INTA 4803 Final Exam

Madi Wickett

```
library(ggplot2)
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(topicmodels)
library(quanteda)
```

```
## Package version: 2.1.2
```

```
## Parallel computing: 2 of 8 threads used.
```

```
## See https://quanteda.io for tutorials and examples.
```

```
##
## Attaching package: 'quanteda'
```

```
## The following object is masked from 'package:utils':
##
##     View
```

```
library(tidytext)
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.0 ──
```

```
## ✓ tibble  3.0.3     ✓ dplyr   1.0.2
## ✓ tidyr   1.1.2     ✓ stringr 1.4.0
## ✓ readr   1.3.1     ✓ forcats 0.5.0
## ✓ purrr   0.3.4
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(tm)
```

```
## Loading required package: NLP
```

```
##
## Attaching package: 'NLP'
```

```
## The following objects are masked from 'package:quanteda':
##
##     meta, meta<-
```

```
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

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

```
## The following objects are masked from 'package:quanteda':
##
##     as.DocumentTermMatrix, stopwords
```

```
library(doMC)
```

```
## Loading required package: foreach
```

```
##
## Attaching package: 'foreach'
```

```
## The following objects are masked from 'package:purrr':
##
##     accumulate, when
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

```
library(SnowballC)
library(textdata)
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(caretEnsemble)
```

```
##
## Attaching package: 'caretEnsemble'
```

```
## The following object is masked from 'package:ggplot2':
##
##     autoplot
```

Question 1 LDA Topic Modeling

```
ira1<-read.csv("IRAhandle_tweets_1.csv", stringsAsFactors = FALSE)
ira2<-read.csv("IRAhandle_tweets_2.csv")
ira3<-read.csv("IRAhandle_tweets_3.csv", stringsAsFactors = FALSE)
```

```
ira<-rbind(ira1, ira2, ira3)
```

```
tidy_r_tweets<-ira %>%
  unnest_tokens(word, content) %>%
  anti_join(stop_words) %>%
  count(word, sort = TRUE)
```

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

```
custom<-add_row(stop_words, word="t.co", lexicon="custom")
custom<-add_row(custom, word="rt", lexicon="custom")
custom<-add_row(custom, word="https", lexicon="custom")
custom<-add_row(custom, word="http", lexicon="custom")
custom<-add_row(custom, word="в", lexicon="custom")
custom<-add_row(custom, word="на", lexicon="custom")
custom<-add_row(custom, word="amp", lexicon="custom")
custom<-add_row(custom, word="и", lexicon="custom")
custom<-add_row(custom, word="с", lexicon="custom")
custom<-add_row(custom, word="не", lexicon="custom")
custom<-add_row(custom, word="2", lexicon="custom")
custom<-add_row(custom, word="1kpxto2hfw", lexicon="custom")
custom<-add_row(custom, word="за", lexicon="custom")
custom<-add_row(custom, word="amb", lexicon="custom")
custom<-add_row(custom, word="о", lexicon="custom")
custom<-add_row(custom, word="1", lexicon="custom")
custom<-add_row(custom, word="di", lexicon="custom")
```

```
tweet_weights<-ira %>%
  unnest_tokens(output = "word", token = "words", input = content) %>%
  anti_join(custom) %>%
  mutate(word=wordStem(word)) %>%
  count(tweet_id, word) %>%
  bind_tf_idf(word, tweet_id, n)
```

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

```
nums <- tweet_weights %>%
  filter(str_detect(word, "^[0-9]")) %>% select(word) %>% unique()

### Filter Numbers ###
tweet_weights <- tweet_weights %>%
  anti_join(nums, by = "word")

### Filter out Non-ACSII Characters (Non-English Characters)
tweet_weights <- tweet_weights %>%
         mutate(word = iconv(word, from = "latin1", to = "ASCII")) %>%
         filter(!is.na(word))
```

```
tweet_matrix <- tweet_weights %>%
  cast_dtm(document = tweet_id, term = word,
           value = n, weighting = weightTf)

# Print the matrix details
tweet_matrix
```

```
## <<DocumentTermMatrix (documents: 722488, terms: 768949)>>
## Non-/sparse entries: 4954360/555551470752
## Sparsity           : 100%
## Maximal term length: 121
## Weighting          : term frequency (tf)
```

```
lda<-LDA(tweet_matrix, k = 10, method = "Gibbs", control = list(verbose=25L, seed = 1
23, burnin = 100, iter = 500))
```

```
## K = 10; V = 768949; M = 722488
## Sampling 600 iterations!
## Iteration 25 ...
## Iteration 50 ...
## Iteration 75 ...
## Iteration 100 ...
## Iteration 125 ...
## Iteration 150 ...
## Iteration 175 ...
## Iteration 200 ...
## Iteration 225 ...
## Iteration 250 ...
## Iteration 275 ...
## Iteration 300 ...
## Iteration 325 ...
## Iteration 350 ...
## Iteration 375 ...
## Iteration 400 ...
## Iteration 425 ...
## Iteration 450 ...
## Iteration 475 ...
## Iteration 500 ...
## Iteration 525 ...
## Iteration 550 ...
## Iteration 575 ...
## Iteration 600 ...
## Gibbs sampling completed!
```

```
tweet_df<-tidy(tweet_matrix)
head(df)
```

```
##
## 1 function (x, df1, df2, ncp, log = FALSE)
## 2 {
## 3     if (missing(ncp))
## 4         .Call(C_df, x, df1, df2, log)
## 5     else .Call(C_dnf, x, df1, df2, ncp, log)
## 6 }
```

I examined each of the topics and related terms, but removed them for the sake of space and clarity. Many of them were nonsensical, or even in another language. However, topic 8 seemed to have a coherent theme around police and violence. I chose this one to look at given the increased conversation around police brutality in 2016-2018. In another context, speaking frequently about police and crime could be a way to stir up racist or xenophobic rhetoric.

```
terms <- get_terms(lda, 15)
paste(terms[,8], collapse = ", ")
```

```
## [1] "polic, break, kill, new, shoot, chicago, fire, offic, woman, arrest, local, d
eath, shot, citi, home"
```

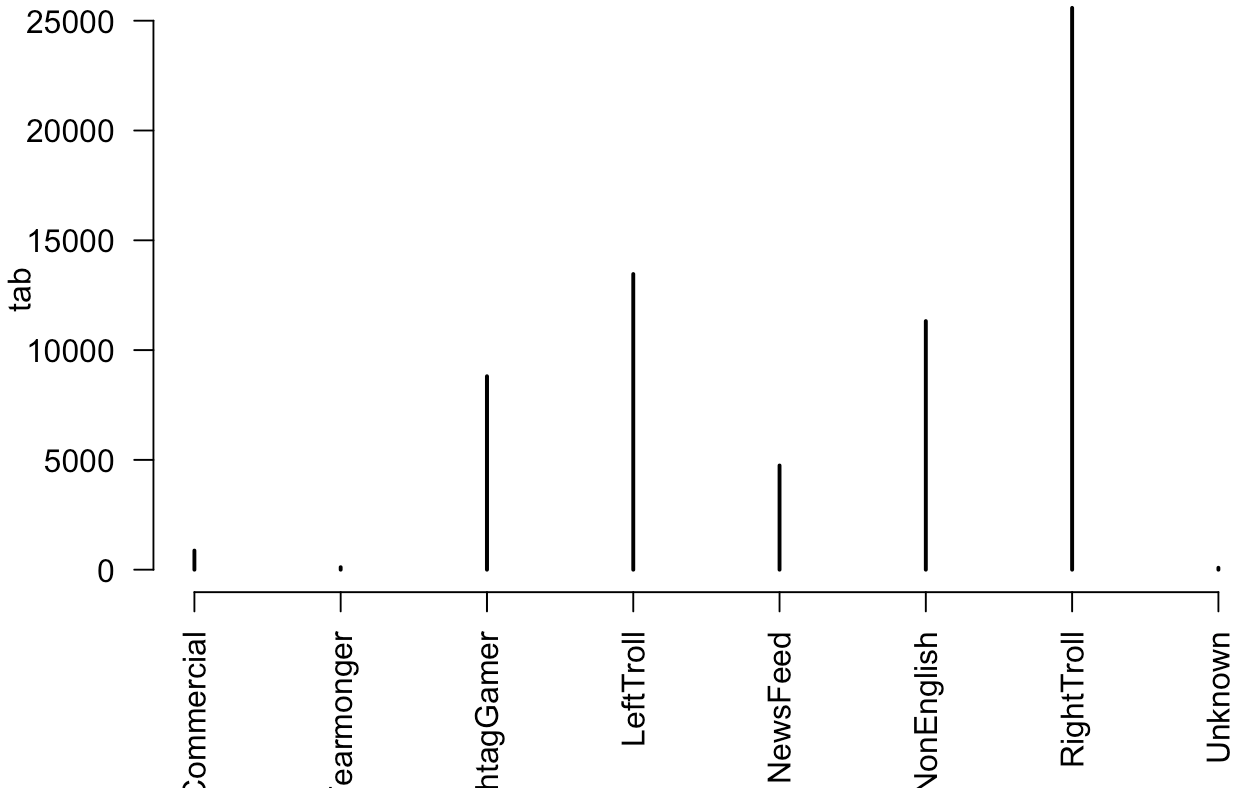
```
ira_trimmed$prob_topic<-lda@gamma[,8]
```

```
topics <- get_topics(lda, 1)
```

```
ira_trimmed$pred_topic<-topics
tab<-table(ira_trimmed$account_category[ira_trimmed$pred_topic == 8])
```

This plot shows that right wing trolls are tweeting most frequently on this topic, with left wing trolls in a distant second. This suggests that tweets in this category will use these words to describe pro-police rhetoric and racist associations with crime.

```
plot(tab, las = 2)
```



Question 2 Sentiment Analysis

I chose to only look at sentiment analysis for tweets that were in topic 8.

```
cont<-subset(ira_trimmed, topics == 8)

tweets<-cont %>%
  select(author, content) %>%
  group_by(author) %>%
  ungroup()
```

```
tweets_words<- tweets %>%
  unnest_tokens(input = "content", token = "words", "word")
```

I chose to look at the dichotomy of trust and fear for this particular category, given that it concerns public safety. Below are the top ten most frequently used words in both the “fear” and “trust” sentiment category. Interestingly, “police” shows up in both categories. It is actually well-suited, as connecting police with fear or trust will align based on political views. Another interesting thing is that “president” and “white” show up in the “trust” category. Based on your political leanings, those words could also just as well fit into a “fear” or “negative” category. This points out a flaw of sentiment analysis based on words alone. Context around those words can change their sentiment drastically, and that is not well reflected here.

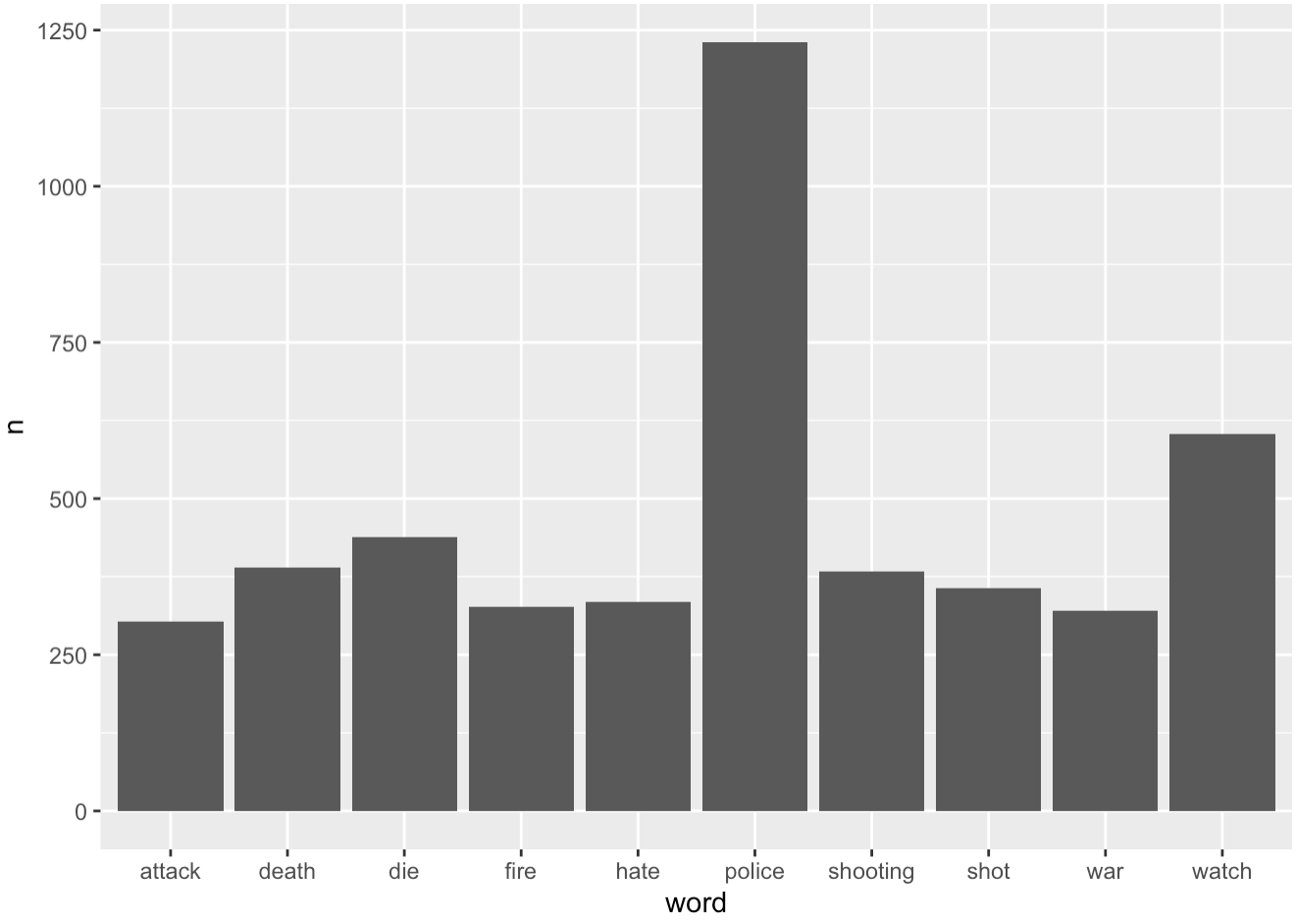
```
nrc_fear<-get_sentiments("nrc") %>%
  filter(sentiment == "fear")

fear <- tweets_words %>%
  inner_join(nrc_fear) %>%
  count(word, sort = TRUE)
```

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

```
top_n(fear, n=10) %>%
          ggplot(., aes(x=word, y=n))+
              geom_bar(stat='identity')
```

```
## Selecting by n
```



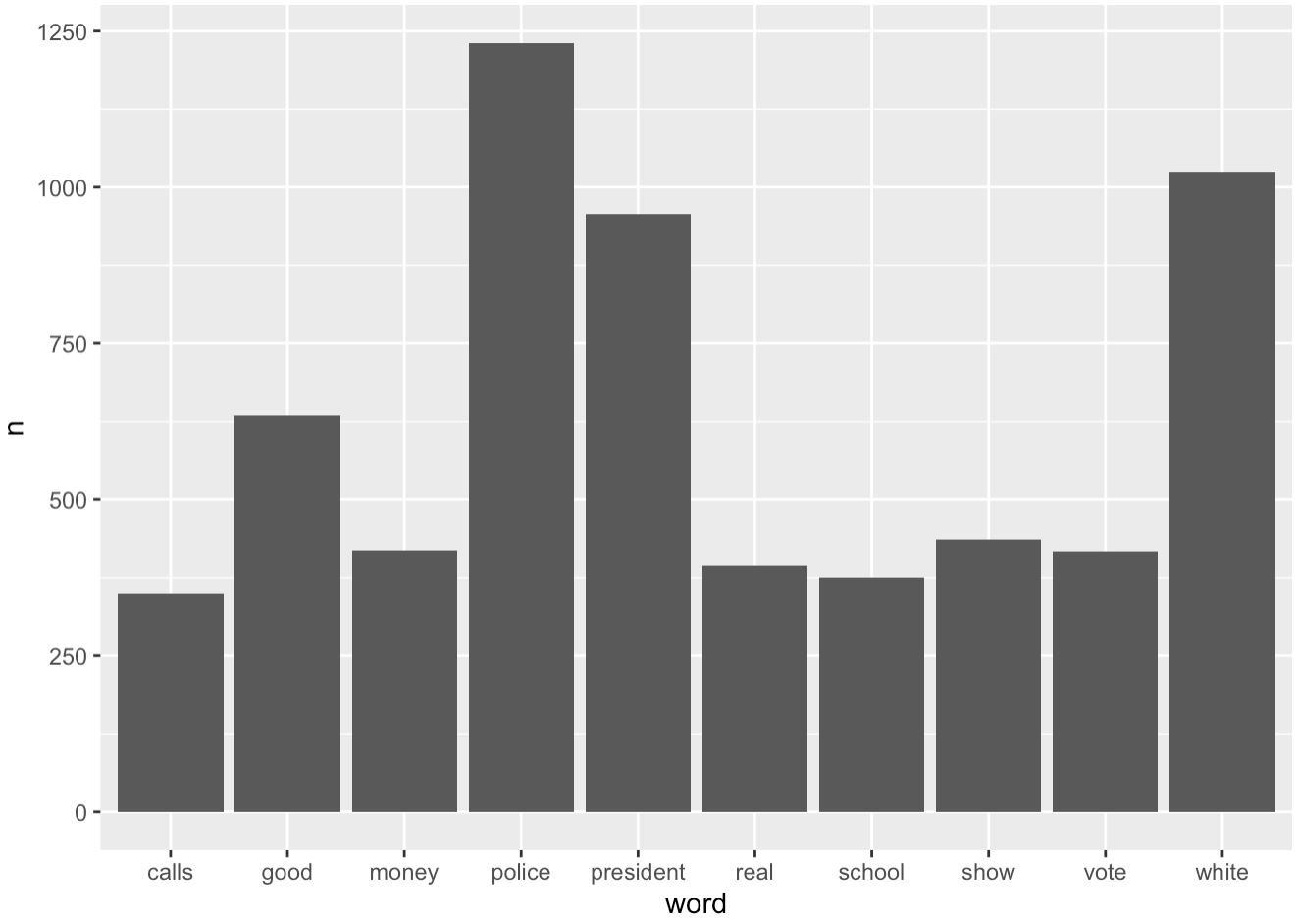

```
nrc_trust<-get_sentiments("nrc") %>%
  filter(sentiment == "trust")

trust <- tweets_words %>%
  inner_join(nrc_trust) %>%
  count(word, sort = TRUE)
```

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

```
top_n(trust, n=10) %>%
          ggplot(., aes(x=word, y=n))+
              geom_bar(stat='identity')
```

```
## Selecting by n
```



Question 3

For the supervised model, I will be using Random Forests. For the predictors, I will mostly be using the tweet information data as well as the topic categorization to predict right vs left trolls.

```
ira_rl<-subset(ira_trimmed, account_category == "LeftTroll" | account_category == "Ri
ghtTroll")
```

The data is split randomly into training and testing. Due to computational limits, the training dataset is significantly smaller.

```
sample_size = nrow(ira_rl)/400
set.seed(24601)
picked = sample(seq_len(nrow(ira_rl)), size = sample_size)
training = ira_rl[picked,]
testing = ira_rl[-picked,]
```

```
fitControl<-trainControl(method = "cv",
                         number = 10,
                         summaryFunction = twoClassSummary,
                         classProb = TRUE,
                         savePredictions = TRUE,
                         allowParallel = TRUE)
```

```
train.rf<-train(account_category ~ following + followers + updates + post_type + retw
eet + new_june_2018 + pred_topic,
                method = "rf",
                trControl = fitControl,
                metric = "ROC",
                ntree = 1000,
                data = training)
train.rf
```

```
## Random Forest
##
## 1010 samples
##    7 predictor
##    2 classes: 'LeftTroll', 'RightTroll'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 909, 909, 909, 909, 909, 909, ...
## Resampling results across tuning parameters:
##
##   mtry  ROC        Sens       Spec
##   2     0.9736337  0.8347619  0.9833100
##   5     0.9881083  0.8916667  0.9772028
##   8     0.9856172  0.8946032  0.9757110
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 5.
```

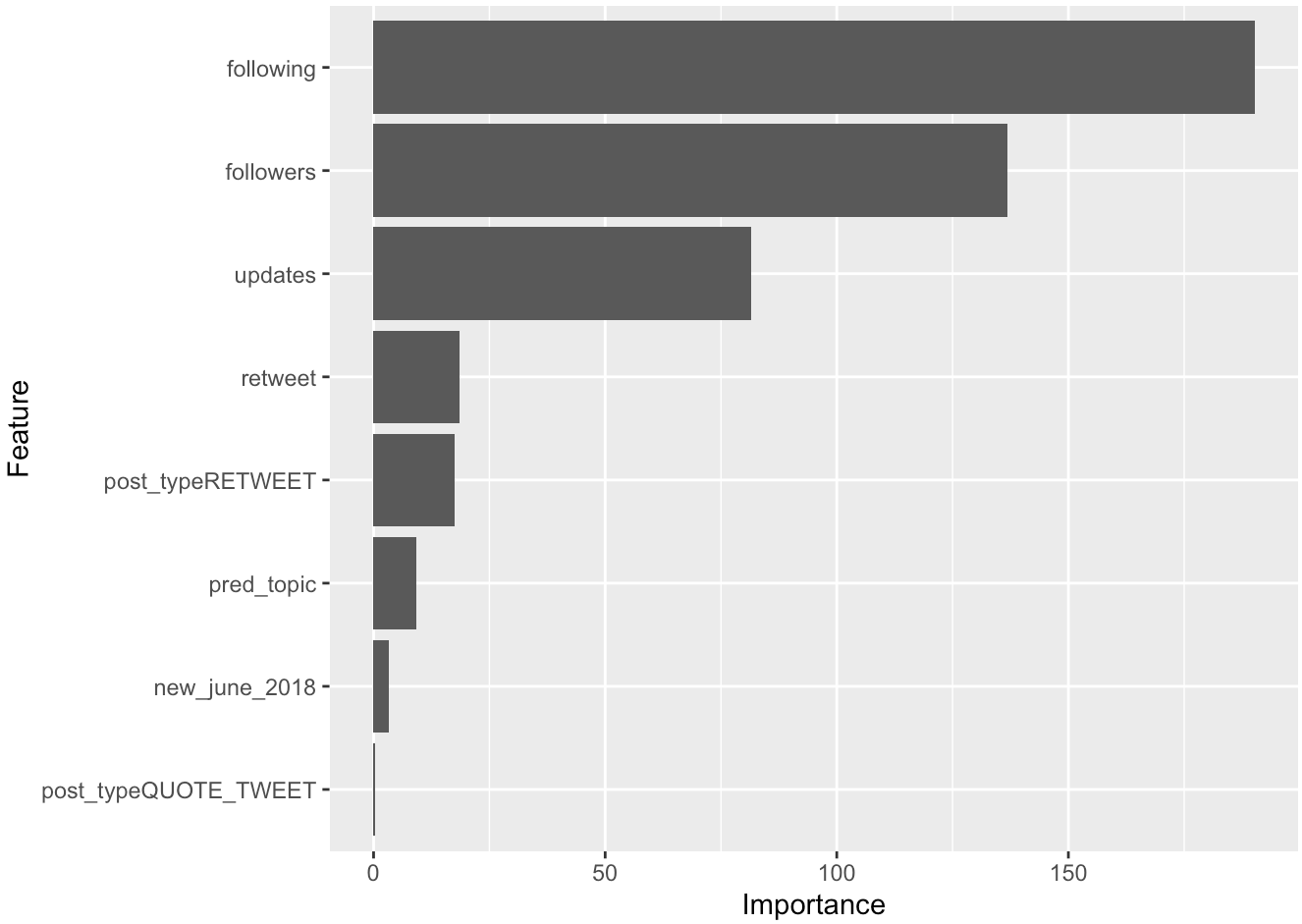
As seen below, the model had fairly good predictive accuracy. There may be a large number of misclassified troll tweets, but given the size of the dataset they make up a small percentage. LeftTroll had slightly higher predictive accuracy than RightTroll.

```
pred.rf<-predict(train.rf, newdata = testing, type = "raw")
confusionMatrix(table(pred.rf, testing$account_category), mode = "everything")
```

```
## Confusion Matrix and Statistics
##
##
## pred.rf      LeftTroll RightTroll
##   LeftTroll     128760      11558
##   RightTroll     14438     248544
##
##                Accuracy : 0.9355
##                  95% CI : (0.9348, 0.9363)
##     No Information Rate : 0.6449
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8586
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8992
##             Specificity : 0.9556
##          Pos Pred Value : 0.9176
##          Neg Pred Value : 0.9451
##               Precision : 0.9176
##                  Recall : 0.8992
##                      F1 : 0.9083
##              Prevalence : 0.3551
##          Detection Rate : 0.3193
##    Detection Prevalence : 0.3479
##       Balanced Accuracy : 0.9274
##
##        'Positive' Class : LeftTroll
##
```

I also wanted to look at a variable importance plot to see what predictors held the most weight. Interestingly, the highest factors were about the level of engagement with tweets. One possible explanation could be that IRA had a deeper focus on impacting right-wing Americans, so they engaged those accounts at a much higher rate.

```
rf.imp<-varImp(train.rf, scale = FALSE)
ggplot(rf.imp)
```


following
followers
updates
retweet
post_typeRETWEET
pred_topic
new_june_2018
post_typeQUOTE_TWEET
Feature
0
50
100
150
Importance