# Action Steps, Twitter Dataset Analysis

Cady Stringer

9/18/2020

## I. Data Cleaning and Preparation

- Assessed number of NAs per column and chose to remove the following columns: airline_sentiment_gold, negativereason_gold, tweet_cord, and tweet_location.
- Changed tweet_created column to only include date to reduce factor levels for modelling.
- Data cleaned separately for each model attempted and completed.

## II. Data Visualization

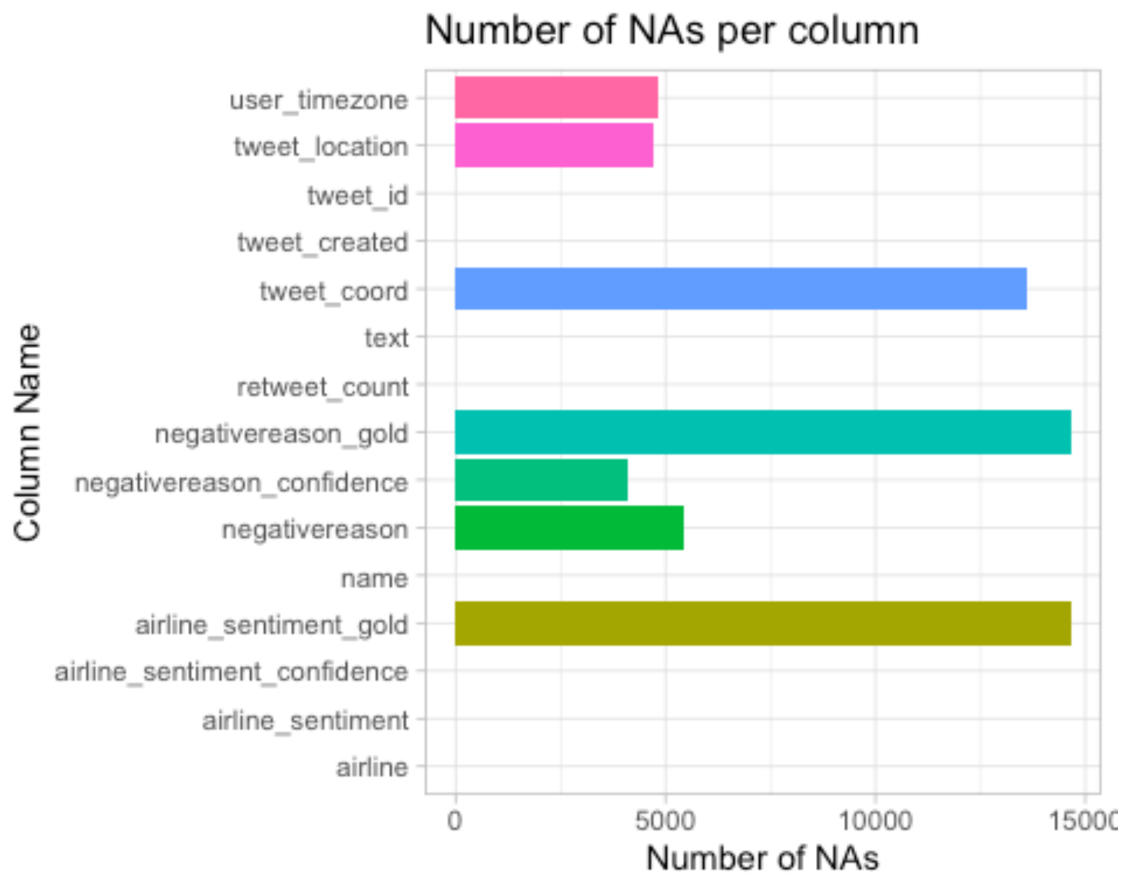
- Created visualizations for NAs per column and sentiment to validate choices during data cleaning.
- Used the cleaned dataset to create an interactive Tableau Dashboard.
- Created additional Tableau visualizations that weren't included in the final Tableau dashboard as not to overcrowd it with repetitive insights.

## III. Models

- Created a moderately successful random forest classifier to predict sentiment, with the goal of identifying the most important variables for predicting sentiment.
- Created a binomial logistic regression model to predict positive or negative sentiment, with a Lasso penalty to identify the most important variables.
- Attempted a multinomial logistic regression to predict negativereason, but there were too many factor levels for meaningful predictions and accuracy was low.
- Attempted a multinomial naive bayes classifier to predict positive, negative, or netural sentiment using up-sampled data, which predicted with only ~35% accuracy.

## IV. Data Undertsanding/Analysis

This is a pretty problematic dataset. The columns airline_sentiment_gold and negativereason_gold are almost all NAs, so I removed both columns entirely. The column tweet_cord has 13621 NAs out of 14640 rows, so I removed that entire column too.



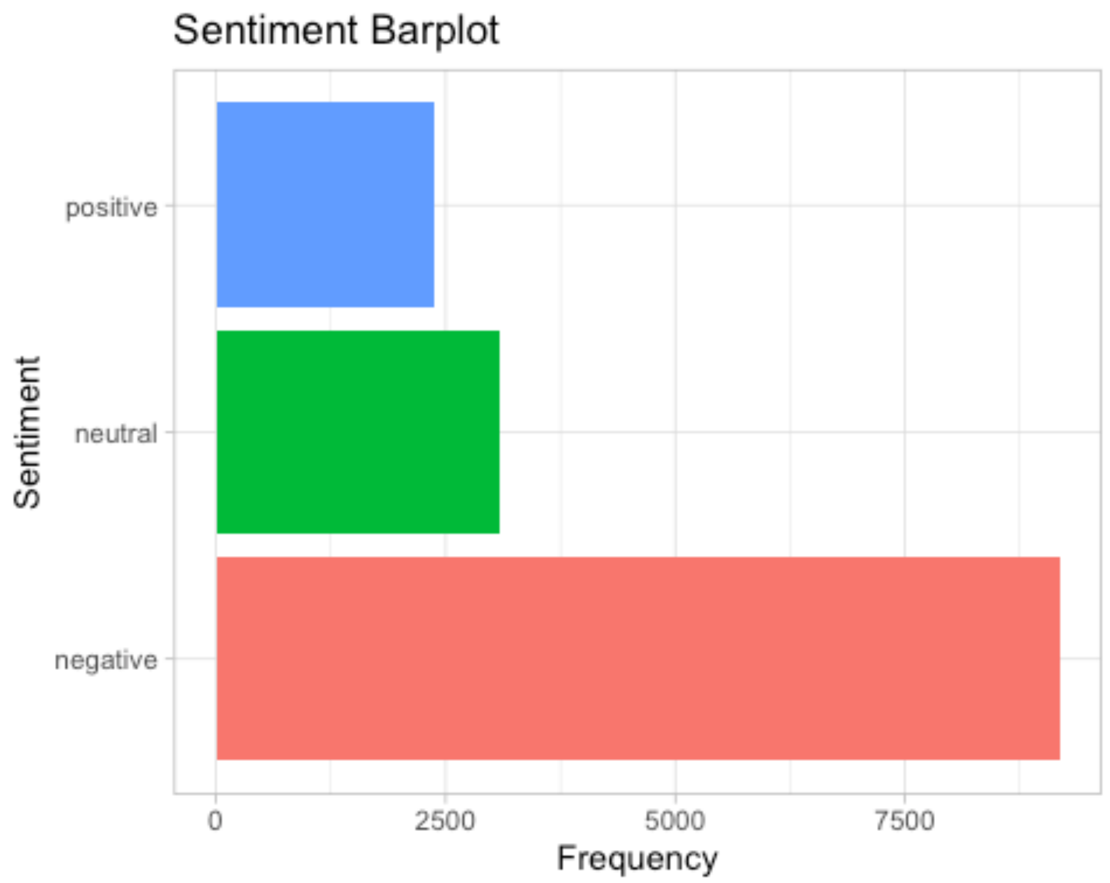
The column tweet_location has 4733 NAs and of those that are filled, many are locations that aren't real places, so I chose to remove this column.

```
tweet <- tweet %>% select(-c(airline_sentiment_gold,
                             negativereason_gold,
                             tweet_coord,
                             tweet_location))
```

The date column has too many factor levels because it lists the exact time each tweet was created. A model would better handle data separated into days, so I used the as.Date function to remove the time.

```
tweet$tweet_created <- as.Date(tweet$tweet_created)
```

The negativereason column has 5462 NAs and the negativereason_confidence column has 4118 NAs, most of which come from rows with positive or neutral sentiment. Dropping NAs would remove all the positive and neutral sentiments, so I chose to manually remove NAs as needed instead.



## Variable Importance: Random Forest Predicton

Data cleaning and preparation:

```
library(randomForest)

library(randomForestExplainer)

rf_tweet <- tweet %>% select(-c(tweet_id,
                                negativereason,
                                negativereason_confidence,
                                name,
                                text
                                )
                             )
view(rf_tweet)
rf_tweet$airline_sentiment <- as.factor(rf_tweet$airline_sentiment)

rf_tweet <- rf_tweet[complete.cases(rf_tweet), ]
rf_tweet <- na.omit(rf_tweet)
```

Class Imbalance: a predictive model would simply choose all negatives, and that wouldn't create a model that has any explanatory power.

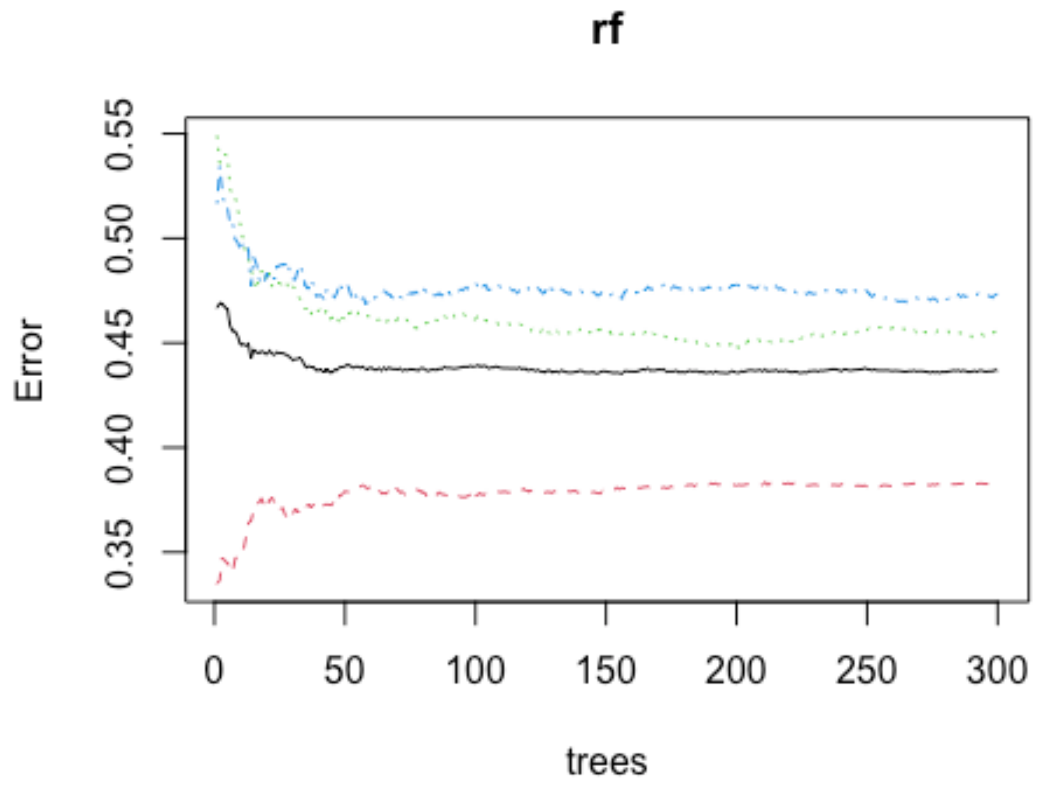
```
sum(rf_tweet$airline_sentiment == "negative")

## [1] 6008

sum(rf_tweet$airline_sentiment == "positive")

## [1] 1684

sum(rf_tweet$airline_sentiment == "neutral")

## [1] 2128
```

To mitigate this problem, I chose to upsample the neutral and positive classes to maintain as much of the negative sentiment data as possible, while balancing the sentiment distribution. To also maintain as much of the data as possible, I chose to cross-validate the results of the model using Out of Bag error instead of a test set.

```
library(caret)

up_tweet <- upSample(x = rf_tweet[, -ncol(rf_tweet)],
                     y = rf_tweet$airline_sentiment)

up_tweet <- up_tweet %>% select(-Class)

rf <- randomForest(airline_sentiment ~.,
                       data = up_tweet,
                       type = classification,
                       ntree = 300,
                       importance = TRUE,
                       localImp = TRUE
                       )
rf

##
## Call:
##  randomForest(formula = airline_sentiment ~ ., data = up_tweet,      type
= classification, ntree = 300, importance = TRUE, localImp = TRUE)
##                Type of random forest: classification
##                      Number of trees: 300
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 43.67%
## Confusion matrix:
##          negative neutral positive class.error
## negative     3707    1063     1238   0.3829893
## neutral      1325    3281     1402   0.4538948
## positive     1619    1224     3165   0.4732024
```

I chose to use 300 trees because it's computationally efficient and also minimizes error. We can see in the plot below that total error levels out around 250.

```
plot(rf)
```



The primary reason I chose to create a random forest model wasn't the predictions themselves, which aren't of much use to airlines looking to improve customer satisfaction, but instead to identify variable importance: which attributes are most essential to predicting a tweet's sentiment, and thus, a customer's satisfaction level? Confidence in the results is the most important, but airline comes in a close second place.

## How can airlines use this information to improve customer satisfaction?

The error rate of this model was about 43%, which is pretty high. We can't let these results alone guide decisions because the model isn't effective enough. The airline_sentiment_confidence variable suggests that the sentiment analysis was more or less confident in predicting a certain sentiment, which isn't useful for airlines to improve service. The importance of the airline variable, however, encourages further exploration of sentiment per airline, and suggests that certain airlines may need to improve sentiment more than others.

## Lasso Logistic Regression

This model predicted positive and negative sentiment with 60% accuracy and identified that retweet count wasn't a useful predictor of sentiment, but confidence, airline, and the date the tweet was created all impacted predictions of sentiment. This further validates the results of the random forest classifier and provides the same insights. Airlines can use this information to decide where to focus time and resources. Running this model with more inputs would create better insights. We should also be wary of such a low accuracy score, because this model was predicting only slightly better than chance.

```r
library(glmnet)

library(glmnetUtils)

train_size <- 0.8
train_idx_2 <- sample(1:nrow(up_tweet), size = train_size*(nrow(up_tweet)))
tweet_train_2 <- up_tweet %>% slice(train_idx_2) %>%
      filter(airline_sentiment !='neutral')
tweet_test_2 <- up_tweet %>% slice(-train_idx_2) %>%
      filter(airline_sentiment !='neutral')

tweet_train_2$airline_sentiment <- as.factor(tweet_train_2$airline_sentiment)

tweet_train_2$airline<- as.factor(tweet_train_2$airline)
tweet_train_2$tweet_created<- as.factor(tweet_train_2$tweet_created)

tweet_train_2 <- tweet_train_2 %>% mutate(airline_sentiment = if_else(
airline_sentiment == "positive", 1, 0))

tweet_train_2$airline <- as.numeric(tweet_train_2$airline)
tweet_train_2$tweet_created <- as.numeric(tweet_train_2$tweet_created)

xs <- as.matrix(tweet_train_2 %>% select(-airline_sentiment))
ys <- as.matrix(tweet_train_2 %>% select(airline_sentiment))

lasso <- cv.glmnet(y = ys,
                   x = xs,
                   family="binomial",
                   alpha = 1)
tweet_test_2$airline_sentiment <- as.factor(tweet_test_2$airline_sentiment)

tweet_test_2$airline<- as.factor(tweet_test_2$airline)

tweet_test_2$tweet_created<- as.factor(tweet_test_2$tweet_created)

tweet_test_2<- tweet_test_2%>% mutate(airline_sentiment =
if_else(airline_sentiment == "positive", 1, 0))

tweet_test_2$airline <- as.numeric(tweet_test_2$airline)
```

```
tweet_test_2$tweet_created <- as.numeric(tweet_test_2$tweet_created)



true <- tweet_test_2$airline_sentiment

tweet_test_2 <- tweet_test_2 %>% select(-airline_sentiment)

lasso_preds <- predict(lasso,as.matrix(tweet_test_2),type = "class")

Accuracy(lasso_preds,true)

## 0.602706
coef(lasso, lasso$lambda.1se)

## 5 x 1 sparse Matrix of class "dgCMatrix"
##                                        1
## (Intercept)                   2.30575111
## airline_sentiment_confidence -1.87620928
## airline                      -0.04619400
## retweet_count                 .
## tweet_created                -0.07918692
```