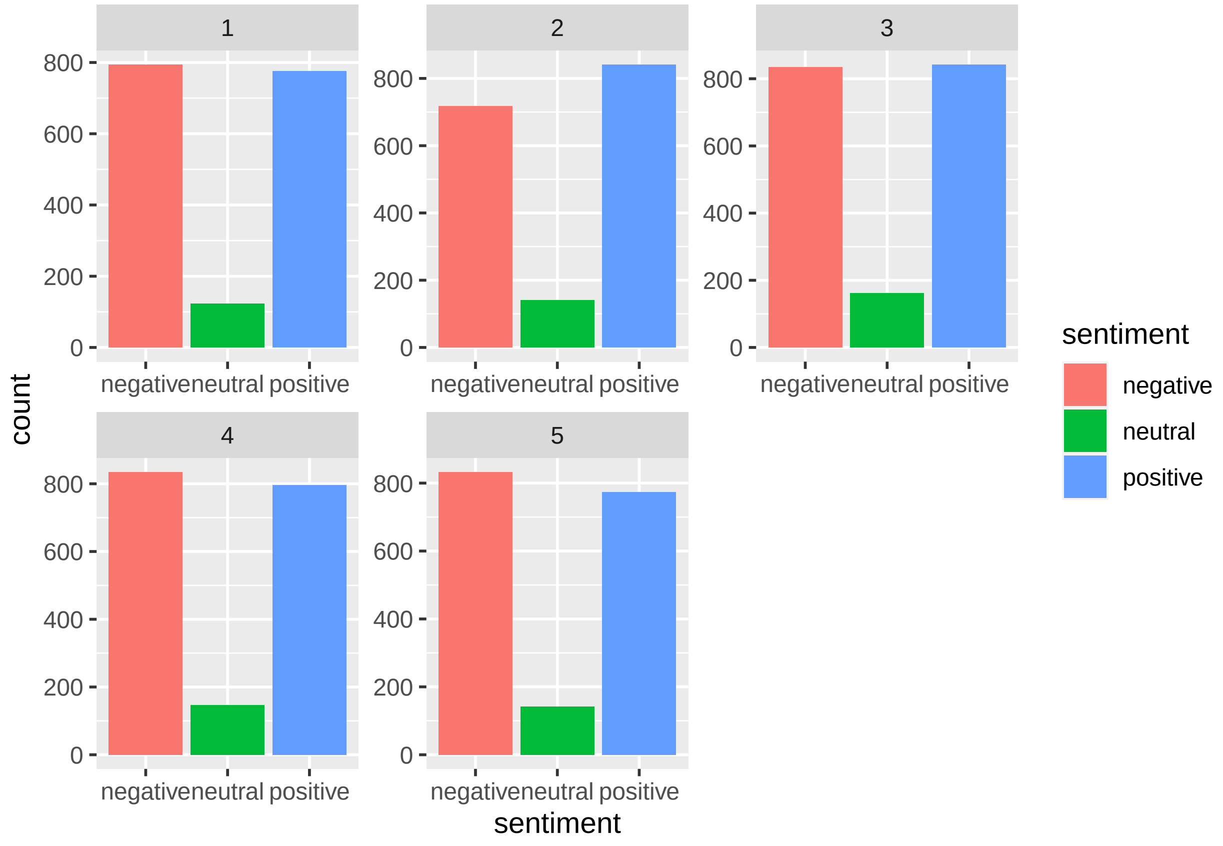
```
beta_topics <- tidy(model_lda, matrix = "beta")
beta_topics
```

```
## # A tibble: 119,445 x 3
##    topic term         beta
##    <int> <chr>      <dbl>
##  1     1 control 0.000633
##  2     2 control 0.000561
##  3     3 control 0.000629
##  4     4 control 0.000478
##  5     5 control 0.000738
##  6     1 ever    0.000480
##  7     2 ever    0.000451
##  8     3 ever    0.000985
##  9     4 ever    0.00109
## 10     5 ever    0.000463
## # ... with 119,435 more rows
```

```
beta_top_terms <- beta_topics %>%
  group_by(topic) %>%
  slice_max(beta, n = 10) %>%
  ungroup() %>%
  arrange(topic, -beta)

beta_top_terms %>%
  mutate(term = reorder_within(term, beta, topic)) %>%
  ggplot(aes(beta, term, fill = factor(topic))) +
  geom_col(show.legend = F) +
  facet_wrap(~ topic, scales = "free") +
  scale_y_reordered()
```

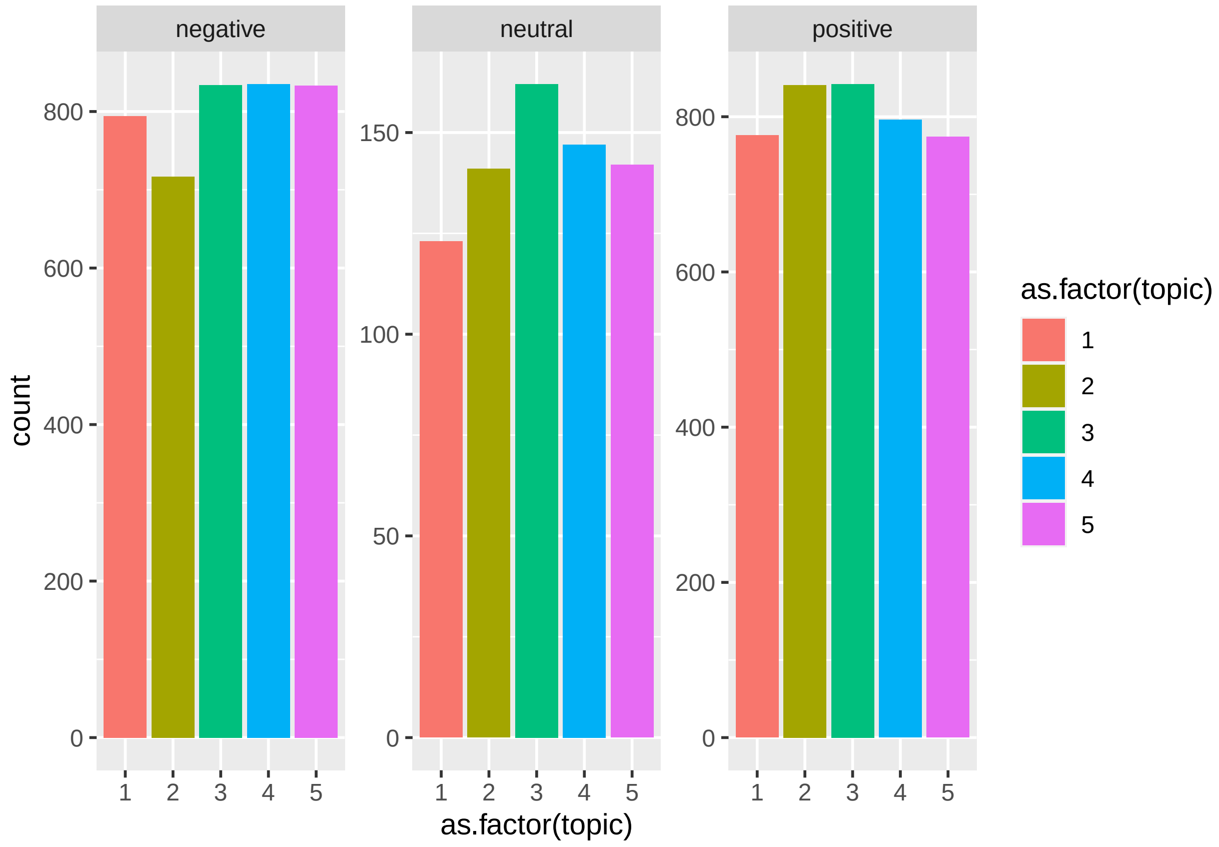


```
topic_probs <- tidy(model_lda, matrix = "gamma")

df_max_gamma <- topic_probs %>%
  group_by(document) %>%
  slice(which.max(gamma))
df_max_gamma$document <- as.numeric(df_max_gamma$document)

climate <- climate %>%
  mutate(document = row_number())

climate <- climate %>%
  left_join(df_max_gamma, by = "document")

climate <- climate %>%
  select(document, clean_tweet, topic, vader, Timestamp, sentiment)

ggplot(climate, aes(sentiment, fill = sentiment)) +
  geom_bar() +
  facet_wrap(~topic, scales = "free")
```



```
sentiment_by_topic <- climate %>%
  group_by(topic, sentiment) %>%
  summarize(count = n()) %>%
  group_by(topic) %>%
  mutate(prop = count / sum(count))
```

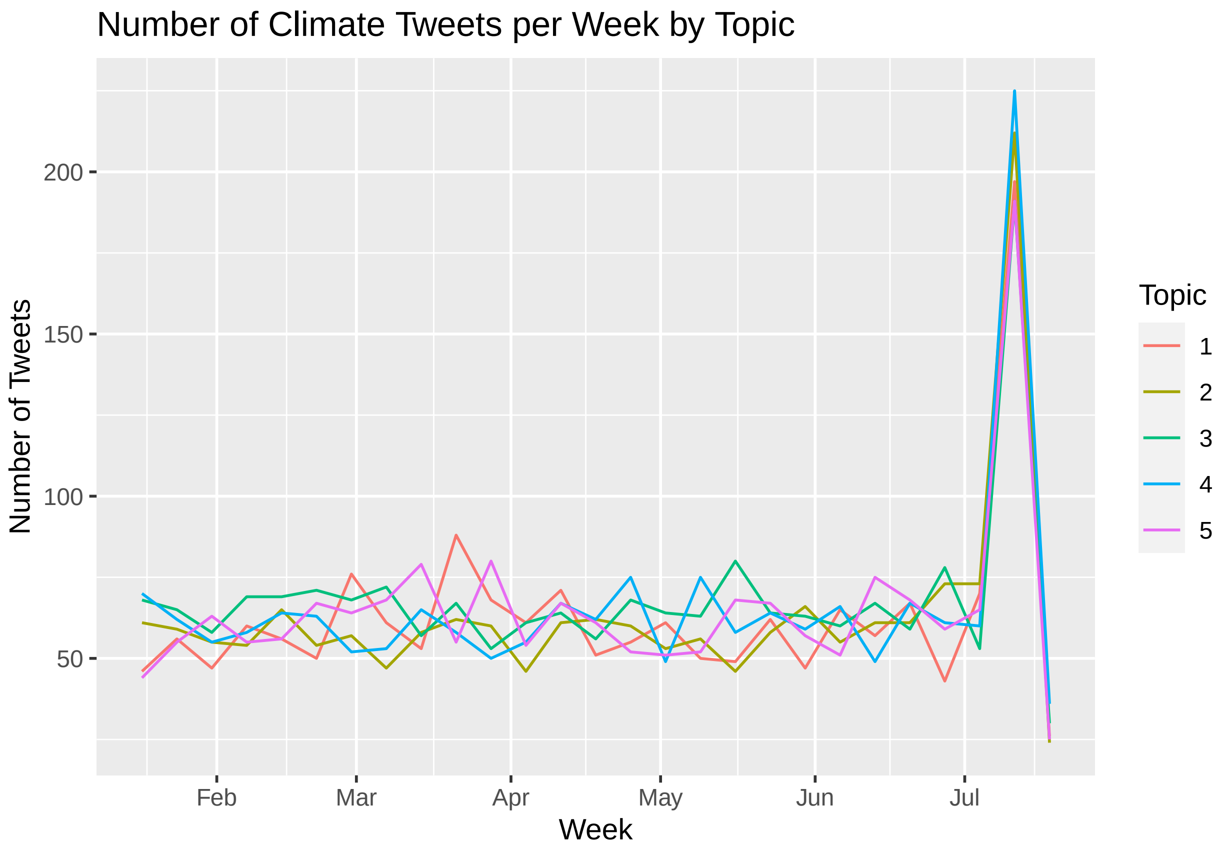
```
ggplot(climate, aes(as.factor(topic), fill = as.factor(topic))) +
  geom_bar() +
  facet_wrap(~sentiment, scales = "free")
```



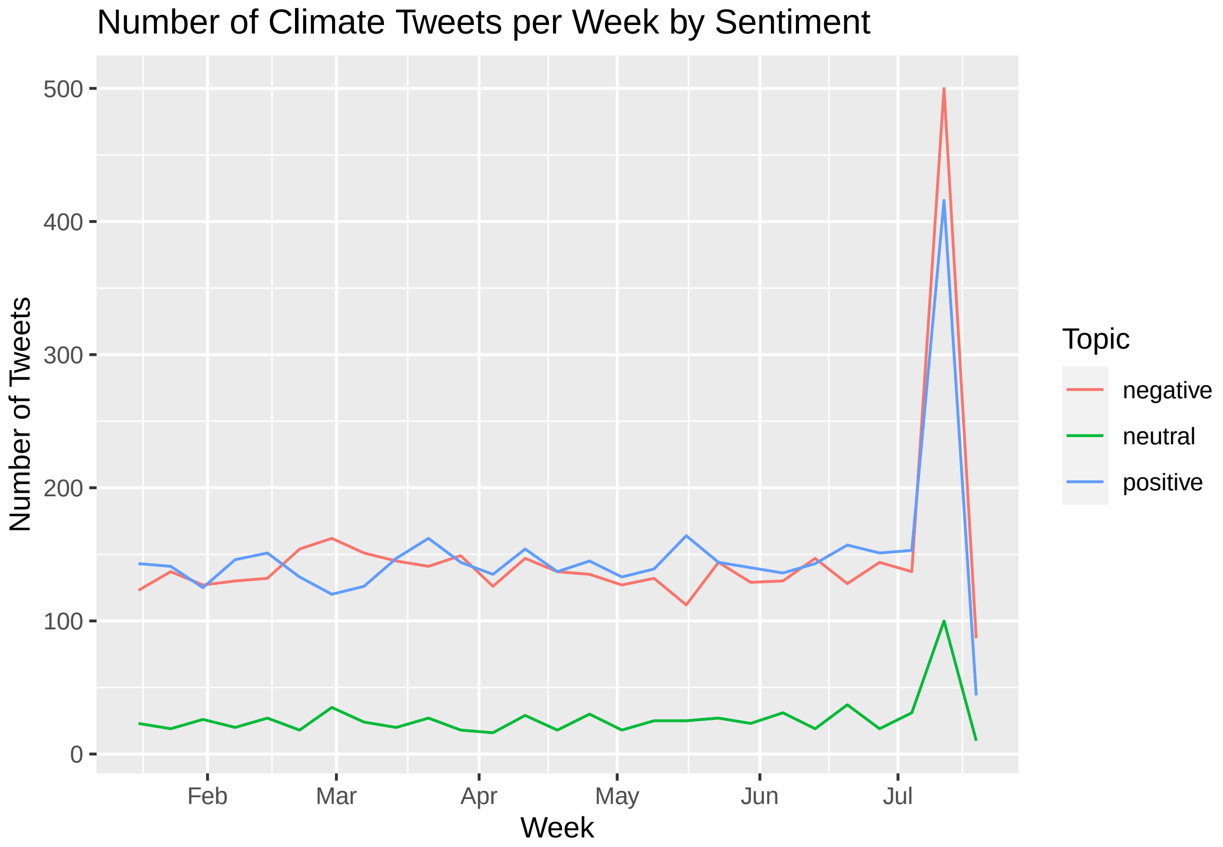
```
topic_by_sentiment <- climate %>%
  group_by(sentiment, topic) %>%
  summarize(count = n()) %>%
  group_by(sentiment) %>%
  mutate(prop = count / sum(count))
```

Temporal Analysis

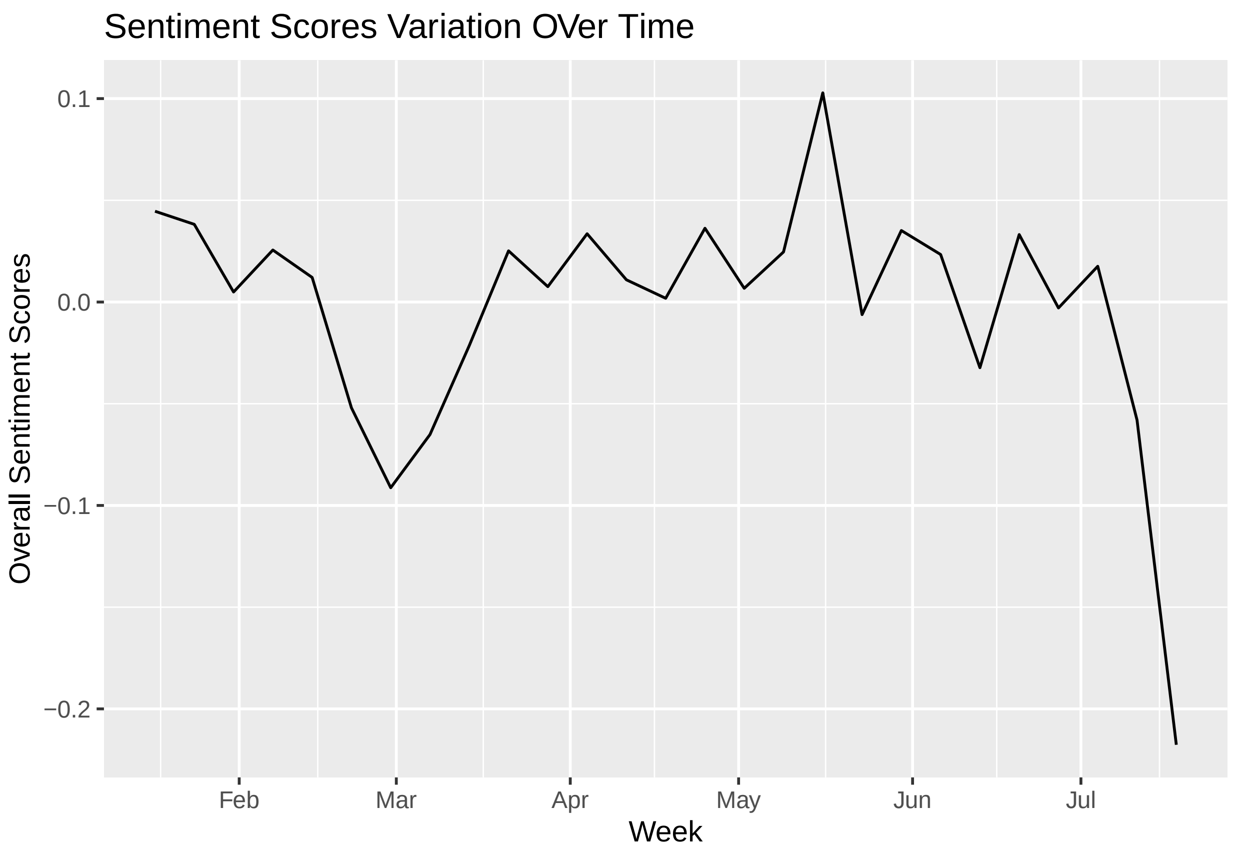
```
climate$Timestamp <- as.POSIXct(climate$Timestamp,
                                format = "%Y-%m-%dT%H:%M:%SZ")

climate$week <- as.Date(cut(climate$Timestamp, breaks = "week"))

tweets_per_week_topic <- climate %>%
  group_by(week, topic) %>%
  summarize(num_tweets = n())

ggplot(data = tweets_per_week_topic, aes(x = week, y = num_tweets,
                                         color = factor(topic))) +
  geom_line() +
  labs(title = "Number of Climate Tweets per Week by Topic",
       x = "Week", y = "Number of Tweets", color = "Topic")
```



```
tweets_per_week_sentiment <- climate %>%
  group_by(week, sentiment) %>%
  summarize(num_tweets = n())

ggplot(data = tweets_per_week_sentiment, aes(x = week, y = num_tweets,
                                             color = factor(sentiment))) +
  geom_line() +
  labs(title = "Number of Climate Tweets per Week by Sentiment",
       x = "Week", y = "Number of Tweets", color = "Topic")
```



```
tweets_per_week_sentiment_scores <- climate %>%
  group_by(week) %>%
  summarize(overall_sentiment_scores = mean(vader))

ggplot(data = tweets_per_week_sentiment_scores,
       aes(x = week, y = overall_sentiment_scores)) +
  geom_line() +
  labs(title = "Sentiment Scores Variation OVer Time", x = "Week",
       y = "Overall Sentiment Scores")
```


Sentiment Scores Variation OVer Time
Overall Sentiment Scores
0.1
0.0
−0.1
−0.2
Feb
Mar
Apr
May
Jun
Jul
Week

THE END