# R for Data Science Project 1 Final Paper

Melanie McCord

## Contents



Note: this is the abbreviated paper. For the full paper, look at FinalPaper.Rmd.

```
reddit_stress_data <- read_feather("reddit_stress_data.feather")
```

# Introduction

## Background

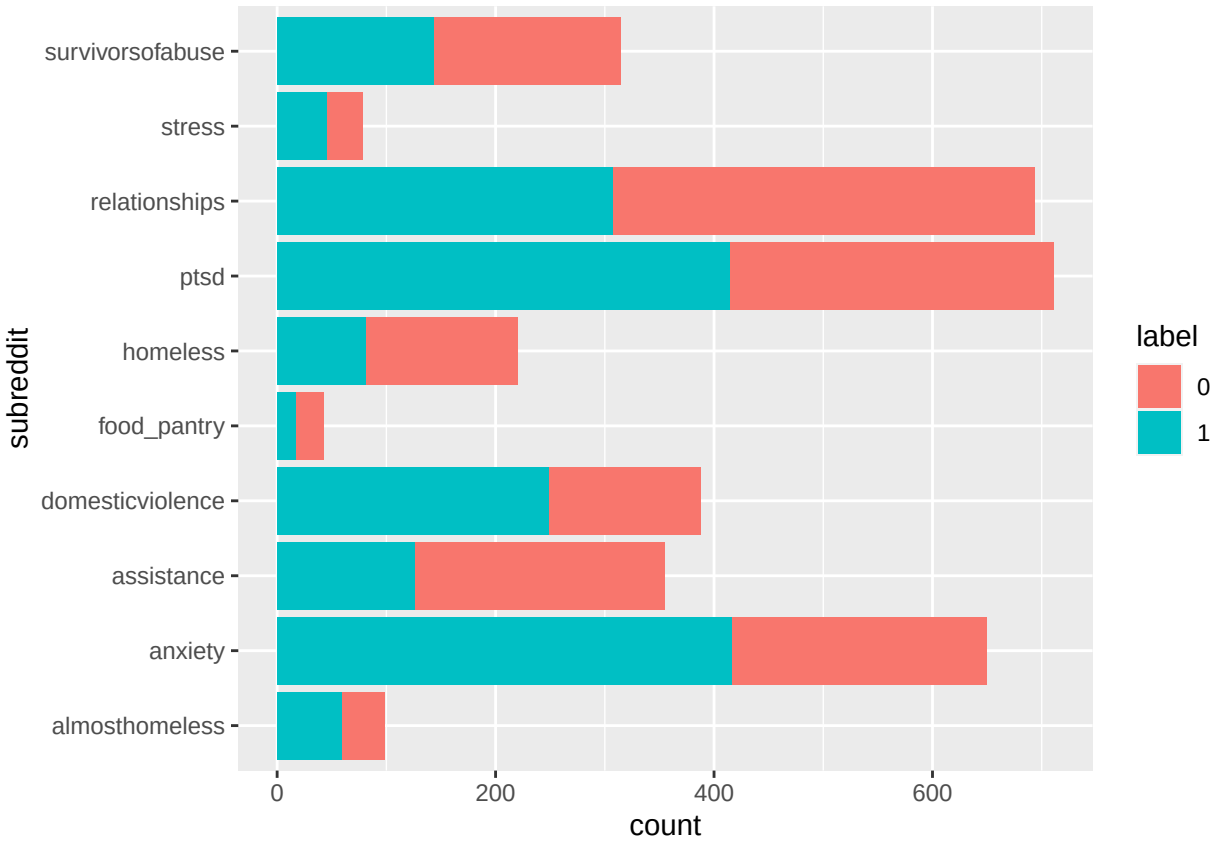
Stress is a common aspect of modern lives that can cause negative health outcomes, such as anxiety, depression, insomnia, and a weakened immune system. If we can predict stress on social media, we can help understand the extent of the problem and perhaps gain insights on how to address it. For this specific 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. Here is the distribution of the data by subreddit and label.

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




The dataset distribution is heavily imbalanced. There are many more posts in r/ptsd than 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.

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.

# Statistical Computing

## 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.

Detecting the rare words was the most complicated task that I had to do. I had to identify a metric for rare words and then convert them into a string. Then, I needed to replace each rare word with the "unk" token. Then, I needed to determine what the ideal criteria for rare words was. I chose n = 15 for the rare words since that seemed to be the point where the rare words were no longer clustered around the lowest possible value.

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.

# Results

I chose to examine the variables sentiment and the words chosen in the text for analysis for how that plays into whether a post is stressed or what subreddit it came from. I studied the relationships between words and label and subreddit using bar graphs of the top 10 words and wordcloud graphs. There were some differences between the top 10 words, but by far the better visualization for the differences was the wordcloud graph. For studying the sentiment variable, I used histograms, boxplots, and ridge plots. I found significant differences between the variables depending on the label and subreddit. My statistical modeling involved running tests to find out if the differences are statistically significant and then running a decision tree model in order to predict whether or not the data was stressed.

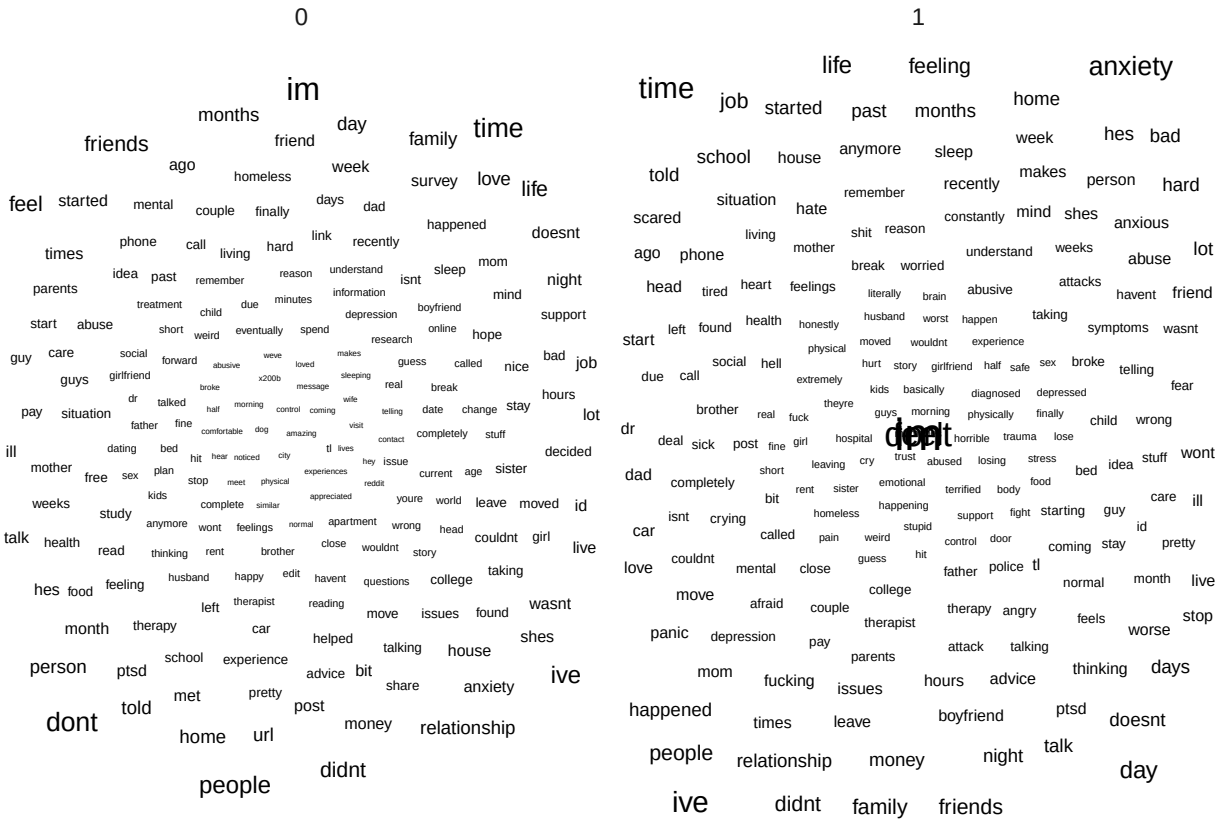
For word count, here is the difference between the words by subreddit.

```
num = 200
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))
```

```
## 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(label = word, size = n)) +
  geom_text_wordcloud() +
  facet_wrap(~ label) +
  theme_minimal()
```

```
## Warning in wordcloud_boxes(data_points = points_valid_first, boxes = boxes, :
## Some words could not fit on page. They have been placed at their original
## positions.
```




From this visualization, you can see that stress free posts have to do with a diversity of topics. There are a significant amount of negative words that appear fairly frequently, such as “homeless”, “ptsd” and anxiety, but the most common words are “I’ve” “don’t”, “time”, and “people”. Since the non-stress-related posts are still picked from subreddits related to stressful topics, this may be people simply talking about how they dealt with these issues or calmly asking for advice, rather than being obviously stressed while writing their posts.

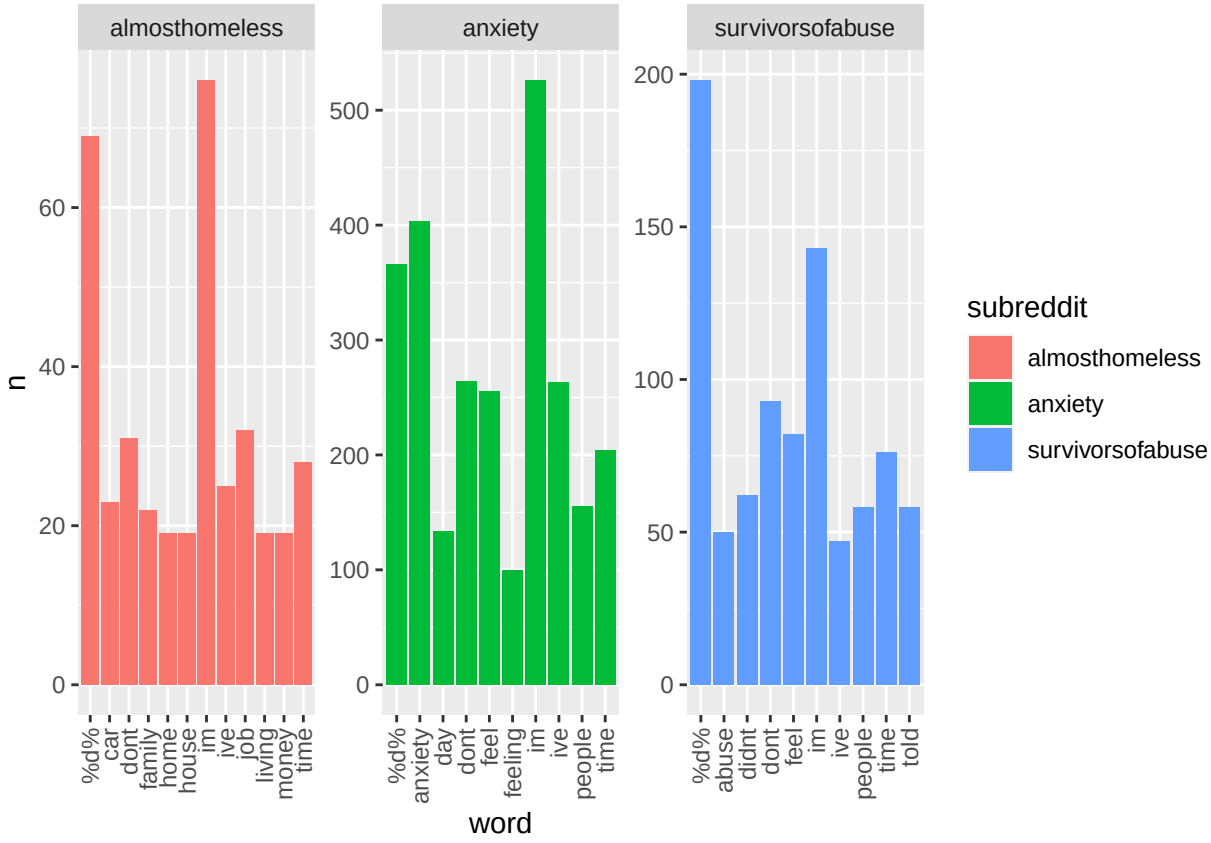
Among the stress-related posts, there is significantly more variation in the words and the only words that appear very frequently are “i’m”, “don’t”, “feel”, “time” and anxiety. There appear to be a lot of posts that seem related to the diversity of posts. Some words that commonly appear are “fucking”, “abuse”, “panic”, and “depression”. Time appears frequently in both the stress-related and non-stress-related subreddits, but proportionately much more frequently in the non-stress-related.
Overall, the stressed posts’ word distributions and the non-stressed posts’ word distributions appear similar, but the stressed posts appear to be more diverse, and the non-stressed posts appear to be more similar and have less very negative words among the top 100 words.

```
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)
```



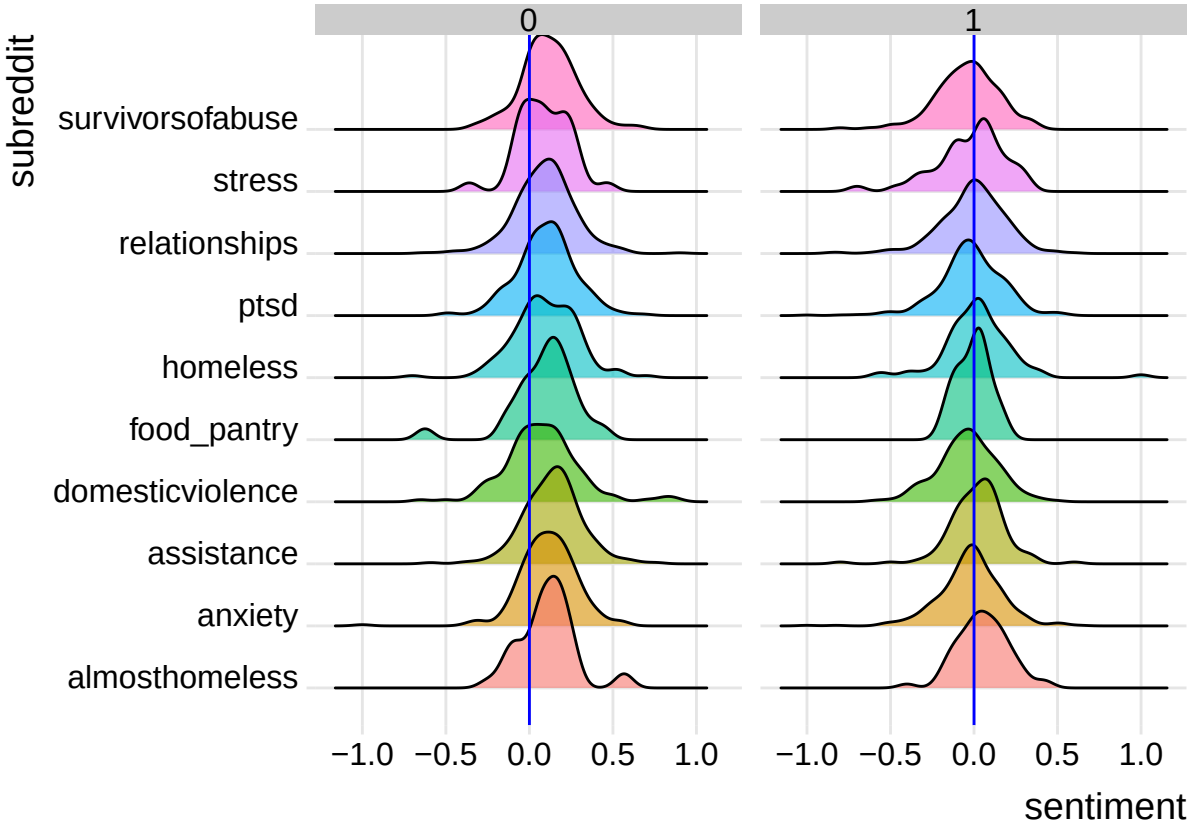

From this visualization, you can see that the most common words differ significantly by subreddit and seem to be related to the major topics of each subreddit.

Now, looking at the sentiment distribution, we can also see major differences.

```
mu = mean(reddit_stress_data$sentiment)
mx <- 0
ggplot(reddit_stress_data, aes(x = sentiment, y = subreddit, fill = subreddit)) +
  geom_density_ridges(alpha = 0.6) +
  theme_ridges() +
  theme(legend.position = "none") + facet_wrap(~ label) + geom_vline(xintercept = mx, col = "blue", lwd
```

```
## Picking joint bandwidth of 0.0542
```

```
## Picking joint bandwidth of 0.052
```




If we look at the differences by label, we can see further differences along the distribution. r/almosthomeless is normally distributed for stressed posts, whereas right skewed for the non-stressed posts. r/domesticviolence is right skewed for non-stressed posts and significantly more normally distributed. r/relationships looks like it is approximately normally distributed for both the stressed posts and the non-stressed posts. r/food_pantry is extremely skewed, which may be influenced by the smaller number of posts overall. Some of this may be affected by the overall number of posts in each category: for example, r/domesticviolence tends to have a higher proportion of stressed posts, whereas r/relationships has the most posts and tends to be more non-stressed than stressed.

## Statistical Analysis

### Chi-Square Test

Let's test to see if the stress data by subreddit and label are associated, and set the p-value to be 0.05.

```
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
```

Since our p-value is less than 0.05, we can reject the null hypothesis that the subreddits do not differ significantly in label by subreddit, and determine that there are significant differences between the label distribution by subreddit.

Now, let's compare the differences in sentiment by label using the analysis of variance test, and again set our p-value to be 0.05.

```
summary(aov(reddit_stress_data$label ~ reddit_stress_data$sentiment))
```

```
##                                Df Sum Sq Mean Sq F value Pr(>F)
## reddit_stress_data$sentiment    1   84.0   83.97   371.6 <2e-16 ***
## Residuals                    3551  802.5    0.23
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value is less than 0.05, we can reject the null hypothesis that the differences between sentiment by label are not significant and say that the differences between sentiment by label are significant.

### Decision Tree Model

Finally, I will run a decision tree model on the dataset and see what variables are the strongest in predicting whether or not a post is stressed.
A decision tree model is a machine learning model that determines classes by similar groups in the data and then uses them to determine rules for the classification. For example, if $(x_1 > 0.5 \land x_2 < 0.5 \implies TRUE)$
For simplicity, I am only going to use the following to predict the result: bag of words model of the posts and sentiment in order to predict the label.
I will read in the data. Since the full dataset has over 1000 variables, I will not display all of them, but just a subset so that you can see some of the variables that were selected.

```
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", "text_cancel", "text_cancer", "text_constantly")) %>%
  head(2)
```

```
## # A tibble: 2 x 5
##   sentiment label text_cancel text_cancer text_constantly
##       <dbl> <dbl>       <dbl>       <dbl>           <dbl>
## 1  -0.00274     1           0           0               0
## 2   0.293       0           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)
```

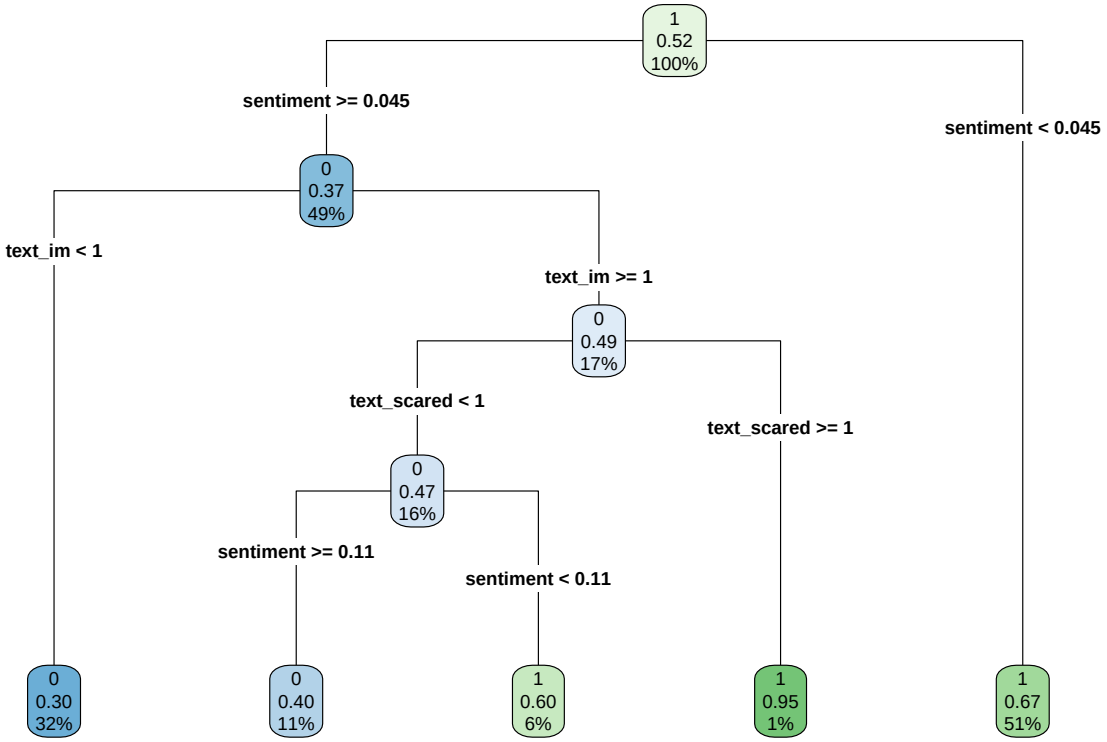
```
##  logi [1:1724] TRUE FALSE TRUE FALSE FALSE TRUE ...
```

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

Now, let's run the decision tree model on the dataset and see what the most significant features are for determining whether a post is stressed or not.

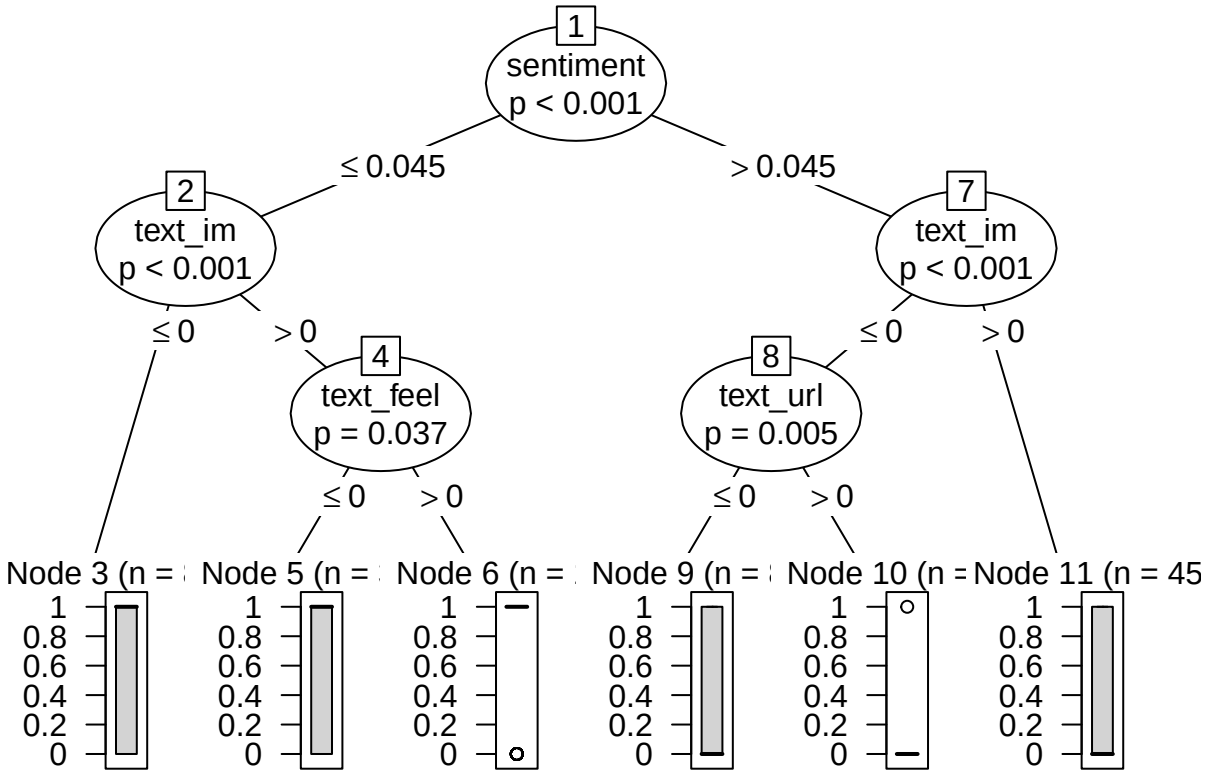
```
rtree <- rpart(label ~ ., data = train_data, method = "class")
```

```
rpart.plot(rtree, yesno = 2, type = 4, clip.right.labs = FALSE)
```




According to this model, the most significant features are the sentiment, using "scared", and using "im". If sentiment $<$ 0.045, it is likely going to be a stressed label. Otherwise, depending on other features, including whether "scared" appears, whether "im" appears, it may or may not be stressed.

```
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
```

First, let's see how well my model performs on the training data.

```
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
##
```

Now, let's check its performance on the test data.

```
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
##
```

For both the training and test data, the classification is accurate around 66% of the time. This means that the model does not predict the training data very well nor the test data.

## Conclusions and Future Work

Overall, there are significant differences in the data depending on the sentiment, words, and the subreddits by label. The data was imbalanced, resulting in a lower classification.

Future work will include testing the decision tree with more parameters, adding in the other variables, and testing multiple different classification methods.