Mashable Dataset: Predicting Article Popularity

Authors: Puneet Auluck, Michele Bradley, Ahsanul Choudhury

Abstract: In this paper, we will attempt to determine proper methods to predict article popularity utilizing the data extracted from K. Fernandes, P. Vinagre, and P. Cortex for a conference on Artificial Intelligence. The methods we will be utilizing are common within Data Mining and Data Science community, and will incorporate the use of the Statistical Language R. We will attempt to build a model that predicts news popularity through utilizing metadata associated with the article.

Key-Words: Data-Mining, Generalized Linear Models, Multivariable Regression, Media

Table of Contents

Introduction

We are going to be analyzing, exploring, and modeling a dataset that contains 39644 URL links uploaded via Mashable.com. The dataset summarizes a heterogeneous set of features pertaining to articles published by Mashable in a two-year period. The goal is to predict the number of shares in social networks (popularity) of an article. Each datapoint contains information regarding a URL of an article (url), the days between the article publication and the dataset acquisition along with many statistics pertaining to the article. Note that the date of acquisition is January 8, 2015 and the data is provided by mashable.com.

This is a topic of interest for mashable.com or any news platform because their news stories are typically labeled as “clickbait”. “Clickbait” is a term used to describe webpages and headlines that are designed to make readers want to click on a hyperlink because they immediately draw your attention. However, once the webpage is clicked on, the content of the article is often quite dubious or silly. The value of the content is subpar or often the “news” featured on the article does not have high quality journalism. The articles have headlines that lure a user in but do not provide high-quality content, like an article on The New York Times might. Therefore, this research would be of interest to many different journalism articles. For example, certain news platforms such as The New York Times might find this interesting because they can learn how to increase the number of users that sign on to their content. News platforms that generate clickbait articles such as Mashable and Buzzfeed may also find this information useful because they can increase revenue through ad content. Since this is their number one source of income it is of interest for them to generate more clicks on their webpage.

The variables associated with the articles are typically “count” data. In addition, there are no missing values in the data-set, and anything that was not known to the researchers was estimated utilizing a Random Forest Classifier and a rolling windows as assessment method.

Our goal is to build multiple linear regression models and count regression models on the target variable shares. In this report, we will be discussing how other researchers performed analysis on this dataset along with all of the methods that we will attempt to generate. This includes all data-manipulation, data-preparation, model building, model selection, and model evaluation techniques.

Literature Review

One paper, Predicting Key Events in the Popularity Evolution of Online Information broke down the main issue regarding article popularity quite well. This paper attempts to utilize hashtags to determine the popularity of an online news article over time. The paper explains that each article has periods of evolution that contains three separate periods for each of the articles: “burst”, “peak”, and “fade”. The “burst” period occurs when the article is first published, and many people view the article. The article starts to spread through shares on social media, and after many people share this content, the article goes into “peak” mode, in which the article

is at its highest viewing/sharing time. Shortly after, the article begins to “fade” and is not shared as often as it used to. The challenges of identifying where these occur is due to the high variation and correlation between these features. The advantage of utilizing time as a dimension for understanding popularity is that we see how an article develops in relation to the amount of shares over time, so we can begin to determine how it morphs. The researchers even talk about the article as if it were an organism going through an evolution period as it is born, reaches its peak, and later dies.

Our dataset does not incorporate this element of time but it is still an interesting way of viewing news articles in particular. They are transformed into a being that can change and alter, as opposed to being stagnant. Our dataset contains metadata associated with an article, keeping it stagnant and only incorporating a timedelta variable that explains the difference in time from publishing to data acquisition. However, there are severe limitations to incorporating a timedelta element and utilizing only metadata metrics to understand the popularity of a news article. Social media develops over time, and that is something to keep in mind as we continue analysis of this dataset.

Another researcher attempted to predict the popularity of TED Talks, a platform that generates conference series via videos as a way to “democratise knowledge” and share interesting, new research for many people. What they found with their research, which utilizes neural networks and a subfield of Machine Learning called latent Dirichlet allocation (LDA), which is a popular probabilistic methods for topic modeling. Their conclusions with this model was that positive articles such as entertainment or motivation tend to have the greatest number of views as opposed to talks about negative topics such as disease, global issues, or war. This is an interesting component to look out for while we pursue our research. One set of variables are: “data_channel_is_lifestyle”, “data_channel_is_entertainment”, “data_channel_is_bus”, “data_channel_is_socmed”, “data_channel_is_tech”, and “data_channel_is_world”. From this research we would expect this variable to have a high influence and that variables such as “lifestyle” and “entertainment” to have the most amount of views, while other channels like “world” would have the fewest.

Methodology

We generated all of our analysis on our dataset using the Open Source Statistics Package, R. We started our analysis by generating basic statistics on all of our variables to better understand the distributions we were working on to determine the best model to choose. For example, we generated means, medians, standard deviations, standard errors, min, max, range, and a kurtosis value. The kurtosis value in particular is useful because it helps describe the measure of peakedness/flatness in our distribution. For example, if the kurtosis number is <0, we say it relatively flat, =0 means there is an approximately normal distribution, while >0 means that there is high peaks. Every variable had a kurtosis >0, which means there are high peaks. These include n_unique_tokens (rate of unique words in the content), n_non_stop_words (rate of non-stop words in the content), n_non_stop_unique_tokens (rate of

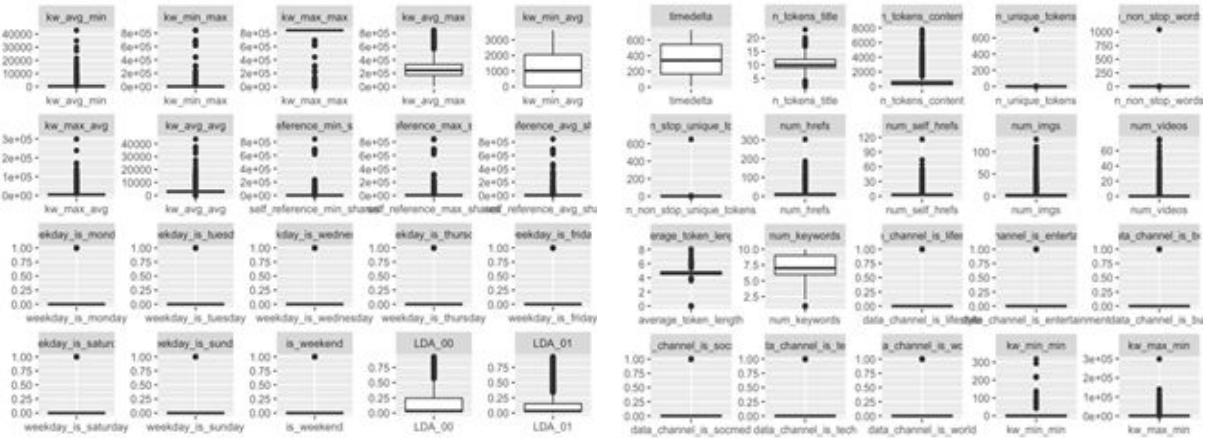
unique non-stop words in the content), kw_min_max (best keyword (min. shares)), kw_avg_min (worst keyword avg. shares), as well as our response variable, shares (number of social media shares). Therefore there is likely a lot of variability within our dataset.

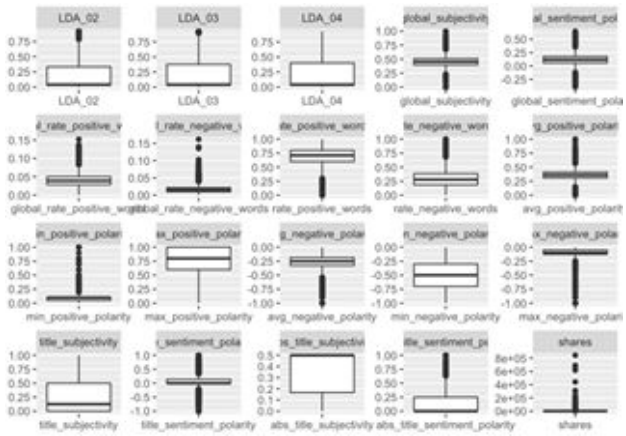
In addition, we analyzed the correlation between the variables in our dataset to ensure that no variables would cause issues when generating a multiple linear regression model. Collinearity is an issue that can affect results because there is overlap in the variables being utilized.

Experimentation and Results

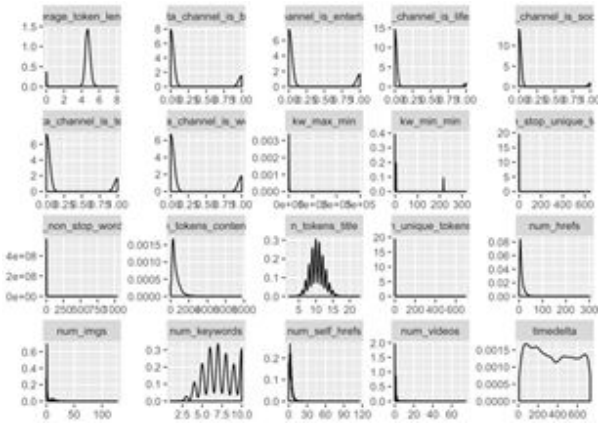
Data Exploration

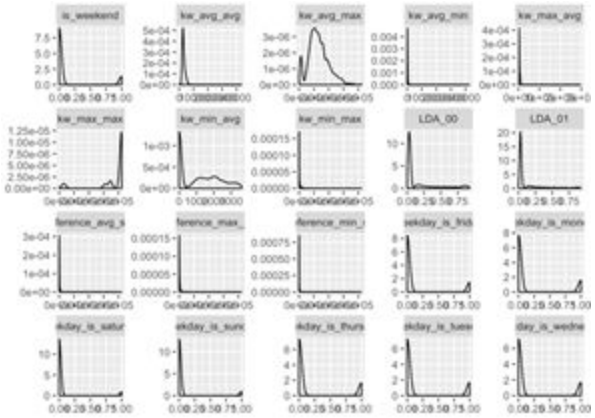
We looked into boxplots that helped us visualize our variables better. We noticed that many of the datasets contained outliers that are incredibly high, so most of our variables were skewed to the right. These included the variables such as n_non_unique_tokens, n_non_stop_unique_tokens, n_non_stop_words. Other variables that appeared to be skewed significantly to the right with one outlier were our binary variables, which indicated what section of the website the article was from or what day of the week the article was posted. Our other variables, such as n_tokens_title, average_token_length, global_subjectivity, global_sentiment_polarity, global_rate_positive_words, and global_rate_negative_wods appeared normally distributed.

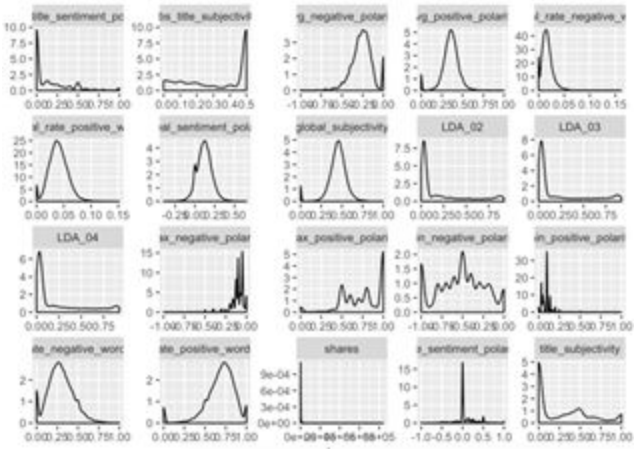




We generated density plots to further understand the distribution of our data. We noticed that many of the variables had a bivariate distribution while many others had a normal count distribution. There were other distributions too, such as, the variables kw_max_max (est keyword (max. shares)) and kw_max_avg (best keyword (avg. shares)) had zero-inflated variables. Most of the other variables appeared relatively normally distributed and did not require much manipulation. The variables in need of preparation included those that are slightly skewed to the left or to the right. These would be manipulated in the data preparation section. The count, bivariate, and zero-inflated variables did not need to be manipulated at all.

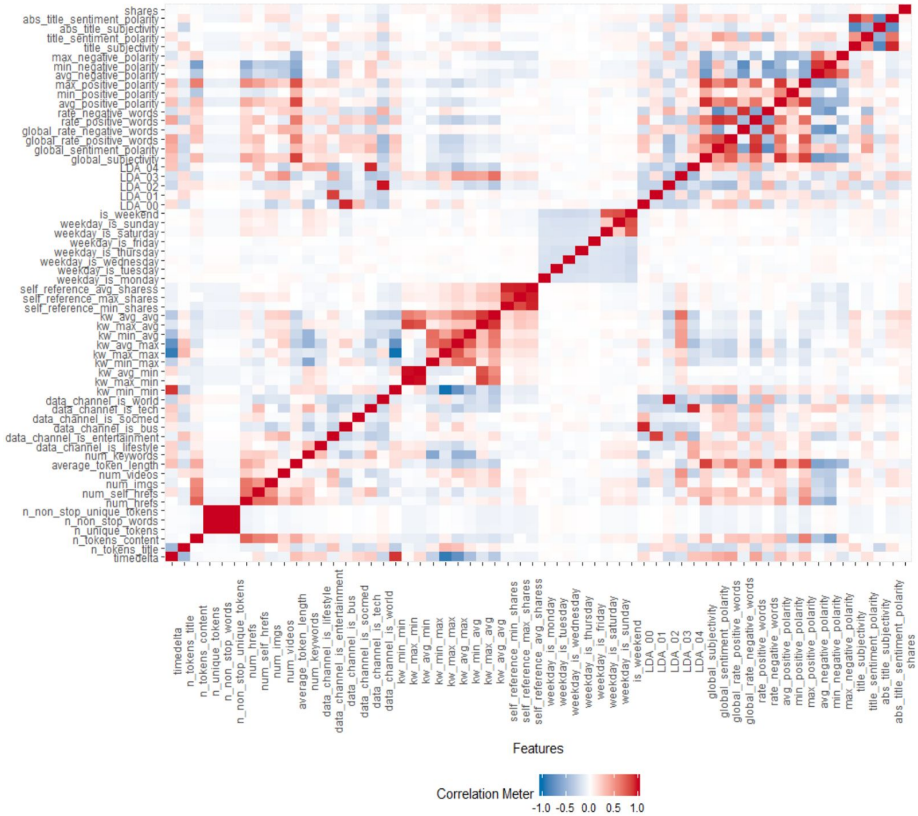




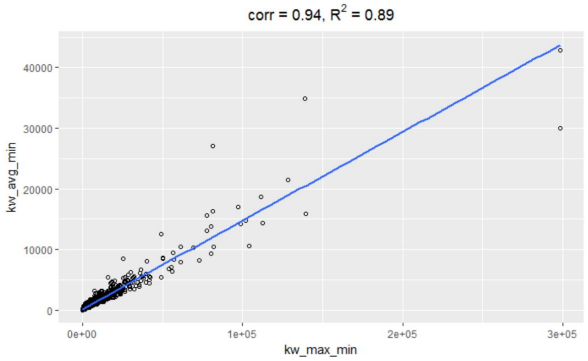


Multicollinearity

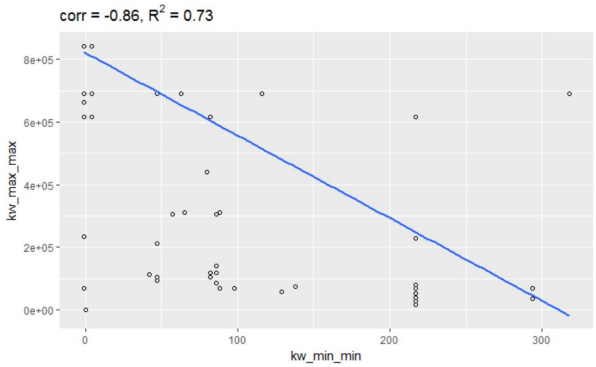
We looked at correlation between the variables to see if there was any multicollinearity among the variables in our dataset. First, we created a correlation matrix heatmap to visualize the correlation matrix. It showed that there were some highly correlated variables in our dataset.



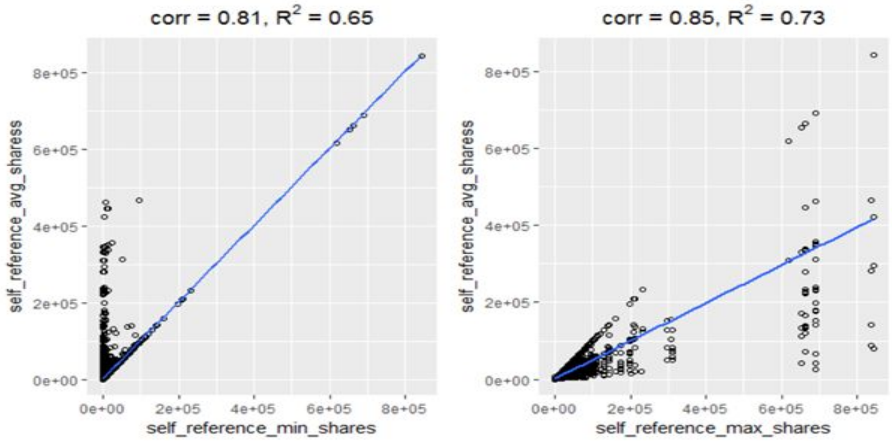
There was a strong positive correlation between kw_avg_min (average share of worst keywords) with kw_max_min (maximum shares of worst keywords). The correlation coefficient was 0.94 and when running regression model, it returned R^2=0.89. The statistic values and the plot indicated that these two variables were correlated. Our strategy was to keep one of the variables and drop the other.



There was negative correlation between kw_min_min (minimum shares of worst keywords) and kw_max_max (maximum shares of best keywords) with correlation coefficient value at -0.86. The summary from regression model between these two predictors resulted in R^2=0.73. The negative relation can be seen on the scatterplot as well.



The predictor variable self_reference_avg_sharess (avg. shares of referenced articles) had high positive correlation with both self_reference_min_shares (min shares of referenced articles) and self_reference_max_shares (max shares of referenced articles) with correlation coefficients 0.81 and 0.85, respectively. When running a model of average shared references against min and max, they produced R^2 value of 0.65 and 0.73. The positive relation was also visible through the scatterplots below. Since there was high correlation, we would keep the average shares and drop the min and max share variables.



All LDA predictor variables, except for LDA_03, had high correlation with data channel variables of different types. The variable LDA_02 had the highest correlation with data_channel_is_world, followed by LDA_00 with data_channel_is_bus. We had kept the data channel variables and dropped LDA variables. It was also more intuitive to interpret data channel than LDA predictors.

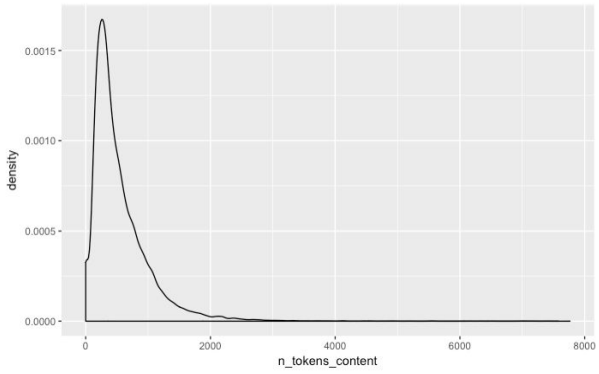
We did not find any independent variables highly correlated with the response variable. The highest correlated variable is kw_avg_avg (average of average keywords) with coefficient of 0.11 and most of the others had values less than 0.05.

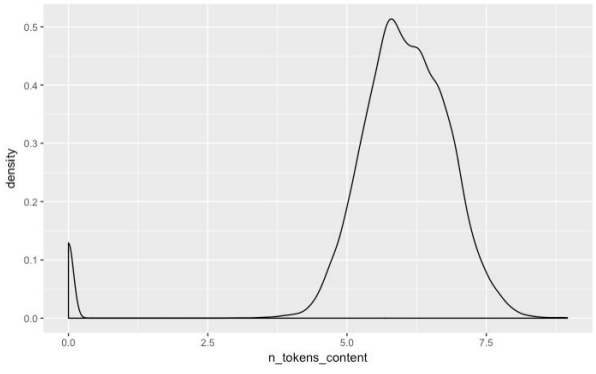
Data Preparation

As mentioned earlier, all of the variables were complete and did not require imputation. In addition, many of the variables had either a bivariate, count, or zero-inflated distributions and therefore did not require much data preparation. The variables in need of preparation included those that were slightly skewed to the left or to the right.

The data channel and day of the week variables were so spread out, it was advantageous to generate two count variables that indicated day_of_week and data_channel as count variables instead of bivariate ones.

Our variable, n_tokens_content was skewed to the right. To normalize this variable, we took the log + 1 of the variable. The new five number summary is shown below the two density plots. The density plot on the left is our original data distribution while the density plot on the right shows our new variable distribution. The variable, n_tokens_content stood for the rate of unique words in the article.

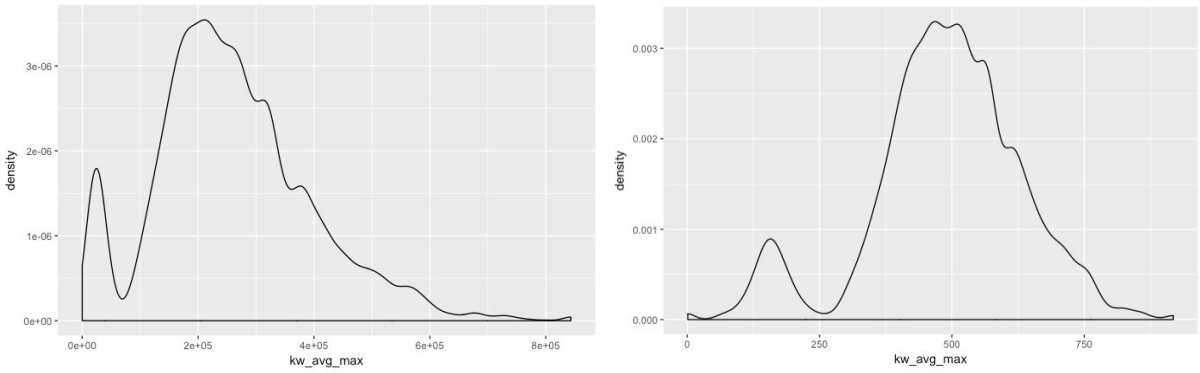





```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.000   5.509   6.019   5.891   6.575   8.957
```

Our variable, kw_avg_max was skewed to the right. To normalize this variable, we took the square root of the variable. The new five number summary is shown below the two density plots. The density plot on the left is our original data distribution while the density plot on the

right shows our new variable distribution. The variable, kw_avg_max stands for a variable that describes an article that utilizes best keywords and min. shares

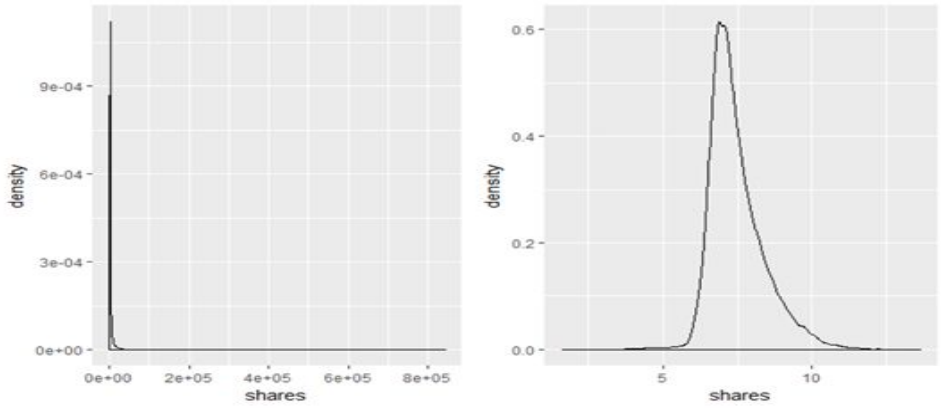


Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
1.0	415.6	495.2	488.3	575.6	918.3

Similarly we also applied square-root transformation to other variables that were skewed to the right and contained many zero values.

We also dropped variables that had strong correlation with other variables. These variables were the LDA levels and min and max values as there were already averages available. We also dropped the absolute values of subjectivity and polarity. Having original values would help us interpret the models better than absolute values where we cannot know if the true values were negative or positive.

As mentioned earlier, our response variable *shares* is heavily skewed to the right. To normalize the data, log transformation was performed on the entire dataset. The plots before and after the transformation are shown below with the summary of the transformed variable. The density plot on the left shows the right skewed distribution and the plot on the right shows symmetric distribution after log transformation.



Models

We generated few regression models to find the right model selection. We looked into multiple linear, poisson and negative binomial regression. The model building was based on the manipulated data as there were variables that do not follow appropriate assumptions. Our response variable is also transformed with use of logs.

➢ *Multiple Linear Regression: Forward Selection*

The forward selection starts with an empty data frame and then variables are added one at a time. It bases the selection using t-value with significance of 0.05%. Initially, this method returned with thirty-plus predictors where some variables had very high p-value and were not significant. For ease of interpretation and predictions, we removed some of the insignificant variables to twenty-four variables.

The adjusted $\mathbf{R^2}$ value was 0.12 and **p-value** was significantly smaller than 0.05. The **AIC** value was returned with **758884.89**. Most independent variables were significant to the model with low p-value, however there were couple that did not appear as substantial and had p-value greater than 0.05. The news channels other than Lifestyle indicated high correlation with number of news shares. The Entertainment and World news channels had negative impact on shares while Social Media channels were shared the most. As for the days, the weekends generated more news shares than the weekdays. The average keywords shared, kw_avg_avg_sqrt, also had high correlation to the response variable. The two variables that did not appear to be highly significant were news_typeLifstyle and news_dayMonday.

➢ *Multiple Linear Regression: Backward Selection*

In linear regression, backward elimination starts with full model and eliminates variables that are not considered significant. We noticed that this model produced slightly lower AIC value (75859.42) than forward selection method. The $\mathbf{R^2}$ value of 0.125 is also higher.

There were total of twenty-eight predictors selected. Similar to forward selection method, variables news_typeLifeStyle and news_dayMonday were insignificant to the model. Also, the weekends produced more shares while weekdays brought the counts down as they had negative coefficients. Along with these, there were other variables that had negative correlation with the response variable: num_self_hrefs, average_toen_length and polarity variables. However, there were a lot more variables that increased the number of shares such as title_subjectivity, timedelta, num_hrefs and num_imgs. The variables num_hrefs and num_imgs were not included within the forward selection method and we believe they should be as someone reading an article would be attracted more towards images and videos before reading an article further. It appeared that the backward elimination provided a better model especially with the choice of variables and its statistic return values for AIC and $\mathbf{R^2}$ values.

➢ *Multiple Tweedie Regression*

The tweedie generalized linear regression model is a form of generalized linear regression that is best used during the presence of zero-inflated variables. In our model, note that the distribution for the response variable, shares, is zero-inflated. This is because there are many articles that do not contain any shares, therefore a big bulk of our values contain the number zero. Therefore, unlike our other regression models, there is no need to take the log of our response variable, shares.

Tweedie distributions are a subset of Exponential Dispersion Models (EDMs). These types of models have two parameter distributions form the linear exponential family that also have a dispersion parameter f. A random variable Y will follow a Tweedie distribution if var(Y) = f * V(m) where m is the mean of the distribution, f is the dispersion parameter, V is a function describing the mean/variance of a relationship, and p is a constant such that V(m) = m^p (Rickert). There are many distributions that can be modeled in this fashion: a p = 0 would be normally distributed, p = 1 would be poisson, while p = 2 would be gamma. A Tweedie model however, has falls between the interval [1, 2] (Rickert). In general, many statisticians performing analysis on zero-inflated distributions highly encourage the use of Tweedie distributions because they often aptly describe the data and generate good regression models (Mullah).

We initially generated two tweedie multivariable generalized linear regression models: one using all of the variables deemed to not have collinearity with other variables, along will all of the variables manipulated to be normally distributed, along with a model that utilizes backward selection procedures using a critical p-value of .1. Interestingly when we compare the AIC values of the two models, even though the AIC criteria is meant to punish using too many variables, the model with the lower AIC contained all of the variables in our dataset. Therefore, we generated a third model using a critical p-value of .2, which generated a model with a slightly lower AIC value than using all variables.

Since we do not have much knowledge of how these variables affect the amount of shares, we can use the following box plots that breakdown the relationship between similar variables to determine if some of our variables are generating output in the correct direction. Below you will see the relationship between shares and the day of the week. Notice that shares are the highest when an article is published on Sunday as opposed to any other day. Therefore, notice in our model that most variables state less shares on a Monday, Tuesday, Wednesday, Thursday, and Friday. This makes sense. In addition, notice that the most popular data channel is social media, which is indicated correctly in the tweedie generalized linear regression model, while other data channels have less amount of shares. This also makes sense. From our literature review, we can also ascertain that articles that are more positive likely have a higher amount of shares. Notice that avg_negative_polarity decreases the amount of shares, which is a good sign. However, global_rate_positive_words_sqrt also appears to reduce the amount of shares, which is not a good sign.

➢ ***Negative Binomial Regression***

Since our data is possibly overdispersed we built negative binomial model next. We built two negative binomial models; first, a full model using all the predictors in our prepared dataset and second, a reduced model by keeping only the significant predictors. For the second model we have used 21 predictors which deemed significant from our previous models.

The AIC value returned from the first model (full model) was 117850.10 and from the second model the AIC value was returned with 117838.40. As expected the reduced model yielded a slightly lower AIC value then the full model. The log-likelihood value for full model of -58886.06 and log-likelihood value for the reduced model of -58886.19 also indicates the reduced binomial model is a better fit than the full binomial model.

AIC Metrics

		AIC
Multiple Linear Regression	Forward Selection	75884.89
	Backward Elimination	75859.42
Tweedie Regression	Backward Elimination (.2)	177662.00
	Backward Elimination (.1)	177697.00
Negative Binomial Regression	Full Model	117850.10
	Reduced model	117838.40

Model Selection and Predictions

We employed multiple linear, tweedie and negative binomial regressions with two models each to assess the best fit for this data. For multiple linear regression, we discovered that backward elimination method gives better results than forward selection in terms of the variables selected and its lower AIC value.

For tweedie, we generated model with all variables and then second model with selected variables that did not have collinearity with other variables. The model with all variables produced lower AIC value than the second variable, even though, tweedie models tend to bring the AIC down as non-useful variables are dropped.

Similarly, we used negative binomial model first with all variables and then second with selected variables that appeared to be independent. The model with reduced variables resulted in lower AIC value than the first model.

When comparing all six models, we observed that multiple linear regression with backward elimination method had the lowest AIC value. There were some reservations on selecting this model because our response variable is a count and this model may produce negative results that would not be useful to us. However, with the limited knowledge we had on the data variables and the large AIC values of other models, we were inclined to pick this model as our model of choice to predict the amount of shares. Below are the predictions we have made using this model.

10	13	18	23	32	33	38	49	50	57	58	64	66	73	81	83	85	87	94	101	114
492	968	588	809	770	846	857	812	801	691	809	762	834	517	819	901	1311	1597	1121	1989	2250
122	123	124	127	133	135	138	141	152	155	158	168	169	177	181	182	189	193	199	200	201
1185	1247	914	1462	1673	1664	2842	1499	900	1422	1547	1285	767	2012	1042	1691	1134	1498	1250	2604	999
202	208	209	213	216	217	218	221	224	225	226	230	235	237	241	244	245	250	255	256	262
1257	1336	984	894	1418	1278	1153	983	1153	1562	1181	1451	1235	1361	1131	1487	1644	1666	1064	1559	2413
265	268	276	283	284	285	290	292	297	307	308	311	313	314	320	329	335	343	347	351	354
2261	1513	1398	1507	1765	1565	1466	1744	1304	1824	1466	2041	2099	2375	1775	2340	1777	3219	1743	2339	2374
362	365	375	376	382	388	389	392	393	398	403	404	405	411	432	439					
1527	1575	1396	1244	1239	1651	2276	1737	1542	2034	1495	1618	2130	1492	2214	1444					

Discussion and Conclusion

From the time of acquiring news popularity data from Mashable.com, we went through rigorous process to explore, analyze and prep its variables to generate regression models. We transformed skewed data using logs and square roots. While building models, we took into consideration of multicollinearity and zero-inflated variables. We generated six models and assessed their outcome using AIC values to select the best model to predict amount of shared news articles.

In conclusion, we believe that even though we had taken the proper steps before making predictions, more improvements can be made to understand and analyze variables. There are techniques, noticed through literature review, other than regression such as decision trees or supported vector machine methods through classification that can be employed for further improvements.

Appendix

➢ *Descriptive Statistics Table*

	n	mean	median	sd	se	min	max	range	kurtosis
timedelta	29733	355.63	340.00	213.90	1.24	8.00	731.00	723.00	2
n_tokens_title	29733	10.40	10.00	2.11	0.01	2.00	23.00	21.00	3
n_tokens_content	29733	547.19	410.00	473.20	2.74	0.00	7764.00	7764.00	22
n_unique_tokens	29733	0.55	0.54	4.06	0.02	0.00	701.00	701.00	29663
n_non_stop_words	29733	1.01	1.00	6.04	0.04	0.00	1042.00	1042.00	29684
n_non_stop_unique_tokens	29733	0.69	0.69	3.77	0.02	0.00	650.00	650.00	29632
num_hrefs	29733	10.92	8.00	11.53	0.07	0.00	304.00	304.00	43
num_self_hrefs	29733	3.30	3.00	3.80	0.02	0.00	116.00	116.00	61
num_imgs	29733	4.54	1.00	8.33	0.05	0.00	128.00	128.00	28
num_videos	29733	1.24	0.00	4.08	0.02	0.00	75.00	75.00	75
average_token_length	29733	4.55	4.66	0.84	0.00	0.00	8.04	8.04	25
num_keywords	29733	7.22	7.00	1.91	0.01	1.00	10.00	9.00	2
data_channel_is_lifestyle	29733	0.05	0.00	0.22	0.00	0.00	1.00	1.00	17
data_channel_is_entertainment	29733	0.18	0.00	0.38	0.00	0.00	1.00	1.00	4
data_channel_is_bus	29733	0.16	0.00	0.37	0.00	0.00	1.00	1.00	4
data_channel_is_socmed	29733	0.06	0.00	0.23	0.00	0.00	1.00	1.00	15
data_channel_is_tech	29733	0.19	0.00	0.39	0.00	0.00	1.00	1.00	4
data_channel_is_world	29733	0.21	0.00	0.41	0.00	0.00	1.00	1.00	3
kw_min_min	29733	26.22	-1.00	69.74	0.40	-1.00	318.00	319.00	7
kw_max_min	29733	1166.29	658.00	4151.06	24.07	0.00	298400.00	298400.00	2014
kw_avg_min	29733	313.81	235.25	642.03	3.72	-1.00	42827.86	42828.86	1362
kw_min_max	29733	13611.07	1400.00	57906.72	335.82	0.00	843300.00	843300.00	127
kw_max_max	29733	752009.54	843300.00	215137.72	1247.66	0.00	843300.00	843300.00	9
kw_avg_max	29733	259545.34	245175.00	135450.56	785.53	0.00	843300.00	843300.00	4
kw_min_avg	29733	1111.92	1014.50	1137.39	6.60	-1.00	3613.04	3614.04	2
kw_max_avg	29733	5667.78	4357.42	6328.02	36.70	0.00	298400.00	298400.00	509
kw_avg_avg	29733	3134.37	2864.56	1334.15	7.74	0.00	43567.66	43567.66	112
self_reference_min_shares	29733	3930.53	1200.00	19065.85	110.57	0.00	843300.00	843300.00	933
self_reference_max_shares	29733	10324.81	2800.00	40905.21	237.22	0.00	843300.00	843300.00	229
self_reference_avg_sharess	29733	6354.32	2200.00	23642.00	137.11	0.00	843300.00	843300.00	446
weekday_is_monday	29733	0.17	0.00	0.38	0.00	0.00	1.00	1.00	4

weekday_is_tuesday	29733	0.19	0.00	0.39	0.00	0.00	1.00	1.00	4
weekday_is_wednesday	29733	0.19	0.00	0.39	0.00	0.00	1.00	1.00	4
weekday_is_thursday	29733	0.18	0.00	0.39	0.00	0.00	1.00	1.00	4
weekday_is_friday	29733	0.14	0.00	0.35	0.00	0.00	1.00	1.00	5
weekday_is_saturday	29733	0.06	0.00	0.24	0.00	0.00	1.00	1.00	15
weekday_is_sunday	29733	0.07	0.00	0.25	0.00	0.00	1.00	1.00	13
is_weekend	29733	0.13	0.00	0.33	0.00	0.00	1.00	1.00	6
LDA_00	29733	0.19	0.03	0.26	0.00	0.00	0.93	0.93	4
LDA_01	29733	0.14	0.03	0.22	0.00	0.00	0.93	0.93	6
LDA_02	29733	0.22	0.04	0.28	0.00	0.00	0.92	0.92	3
LDA_03	29733	0.22	0.04	0.30	0.00	0.00	0.93	0.93	3
LDA_04	29733	0.23	0.04	0.29	0.00	0.00	0.93	0.93	3
global_subjectivity	29733	0.44	0.45	0.12	0.00	0.00	1.00	1.00	8
global_sentiment_polarity	29733	0.12	0.12	0.10	0.00	-0.39	0.65	1.04	4
global_rate_positive_words	29733	0.04	0.04	0.02	0.00	0.00	0.15	0.15	4
global_rate_negative_words	29733	0.02	0.02	0.01	0.00	0.00	0.16	0.16	9
rate_positive_words	29733	0.68	0.71	0.19	0.00	0.00	1.00	1.00	6

➢ *Correlation Table*

Index	Var1	Var2	value
1	kw_max_max	timedelta	-0.64
2	global_subjectivity	average_token_length	0.60
3	LDA_01	data_channel_is_entertainment	0.60
4	LDA_00	data_channel_is_bus	0.77
5	LDA_04	data_channel_is_tech	0.75
6	LDA_02	data_channel_is_world	0.84
7	kw_max_max	kw_min_min	-0.86
8	kw_avg_min	kw_max_min	0.94
9	kw_max_avg	kw_max_min	0.62
10	kw_avg_avg	kw_max_avg	0.81
11	self_reference_avg_sharess	self_reference_min_shares	0.81
12	self_reference_avg_sharess	self_reference_max_shares	0.85
13	is_weekend	weekday_is_saturday	0.66
14	is_weekend	weekday_is_sunday	0.70
15	avg_positive_polarity	global_subjectivity	0.63
16	rate_positive_words	global_sentiment_polarity	0.73
17	rate_negative_words	global_sentiment_polarity	-0.65
18	rate_positive_words	global_rate_positive_words	0.63
19	rate_negative_words	global_rate_negative_words	0.78
20	max_positive_polarity	avg_positive_polarity	0.70
21	min_negative_polarity	avg_negative_polarity	0.75
22	abs_title_sentiment_polarity	title_subjectivity	0.71

➢ *R Code*

Import Libraries

```
library(dplyr)
library(knitr)
library(moments)
library(kableExtra)
library(reshape2)
library(ggplot2)
library(gridExtra)
```

Read in data

```
news_popularity =
read.csv("https://raw.githubusercontent.com/Michelebradley/data621-mashable/master/OnlineNewsPopularity.csv", as.is = TRUE)
```

Generate Train and Test Datasets

```
smp_size <- floor(0.75 * nrow(news_popularity))
## set the seed to make your partition reproducible
set.seed(123)
train_ind <- sample(seq_len(nrow(news_popularity)), size = smp_size)
train <- news_popularity[train_ind, ]
test <- news_popularity[-train_ind, ]
```

Generate Table of Descriptive Statistics

```
qualitiative_train = select_if(train, is.numeric)
training_summary_df <- do.call(data.frame,
           list(n = sapply(qualitiative_train, function(x) length(x[!is.na(x)])),
                mean = round(sapply(qualitiative_train, function(x) mean(x, na.rm=TRUE)), 2),
                median = round(sapply(qualitiative_train, function(x) median(x, na.rm=TRUE)),
2),
                sd = round(sapply(qualitiative_train, function(x) sd(x, na.rm=TRUE)), 2),
                se = round(sapply(qualitiative_train, function(x)
sd(x,na.rm=TRUE)/(sqrt(length(x[!is.na(x)])))), 2),
                min = round(sapply(qualitiative_train, function(x) min(x, na.rm=TRUE)), 2),
                max = round(sapply(qualitiative_train, function(x) max(x, na.rm=TRUE)), 2),
                range = round(sapply(qualitiative_train, function(x) max(x, na.rm=TRUE) -
min(x, na.rm=TRUE)), 2),
                kurtosis = round(sapply(qualitiative_train, function(x) kurtosis(x,
na.rm=TRUE))), 2))
kable(training_summary_df, format = "html") %>%
  kable_styling(c("striped", "bordered"))
```

Generate Boxplots

```
twenty <- qualitiative_train[,1:20]
fourty <- qualitiative_train[,21:40]
sixty <- qualitiative_train[,41:60]
meltData <- melt(twenty)
p <- ggplot(meltData, aes(factor(variable), value))
p + geom_boxplot() + facet_wrap(~variable, scale="free")
meltData <- melt(fourty)
p <- ggplot(meltData, aes(factor(variable), value))
p + geom_boxplot() + facet_wrap(~variable, scale="free")
meltData <- melt(sixty)
p <- ggplot(meltData, aes(factor(variable), value))
p + geom_boxplot() + facet_wrap(~variable, scale="free")
```

Same Data Boxplots

```
train$news_type <- rep("Lifestyle", nrow(train))
train$news_type[train$data_channel_is_entertainment==1] <- "Entertainment"
train$news_type[train$data_channel_is_socmed==1] <- "Social Media"
train$news_type[train$data_channel_is_tech==1] <- "Tech"
train$news_type[train$data_channel_is_world==1] <- "World"
p1 <- ggplot(data=train, aes(as.factor(news_type), log(shares)))
p1 + geom_boxplot()
train$news_day <- rep("Sunday", nrow(train))
train$news_day[train$weekday_is_monday==1] <- "Monday"
train$news_day[train$weekday_is_tuesday==1] <- "Tuesday"
train$news_day[train$weekday_is_wednesday==1] <- "Wednesday"
train$news_day[train$weekday_is_thursday==1] <- "Thursday"
train$news_day[train$weekday_is_friday==1] <- "Friday"
train$news_day[train$weekday_is_saturday==1] <- "Saturday"
p2 <- ggplot(data=train, aes(as.factor(news_day), log(shares)))
p2 + geom_boxplot()

training_manipulation = train
```

Generate Density Plots

```
density <- twenty %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_density()
density <- fourty %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_density()
density <- sixty %>%
```

```
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_density()
```

Multicollinearity

```
cormat <- round(cor(train[-1]),2)
plot_correlation(cormat)
cormatdiag <- cormat

for(x in c(1:(ncol(cormatdiag)-1))){

  cormatdiag[x,c((x+1):ncol(cormatdiag))] <- 0
}

melted_cormat <- melt(cormatdiag)
melted_cormat <- subset(melted_cormat, abs(value) >=.6)
melted_cormat[melted_cormat==1] <- NA
melted_cormat<-melted_cormat[complete.cases(melted_cormat),]
melted_cormat <- cbind.data.frame(Index=c(1:nrow(melted_cormat)), melted_cormat)
row.names(melted_cormat) <- NULL
#kable(melted_cormat)

kable(melted_cormat, "html")  %>%
  kable_styling(bootstrap_options = "striped", full_width = F, position = "left",  font_size
=10)
```

Scatterplots

```
# KW_AVG_MIN & KW_MAX_MIN
kw_avgmin_maxmin <- lm(kw_avg_min ~ kw_max_min, data=train)
summary(kw_avgmin_maxmin)

ggplot(train, aes(x=train$kw_max_min, y=train$kw_avg_min)) +
    geom_point(shape=1) +    # Use hollow circles
    geom_smooth(method=lm,   # Add linear regression line
                se=FALSE) +
  scale_x_continuous("kw_max_min") +
  scale_y_continuous("kw_avg_min") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle(expression(paste("corr = 0.94", ", ", R^2," = 0.89" ,sep="")))


#KW_MAX_MAX & KW_MIN_MIN
kw_maxmax_minmin <- lm((kw_max_max) ~ kw_min_min, data=train)
summary(kw_maxmax_minmin)
plot(train$kw_min_min,(train$kw_max_max), xlab="kw_min_min", ylab="kw_max_max" )
abline(kw_maxmax_minmin , col="red")
```

```
ggplot2::ggplot(train, aes(x=train$kw_min_min, y=train$kw_max_max)) +
    geom_point(shape=1) +    # Use hollow circles
    geom_smooth(method=lm,   # Add linear regression line
                se=FALSE) +
  scale_x_continuous("kw_min_min") +
  scale_y_continuous("kw_max_max") +
  #theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle(expression(paste("corr = -0.86", ", ", R^2," = 0.73" ,sep="")))

# SELF_REFERENCE_AVG_SHARESS with SELF_REFERENCE_MIN_SHARES & SELF_REFERENCE_MAX_SHARES
self_avg_min <- lm(self_reference_avg_sharess ~ self_reference_min_shares, data=train)
self_avg_max <- lm(self_reference_avg_sharess ~ self_reference_max_shares, data=train)

summary(self_avg_min)
summary(self_avg_max)

par(mfrow=c(1,2))

plot(train$self_reference_min_shares,(train$self_reference_avg_sharess),
xlab="self_reference_min_shares", ylab="self_reference_avg_sharess" )
abline(self_avg_min, col="red")

plot(train$self_reference_max_shares,(train$self_reference_avg_sharess),
xlab="self_reference_max_shares", ylab="self_reference_avg_sharess" )
abline(self_avg_max, col="red")

refplot1 <- ggplot(train, aes(x=train$self_reference_min_shares,
y=train$self_reference_avg_sharess)) +
    geom_point(shape=1) +    # Use hollow circles
    geom_smooth(method=lm,   # Add linear regression line
                se=FALSE) +
  scale_x_continuous("self_reference_min_shares") +
  scale_y_continuous("self_reference_avg_sharess") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle(expression(paste("corr = 0.81", ", ", R^2," = 0.65" ,sep="")))

refplot2 <- ggplot(train, aes(x=train$self_reference_max_shares,
y=train$self_reference_avg_sharess)) +
    geom_point(shape=1) +    # Use hollow circles
    geom_smooth(method=lm,   # Add linear regression line
                se=FALSE) +
  scale_x_continuous("self_reference_max_shares") +
  scale_y_continuous("self_reference_avg_sharess") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle(expression(paste("corr = 0.85", ", ", R^2," = 0.73" ,sep="")))
refplot2

grid.arrange(refplot1, refplot2, ncol=2)
```

```
### LDA's with DATA CHANNELS
dcldplot1 <- ggplot(train, aes(x=factor(train$data_channel_is_bus), y=train$LDA_00)) +
    geom_boxplot() +    # Use hollow circles
  xlab("data_channel_is_bus") +
  ylab("LDA_00") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle("corr = 0.77")

dcldplot2 <- ggplot(train, aes(x=factor(train$data_channel_is_entertainment), y=train$LDA_01))
+
    geom_boxplot() +    # Use hollow circles
  xlab("data_channel_is_entertainment") +
  ylab("LDA_01") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle("corr = 0.60")

dcldplot3 <- ggplot(train, aes(x=factor(train$data_channel_is_world), y=train$LDA_02)) +
    geom_boxplot() +    # Use hollow circles
  xlab("data_channel_is_world") +
  ylab("LDA_02") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle("corr = 0.84")


dcldplot4 <- ggplot(train, aes(x=factor(train$data_channel_is_tech), y=train$LDA_04)) +
    geom_boxplot() +    # Use hollow circles
  xlab("data_channel_is_tech") +
  ylab("LDA_04") +
  theme(plot.title = element_text(hjust = 0.5)) +
  ggtitle("corr = 0.75")

grid.arrange(dcldplot1, dcldplot2,dcldplot3, dcldplot4, ncol=4)
```

Data Preparation

```
training_manipulation$kw_avg_max = sqrt(training_manipulation$kw_avg_max)
training_manipulation$n_tokens_content = log(training_manipulation$n_tokens_content + 1)
density <- train %>%
  ggplot(aes(n_tokens_content)) +
    geom_density()
density
density <- training_manipulation %>%
  ggplot(aes(n_tokens_content)) +
    geom_density()
density
summary(train$n_tokens_content)
summary(training_manipulation$n_tokens_content)

match(0, training_manipulation$self_reference_min_shares)
```

```
training_manipulation$self_reference_min_shares =
sqrt(training_manipulation$self_reference_min_shares)
density1 <- train %>%
  ggplot(aes(self_reference_min_shares)) +
    geom_density()
density2 <- training_manipulation %>%
  ggplot(aes(self_reference_min_shares)) +
    geom_density()
grid.arrange(density1, density2, ncol=2)

#summary(train$self_reference_min_shares)
summary(training_manipulation$self_reference_min_shares)

match(0, training_manipulation$self_reference_max_shares)
training_manipulation$self_reference_max_shares =
sqrt(training_manipulation$self_reference_max_shares)
density1 <- train %>%
  ggplot(aes(self_reference_max_shares)) +
    geom_density()
density2 <- training_manipulation %>%
  ggplot(aes(self_reference_max_shares)) +
    geom_density()
grid.arrange(density1, density2, ncol=2)
#dev.off()
#summary(train$self_reference_max_shares)
summary(training_manipulation$self_reference_max_shares)

match(0, training_manipulation$self_reference_avg_sharess)
training_manipulation$self_reference_avg_sharess =
sqrt(training_manipulation$self_reference_avg_sharess)
density1 <- train %>%
  ggplot(aes(self_reference_avg_sharess)) +
    geom_density()
density2 <- training_manipulation %>%
  ggplot(aes(self_reference_avg_sharess)) +
    geom_density()
grid.arrange(density1, density2, ncol=2)
#dev.off()
#summary(train$self_reference_avg_sharess)
summary(training_manipulation$self_reference_avg_sharess)


# global_rate_positive_words (use)
match(0, training_manipulation$global_rate_positive_words)
training_manipulation$global_rate_positive_words <- train$global_rate_positive_words
training_manipulation$global_rate_positive_words =
sqrt(training_manipulation$global_rate_positive_words)
density1 <- train %>%
  ggplot(aes(global_rate_positive_words)) +
    geom_density()
```

```
density2 <- training_manipulation %>%
  ggplot(aes(global_rate_positive_words)) +
    geom_density()
grid.arrange(density1, density2, ncol=2)


summary(training_manipulation$global_rate_positive_words)

##global_rate_negative_words (use)

match(0, training_manipulation$global_rate_negative_words)
training_manipulation$global_rate_negative_words <- train$global_rate_negative_words
training_manipulation$global_rate_negative_words =
sqrt(training_manipulation$global_rate_negative_words)
density1 <- train %>%
  ggplot(aes(global_rate_negative_words)) +
    geom_density()
density2 <- training_manipulation %>%
  ggplot(aes(global_rate_negative_words)) +
    geom_density()
grid.arrange(density1, density2, ncol=2)
summary(training_manipulation$global_rate_negative_words)

##shares (use)
match(0, training_manipulation$shares)
training_manipulation$shares <- train$shares
training_manipulation$shares = log(training_manipulation$shares)
density1 <- train %>%
  ggplot(aes(shares)) +
    geom_density()
density2 <- training_manipulation %>%
  ggplot(aes(shares)) +
    geom_density()
grid.arrange(density1, density2, ncol=2)
summary(training_manipulation$shares)

## variables dropped within 40-60

drops_in_sixty <- c("LDA_00", "LDA_01", "LDA_02", "LDA_03", "LDA_04",
                    "rate_positive_words","rate_negative_words",
                    "min_positive_polarity","min_negative_polarity",
                    "max_positive_polarity","max_negative_polarity",
                    "abs_title_subjectivity","abs_title_sentiment_polarity")
training_manipulation <- training_manipulation[ , !(names(training_manipulation) %in%
drops_in_sixty)]

drops_extra <- c("data_channel_is_lifestyle", "data_channel_is_entertainment",
"data_channel_is_bus",
                 "data_channel_is_socmed", "data_channel_is_tech",
                  "data_channel_is_world")
```

```
training_manipulation <- training_manipulation[ , !(names(training_manipulation) %in%
drops_extra)]
```

Model Building

```
#### Multiple Linear Regression: Model 1, Forward Selection

owerstep <- lm(shares_log ~ 1, data = training_manipulation)
upperstep <- lm(shares_log ~ . -self_reference_max_shares_sqrt
                -self_reference_min_shares_sqrt
                -num_imgs
                -n_non_stop_words
                -n_non_stop_unique_tokens
                -global_sentiment_polarity
                -n_unique_tokens
                -avg_positive_polarity
                -avg_negative_polarity
                -n_tokens_title
                -global_rate_positive_words_sqrt
                , data = training_manipulation)

forward <- step(lowerstep, scope=list(lower=lowerstep, upper=upperstep), direction =
"forward", k=2, trace=F)
summary(forward)

c("Multiple LR Forward Selection: AIC"=AIC(forward))


#### Multiple Linear Regression: Model 2, Backward Elimination

lowerstep <- lm(shares_log ~1, data = training_manipulation)
upperstep <- lm(shares_log ~ . -n_non_stop_words -n_tokens_content -num_videos
-n_non_stop_unique_tokens -n_unique_tokens, data = training_manipulation)

backward <- step(upperstep, scope=list(lower=lowerstep, upper=upperstep), direction =
"backward", k=2, trace=F)
summary(backward)
c("Multiple LR Backward Elimination: AIC"=AIC(backward))


#### Tweedie Regression Backward Elimination with p-critical value of .2

cpglm(shares ~ timedelta + n_tokens_title + n_tokens_content + n_unique_tokens +
n_non_stop_unique_tokens + num_hrefs + num_self_hrefs + 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_world + kw_avg_min + weekday_is_monday +
weekday_is_tuesday + weekday_is_wednesday + weekday_is_thursday + weekday_is_friday +
global_subjectivity + global_sentiment_polarity + avg_negative_polarity +
title_sentiment_polarity + kw_avg_avg_sqrt + self_reference_min_shares_sqrt +
self_reference_max_shares_sqrt + self_reference_avg_sharess_sqrt +
global_rate_positive_words_sqrt, data = training_manipulation)
```

```
#### Tweedie Regression Backward Elimination with p-critical value of .1

cpglm(shares ~ timedelta + n_tokens_title + n_tokens_content + n_unique_tokens +
n_non_stop_unique_tokens + num_hrefs + num_self_hrefs + average_token_length +
data_channel_is_entertainment + data_channel_is_bus + data_channel_is_world + kw_avg_min +
weekday_is_monday + weekday_is_tuesday + weekday_is_wednesday + weekday_is_thursday +
weekday_is_friday + global_subjectivity + title_sentiment_polarity + kw_avg_avg_sqrt +
self_reference_min_shares_sqrt + self_reference_max_shares_sqrt +
global_rate_positive_words_sqrt, data = training_manipulation)

#### Negative Binomial Full Model

nb1 <- glm.nb(shares_log ~ . , data=training_manipulation, trace = FALSE)
# summary(nb1)

#### Negative Binomial Reduced

nb2 <- glm.nb(shares_log ~ timedelta + n_tokens_title + n_tokens_content +
    n_non_stop_words + n_non_stop_unique_tokens + num_hrefs +
    num_self_hrefs + num_imgs + average_token_length + num_keywords +
    kw_avg_min + kw_avg_max + global_subjectivity + avg_positive_polarity +
    avg_negative_polarity + title_subjectivity + title_sentiment_polarity +
    news_type + news_day + kw_avg_avg_sqrt + self_reference_min_shares_sqrt +
    self_reference_avg_sharess_sqrt, data = training_manipulation, trace = FALSE)
# summary(nb2)
```

Resources

Hu Y, Hu C, Fu S, Fang M, Xu W. Predicting Key Events in the Popularity Evolution of Online Information. Du W-B, ed. *PLoS ONE*. 2017;12(1):e0168749. doi:10.1371/journal.pone.0168749.

Alvarez, T. Predicting the Popularity of TED Talks. DZone.com 2017

Kumar, Sunil, Advilkar, Shivali , Ben, Chendong, Zaky, Hebatalla and Patel, Manan, Online News Popularity Dataset, slideshare.net, April 2016

Mullah, M, Parveen, N. Tweedie Model for Analyzing Zero-Inflated Continuous Response: An Application to Job Training Data. British Journal of Economics. 2016.

Ren, He and Yang Quan, Predicting and Evaluating the Popularity of Online News, Department of Electrical Engineering, stanford.edu, 2015

Rickert, J. A Note on Tweedie. R-bloggers. October 2014