# Virality of Articles

Mashable is interested in building a model to predict for whether the article goes viral or not. The criteria of "virality" depends on the following constraint:

1. Shares of an article is greater than 1400.

They want to understand the variables that can improve an article's chance of reaching this threshold.

## Libraries and Loading Dataset:

For the analysis, we used the 'online_news.csv' file and the tidyverse, gamlr, and dplyr libraries.

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------------ tidyverse 1.3.0 --

## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.4
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

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

```
library(gamlr)
```

```
## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
library(dplyr)
news = read.csv(params$online_news)
news = na.omit(news)
```

For this problem, we built a KNN model with a k value of 3 and the following variables:

1. num_keywords
2. data_channel_is_entertainment
3. self_reference_avg_sharess
4. global_rate_positive_words
5. weekday_is_saturday

For the first iteration, the approach to apply the regression and threshold second was applied. For this approach, we also did 100 different iterations and averaged the counts for "viral" and "not viral".

# Results:

### Approach 1: Regression then Thresholding | K = 99

After running the knn model with different train/test splits for 100 iterations. We averaged over the counts of "viral" and "not viral". With using the parameters stated above, we were able to get an accuracy for this approach ranging from 48% - 53%.

Below, it shows the table for the confusion matrix, accuracy, true positive rate, and false positive rate.

### Confusion Matrix:

```
##                    Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                 3518              496
## Actual: Viral                     3230              683
```

### Accuracy:

```
## [1] 52.98272
```

### True Positive Rate:

```
## [1] 0.1745464
```

### False Positive Rate:

```
## [1] 0.1235675
```

The Null Model tended to do worse than the KNN model, and ranged in accuracy of 40% - 46%.

### Null Model

```
##                    Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                 4014                0
## Actual: Viral                     3914                0
```

### Accuracy: Null Model

```
## [1] 50.62429
```

### Approach 2: Thresholding then Regression | K = 99

With this approach, we created a column variable 'viral' which converts the 'shares' column of 'online_news.csv' to 1 or 0. This conversion will be based on the case if the number of 'shares' is greater than 1400. Before we do any regression and make any knn models, we simplify the shares from numbers ranging in the thousands to a binary value.

This type of binary model did better than the first approach. There was an average increase of accuracy percentage by around 5-7%. We believe this was the case because it simplifies the guess that the model has to make. Instead of trying to guess a certain number of shares based on the training data, the model can choose 1 or 0. This simpler model provides a better accuracy in both the KNN3 model and the Null Model when compared to Approach 1.

Below, it shows the table for the confusion matrix, accuracy, true positive rate, and false positive rate.

We can see below that this model was overall a better predictor to whether or not an article would become viral.

### Confusion Matrix

```
##                    Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                  2013             1687
## Actual: Viral                      1464             2763
```

### Accuracy:

```
## [1] 60.23458
```

### True Positive Rate:

```
## [1] 0.6536551
```

### False Positive Rate:

```
## [1] 0.4559459
```

The Null Model tended to do worse than the KNN model, and ranged in accuracy of 40% - 43%.

### Null Model

```
##                    Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                  3701                0
## Actual: Viral                      4227                0
```

### Accuracy: Null Model

```
## [1] 46.67676
```

### Using Step-Wise Selection for Feature Selection

For this iteration of Approach 2, in order to find the best feature selection, I implemented the step-wise selection to find the best possible combination for these variables:

We chose 11 random variables to create the first baseline model.

```
# going to try and find best variables for linear model and implement those into the knn model

# baseline medium model with 11 main effects
lm_medium = lm(shares ~ n_tokens_title + n_tokens_content + num_hrefs + num_self_hrefs + num_imgs +
                 num_videos + average_token_length + num_keywords + data_channel_is_lifestyle +
                 data_channel_is_entertainment + data_channel_is_bus + data_channel_is_socmed + data_cha
                 data_channel_is_world, data=news)
```

Next, we used the rest of the variables to include the pair-wise interactions between those 11 variables and the rest:

With this selection, it gave us the following features to include:

```
#select features to include in knn model
getCall(lm_step)
```

```
## lm(formula = shares ~ n_tokens_title + n_tokens_content + num_hrefs +
##     num_self_hrefs + num_imgs + num_videos + average_token_length +
##     num_keywords + data_channel_is_lifestyle + data_channel_is_entertainment +
##     data_channel_is_bus + data_channel_is_socmed + data_channel_is_tech +
##     data_channel_is_world + self_reference_avg_sharess + avg_negative_polarity +
##     self_reference_min_shares + weekday_is_monday + is_weekend +
##     abs_title_sentiment_polarity + self_reference_max_shares +
##     min_positive_polarity + num_hrefs:data_channel_is_tech +
##     num_self_hrefs:num_imgs + num_videos:data_channel_is_bus +
##     data_channel_is_bus:avg_negative_polarity + num_videos:data_channel_is_lifestyle +
##     average_token_length:self_reference_min_shares + n_tokens_title:self_reference_min_shares +
##     n_tokens_content:self_reference_min_shares + num_self_hrefs:self_reference_min_shares +
##     num_imgs:data_channel_is_world + n_tokens_content:num_keywords +
##     num_self_hrefs:average_token_length + avg_negative_polarity:self_reference_min_shares +
##     n_tokens_title:self_reference_avg_sharess + self_reference_min_shares:weekday_is_monday +
##     num_hrefs:self_reference_min_shares + avg_negative_polarity:weekday_is_monday +
##     average_token_length:weekday_is_monday + n_tokens_content:num_videos +
##     n_tokens_content:num_imgs + n_tokens_title:average_token_length +
##     n_tokens_content:average_token_length + data_channel_is_tech:self_reference_min_shares +
##     data_channel_is_lifestyle:self_reference_avg_sharess + num_videos:data_channel_is_tech +
##     num_imgs:data_channel_is_bus + num_keywords:avg_negative_polarity +
##     data_channel_is_socmed:self_reference_min_shares + self_reference_min_shares:is_weekend +
##     n_tokens_title:num_self_hrefs + num_videos:abs_title_sentiment_polarity +
##     num_hrefs:abs_title_sentiment_polarity + average_token_length:data_channel_is_bus +
##     num_hrefs:data_channel_is_socmed + num_self_hrefs:weekday_is_monday +
##     n_tokens_content:weekday_is_monday + num_hrefs:num_self_hrefs +
##     num_hrefs:num_videos + self_reference_avg_sharess:self_reference_max_shares +
##     num_keywords:self_reference_max_shares + num_keywords:self_reference_avg_sharess +
##     n_tokens_content:data_channel_is_lifestyle + data_channel_is_tech:weekday_is_monday +
##     data_channel_is_entertainment:weekday_is_monday + num_self_hrefs:abs_title_sentiment_polarity +
##     num_imgs:data_channel_is_entertainment + n_tokens_content:data_channel_is_tech +
##     n_tokens_content:data_channel_is_socmed + n_tokens_title:avg_negative_polarity +
##     n_tokens_content:min_positive_polarity + data_channel_is_entertainment:min_positive_polarity +
##     self_reference_avg_sharess:min_positive_polarity + data_channel_is_world:min_positive_polarity +
##     num_imgs:min_positive_polarity + num_hrefs:min_positive_polarity +
##     weekday_is_monday:min_positive_polarity + num_keywords:self_reference_min_shares +
##     n_tokens_content:data_channel_is_entertainment + data_channel_is_tech:min_positive_polarity +
##     data_channel_is_lifestyle:min_positive_polarity + average_token_length:data_channel_is_world +
##     num_hrefs:num_keywords + n_tokens_content:is_weekend + is_weekend:min_positive_polarity,
##     data = news)
```

After, we used these variables and pair-wise interactions to train the knn model. This gave us the following numbers:

### Confusion Matrix

```
##                    Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                  2221             1485
## Actual: Viral                      1407             2814
```

### Accuracy:

```
## [1] 63.50107
```

**True Positive Rate:**

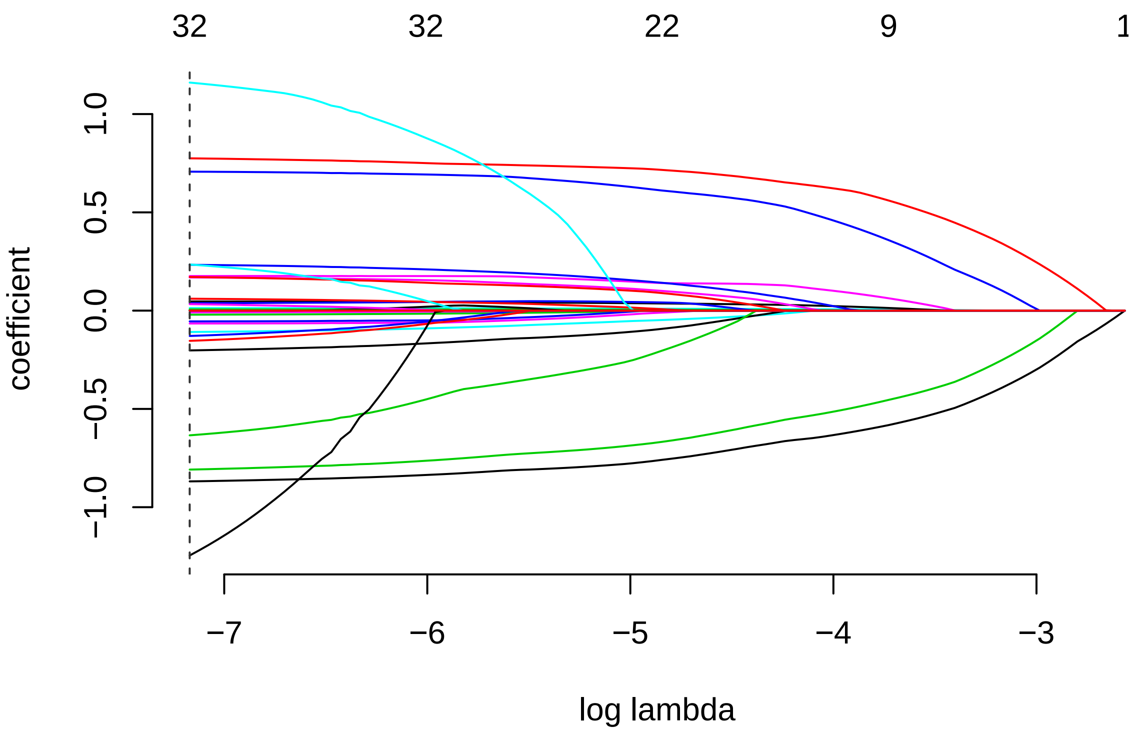
```
## [1] 0.6666667
```

**False Positive Rate:**

```
## [1] 0.4007016
```

### Using Lasso Regression to choose most important feature variables

Lasso Regression uses a shrinkage method to zero the non-important feature variables to include in the machine learning model. Combining this with AICc, we can find the optimal lambda (the tuning factor) in order to create the largest coeffecients for the most important features to include. Although, AIC gets larger as the lambda increases, so the best model has the largest lambda with the lowest AICc value.




These were the following variables that were not zero'd out from lasso regression:

Out of the 36 features, lasso regression outputted 19 important features to use.

These features, in no particular order, are listed below:

1. n_tokens_title
2. n_tokens_content
3. num_self_hrefs
4. average_token_length
5. num_keywords
6. data_channel_is_entertainment
7. data_channel_is_bus
8. data_channel_is_socmed
9. data_channel_is_tech
10. data_channel_is_world
11. self_reference_min_shares

12. weekday_is_monday
13. weekday_is_tuesday
14. weekday_is_friday
15. weekday_is_saturday
16. is_weekend
17. global_rate_positive_words
18. title_subjectivity
19. title_sentiment_polarity

Using only these features I created another model.

#### 19 Important Features

#### Confusion Matrix

```
##                     Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                   2171             1527
## Actual: Viral                       1358             2871
```

#### Accuracy:

```
## [1] 63.58936
```

#### True Positive Rate:

```
## [1] 0.6788839
```

#### False Positive Rate:

```
## [1] 0.4129259
```

#### 10 Most Important Features

Then, I chose 10 largest features in terms of magnitude:

```
x = dplyr::select(news, weekday_is_saturday, global_rate_positive_words, data_channel_is_socmed, is_weel
```

#### Confusion Matrix

```
##                     Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                   2049             1646
## Actual: Viral                       1283             2949
```

#### Accuracy:

```
## [1] 63.03443
```

#### True Positive Rate:

```
## [1] 0.6968336
```

**False Positive Rate:**

```
## [1] 0.4454668
```

**5 Most Important Features**

Then, the largest 5 features:

**Confusion Matrix**

```
##                    Predicted: Not Viral Predicted: Viral
## Actual: Not Viral                  1768             1935
## Actual: Viral                      1357             2867
```

**Accuracy:**

```
## [1] 58.4563
```

**True Positive Rate:**

```
## [1] 0.6787405
```

**False Positive Rate:**

```
## [1] 0.5225493
```

# Conclusion:

In conclusion, the second approach provided a better model to predict the virality of an article and a k value between 61 - 117 seemed to increase the accuracy of the model. In both approaches, the null model gave an accuracy around the lower 42% - 45% mark. The first approach had a lower accuracy on average against multiple second approaches.

On average, the best machine learning model that we were able to get was using the full step-wise pairing or the 19 important variables. These models gave around a 58% - 62% accuracy, and surprisingly creating a simpler model with 10 and 5 important variables lowered the accuracy.

For Mashable, when writing an article, they should consider the top 19 important variables, some of which were weekday_is_saturday, global_rate_positive_words, and data_channel_is_socmed. From these variables, they can conclude that people tend to read positive posts about social media on the weekend. When writing viral articles, they can take these factors into account.