Online News Popularity Multiple Linear Regression Model

—-------Lydia Zhang, Sarabjit singh, Qingyan Liang

Introduction

More and more people nowadays are paying attention to online articles since news on the internet can be able to broadcast news real-time without having people to wait an extra longer time to know the details.

Purpose

The main purpose of this project is to predict the popularity of new articles ; in other words, based on its 60 features, we want to predict how many times an article will be shared online and be viewed by people.

The reason why we chose this topic is because predicting the popularity of internet news has a wide range of applications nowadays. We want to get a deeper understanding of the people who read online news. Consequently, it allows news organizations to deliver more relevant and compelling content and the company can allocate resources more wisely to prepare stories over their life cycle. Additionally, the prediction of news content is also beneficial for trend forecasting, and in this way can help advertisers propose more profitable monetization techniques, and assist readers filter the huge amount of information quickly and efficiently.

Data Description

The data set

The dataset summarizes a heterogeneous set of features about articles published by Mashable in a period of two years.Our main goal is to predict the number of shares in social networks. The features broadly come from 6 areas including Words, Links, Digital Media, Time, Key words and Natural Language Processing.

figure(1)

Feature	Type (#)
Words	
Number of words in the title	number (1)
Number of words in the article	number (1)
Average word length	number (1)
Rate of non-stop words	ratio (1)
Rate of unique words	ratio (1)
Rate of unique non-stop words	ratio (1)
Links	
Number of links	number (1)
Number of Mashable article links	number (1)
Minimum, average and maximum number of shares of Mashable links	number (3)
Digital Media	
Number of images	number (1)
Number of videos	number (1)
Time	
Day of the week	nominal (1)
Published on a weekend?	bool (1)

Feature	Type (#)
Keywords	
Number of keywords	number (1)
Worst keyword (min./avg./max. shares)	number (3)
Average keyword (min./avg./max. shares)	number (3)
Best keyword (min./avg./max. shares)	number (3)
Article category (Mashable data channel)	nominal (1)
Natural Language Processing	
Closeness to top 5 LDA topics	ratio (5)
Title subjectivity	ratio (1)
Article text subjectivity score and its absolute difference to 0.5	ratio (2)
Title sentiment polarity	ratio (1)
Rate of positive and negative words	ratio (2)
Pos. words rate among non-neutral words	ratio (1)
Neg. words rate among non-neutral words	ratio (1)
Polarity of positive words (min./avg./max.)	ratio (3)
Polarity of negative words (min./avg./max.)	ratio (3)
Article text polarity score and its absolute difference to 0.5	ratio (2)

Target	Type (#)
Number of article Mashable shares	number (1)

Data Cleaning

Before starting our model selection process, we have to check the data quality first. It takes a lot of effort to check the data quality, for in this data set a lot of problems are just hidden under the ice. We find a lot of missing values were set to 0(figure 2), so we must use the background knowledge to tell if they are true value or missing value. For example, it makes no sense if the content of an article has no words in it. We also find this data has some wrong values, for example it is really weird to have a negative value for the min/max shares(figure 3). We deleted these cases with missing or wrong values, which are 1182 cases accounting for 3% of the original data. We also find the definition in the dictionary is not correct, so we recalculate the value accordingly(figure 4).
See appendix 1 to clean our data set for regression analysis.

figure(2)

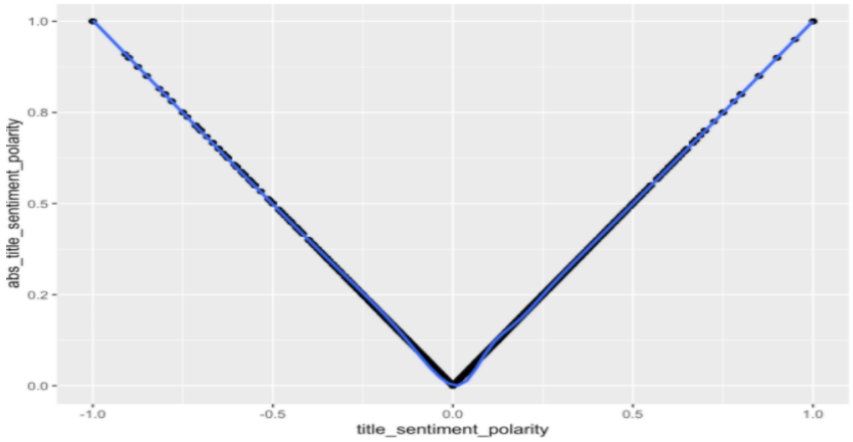
	min <dbl>	max <dbl>	median <dbl>	mean <dbl>
n_tokens_title	2.0	23.0	10.00	10.40
n_tokens_content	0.0	8474.0	409.00	546.51
n_unique_tokens	0.0	701.0	0.54	0.55
n_non_stop_words	0.0	1042.0	1.00	1.00
n_non_stop_unique_tokens	0.0	650.0	0.69	0.69

figure(3)

	min <dbl>	max <dbl>	median <dbl>	mean <dbl>
num_keywords	1.0	10.0	7.00	7.22
kw_min_min	-1.0	377.0	-1.00	26.11
kw_max_min	0.0	298400.0	660.00	1153.95
kw_avg_min	-1.0	42827.9	235.50	312.37
kw_min_max	0.0	843300.0	1400.00	13612.35
kw_max_max	0.0	843300.0	843300.00	752324.07
kw_avg_max	0.0	843300.0	244572.22	259281.94
kw_min_avg	-1.0	3613.0	1023.64	1117.15
kw_max_avg	0.0	298400.0	4355.69	5657.21
kw_avg_avg	0.0	43567.7	2870.07	3135.86

figure(4)

abs_title_sentiment_polarity is title_sentiment_polarity absolute difference to 0(not 0.5), the definition in the document is not correct

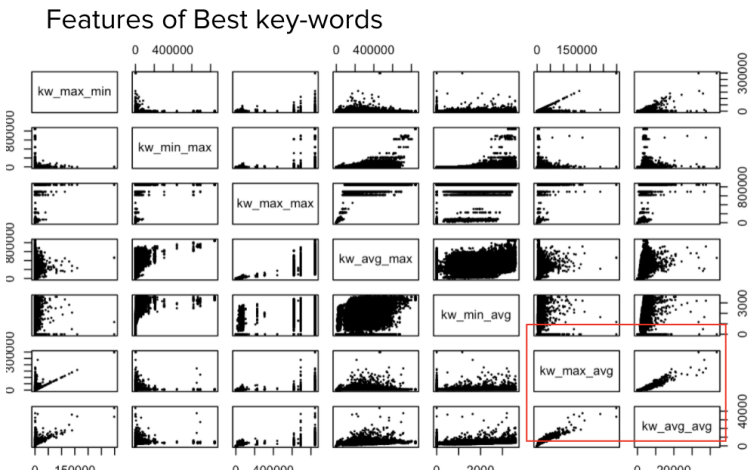


Data Exploration

Since the data set is composed of 6 heterogeneous sets of features, we can explore the relationship among different features in each set. We use the scatter matrix plot to find if there exists a linear relationship and use correlation analysis to define how strong the relationship is. In this way we can remove predictor variables that are either not useful for our analysis or redundant.

Here we choose Key-Words Features for example(figure 5).

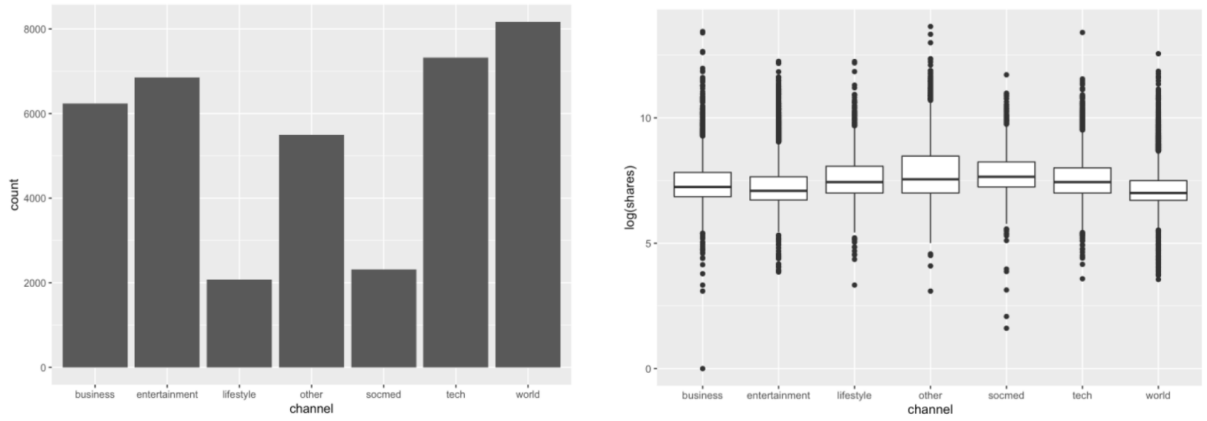
figure(5)



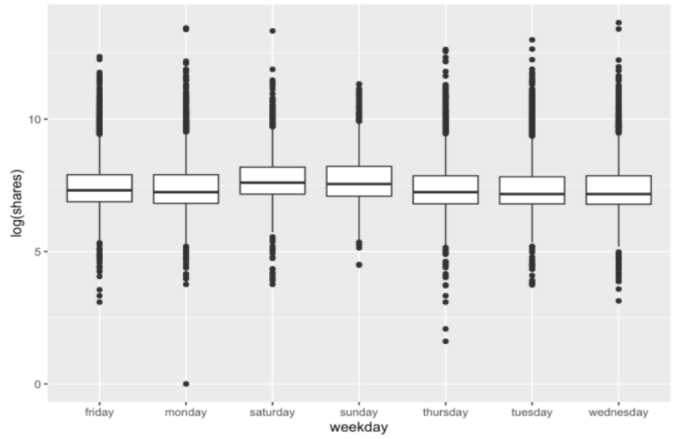
we find kw_max_avg and kw_avg_avg are high correlated (0.8164474),we will choose kw_avg_avg, drop kw_max_avg

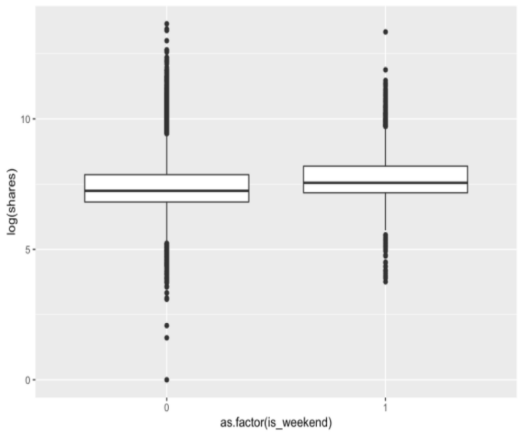
For the categorical variables, we recode the dummy variables to factors and remove the redundant variables and the not useful variables. After we record the 6 dummy channel variables into a factor variable, we find different channels have different performance on log(shares), so we keep the channel predictor(figure 6). There are no differences through Monday to Friday on log(shares), median log(shares) for weekends is obviously higher than not weekends(figure7).
See appendix 2 and appendix 3 for categorical variables.

figure(6)



figure(7)





Summary

After the data dangling process, our final cleaned and trimmed dataset has 35 predictor variables,38642 cases, this is the summary of the predictors we kept (Table 1) .

Table(1)

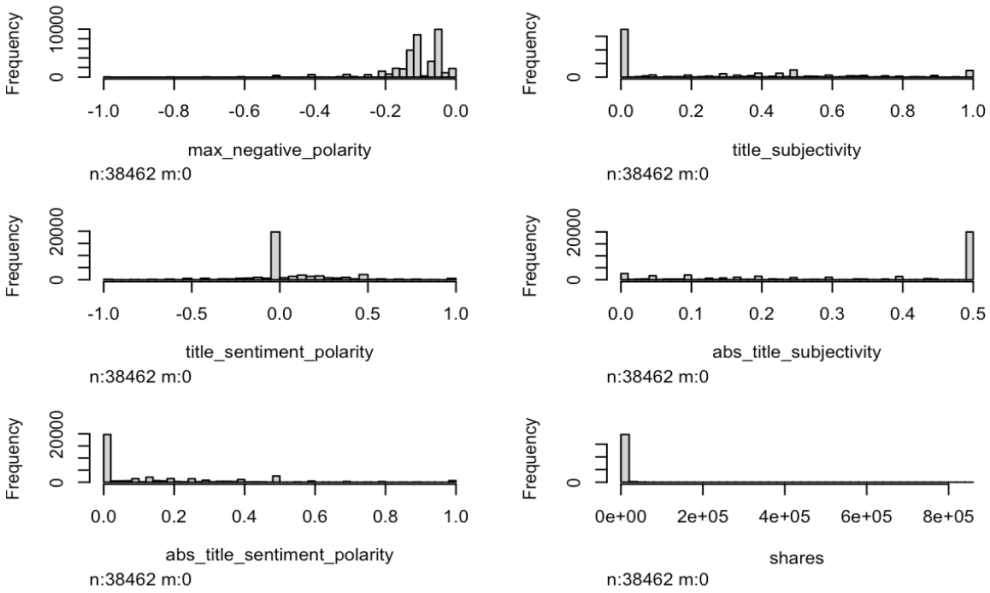
Areas	**Variables kept in the model**	**Dropped in the model**	**Dropping reasons**
Words	n_tokens_title , n_tokens_content n_non_stop_unique_tokens,average_token_length	url, timedelta, n_non_stop_words,n_unique_tokens	Singular or almost singular value/Redundant
Links & Reference	num_hrefs, num_self_hrefs,self_reference_avg_sharess	self_reference_min_shares self_reference_max_shares	Redundant
Digital Media	num_imgs, num_videos	—---	
Time	is_weekend	Weekday_is_monday to weekday_is_sunday	No difference in performance/ Redundant
Keywords	num_keywords, kw_avg_min, kw_min_max , kw_max_max, kw_avg_max, kw_min_avg, kw_avg_avg	kw_min_min, kw_max_min, kw_max_avg	Too many wrong values /Redundant
NLP	LDA-01, LDA-02, LDA-04,LDA-04	LDA-00	Redundant
	Channel	The original 6 channel dummy variables	Recode to 1 factor variable
	Variables from Sentiment analysis models and Subjectivity analysis models: global_subjectivity, global_sentiment_polarity,global_rate_positive_words, global_rate_negative_words and etc	title_subjectivity,title_sentiment_polarity,abs_title_subjectivity,abs_title_sentiment_polarity,rate_negative_words	Recode absolute distance to the original value/ Redundant

Methods and Results

Modeling

We can see most of our variables are obviously left or right screwed(figure 8), transformations would be needed to both y variable and Xs variables.

figure(8)



We randomly divided the data set into two parts: a training set, and a validation or test set. In our model building process, we randomly selected 70% of the data as training and the last 30% as testing data set. See appendix 4.

In this paper, we build 4 different multiple linear regression models. We use shares as our response variable to build our 1st (with no transformation)regression model, we use log(shares) to build our Log-transformed model, and Box-Cox transformed shares(((Y^-0.22)-1)/(-0.22)) to build Box-cox transformed model. For our last model, in order to further reduce the skewness of our data, we transformed some of our predictor variables and also added some poly nominal effects to the model to improve the model performance. Our final model has the best performance compared with the other 3 models(table 2)

Table 2

Model	Multiple R-Squared	Adjusted Multiple R-Squared
No-Trans model	0.0212	0.0204
Log-transformed model	0.118	0.117
Box-cox transformed model	0.122	0.121

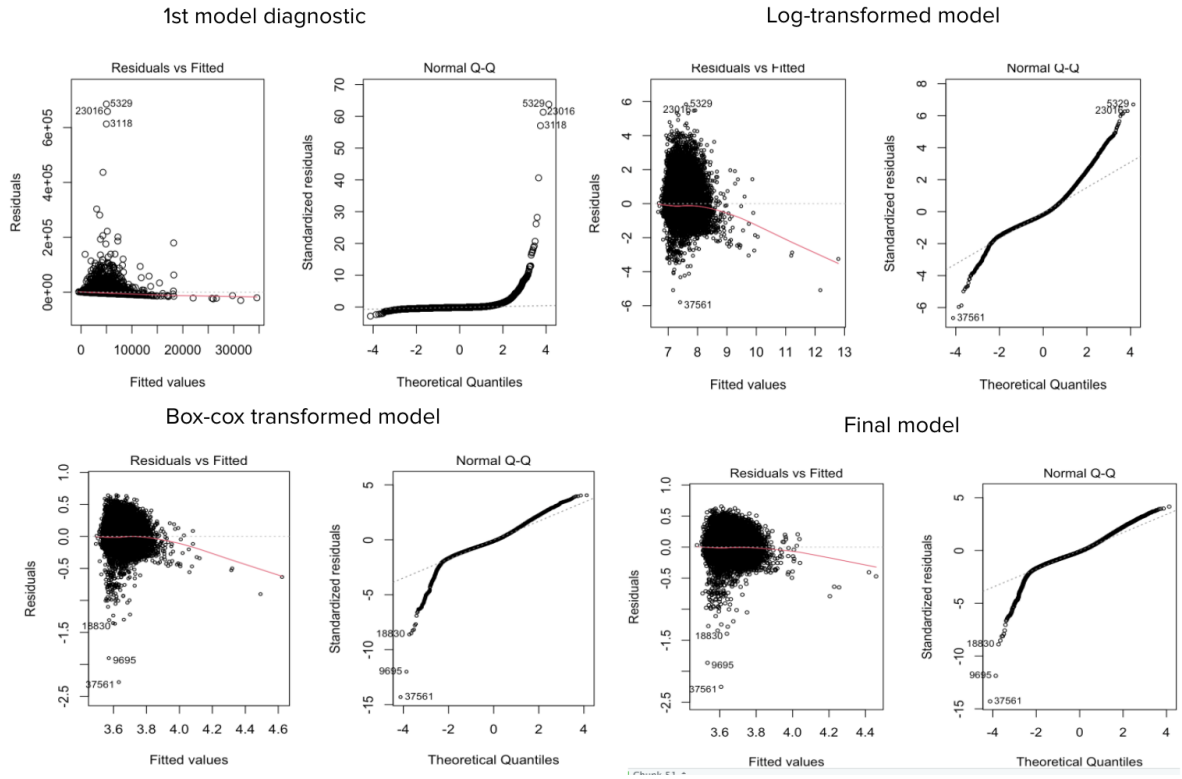
Final model	0.138	0.137

Diagnostic

Model Assumption diagnostic

From the first model to the final model the variance looks more and more constant and the Normal QQ plot looks more and more close to normal distribution(figure 9). We can say the constant variance and the normality assumptions are approximately met in the final model.(code see appendix 5)

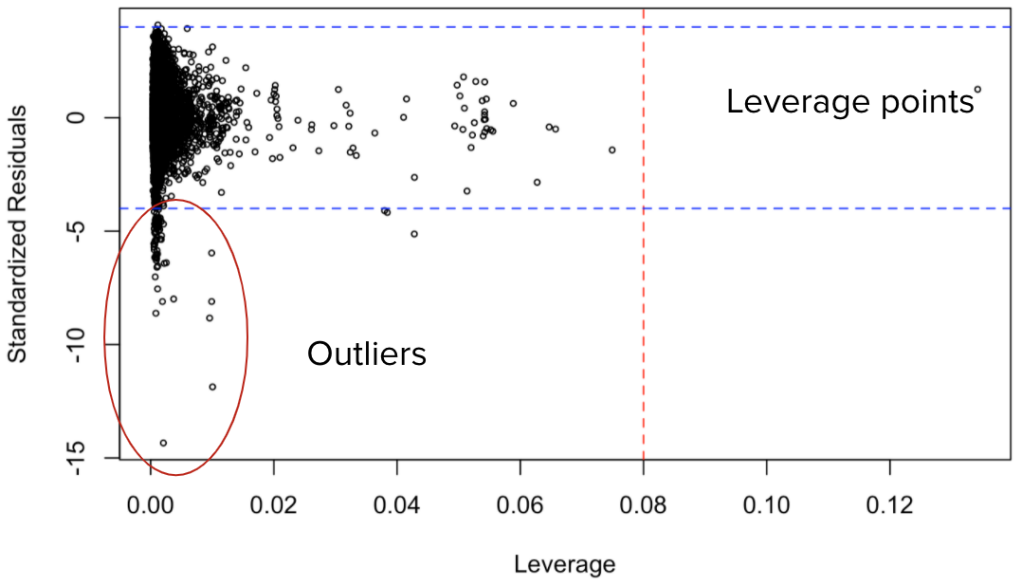
figure(9)



Outliers

Our data set is pretty large (26,923 cases), we take any points which standard residuals fall outside of the interval[-4,4] as outliers(Figure 10).We find 75 outliers account for 0.279% of the total training data.

figure(10)



Multicollinearity Diagnostic

Multicollinearity may cause the signs of the coefficients to be the opposite of what we expect.The standard errors are inflated so the t-tests may fail to reveal significant predictors. To avoid these problems, we need further check if our final model exists the multicollinearity in the predictors variables. As a rule of thumb, we use variance inflation factors that are large and exceed the 5 as a cut-off. We can see there are no severe multicollinearity problems in our final model(figure 11). In this way the signs of the coefficients and the standard errors are reliable.

figure(11)

	GVIF	Df	GVIF^(1/(2*Df))
poly(sqrt(n_tokens_content), 2)	3.29	2	1.35
n_non_stop_unique_tokens	2.42	1	1.56
sqrt(num_hrefs)	1.80	1	1.34
poly(sqrt(num_self_hrefs), 2)	2.09	2	1.20
poly(sqrt(num_imgs), 2)	2.23	2	1.22
poly(sqrt(num_videos), 3)	1.50	3	1.07
average_token_length	1.29	1	1.13
channel	82.00	6	1.44
sqrt(kw_max_min)	1.41	1	1.19
kw_min_max	1.24	1	1.11
kw_avg_max	1.86	1	1.36
kw_avg_avg	2.11	1	1.45
poly(sqrt(self_reference_avg_sharess), 3)	1.75	3	1.10
is_weekend	1.02	1	1.01
LDA_01	4.12	1	2.03
LDA_02	5.83	1	2.41
LDA_03	6.29	1	2.51
LDA_04	5.22	1	2.28
global_subjectivity	1.32	1	1.15
min_positive_polarity	1.28	1	1.13
avg_negative_polarity	1.19	1	1.09
title_subjectivity_dis	1.11	1	1.05
title_sentiment_polarity_dis	1.08	1	1.04

Evaluation

In this section, we use the 4 models we built in the previous section, we find the final model still has the best performance and can be used to predict the future new data.
figure(12)

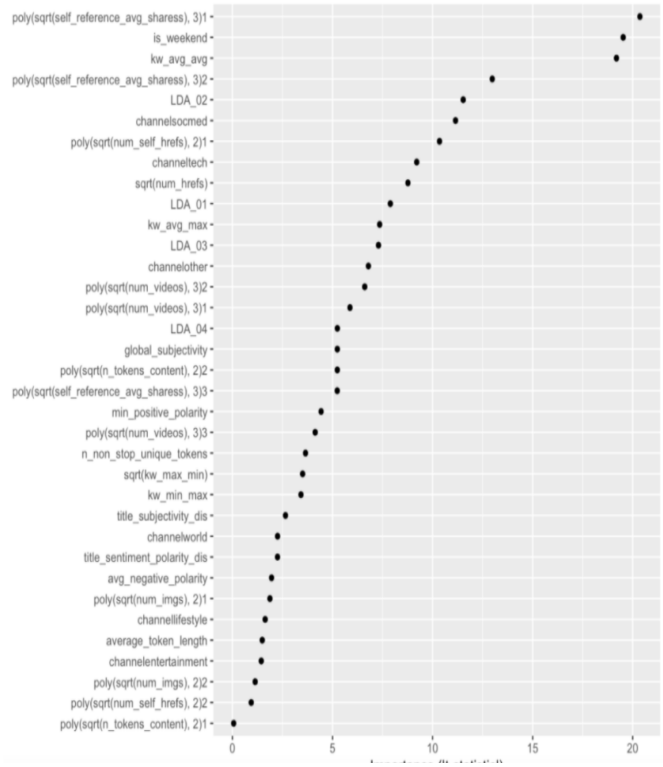
Model <chr>	Rsquared <dbl>	adjust_Rsquared <dbl>
No_Trans Model	0.0156	0.0138
Log_Trans Model	0.1056	0.1033
Box_Cox_Trans Model	0.1076	0.1052
Final Model	0.1266	0.1240

Conclusion

Model Interpretation

Based on the predictors importance rank and the sign of their coefficients (figure 13), we can give some critical advices to the website authors to help them increase the shares of their articles:
The average shares of the referenced articles in Mashable is a very important benchmark.
If the author wants to get more shares, it is better to publish the article during the weekends.
The keyword is also very important, opposite to the best keyword or the worst keyword, normal keywords that with high average shares would get more shares.
The article's shares have a negative relationship to the closeness to LDA topic 1 to topic 4, otherwise choosing LDA topic 0 would get more shares.
Articles that belong to social media channel will get more shares, followed by tech channel, other channel, world channel, life channel and business channel. Articles belonging to the Entertainment channel will get the least shares.
If the author wants to get more shares, it is suggested to put less links to other articles published by Mashable.
We also find adding videos will help to increase shares.

figure(13)



Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	3.65e+00	2.27e-02	160.64	< 2e-16	***
poly(sqrt(n_tokens_content), 2)1	1.61e-02	2.61e-01	0.06	0.95086	
poly(sqrt(n_tokens_content), 2)2	9.30e-01	1.78e-01	5.24	1.6e-07	***
n_non_stop_unique_tokens	-5.42e-02	1.49e-02	-3.65	0.00026	***
sqrt(num_hrefs)	8.42e-03	9.60e-04	8.77	< 2e-16	***
poly(sqrt(num_self_hrefs), 2)1	-2.08e+00	2.01e-01	-10.35	< 2e-16	***
poly(sqrt(num_self_hrefs), 2)2	-1.69e-01	1.81e-01	-0.94	0.34872	
poly(sqrt(num_imgs), 2)1	4.07e-01	2.18e-01	1.87	0.06131	.
poly(sqrt(num_imgs), 2)2	-1.94e-01	1.71e-01	-1.13	0.25703	
poly(sqrt(num_videos), 3)1	1.09e+00	1.86e-01	5.87	4.4e-09	***
poly(sqrt(num_videos), 3)2	-1.07e+00	1.62e-01	-6.61	4.0e-11	***
poly(sqrt(num_videos), 3)3	6.62e-01	1.60e-01	4.13	3.6e-05	***
average_token_length	-5.71e-03	3.84e-03	-1.49	0.13687	
channelentertainment	-8.68e-03	6.03e-03	-1.44	0.14967	
channellifestyle	9.95e-03	6.08e-03	1.64	0.10198	
channelother	4.29e-02	6.32e-03	6.79	1.1e-11	***
channelsocmed	5.72e-02	5.14e-03	11.15	< 2e-16	***
channeltech	5.04e-02	5.47e-03	9.21	< 2e-16	***
channelworld	1.32e-02	5.87e-03	2.25	0.02422	*
sqrt(kw_max_min)	-2.20e-04	6.29e-05	-3.51	0.00045	***
kw_min_max	-6.61e-08	1.93e-08	-3.42	0.00062	***
kw_avg_max	-7.33e-08	9.96e-09	-7.36	2.0e-13	***
kw_avg_avg	2.09e-05	1.09e-06	19.19	< 2e-16	***
poly(sqrt(self_reference_avg_sharess), 3)1	3.65e+00	1.79e-01	20.36	< 2e-16	***
poly(sqrt(self_reference_avg_sharess), 3)2	-2.24e+00	1.73e-01	-12.98	< 2e-16	***
poly(sqrt(self_reference_avg_sharess), 3)3	9.13e-01	1.74e-01	5.24	1.7e-07	***
is_weekend	5.62e-02	2.88e-03	19.52	< 2e-16	***
LDA_01	-6.94e-02	8.80e-03	-7.89	3.1e-15	***
LDA_02	-9.48e-02	8.23e-03	-11.53	< 2e-16	***
LDA_03	-6.11e-02	8.38e-03	-7.30	3.0e-13	***
LDA_04	-3.95e-02	7.54e-03	-5.24	1.6e-07	***
global_subjectivity	6.56e-02	1.25e-02	5.24	1.6e-07	***
min_positive_polarity	-6.85e-02	1.54e-02	-4.43	9.4e-06	***
avg_negative_polarity	-1.69e-02	8.65e-03	-1.95	0.05061	.
title_subjectivity_dis	8.29e-03	3.13e-03	2.65	0.00808	**
title_sentiment_polarity_dis	8.60e-03	3.82e-03	2.25	0.02439	*

Model Improving

Improving based on the current variables

It would make sense that the article belonging to the entertainment channel published on weekends would get more shares,so that we can add an interaction effect between the publish time and channel. Following this direction, we can still explore if there are interaction effects between different variables.
We can also find in our last model QQ plot, the residual plot is still left-skewed, meaning that some outliers exist. We need to further investigate these outliers to get more information.
It is very hard to interpret the model results after taking Box-Cox transformation, in this way bootstrap regression method would be a good choice, which does not need normality assumptions.

Improving predictor frame

We find even if our model is robust and the result is statistically significant, the model in total can only explain about 13.8% of the variance of the response variable. That means these variables cannot explain the response variable perfectly; If Mashable wants to get a more

accurate predictor of shares, it needs to continue working on refining the predictor frame from a professional angle.

Appendix

Data:
https://archive.ics.uci.edu/ml/datasets/online+news+popularity

Code:

https://github.com/tracedata1/regression-project/blob/main/code

Appendix 1

```r
#  n_non_stop_words
round(sum(news0$n_non_stop_words==0)/length(news0$n_non_stop_words),2)# 0.97, this variable is almost a single value, we will drop this variable in our model.
```

```r
summary(news0$n_non_stop_words)
boxplot(news0$n_non_stop_words,xlab=" n_non_stop_words")
```

```r
#install.packages("pastecs")
library(pastecs)
news0n0 <-news0[,-c(14:19)]
names(news0n0)
news0n1 <-news0n0[,-c(26:33)]
sumad <- stat.desc(news0n1)

options(scipen = 999,digits=1)
summary_new<-as.data.frame(t(sumad[c(4,5,8,9),c(-1,-2)]))
summary_new
```

```{r}
# kw_min_min:Worst Keyword (Min. Shares). It is weird to have negative value in this variable, we will drop this variable since there are too many wrong values.
sum(news0$kw_min_min==-1)  #22980 cases
sum(news0$kw_avg_min==-1) # 694
sum(news0$kw_avg_max==-1) # 0
cor(news0[20:22]) # since kw_max_min and kw_avg_min are highly related(r=0.94), we will choose kw_avg_max in our model
variable_name <- c("kw_min_min","kw_avg_min","kw_avg_max")
wrong_value <- c(22980,694,0)

missing_percent <- c(22980/nrow(news0),694/nrow(news0),0)


# n_tokens_content
boxplot(news0$n_tokens_content,xlab="n_tokens_content")


sum(news0$n_tokens_content==0)/length(news0$n_tokens_content)
# 0.02979013, 1181 cases
# we will treat 0 as missing value
```

```{r}
# n_unique_tokens is Rate Of Unique Words In The Content, the value range should be (0,1),any value great than 1 should be considered as wrong.
 indt <- which(news0$n_unique_tokens>1)
 wr <- news0[indt,]
wr[,-wr$timedelta] # we can drop this case in our data

library(tidyverse)

sum(news0$n_unique_tokens>1)
sum(news0$n_non_stop_unique_tokens>1)

news<-news0%>%
  dplyr::filter( n_tokens_content>0 & n_unique_tokens<=1) %>%
 dplyr::select(-kw_min_min,-kw_avg_min,-url,-timedelta,-n_non_stop_words ) # drop non-predictive variables, also singlar variable n_non_stop_words

Missing_percent <- 1-nrow(news)/nrow(news0) #  0.02981536
Missing_case<- nrow(news0)-nrow(news) # 1182 the total cases we delete
Total_case <-nrow(news0)
Model_case <- nrow(news)

Dt <- data.frame(Missing_percent,Missing_case,Total_case,Model_case)
Dt


```

Appendix 2

```
## data relationship exploration

```{r}
# since they are heterogeneous set of features, we will explore the relationship among the predictor variables in each feature.
names(news)
# Word features
pairs(news[1:4],cex=0.2)
cor(news[3:4])
cor(news$n_unique_tokens , news$n_non_stop_unique_tokens) # 0.885152 we will choose n_non_stop_unique_tokens in our model

# Links & References features
#"self_reference_min_shares",self_reference_max_shares" and "self_reference_avg_sharess", "num_hrefs","num_self_hrefs"
pairs(~self_reference_min_shares+self_reference_max_shares+self_reference_avg_sharess+num_hrefs+num_self_hrefs,data=news,cex=0.2) # we can see there is obviously linear relationship among these 3 variables: "self_reference_min_shares", self_reference_max_shares" and "self_reference_avg_sharess"

cor(news[24:26]) #  in this case we will keep "self_reference_avg_sharess" and drop "self_reference_min_shares", self_reference_max_shares"
cor(news[5:6]) # we will keep these 2 variables


# Keyword features
pairs(news[17:23],cex=0.2) # for variables kw_max_min,kw_min_max,kw_max_max,kw_avg_max,kw_min_avg, kw_max_avg and kw_avg_avg.
cor(news[17:23]) #we find kw_max_avg and kw_avg_avg are high correlated (0.8164474),we will choose kw_avg_avg, drop kw_max_avg

```
```

Appendix 3

```
### set up training and test data set

```{r}
set.seed(123)
n<-nrow(news1)


index_train <- sample(1:n, round(0.7*n)) # we randomly choose 70% of the data as training data
newstrain <- news1[index_train,]
newstest <- news1[-index_train,]
```

```{r}
options(scipen=0)
lmnew_full <- lm(shares~.,data=newstrain)
lmnew_full_step <- step(lmnew_full,trace=F)
summary(lmnew_full_step)

par(mfrow=c(1,2))
plot(lmnew_full_step,1:2)
```
```

```
## refit the model

### using log- transformation

```{r}
lmnew_full_log <- lm(log(shares)~.,data=newstrain)
summary(lmnew_full_log )

par(mfrow=c(1,2))
plot(lmnew_full_log,1:2)
```

```{r}
table(newstrain$channel)
```

```{r}
dim(newstrain)
```

```{r}

lmnew_full_logs <- step(lmnew_full_log,trace=F)
summary(lmnew_full_logs)
par(mfrow=c(1,2))
plot(lmnew_full_logs,1:2,cex=0.5)
```
```

Appendix 4

```
```{r}
library(ggplot2)
# recoding the dummy variables
news$channel[news$data_channel_is_lifestyle==1] <-"lifestyle"
news$channel[news$data_channel_is_entertainment==1] <-"entertainment"
news$channel[news$data_channel_is_socmed==1] <-"socmed"
news$channel[news$data_channel_is_tech==1] <-"tech"
news$channel[news$data_channel_is_world==1] <-"world"
news$channel[news$data_channel_is_bus==1] <-"business"
news$channel[news$data_channel_is_world==0 & news$data_channel_is_lifestyle==0 &
news$data_channel_is_entertainment==0 & news$data_channel_is_socmed==0 & news$data_channel_is_tech==0 &
news$data_channel_is_bus==0] <-"other"
news$channel <- factor(news$channel)
ggplot(news,aes(channel))+geom_bar()
ggplot(news, aes(channel,log(shares)))+
  geom_boxplot()

```

```{r}

# recoding the dummy variables
news$weekday[news$weekday_is_monday==1] <-"monday"
news$weekday[news$weekday_is_tuesday==1] <-"tuesday"
news$weekday[news$weekday_is_wednesday==1] <-"wednesday"
news$weekday[news$weekday_is_thursday==1] <-"thursday"
news$weekday[news$weekday_is_friday==1] <-"friday"
news$weekday[news$weekday_is_saturday==1] <-"saturday"
news$weekday[news$weekday_is_sunday==1] <-"sunday"
news$weekday <- factor(news$weekday)

ggplot(news, aes(weekday,log(shares)))+
  geom_boxplot()
ggplot(news, aes(as.factor(is_weekend),log(shares)))+
  geom_boxplot()
# comment: There are no differences through Monday to Friday on log(shares), median log(shares) for
weekends is abviously higher than not weekends. We can drop redundant variables and keep is_weekend
instead.

```
```

Appendix 5

```r
# we will refit our model using the findings above,add the polynomial effect and drop the none significant variable n_tokens_title.

lmnew_full_ts1 <- lm(formula = (((shares^-0.22) - 1)/(-0.22)) ~
    poly(sqrt(n_tokens_content),2)  + n_non_stop_unique_tokens +
    sqrt(num_hrefs) + poly(sqrt(num_self_hrefs),2) + poly(sqrt(num_imgs),2) + poly(sqrt(num_videos),3) + average_token_length +
    num_keywords +channel +sqrt(kw_max_min)  + +
    kw_min_max + kw_avg_max  + kw_avg_avg  + poly(sqrt(self_reference_avg_sharess),3) + is_weekend + LDA_01 + LDA_02 + LDA_03 + LDA_04 + global_subjectivity +
    global_sentiment_polarity + global_rate_positive_words +
    rate_positive_words + min_positive_polarity + avg_negative_polarity +
    title_subjectivity_dis + title_sentiment_polarity_dis ,
    data = newstrain)

summary(lmnew_full_ts1)
plot(lmnew_full_ts1)

```

```r
# in our final model, we can see multiple R-squared:0.1362, Adjusted R-squared:  0.1351 both of these two values are higher than the model without polynomial effect model multiple R-squared:  0.1195, Adjusted R-squared:  0.1185
options(scipen=0)
lmnew_final <- step(lmnew_full_ts1,trace=F)
summary(lmnew_final)
plot(lmnew_final,1:4,cex=0.5)


newstrain[c(18830,17019,9695),]

```

```r
options(scipen = 999,digits=3)
#library(faraway)
## check collinearity vif
## there are no serious collinearity problems in our model
car::vif(lmnew_final)

```

Appendix 6

```r

### check outliers
p<-35
n<-nrow(newstrain)
plot(hatvalues(lmnew_final), rstandard(lmnew_final),cex=0.5,
xlab='Leverage', ylab='Standardized Residuals')

abline(v=0.08,col="red",lty=2)
abline(h=c(-4,4),col="blue",lty=2)

ind <- which(hatvalues(lmnew_final)>0.08)
newstrain[ind,]


sum(abs(rstandard(lmnew_final))>4) # absolute value that greater than 4
sum(abs(rstandard(lmnew_final))>4)/nrow(newstrain)*100
```

```r
# variable importance
library(vip)
 vip(lmnew_final, num_features = 35, geom = "point", include_type = TRUE)

```