R for Data Science Project 1: Stress Analysis On Reddit

Melanie McCord

Introduction

Background

Stress is a common aspect of modern lives that can cause negative health outcomes, such as anxiety, depression, insomnia, and a weakened immune system. If we can predict stress on social media, we can help understand the extent of the problem and perhaps gain insights on how to address it. For this specific project, the focus is on Reddit. Reddit is a social media platform where users post questions and can get advice. Importantly, Reddit data is concentrated on specific subreddits, divided by topic. This makes it easier to filter by specific topic and analyze the data from each specific topic. If we mine from stress-related subreddits, we can get a more detailed overview of the problem.

Data Structure and Source

This dataset was collected by mining all posts from 10 subreddits between January 1, 2017 and November 19, 2018. Then, the data was annotated by human annotators as being either "stressed", "non-stressed", or "unclear". The posts that were unclear as to whether or not they were stressed were discarded. For each post, the features are the text, the label, and some syntactic features, lexical features, and social media features, as well as the sentiment (how overly positive or negative the posts are).

For more information about the dataset, see this paper: Dreaddit: A Reddit Dataset for Stress Analysis On Social Media.

Major Variables I Will Be Exploring

- Label: stressed (1) or non-stressed (0)
- Sentiment: the intensity of positive or negative a particular post is (-1 being completely negative, 0 being completely neutral, and 1 being completely negative)
- Subreddit: the source of the particular post
- The words that appear in each post (from the text)

Future work may include exploring the other variables, which include social media features, lexical features, and syntactic features.

```
train_data <- read_csv("reddit_stress_data/dreaddit-train.csv", show_col_types = FALSE)
test_data <- read_csv("reddit_stress_data/dreaddit-test.csv", show_col_types = FALSE)
reddit_stress_data <- add_row(train_data, test_data)
```

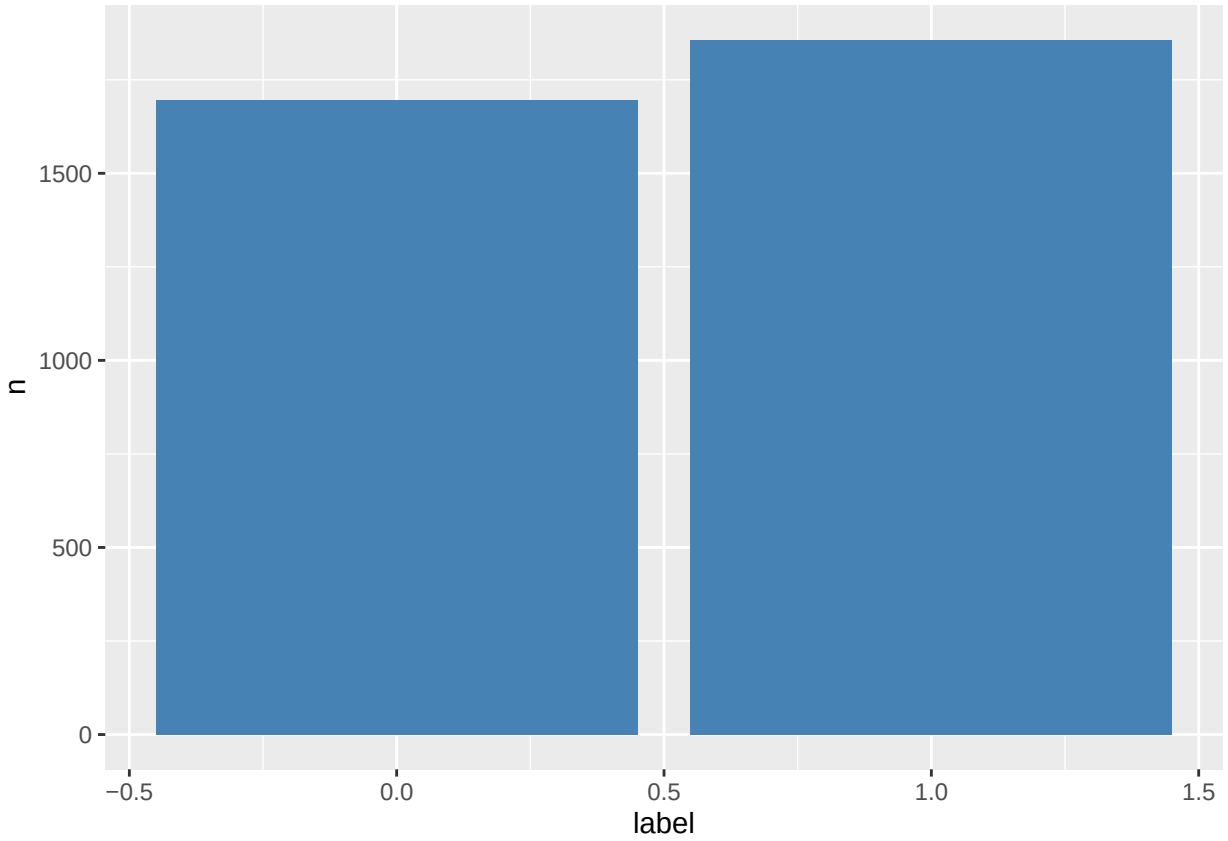
Research Questions

The topics I am exploring in this paper are: How does the stress label differ among subreddits? How can we predict stress given words in the data and sentiment? Is there an association between stress-related data and subreddits?

Distribution of Major Features Among the Dataset

Dataset Label Distribution

```
label_counts <- reddit_stress_data %>%
  group_by(label) %>%
  count()
ggplot(label_counts, aes(x = label, y = n)) + geom_col(fill = "steelblue")
```



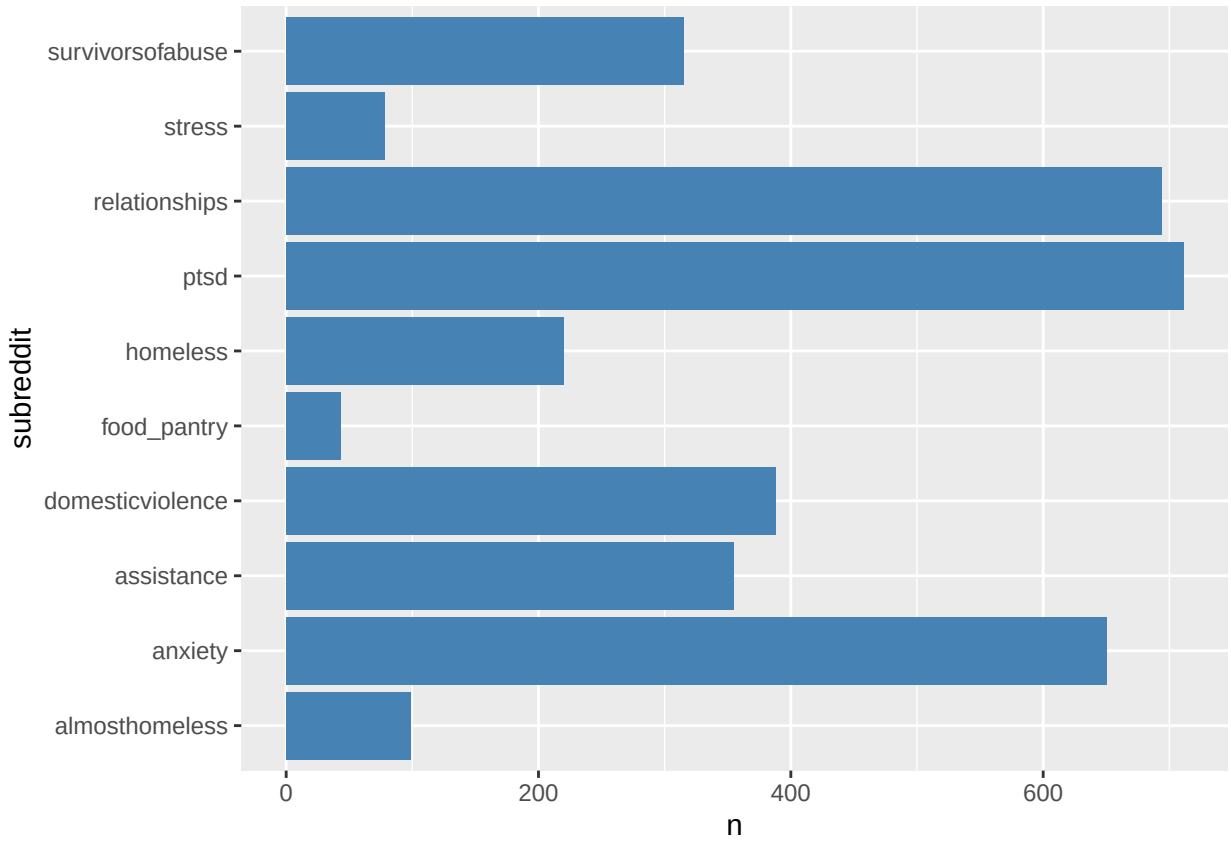

```
prop.table(table(reddit_stress_data$label))
```

```
##
##         0         1
## 0.4773431 0.5226569
```

The data is slightly biased in favor of stress-related posts, but not overly so (52% stressed versus 48% non-stressed).

Distribution of Data By Subreddit

```
reddit_stress_data %>%
  group_by(subreddit) %>%
  count() %>%
  arrange(desc(n)) %>%
  ggplot(aes(y = subreddit, x = n)) + geom_col(fill = "steelblue")
```



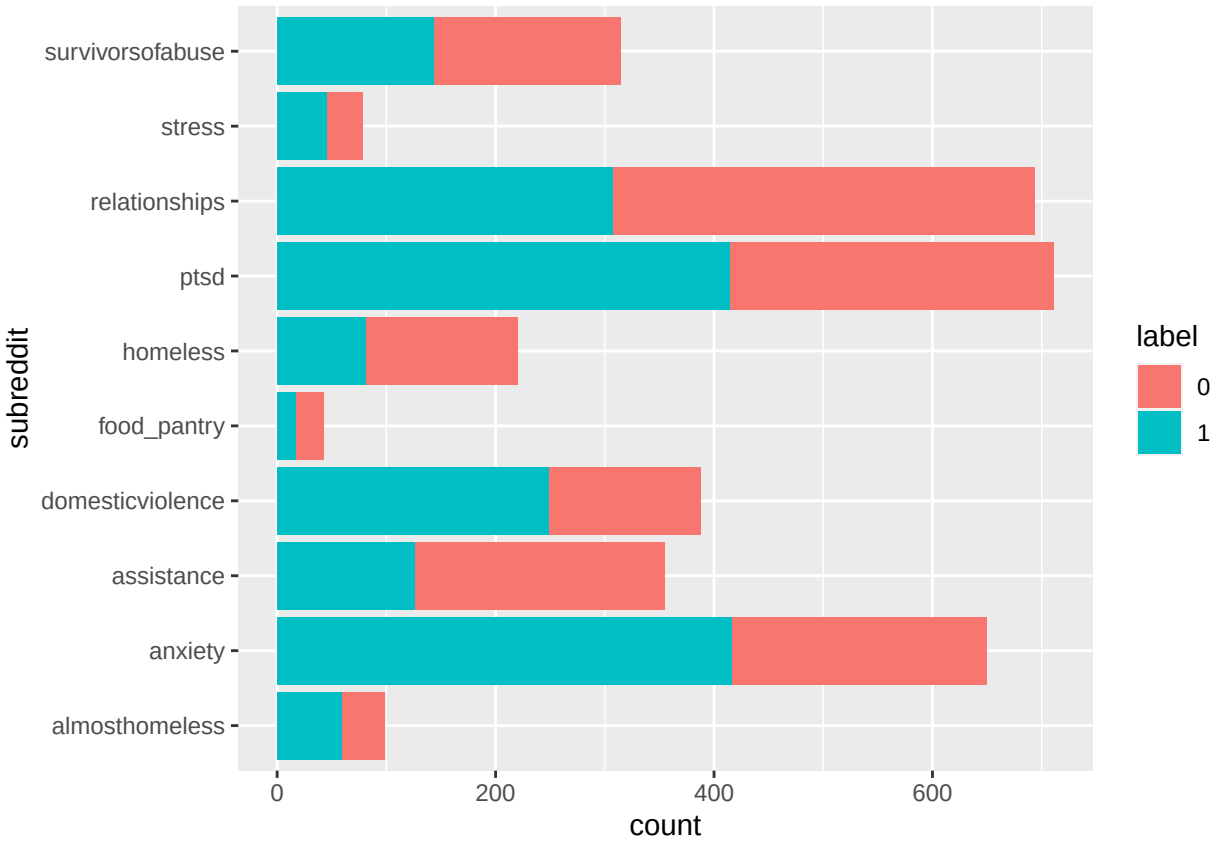

```
prop.table(table(reddit_stress_data$subreddit))
```

```
##
##   almosthomeless          anxiety       assistance domesticviolence
##       0.02786378       0.18294399       0.09991556       0.10920349
##      food_pantry         homeless             ptsd    relationships
##       0.01210245       0.06191950       0.20011258       0.19532789
##           stress survivorsofabuse
##       0.02195328       0.08865747
```

The dataset distribution is heavily imbalanced. 20% of the posts sampled are from r/ptsd, whereas only 1% of the posts sampled are from r/food_pantry. This may be because certain subreddits are less active than other subreddits, or certain subreddits are harder to label as being stressed or not stressed. Regardless, since the data is heavily biased toward certain subreddits, any classification model we will fit to this dataset is likely going to be able to predict stress labels from r/relationships significantly better than labels from r/food_pantry.

Distribution of Posts By Label and Subreddit

```
reddit_stress_data %>%
  mutate(label = as.factor(label)) %>%
  ggplot(aes(y=subreddit)) + geom_bar(aes(fill = label), position="stack")
```



Additionally, for each particular subreddit, there are uneven distributions in the percentage of posts that are stress or not stress related. Assistance appears to be imbalanced towards non-stress related posts, whereas domesticviolence is biased towards stress-related posts. One possible explanation could be that although people are seeking advice from these subreddits, assistance may simply be asking what resources are available, whereas domesticviolence or anxiety may be more focused on stressful incidents or needing support.

Methodology

Data Preprocessing

Another detail about the statistical analysis is that since I was primarily focusing on text mining, there were a number of things that I had to consider that are unique to text data. Text data is inherently messy. Firstly, computers can only process numbers, meaning that any text data needs some way to be converted to numbers. There are a lot of different ways to do this, but the model I chose to focus on is a bag of words model, which counts the number of times a particular word appears and adds each word as a feature. However, in doing so there were some things I needed to clean. Stopwords, such as "and", "so", and "to", appear frequently but don't matter much when understanding the meaning of text. Additionally, punctuation, and capitalization causes issues. A computer considers "Don't", "don't," and "DONT" to be separate words. As part of my data wrangling, I needed to deal with this. Also, some words only appear in one post. For example, one post may ask about resources available in Louisiana, but no other post mentions Louisiana. We need a way to deal with these issues. It's possible that rare words may indicate some information about the text, so we can't just remove rare words. Additionally, the rare words cause the number of words that appear in the post to be significantly higher.

```
word_counts = CleanText(reddit_stress_data)
length(unique(word_counts$word))
```

```
## [1] 12059
```

Without removing any of the rare words, only removing the stopwords, there are 12059 words in the text. Let's look at the distribution of the count of these words.

```
word_counts <- word_counts %>%
  count(word)
favstats(word_counts$n)
```

```
##  min Q1 median Q3  max     mean       sd     n missing
##    1  1      2  5 2235 8.049092 41.04796 12059       0
```

```
nrow(reddit_stress_data)
```

```
## [1] 3553
```

Clearly, there is a large disparity among the most common words and the least common words. The median distribution is 2, and the third quartile is 5 words. There are 3553 posts, and 5 words is still quite rare. If we adjust to remove the number of words, we will be significantly benefitted from this.
Let's set the definition of rare words to be 15 and compare the result.

```
bag_of_words = CreateBagOfWordsWithUnknowns(reddit_stress_data, 15)
ncol(bag_of_words)
```

```
## [1] 1723
```

Since we have eliminated a large percentage of words, let's look at the overall distribution of the rare words and compare it to the words that did not get removed.

```
rare_defn = 15
rare_words <- CleanText(reddit_stress_data) %>%
  count(word) %>%
  filter(n < rare_defn)
favstats(rare_words$n)
```

```
##  min Q1 median Q3 max     mean       sd     n missing
##    1  1      1  3  14 2.738706 2.791588 10869       0
```

```
non_rare_words <- CleanText(reddit_stress_data) %>%
  count(word) %>%
  filter(n >= rare_defn)
favstats(non_rare_words$n)
```

```
##  min Q1 median Q3  max    mean       sd    n missing
##   15 20     30 51 2235 56.5521 120.0162 1190       0
```

Now, the min is significantly larger. There are significantly more rare words than common words. But among the non-rare words, there is significantly more variation and you can see the overall spread of words better.

Another related issue I ran into was dealing with the scenario where "id" and "subreddit" both appeared in the dataset of words and in the original dataset. I dealt with this by adding "text_" to each word after the removal of stopwords and punctuation.

I also joined the training and test data in order to reduce potential bias between the data selected as training and the data selected as test data and added the bag of words column to the original data.

Data Exploration

I was interested in exploring the differences between common words among each of the subreddits, and the differences between the most common words by label and by subreddit. A future data exploration may include exploring the differences among the most common words by both label and subreddit, but since this visualization was difficult to read, I chose not to include it.

Additionally, I chose to explore the distribution of sentiment by subreddit, label, and both. This can shed a light on important patterns between the data and the subreddit column.

Statistical Modeling and Analysis

For the statistical analysis portion, I am doing a chi-square test of statistical significance of the differences among label and subreddit, and an analysis of variance between subreddit and sentiment and label and sentiment.

I will also run a decision tree model on the dataset in order to see what the strongest predictors of label are from the words and sentiment and compare its performance on stressed versus non-stressed data.

Results

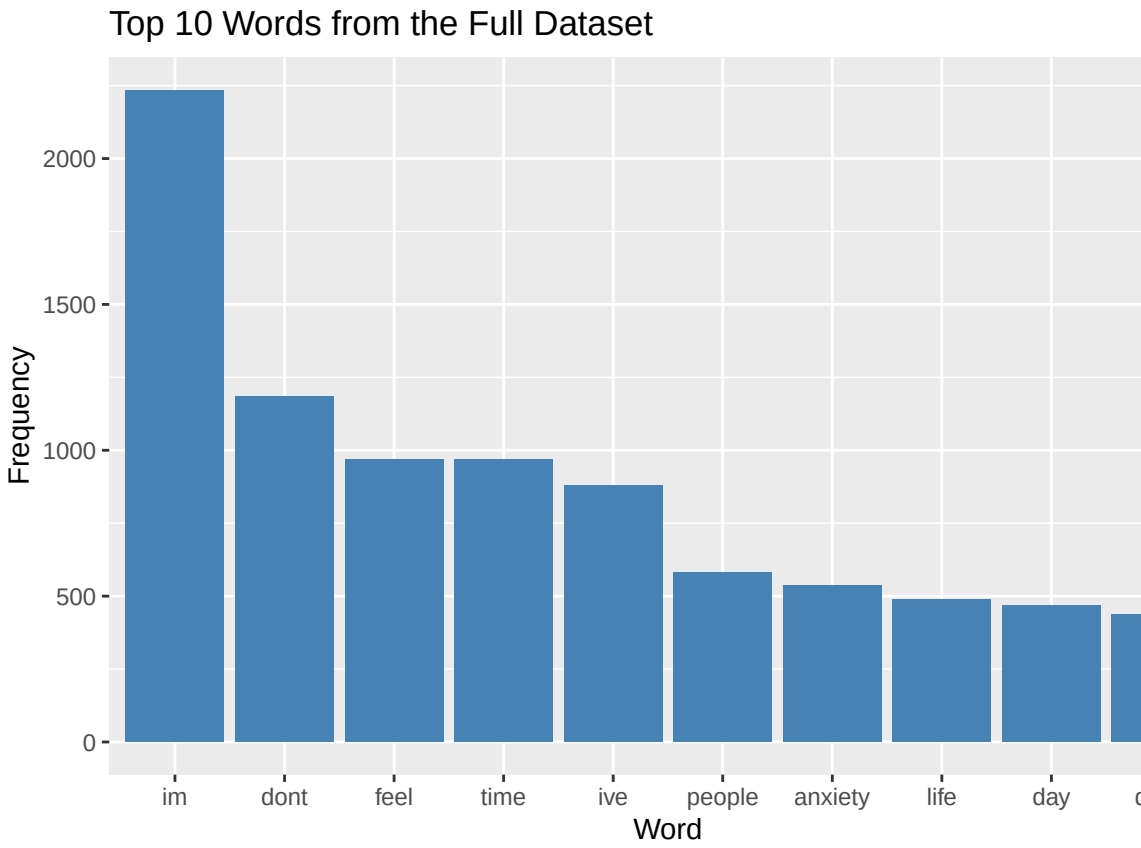
Data Exploration

Top 10 Words

```
words_tokenized <- reddit_stress_data %>%
  select(text, id) %>%
    unnest_tokens(word, text) %>%
    mutate(word = gsub('[[:punct:]]+','', word)) %>%
    mutate(word = gsub('\\<[[:digit:]]+\\>', '%d%', word)) %>%
    anti_join(stop_words, by = c("word" = "word")) %>%
  filter(word != "%d%")
GetTopNMostCommonWords <- function(df, num) {
  top_word_counts <- df %>%
    count(word) %>%
    arrange(desc(n))
  return (head(top_word_counts, num))
}
num <- 10
```

Since I'm interested in exploring the Reddit Stress dataset by counting the words that appear, my first step is to do some visualizations of the top 10 most common words.

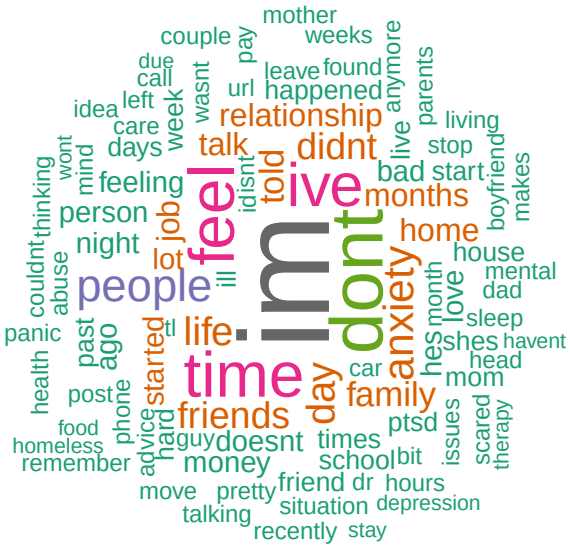
```
top_10_full_data <- GetTopNMostCommonWords(words_tokenized, num)
ggplot(top_10_full_data, aes(x = reorder(word, desc(n)), y = n)) + geom_col(fill = "steelblue") + labs(
```



Top 15 Words Overall

For the whole dataset, the most common words appear to be neutral, possibly linked to the diversity of posts reflected. Anxiety feel and people appear frequently, which is understandable considering the focus is stress-related subreddits. Another way to visualize this is to use a wordcloud.

```
words_tokenized_counts <- words_tokenized %>%
  filter(word != "text_%d%") %>%
  count(word)
wordcloud(words = words_tokenized_counts$word, freq = words_tokenized_counts$n, min.freq = 1, max.words
```

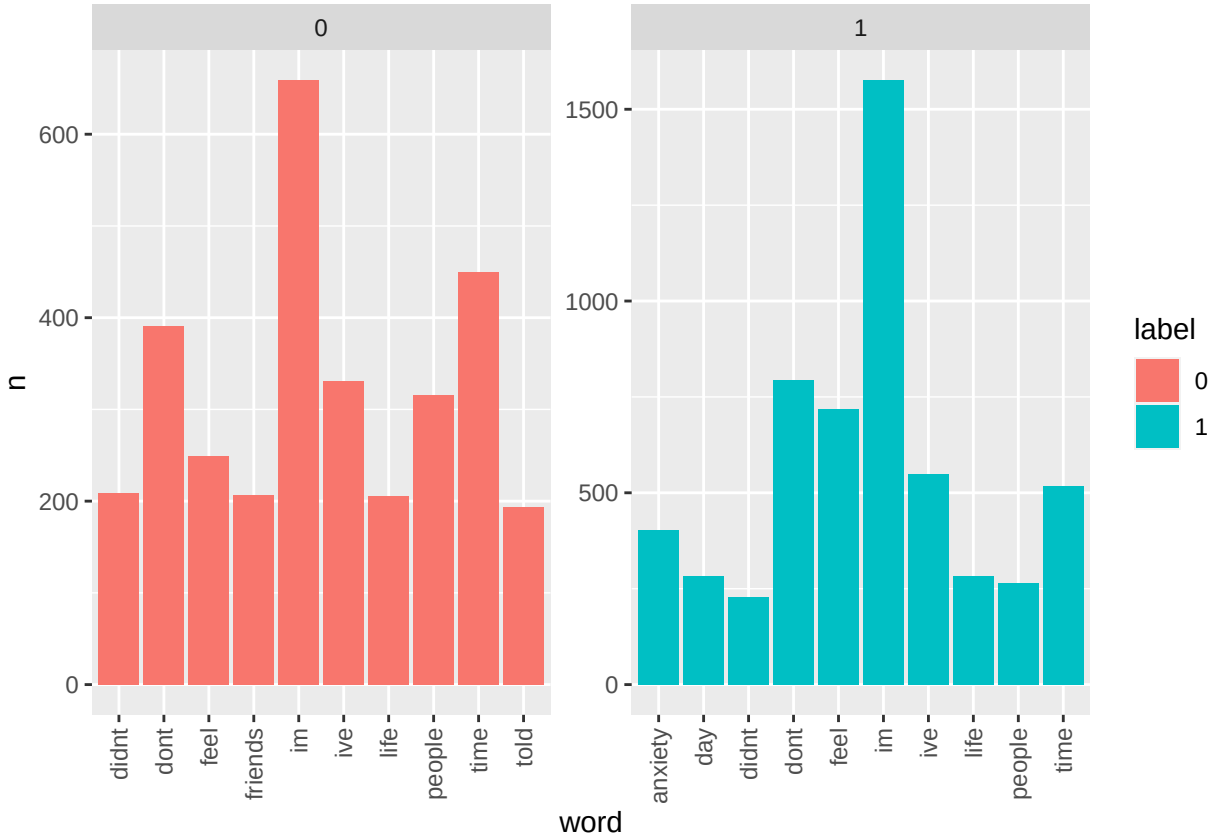


```
words_tokenized_by_label_counts <- reddit_stress_data %>%
  select(c("id", "text", "label", "subreddit")) %>%
  unnest_tokens(word, text) %>%
  mutate(word = gsub('[[:punct:]]+','', word)) %>%
  mutate(word = gsub('\\<[[:digit:]]+\\>', '%d%', word)) %>%
  anti_join(stop_words) %>%
  group_by(label) %>%
  count(word) %>%
  filter(word != "%d%") %>%
  arrange(label, desc(n))
```

Top 10 Words By Label

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

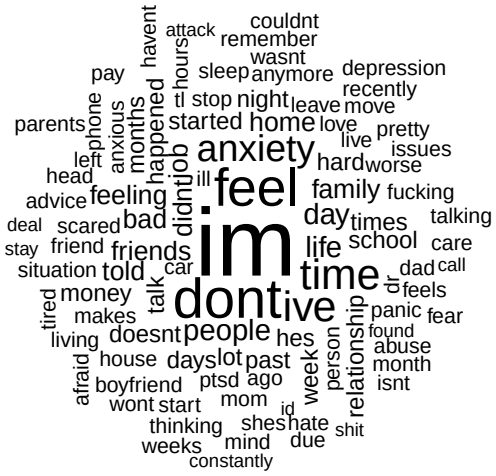
```
words_tokenized_by_label_counts %>%
  top_n(num, n) %>%
  ungroup() %>%
  mutate(label = as.factor(label)) %>%
  arrange(label, n) %>%
  mutate(topic_r = row_number()) %>%
  ggplot(aes(word, n, fill = label)) +
  geom_col() +
  facet_wrap(~ label, scales = "free") + theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```



Now, let's look at a word cloud of the words by label.

```
toPlot <- words_tokenized_by_label_counts %>%
  filter(label == 0)
wordcloud(toPlot$word, toPlot$n, max.words=100, random.order=FALSE, scale=c(3,0.5))
```


```
toPlot2 <- words_tokenized_by_label_counts %>%
  filter(label == 1)
wordcloud(toPlot2$word, toPlot2$n, max.words=100, random.order=FALSE, scale=c(3,0.5))
```

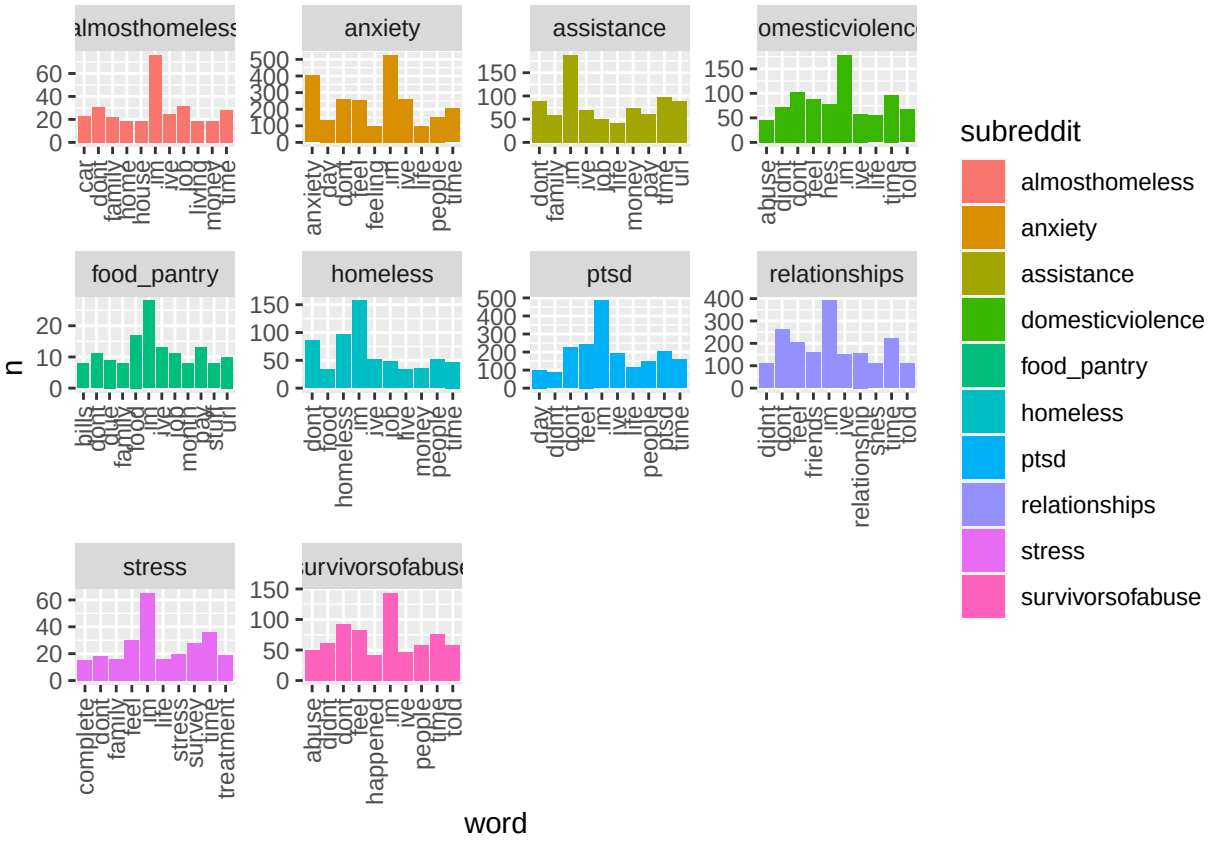



```
words_tokenized_by_subreddit_counts <- reddit_stress_data %>%
  select(c("id", "text", "label", "subreddit")) %>%
  unnest_tokens(word, text) %>%
  mutate(word = gsub('[[:punct:]]+','', word)) %>%
  mutate(word = gsub('\\<[[:digit:]]+\\>', '%d%', word)) %>%
  anti_join(stop_words) %>%
  group_by(subreddit) %>%
  count(word) %>%
  filter(word != "%d%") %>%
  arrange(subreddit, desc(n))
```

Top 10 Words By Subreddit

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

```
words_tokenized_by_subreddit_counts %>%
  top_n(num, n) %>%
  ungroup() %>%
  arrange(subreddit, n) %>%
  mutate(topic_r = row_number()) %>%
  ggplot(aes(word, n, fill = subreddit)) +
  geom_col() +
  facet_wrap(~ subreddit, scales = "free") + theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```



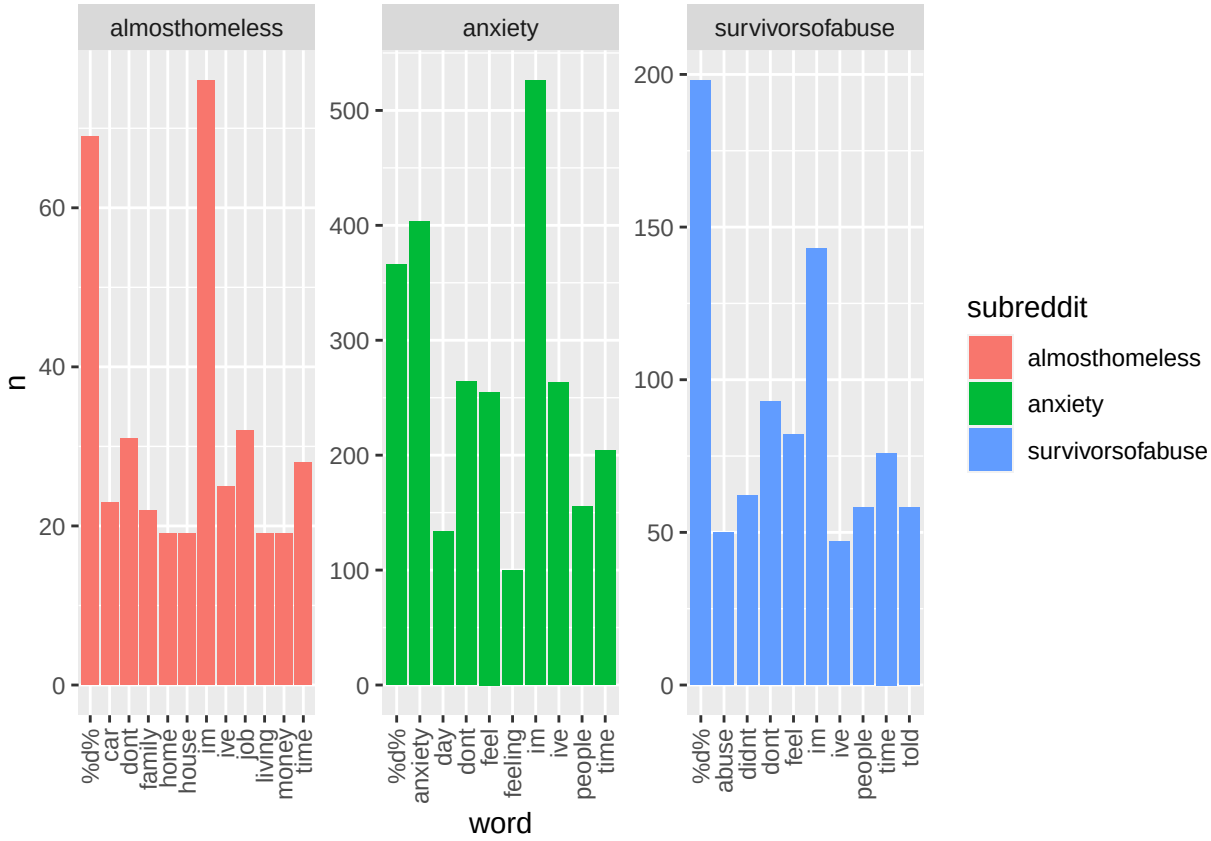
Looking at a smaller subset of the subreddits

```
words_tokenized_by_subreddit_counts_partial <- reddit_stress_data %>%
  select(c("id", "text", "label", "subreddit")) %>%
  filter(subreddit == "almosthomeless" | subreddit == "anxiety" | subreddit == "survivorsofabuse") %>%
  unnest_tokens(word, text) %>%
  mutate(word = gsub('[[:punct:]]+','', word)) %>%
  mutate(word = gsub('\\<[[:digit:]]+\\>', '%d%', word)) %>%
  anti_join(stop_words) %>%
  group_by(subreddit) %>%
  count(word) %>%
  arrange(subreddit, desc(n))
```

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

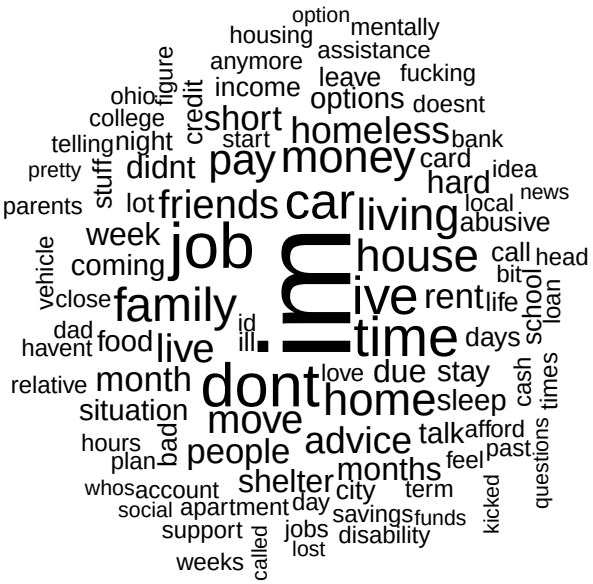
```
words_tokenized_by_subreddit_counts_partial %>%
  top_n(10, n) %>%
  ungroup() %>%
  arrange(subreddit, n) %>%
```

```
  mutate(topic_r = row_number()) %>%
  ggplot(aes(word, n, fill = subreddit)) +
  geom_col() +
  facet_wrap(~ subreddit, scales = "free") + theme(axis.text.x=element_text(angle=90,hjust=1,vjust=0.5))
```

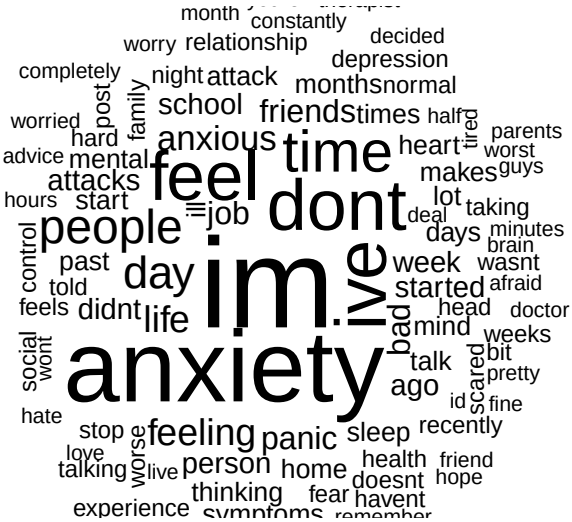


Now looking at each subreddit's individual word cloud.

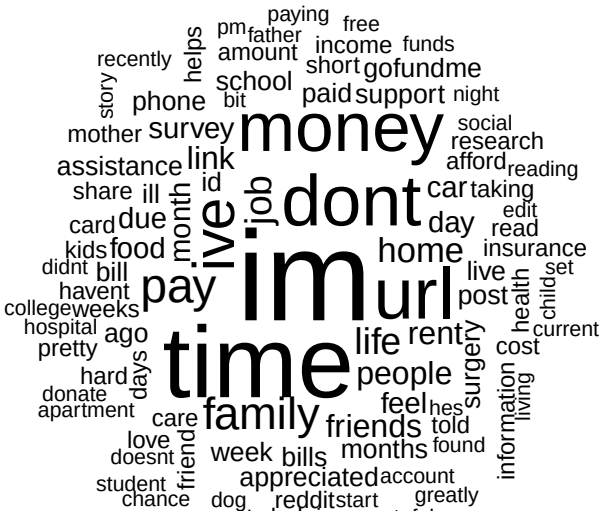
```
toPlot <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "almosthomeless")
wordcloud(toPlot$word, toPlot$n, max.words=100, random.order=FALSE)
```



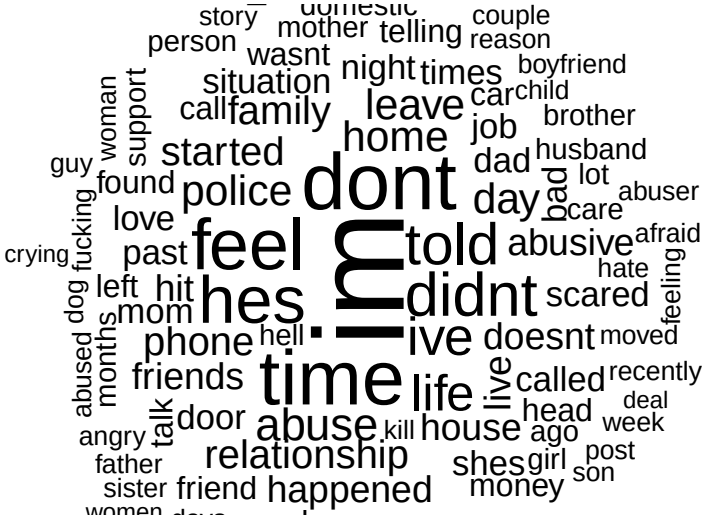

```
#wordcloud2
toPlot2 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "anxiety")
wordcloud(toPlot2$word, toPlot2$n, max.words=100, random.order=FALSE)
```



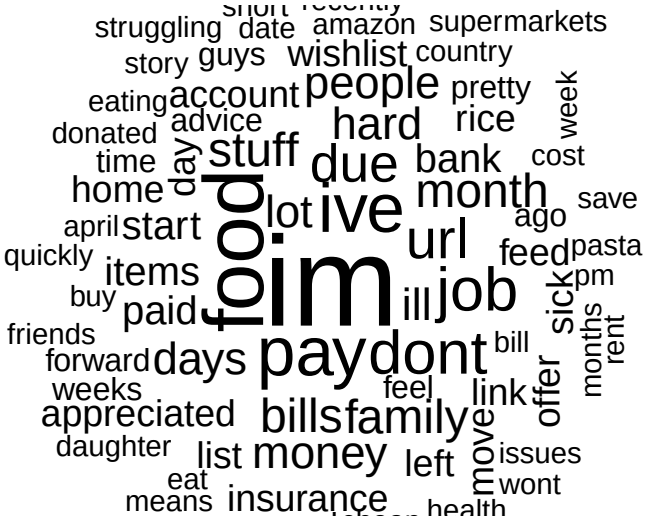

```
#wordcloud4
toPlot4 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "assistance")
wordcloud(toPlot4$word, toPlot2$n, max.words=100, random.order=FALSE)
```



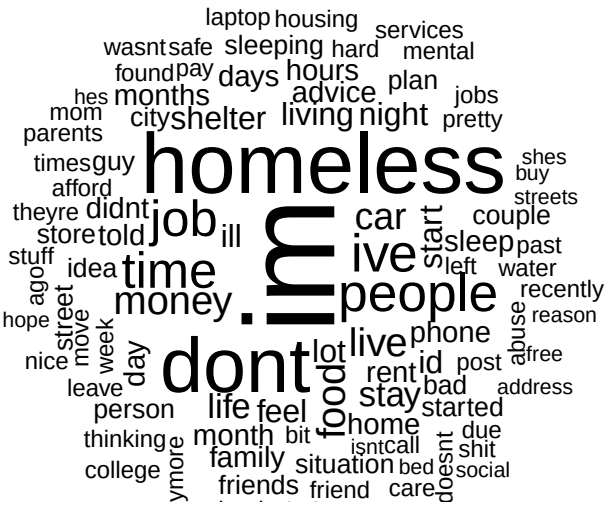

```
toPlot5 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "domesticviolence")
wordcloud(toPlot5$word, toPlot5$n, max.words=100, random.order=FALSE)
```




```
#wordcloud2
toPlot6 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "food_pantry")
wordcloud(toPlot6$word, toPlot6$n, max.words=100, random.order=FALSE)
```

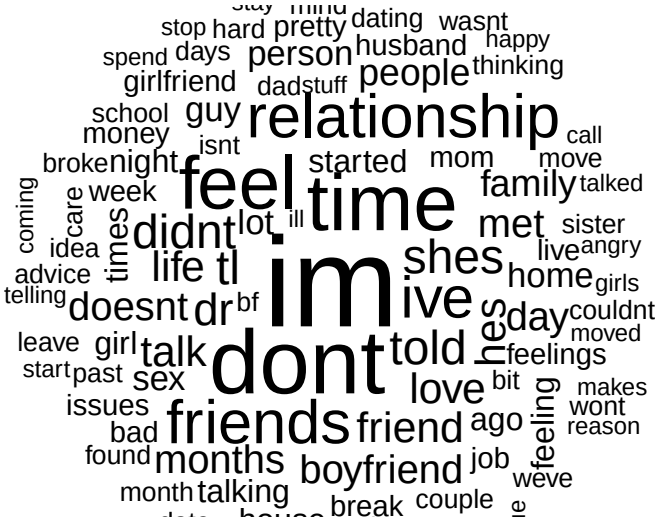



```
#wordcloud3
toPlot7 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "homeless")
wordcloud(toPlot7$word, toPlot7$n, max.words=100, random.order=FALSE)
```




```
#wordcloud4
toPlot8 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "ptsd")
wordcloud(toPlot8$word, toPlot8$n, max.words=100, random.order=FALSE)
```


```
toPlot9 <-words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "relationships")
wordcloud(toPlot9$word, toPlot9$n, max.words=100, random.order=FALSE)
```




```
#wordcloud2
toPlot10 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "stress")
wordcloud(toPlot10$word, toPlot10$n, max.words=100, random.order=FALSE)
```

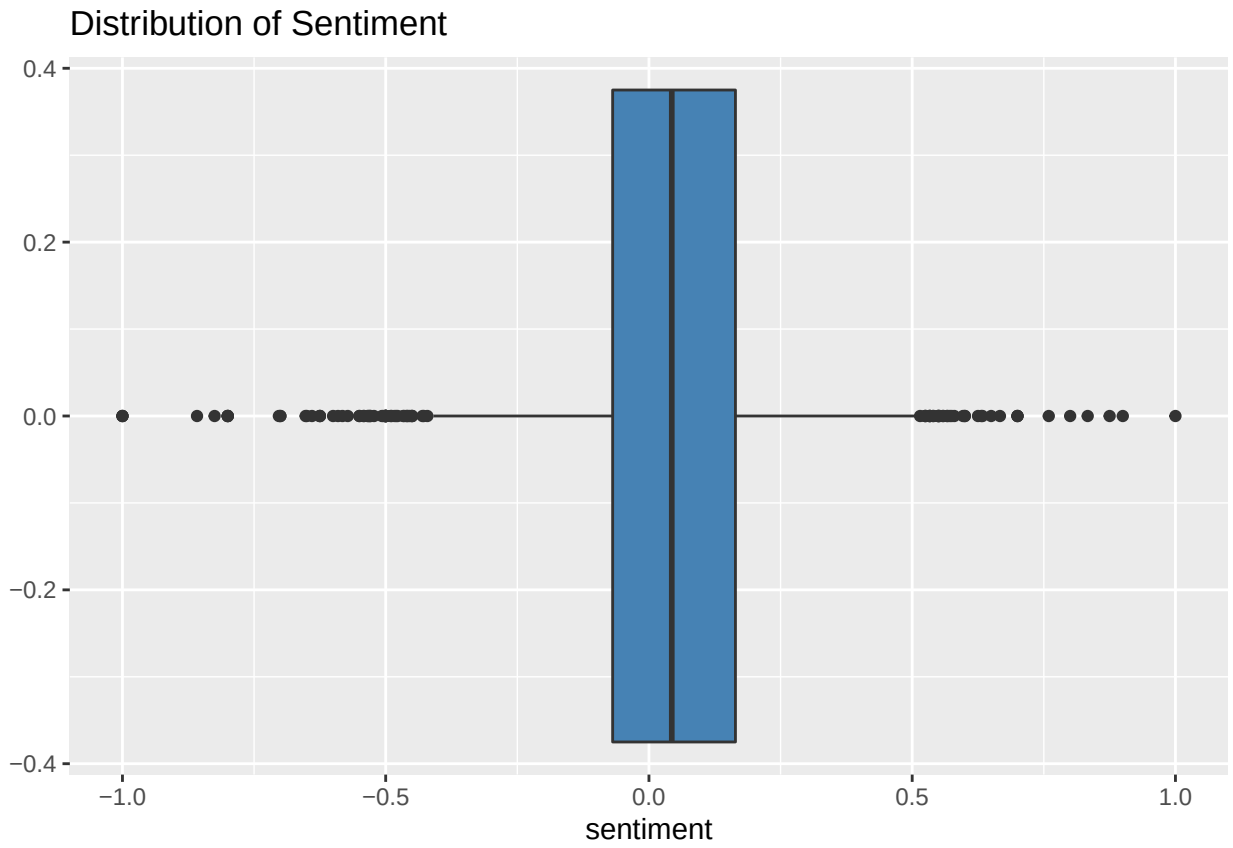

```
#wordcloud3
toPlot11 <- words_tokenized_by_subreddit_counts %>%
  filter(subreddit == "survivorsofabuse")
wordcloud(toPlot11$word, toPlot11$n, max.words=100, random.order=FALSE)
```

Sentiment Distribution

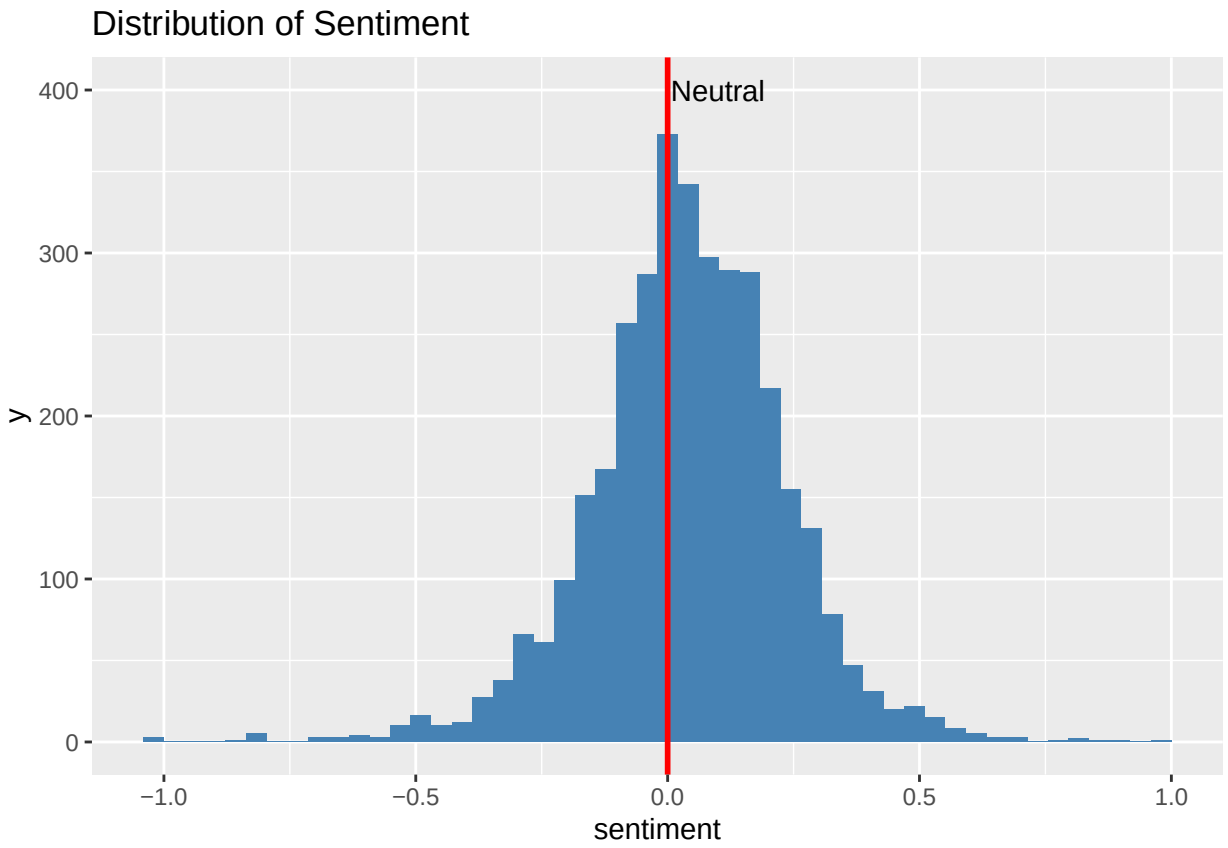
```
ggplot(reddit_stress_data, aes(x = sentiment)) + geom_boxplot(fill = "steelblue", bins = 50) + labs(tit
```

Sentiment Distribution Overall

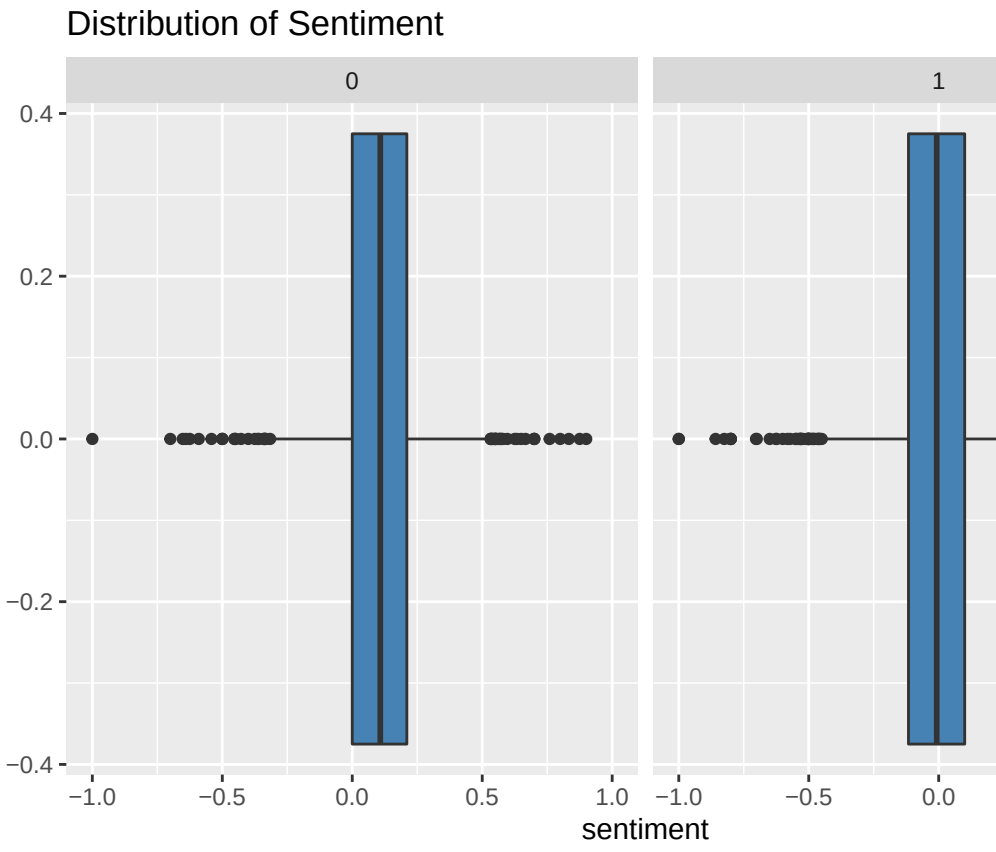
```
## Warning: Ignoring unknown parameters: bins
```




```
mx <- 0
ggplot(reddit_stress_data, aes(x = sentiment)) + geom_histogram(fill = "steelblue", bins = 50) + labs(t
```

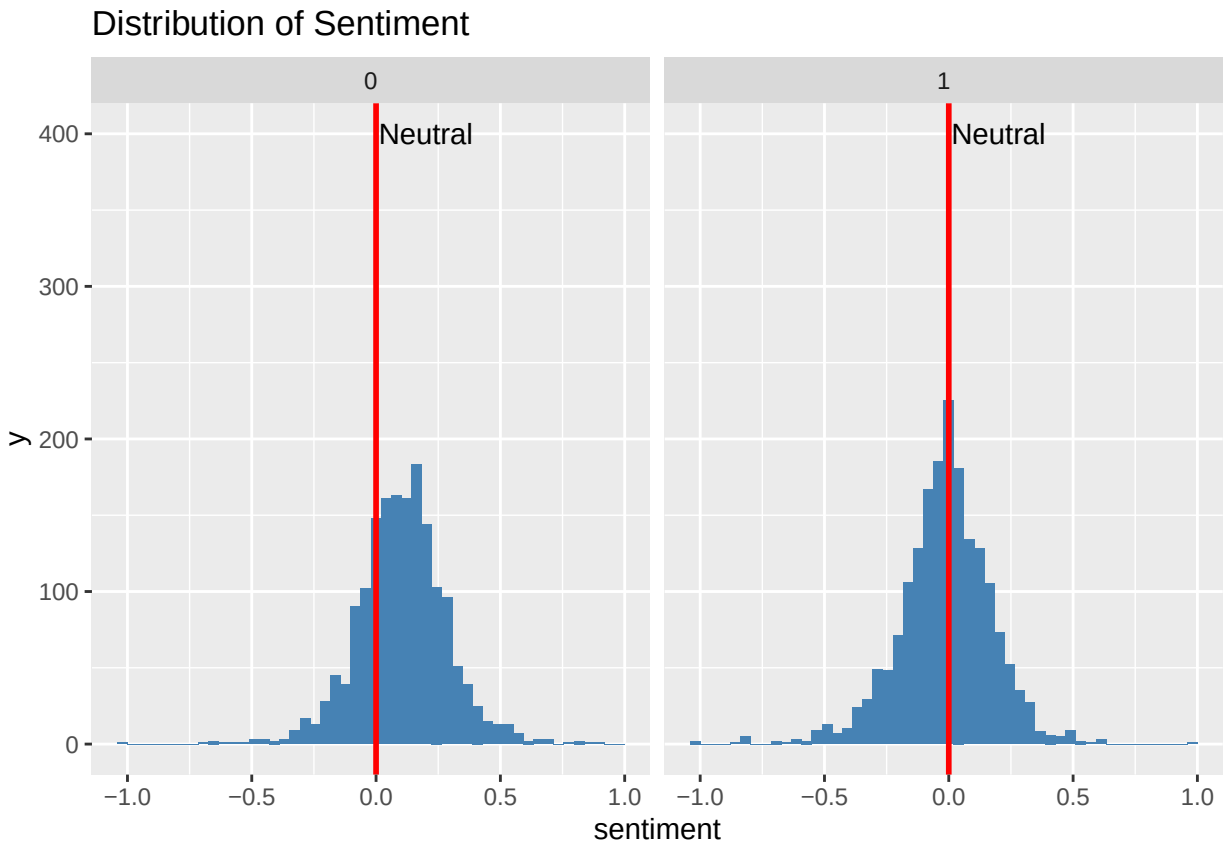



```
ggplot(reddit_stress_data, aes(x = sentiment)) + geom_boxplot(fill = "steelblue") + labs(title = "Distri
```

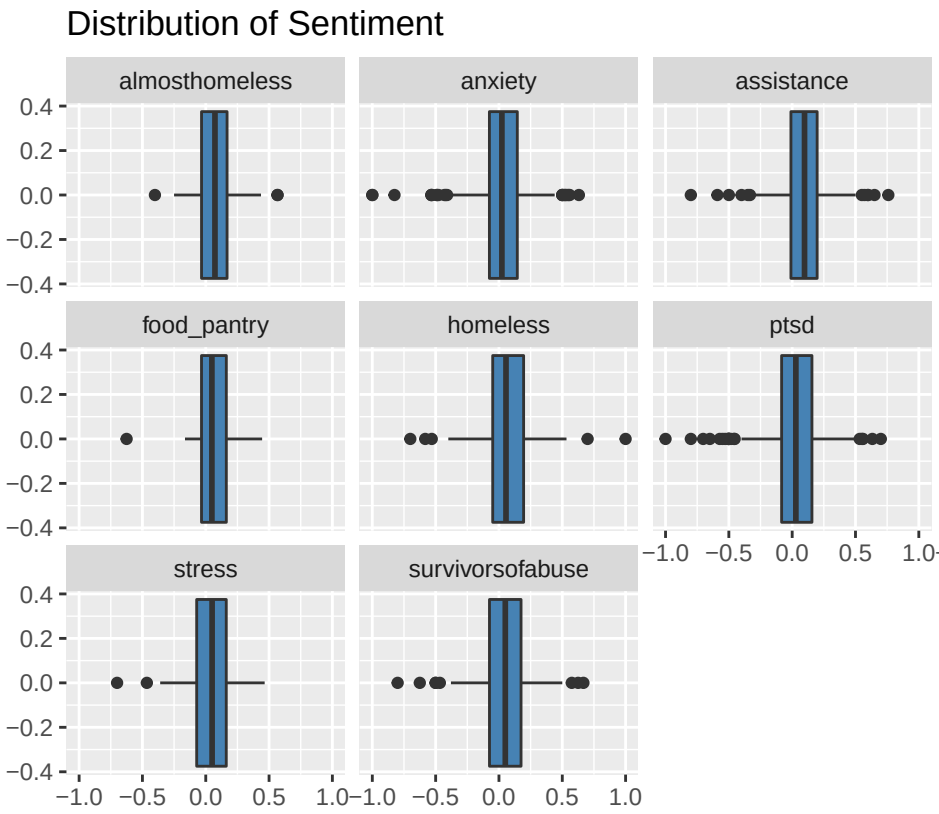


Sentiment Distribution By Label

```
ggplot(reddit_stress_data, aes(x = sentiment)) + geom_histogram(fill = "steelblue", bins = 50) + labs(t
```

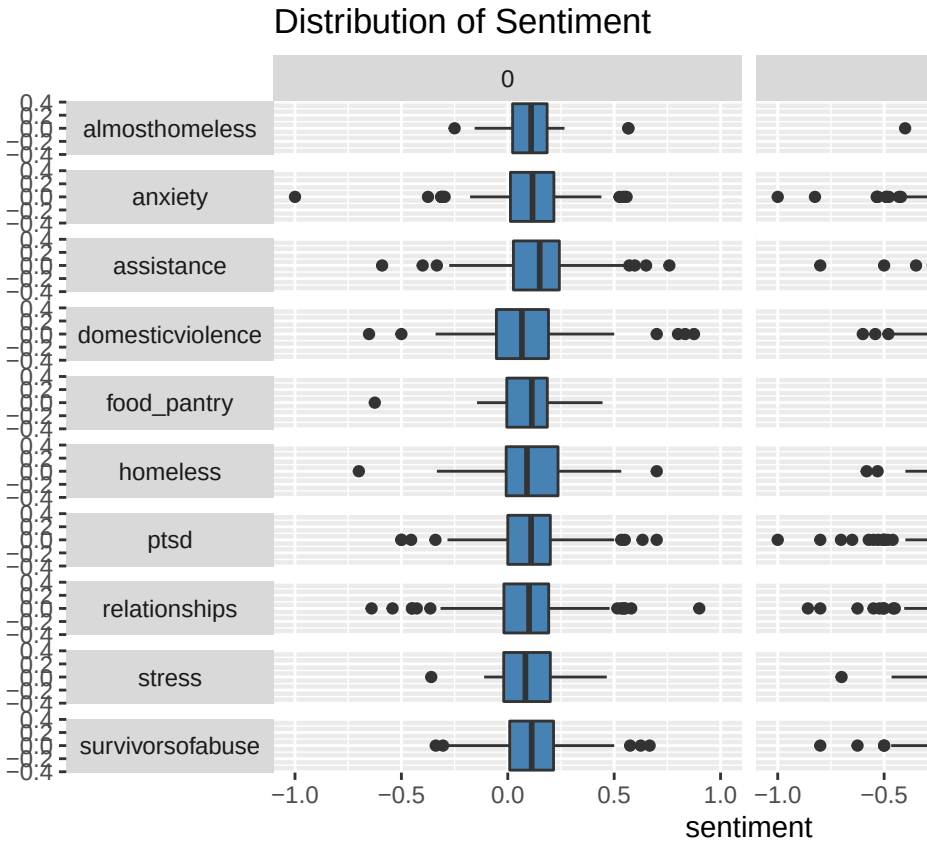



```
ggplot(reddit_stress_data, aes(x = sentiment)) + geom_boxplot(fill = "steelblue") + labs(title = "Distri
```



Sentiment Distribution By Subreddit

```
ggplot(reddit_stress_data, aes(x = sentiment)) + geom_boxplot(fill = "steelblue") + labs(title = "Distri
```



Top 10 Words By Label and Subreddit

Statistical Analysis

Chi-Square Test

```
chisq.test(reddit_stress_data$subreddit, reddit_stress_data$label)
```

```
##
##  Pearson's Chi-squared test
##
## data:  reddit_stress_data$subreddit and reddit_stress_data$label
## X-squared = 158.87, df = 9, p-value < 2.2e-16
```

```
aov(reddit_stress_data$sentiment ~ as.factor(reddit_stress_data$label))
```

```
## Call:
##    aov(formula = reddit_stress_data$sentiment ~ as.factor(reddit_stress_data$label))
##
## Terms:
##                 as.factor(reddit_stress_data$label) Residuals
## Sum of Squares                             12.82015 122.51554
## Deg. of Freedom                                   1      3551
##
## Residual standard error: 0.1857463
## Estimated effects may be unbalanced
```

Decision Tree Model

```
bag_of_words_with_unknowns <- CreateBagOfWordsWithUnknowns(reddit_stress_data, rare_defn = 15)
stress_data_with_bow <- reddit_stress_data %>%
  select(sentiment, label, id) %>%
  left_join(bag_of_words_with_unknowns, by = "id")
stress_data_with_bow %>%
  select(c("sentiment", "label", "id", "text_cancel", "text_cancer", "text_constantly")) %>%
  head(2)
```

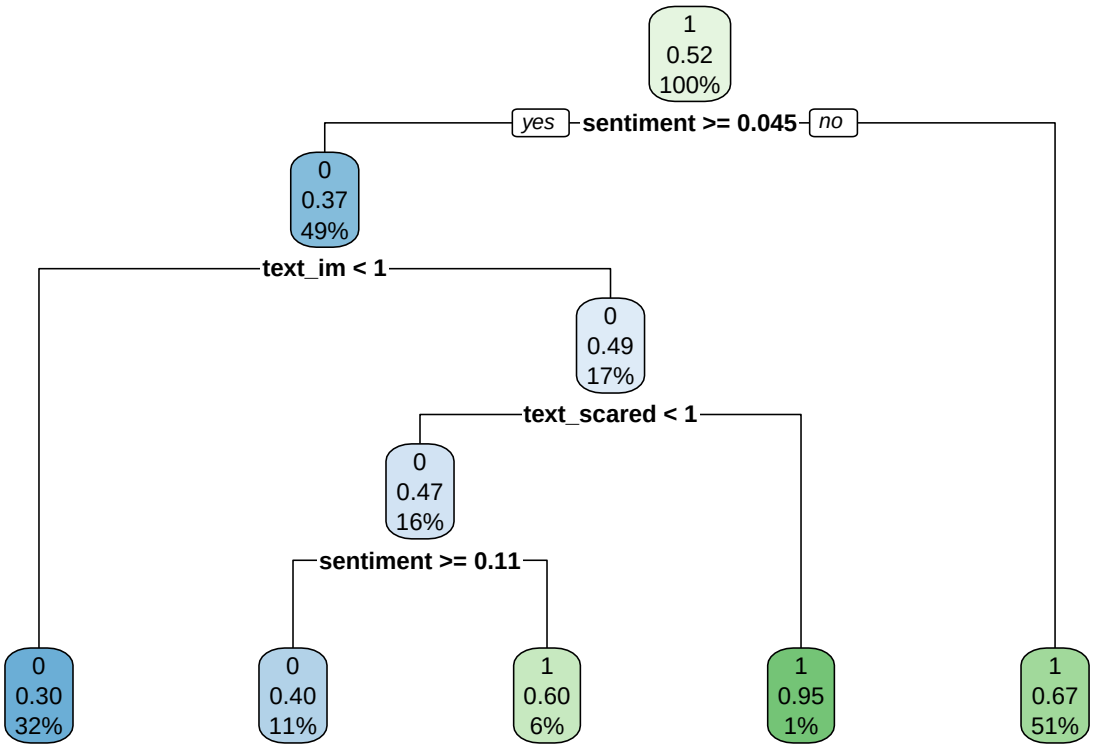
```
## # A tibble: 2 x 6
##   sentiment label    id text_cancel text_cancer text_constantly
##       <dbl> <dbl> <dbl>       <dbl>       <dbl>           <dbl>
## 1  -0.00274     1 33181           0           0               0
## 2   0.293       0  2606           0           0               0
```

```
stress_data_for_analysis <- stress_data_with_bow %>%
  select(-c(id))
stress_data_for_analysis <- data.frame(stress_data_for_analysis)
set.seed(123)
sample_data = sample.split(stress_data_for_analysis, SplitRatio = 0.75)
str(sample_data)
```

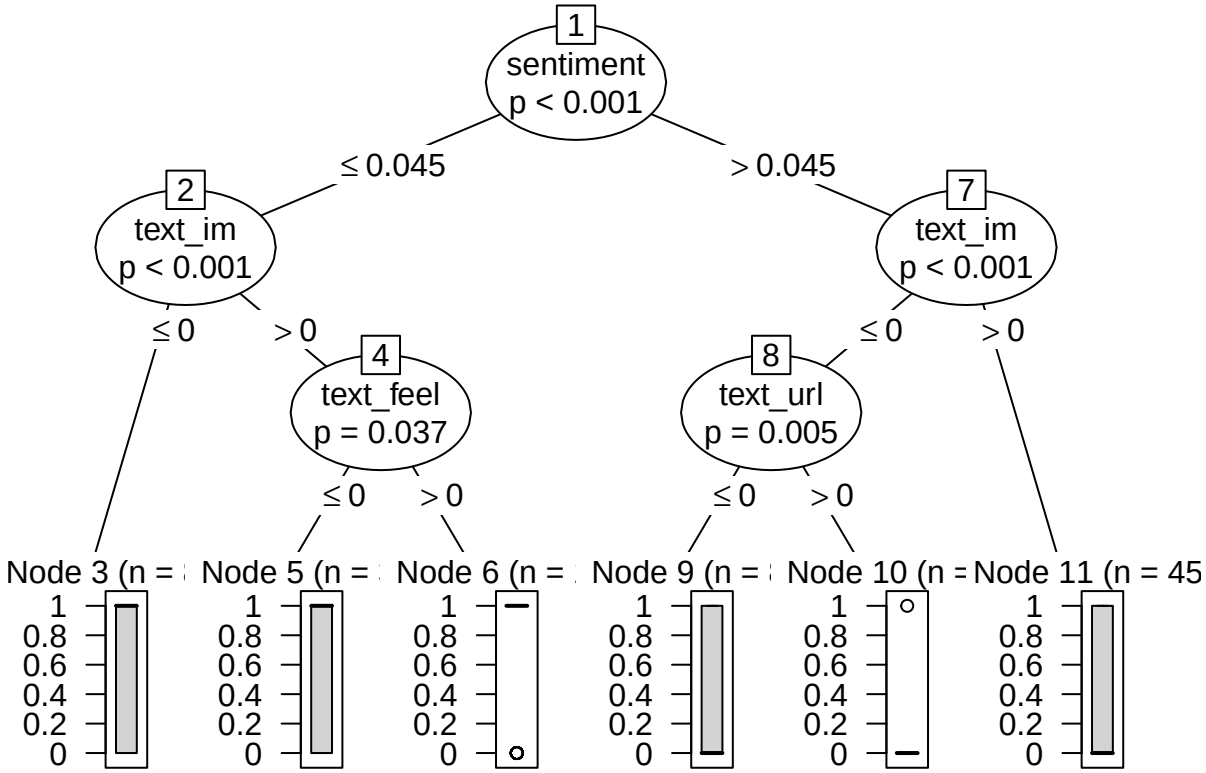
```
##  logi [1:1724] TRUE FALSE TRUE FALSE FALSE TRUE ...
```

```
train_data <- tibble(subset(stress_data_for_analysis, sample_data == TRUE))
test_data <- tibble(subset(stress_data_for_analysis, sample_data == FALSE))
```

```
rtree <- rpart(label ~ ., data = train_data, method = "class")
rpart.plot(rtree)
```



```
ctree_ <- ctree(label ~ ., train_data)
plot(ctree_)
```




```
rpart.rules(rtree)
```

```
##  label
##   0.30 when sentiment >=          0.045 & text_im <  1
##   0.40 when sentiment >=          0.114 & text_im >= 1 & text_scared <  1
##   0.60 when sentiment is 0.045 to 0.114 & text_im >= 1 & text_scared <  1
##   0.67 when sentiment <  0.045
##   0.95 when sentiment >=          0.045 & text_im >= 1 & text_scared >= 1
```

```
y_pred = predict(rtree, train_data, type = "class")
confusionMatrix(y_pred,as.factor(train_data$label))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  769  373
##          1  513 1008
##
##                Accuracy : 0.6673
##                  95% CI : (0.649, 0.6852)
##     No Information Rate : 0.5186
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.331
##
##  Mcnemar's Test P-Value : 3.015e-06
```

```
##
##             Sensitivity : 0.5998
##             Specificity : 0.7299
##          Pos Pred Value : 0.6734
##          Neg Pred Value : 0.6627
##              Prevalence : 0.4814
##          Detection Rate : 0.2888
##    Detection Prevalence : 0.4288
##       Balanced Accuracy : 0.6649
##
##        'Positive' Class : 0
##
```

```
y_pred = predict(rtree, test_data, type = "class")
confusionMatrix(y_pred, as.factor(test_data$label))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 250 138
##          1 164 338
##
##                Accuracy : 0.6607
##                  95% CI : (0.6285, 0.6918)
##     No Information Rate : 0.5348
##     P-Value [Acc > NIR] : 1.755e-14
##
##                   Kappa : 0.3152
##
##  Mcnemar's Test P-Value : 0.1503
##
##             Sensitivity : 0.6039
##             Specificity : 0.7101
##          Pos Pred Value : 0.6443
##          Neg Pred Value : 0.6733
##              Prevalence : 0.4652
##          Detection Rate : 0.2809
##    Detection Prevalence : 0.4360
##       Balanced Accuracy : 0.6570
##
##        'Positive' Class : 0
##
```

Conclusions and Future Work