Secret Behind Online News Popularity

Fan liu, Xinyan Yang, Yang Yu

December 13, 2017

Abstract

Due to the development of internet, browsing news online is increasingly popular. In this report, we are going to figure out what are the most important factors of online news popularity. Five different regression models were built and LASSO was selected for its low test RMSE and easy-to-explain feature.

Introduction

With the development of internet, people browse the latest online news with an increasing frequency. Thus, predicting the online news popularity has become a social research trend since it is valuable for authors, content providers, advertisers and even politicians (e.g., to understand or influence public opinion). What's more, the machine learning including supervised methods and unsupervised methods breaking stricts of traditional statistical models can get a more higher accuracy of the prediction. So, we consider four most common models which are K Nearest Neighbors, Random Forest, Boosting and Elastic Net to build a model to predict the online news popularity.

Our dataset comes from UCI (http://archive.ics.uci.edu/ml/datasets/Online+News+Popularity). It summarizes a heterogeneous set of features about articles published by Mashable (www.mashable.com) on January 8, 2015 in a period of 2013-2014. Mashable is a digital media website founded in 2005. As one of the most influential media in America, Mashable was described as a "one stop shop" for social media. There are 7 channels on Mashable including Video, Entertainment, Culture, Tech, Science, Business and Social good. The data does not share the original content of these articles but some statistics associated with it. The original content be publicly accessed and retrieved using the provided urls.

The whole data contains a response variable "shares" and 60 predictors which describe the different features of the news. We choose to build regression models to get some sense in what are most important predictors that would influence online news popularity. And we choose the most recent data in 2014 because of the quick development of online news and the most recent data can show more information. Finally, since Lasso has low test RMSE among all four models and the coefficients of predictors make it more easier to explain compared to the random forest model, we choose Lasso as our final model and we will explain some important predictors selected by the model in detail.

Materials

Our raw data coming from UCI contains 39,797 observations and 61 variables including the response "shares" and other 60 variables representing different features of articles published on Mashable. And these variables are extracted from HTML code by Python. Each row represents an article.

News changed so fast due to convenient media tools. And people is becoming more and more willing to open mind to new things. Therefore, we think the most updatest data would give us more correct guide of online news popularity. In the end, we decided to choose data from 2014 which contains 21,445 observations.

Independent Variables

During the data processing, we also found that "timedelta" vaiable is wrongly computed according to the date we extract from the "url" varaible. So we self-compute the "timedelta" column to replace the original one and remove the “url” variable which is website information. The following table shows the descriptions of some of our variables.

Variable Table

Variable	Description
url	URL of the article (non-predictive)
n_tokens_title	Number of words in the title
n_tokens_content	Number of words in the content
n_unique_tokens	Rate of unique words in the content
n_non_stop_unique_tokens	Rate of unique non-stop words in the content
num_hrefs	Number of links
num_self_hrefs	Number of links to other articles published by Mashable
num_imgs	Number of images
num_videos	Number of videos
average_token_length	Average length of the words in the content
global_rate_positive_words	Rate of positive words in the content
global_rate_negative_words	Rate of negative words in the content
title_subjectivity	Title subjectivity
shares	Number of shares (target)

Here are the column names of the dataset after all data preprocessing.

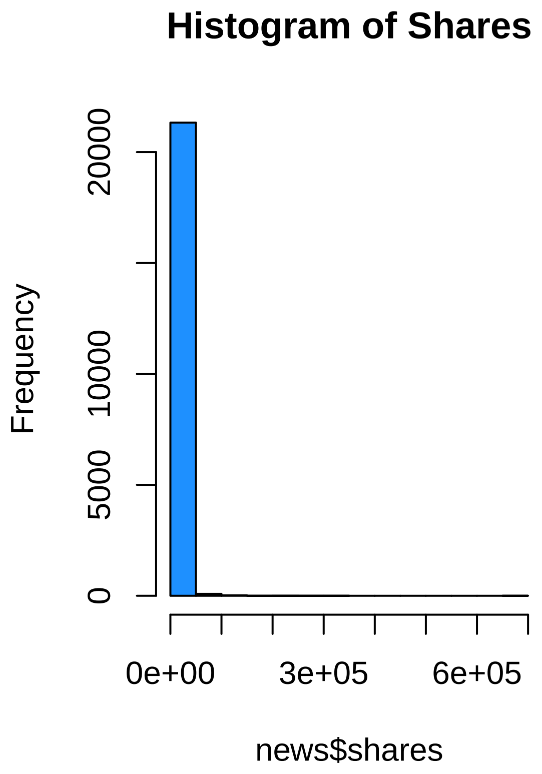
```r
names(news)
```

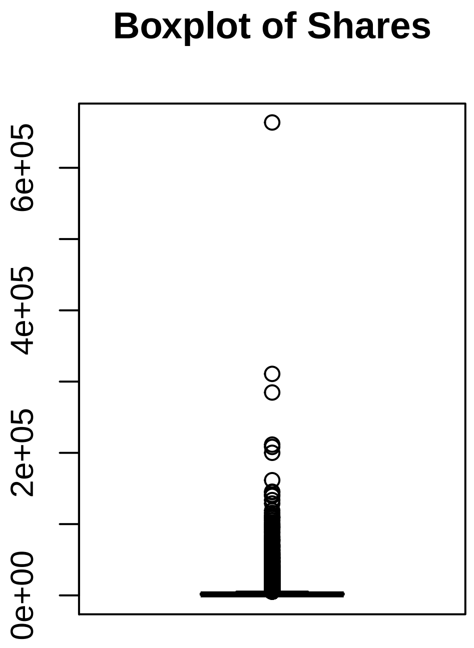
```
##  [1] "timedelta"                     "n_tokens_title"
##  [3] "n_tokens_content"              "n_unique_tokens"
##  [5] "n_non_stop_words"              "n_non_stop_unique_tokens"
##  [7] "num_hrefs"                     "num_self_hrefs"
##  [9] "num_imgs"                      "num_videos"
## [11] "average_token_length"          "num_keywords"
## [13] "data_channel_is_lifestyle"     "data_channel_is_entertainment"
## [15] "data_channel_is_bus"           "data_channel_is_socmed"
## [17] "data_channel_is_tech"          "data_channel_is_world"
## [19] "kw_min_min"                    "kw_max_min"
## [21] "kw_avg_min"                    "kw_min_max"
## [23] "kw_max_max"                    "kw_avg_max"
## [25] "kw_min_avg"                    "kw_max_avg"
## [27] "kw_avg_avg"                    "self_reference_min_shares"
## [29] "self_reference_max_shares"     "self_reference_avg_sharess"
## [31] "weekday_is_monday"             "weekday_is_tuesday"
## [33] "weekday_is_wednesday"          "weekday_is_thursday"
## [35] "weekday_is_friday"             "weekday_is_saturday"
## [37] "weekday_is_sunday"             "is_weekend"
## [39] "LDA_00"                        "LDA_01"
## [41] "LDA_02"                        "LDA_03"
## [43] "LDA_04"                        "global_subjectivity"
## [45] "global_sentiment_polarity"     "global_rate_positive_words"
## [47] "global_rate_negative_words"    "rate_positive_words"
## [49] "rate_negative_words"           "avg_positive_polarity"
## [51] "min_positive_polarity"         "max_positive_polarity"
## [53] "avg_negative_polarity"         "min_negative_polarity"
## [55] "max_negative_polarity"         "title_subjectivity"
## [57] "title_sentiment_polarity"      "abs_title_subjectivity"
## [59] "abs_title_sentiment_polarity"  "shares"
```

Dependent Variables

Before building the models, we feel it necessary to check the distribution of the response variable shares.

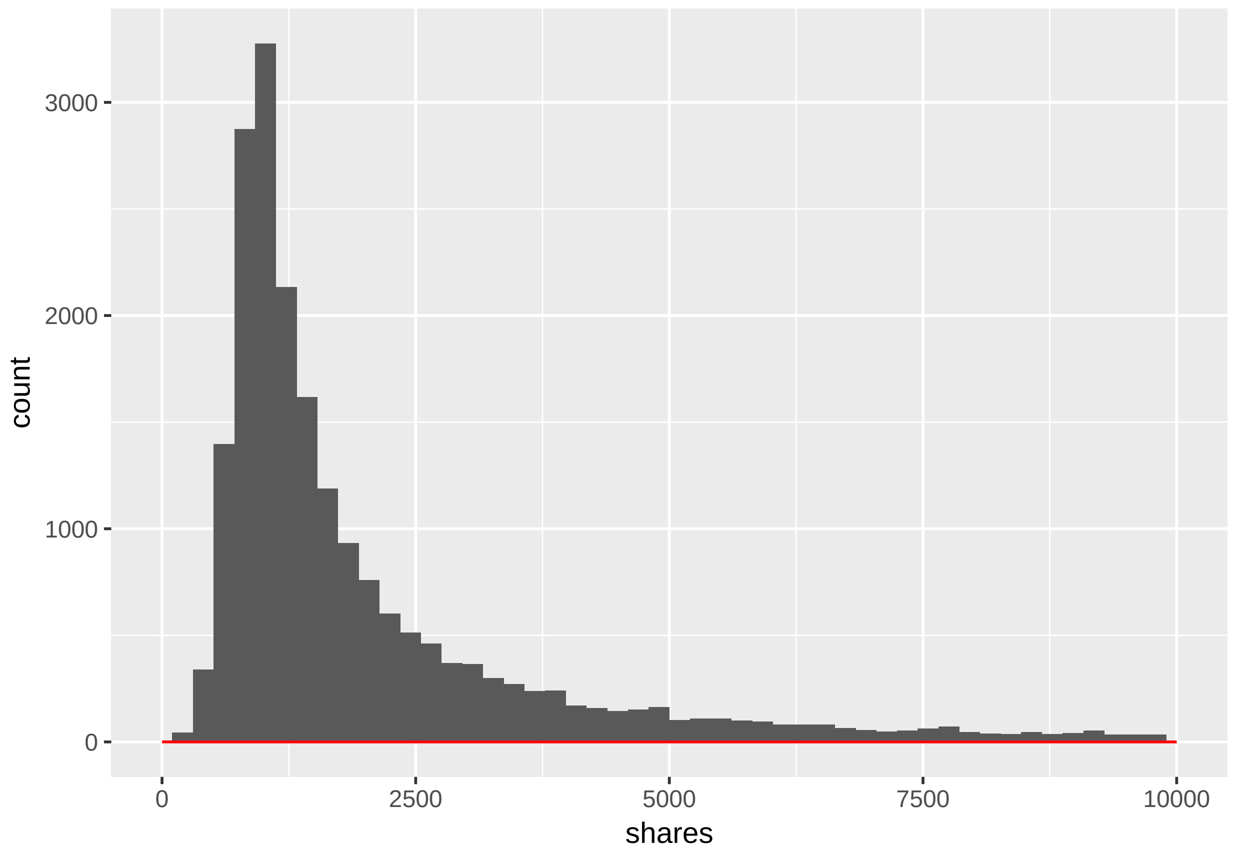
```r
par(mfrow = c(1,2))
hist(news$shares, main = "Histogram of Shares", col = "dodgerblue")
boxplot(news$shares, main = "Boxplot of Shares", col = "darkorange")
```


Histogram of Shares


Boxplot of Shares

Obviously, we can see that there is an outlier with extremely large value, and it's too large that we even cannot see the distribution of shares clearly. Therefore we decided to delete this observation.

Since the distribution of shares is extremely right-skewed, we choose to show the distribution of shares within 0 to 10000.



Methods

Our goal is to explore which factors would affect articles' popularity, so we used shares as our response and other variables as our predictors. In order to choose a best performance model, we tried five methods, K Nearest Neighbors, Random Forest, Boosting, Lasso and Elastic Net. These methods would deal with different kinds of situations, such as linear, nonlinear and other complicated relationships.

In order to avoid overfitting and test different methods, we splitted our dataset into train data and test data, and used test RMSE to compare these models.

```
set.seed(9)
train_index = sample(nrow(news), size = trunc(0.60 * nrow(news)))
news_trn = news[train_index, ]
news_tst = news[-train_index, ]
```

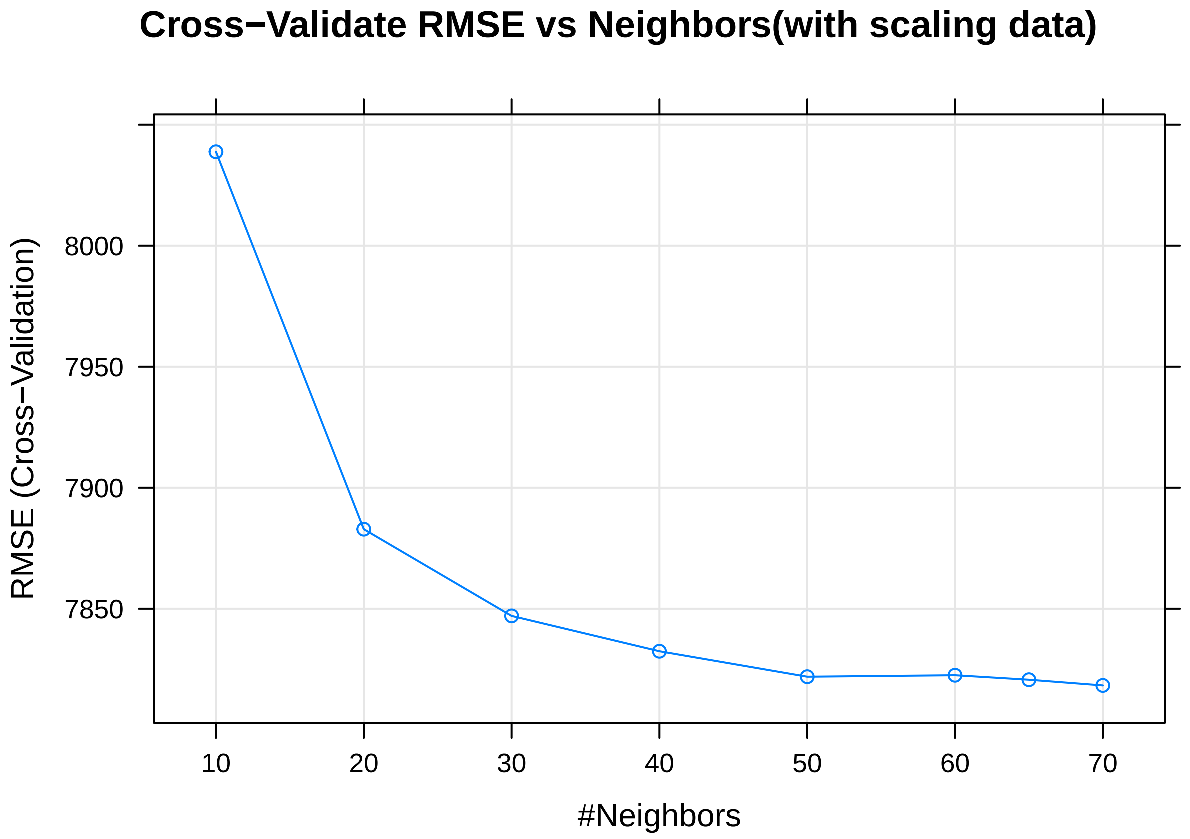
$$\text{RMSE}\left(f(x), \hat{f}(x)\right) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left[\left(f(x) - \hat{f}(x)\right)^2\right]}$$

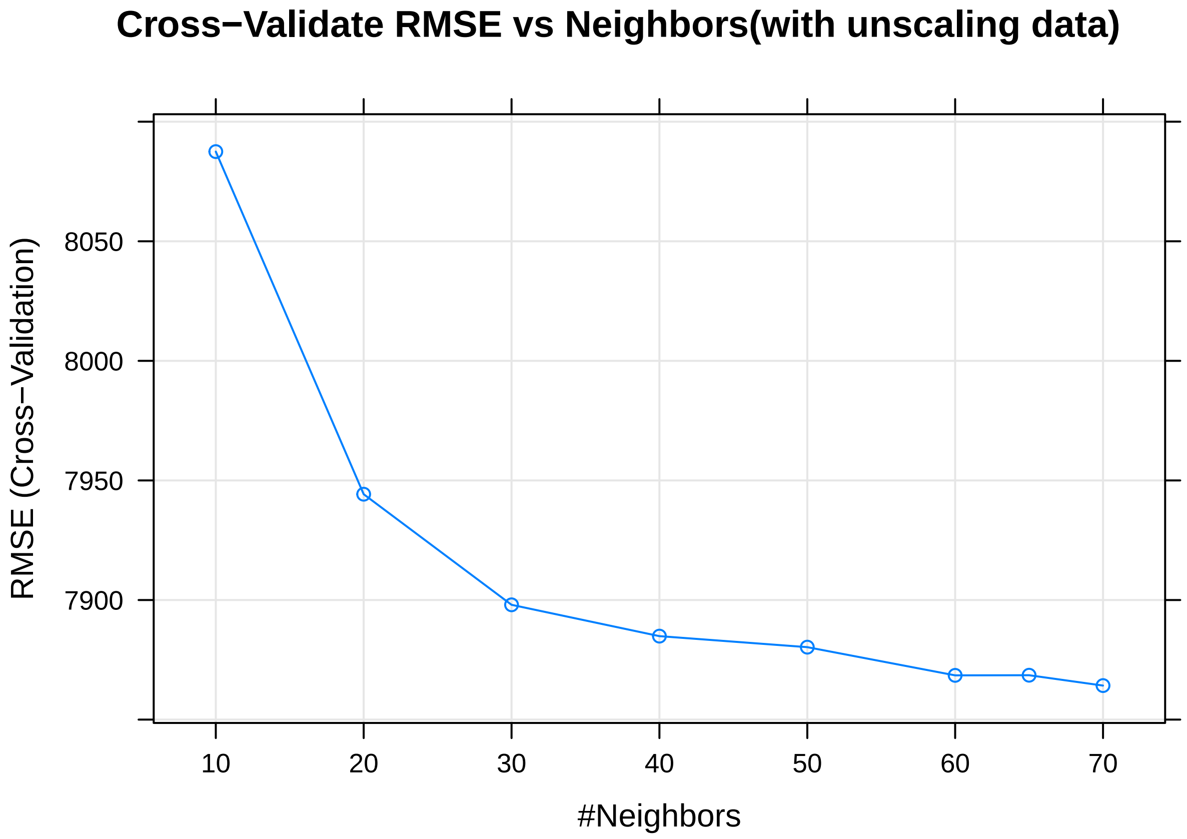
```
#write a function to compute RMSE
calc_rmse = function(pre_mod, actual){
    sqrt(mean((pre_mod - actual)^2))
}
```

KNN

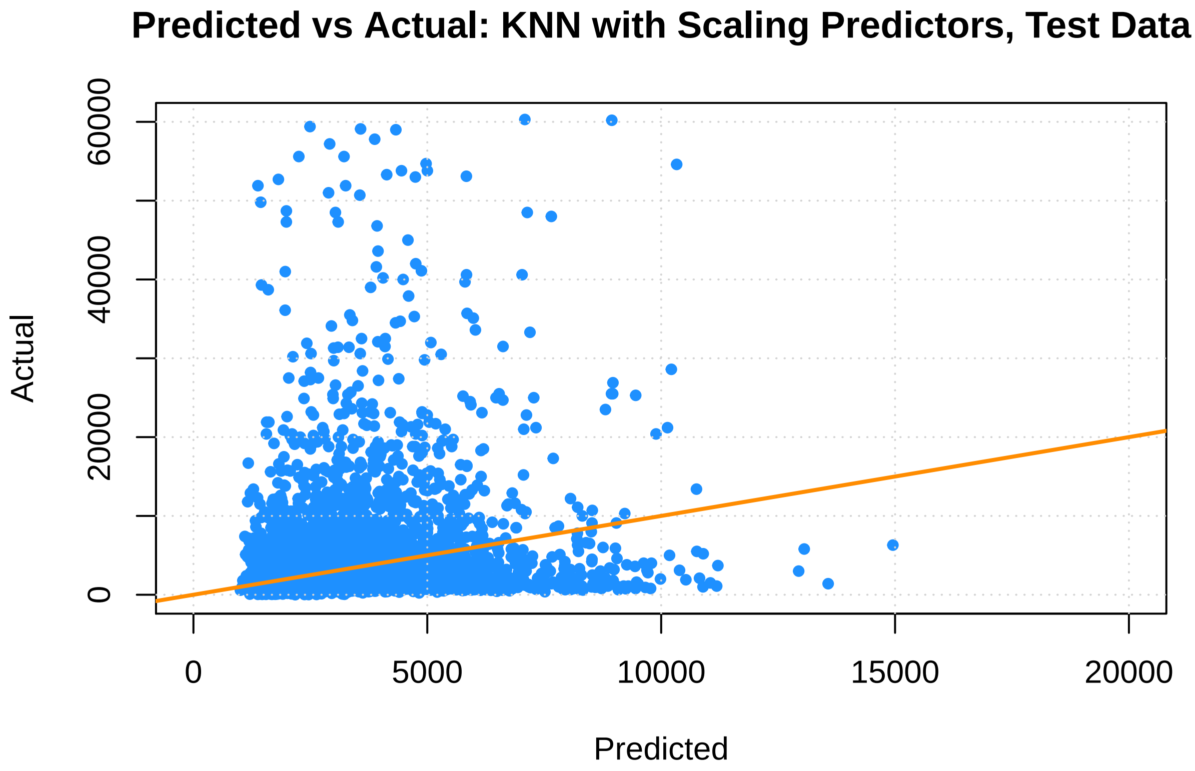
For the k-nearest neighbours model, we consider both the scaling case and unscaling case and consider k belongs to {10, 20, 30, 40, 50, 60, 65, 70}. We used caret package to train models and used 5-fold cross-validation as resampling method.

```r
set.seed(9)

#scaling KNN
knn_scale_mod = train(
  form = shares ~ .,
  data = news_trn,
  trControl = trainControl(method = "cv", number = 5),
  method = "knn",
  preProcess = c("center", "scale"),
  tuneGrid = expand.grid(k = c(10, 20, 30, 40, 50, 60, 65, 70))
)

#unscaling KNN
set.seed(9)
knn_unscale_mod = train(
  form = shares ~ .,
  data = news_trn,
  trControl = trainControl(method = "cv", number = 5),
  method = "knn",
  tuneGrid = expand.grid(k = c(10, 20, 30, 40, 50, 60, 65, 70))
)
```


Cross−Validate RMSE vs Neighbors(with scaling data)
RMSE (Cross−Validation)
8000
7950
7900
7850
10
20
30
40
50
60
70
#Neighbors

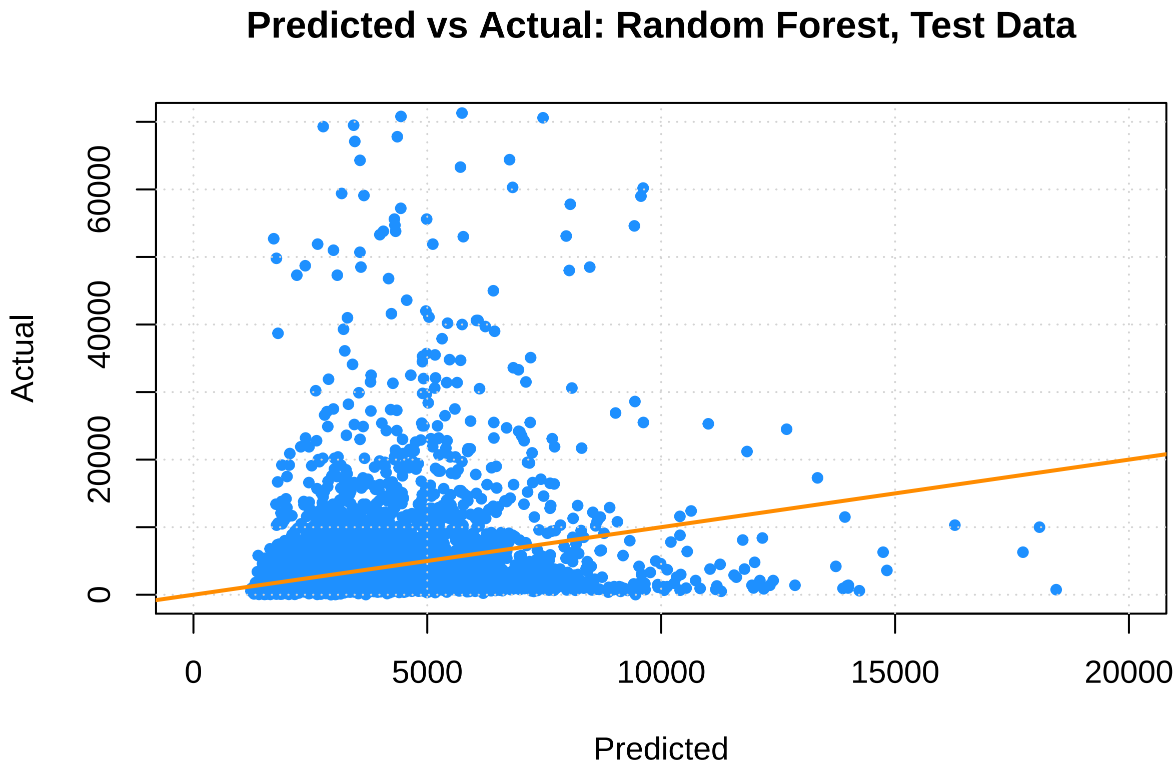


The plot shows that the best k for the scaling case would lies between 60 to 65 and that the scaling case performs better than the unscaling case.



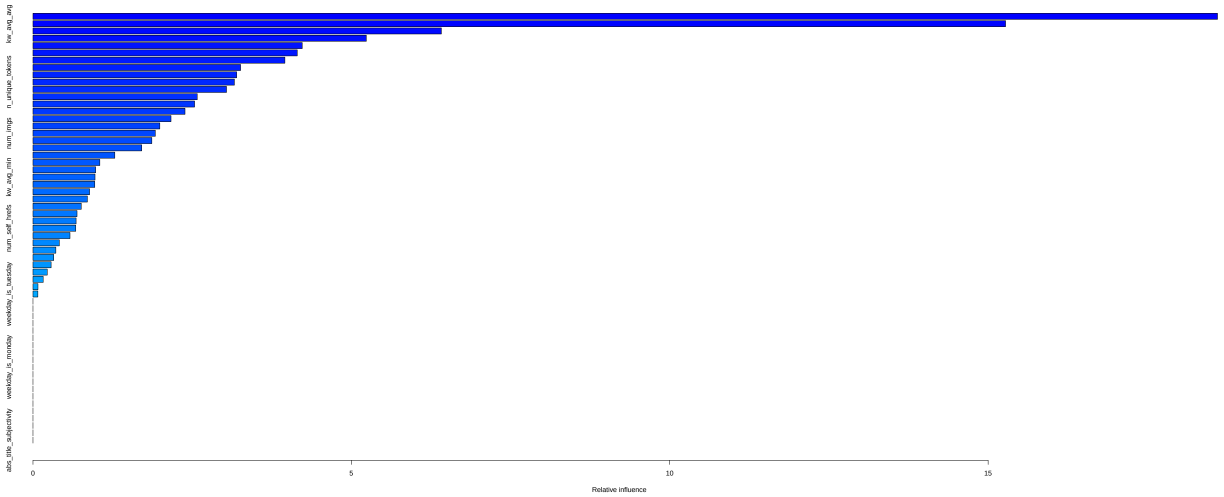
Random Forest

```r
set.seed(9)
rf_mod = train(
    form = shares~.,
    data = news_trn,
    trControl = trainControl(method = "cv", number = 5),
    method = "ranger",
    tuneGrid = expand.grid(.mtry = c(5, 10, 17, 20), .splitrule = "extratrees")
)
```


Predicted vs Actual: Random Forest, Test Data

Boosting

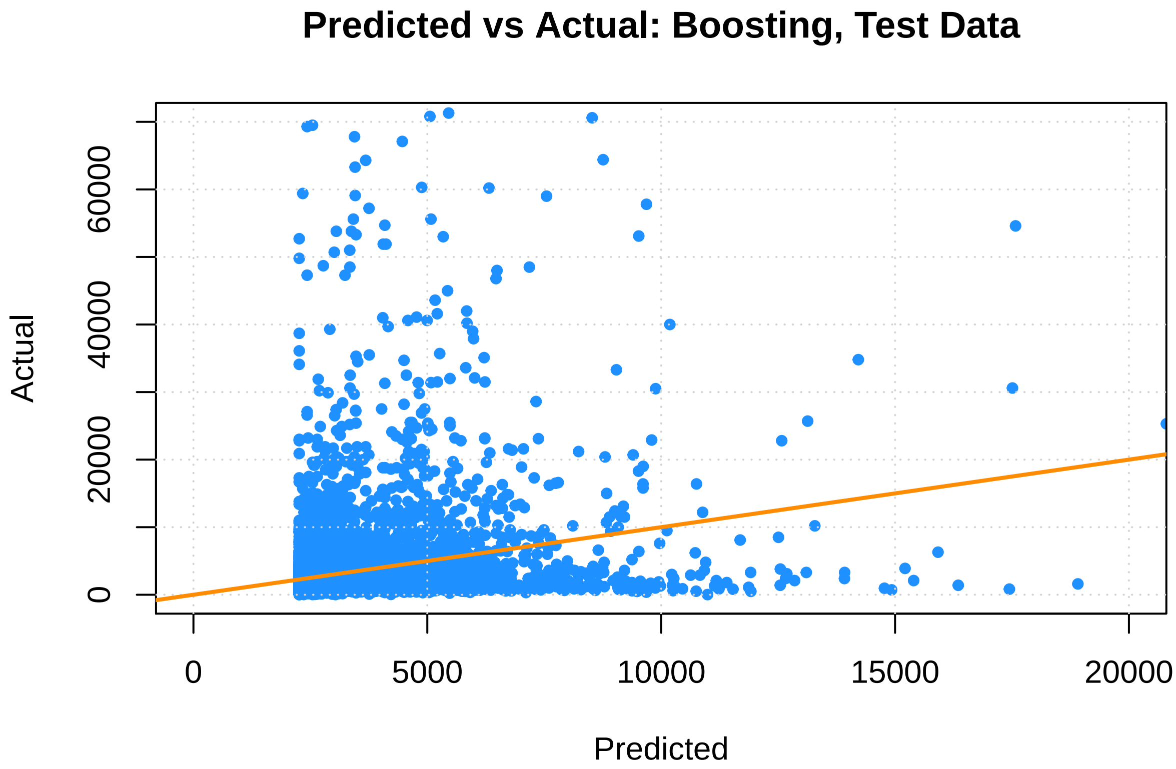
```r
set.seed(9)
boost_mod = gbm(shares ~ .,
                data = news_trn,
                distribution = "gaussian",
                n.trees = 100,
                interaction.depth = 3:5,
                n.minobsinnode = 10,
                shrinkage = 0.1,
                cv.folds = 5)

tibble::as_tibble(summary(boost_mod))
```



```
## # A tibble: 59 x 2
##                       var   rel.inf
##  *                 <fctr>     <dbl>
##  1             timedelta 18.602281
##  2            kw_avg_avg 15.273790
##  3            kw_max_avg  6.414129
##  4            kw_min_avg  5.235553
##  5                LDA_00  4.228737
##  6                LDA_03  4.150843
##  7                LDA_04  3.956309
##  8   rate_negative_words  3.260347
##  9 avg_negative_polarity  3.198447
## 10       n_unique_tokens  3.163585
## # ... with 49 more rows
```

```
## Using 25 trees...
```

```
## Using 25 trees...
```

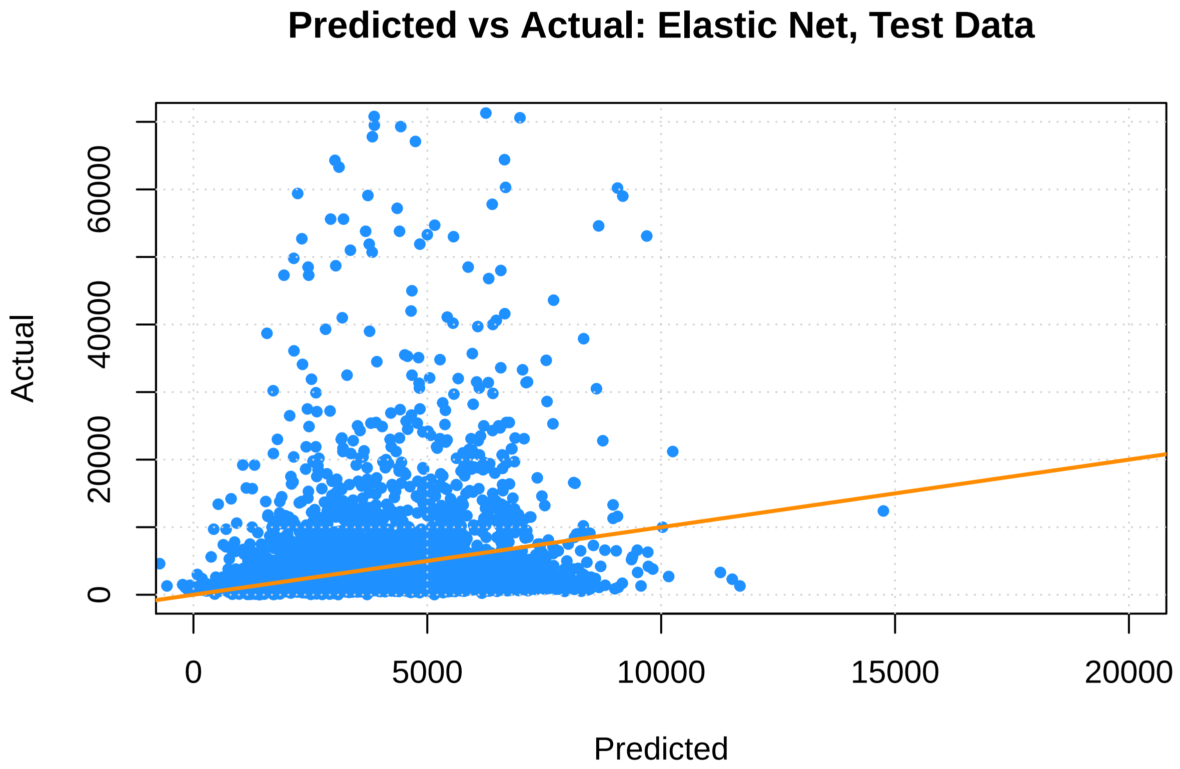


We can see that the boosting plot is different from the obvious two plots in that it barely has any predicted values fall in (0, 2500).

Elastic Net

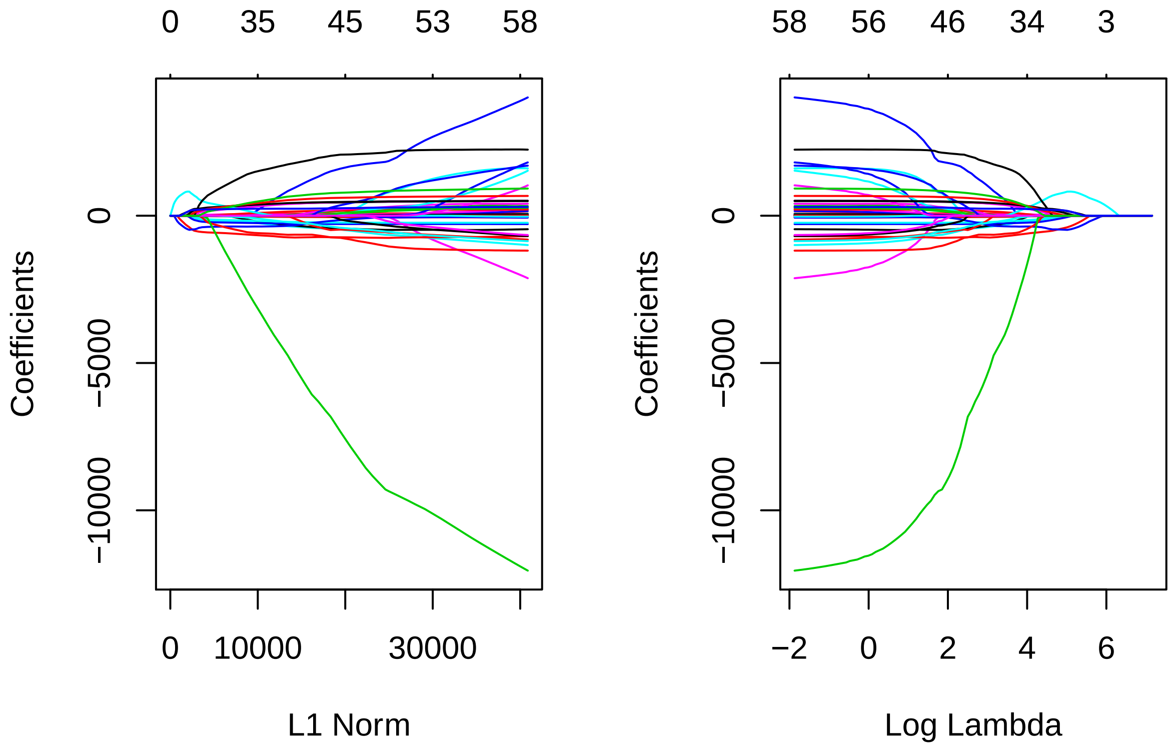
```
set.seed(9)
elnet_mod = train(
  form = shares ~ .,
  data = news_trn,
  method = "glmnet",
  trControl = trainControl(method = "cv", number = 5),
  tuneLength = 10
)

elnet_mod$bestTune
```

```
##    alpha   lambda
## 95     1 16.83063
```



Lasso

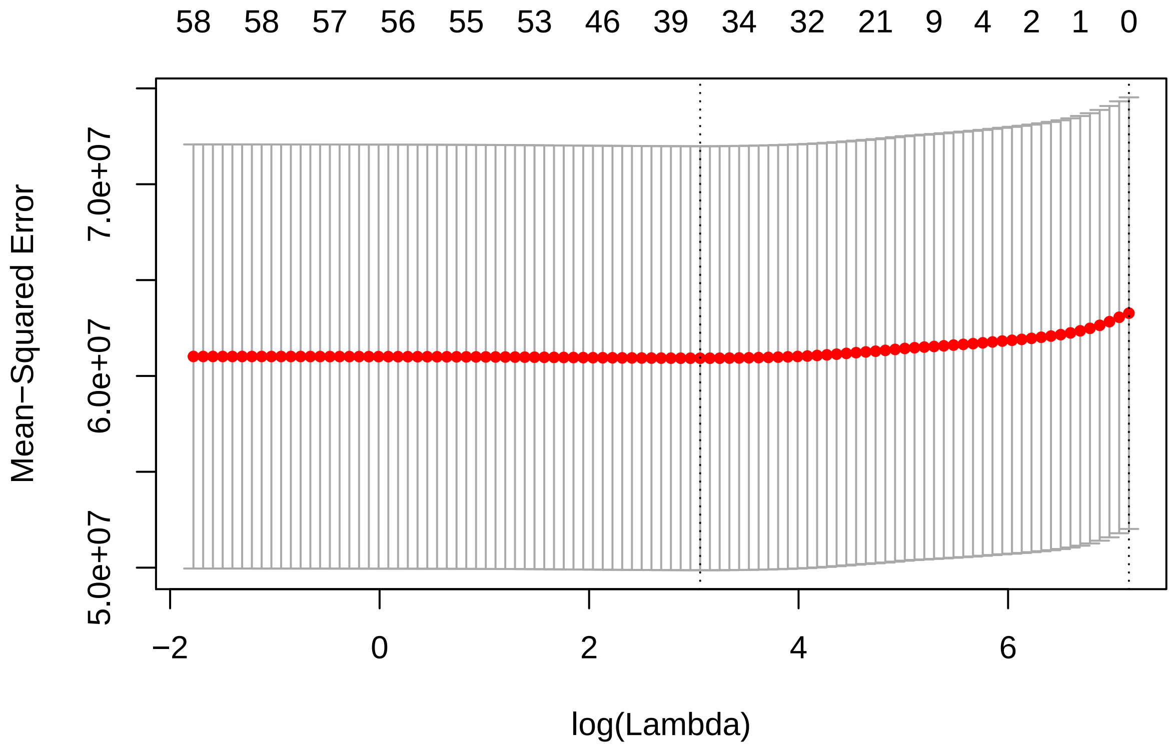
```
set.seed(9)
X = model.matrix(shares ~ ., news_trn)[, -1]
y = news_trn$shares

par(mfrow = c(1, 2))
lasso_mod2 = glmnet(X, y, alpha = 1)
plot(lasso_mod2)
plot(lasso_mod2, xvar = "lambda", label = TRUE)
```



The two plots illustrate how much the coefficients are penalized for different values of lambda. We can notice that only 4 variables have large coefficients at first, with the growth of lambda, most variables become zero.

We use cross-validation to select a good lambda value. The plot illustrates the MSE for the lambda considered. Two lines are drawn. The first is the lambda that gives the smallest MSE. The second is the lambda that gives an MSE within one standard error of the smallest.

```r
set.seed(9)
lasso_mod = cv.glmnet(X, y, alpha = 1)
lasso_cv_rmse = calc_rmse(predict(lasso_mod, newx = X), news_trn$shares)
plot(lasso_mod)
```



The above plot shows that there is no significant differences between different lambda, and we could obtain 32 variables if we pick the lambda that gives the smallest MSE.

```r
#fitted coefficients, using minimum lambda
lasso_coef = coef(lasso_mod, s = "lambda.min")
name = rownames(coef(lasso_mod, s = "lambda.min"))[which(coef(lasso_mod, s = "lambda.min") != 0)]
name
```

```
##  [1] "(Intercept)"                    "timedelta"
##  [3] "n_tokens_title"                 "n_unique_tokens"
##  [5] "num_hrefs"                      "num_self_hrefs"
##  [7] "num_imgs"                       "num_videos"
##  [9] "average_token_length"           "num_keywords"
## [11] "data_channel_is_lifestyle1"     "data_channel_is_entertainment1"
## [13] "data_channel_is_bus1"           "kw_avg_min"
## [15] "kw_min_max"                     "kw_max_max"
## [17] "kw_avg_max"                     "kw_min_avg"
## [19] "kw_max_avg"                     "kw_avg_avg"
## [21] "self_reference_max_shares"      "self_reference_avg_sharess"
## [23] "weekday_is_monday1"             "weekday_is_tuesday1"
## [25] "weekday_is_thursday1"           "weekday_is_friday1"
## [27] "weekday_is_sunday1"             "LDA_02"
## [29] "LDA_04"                         "global_subjectivity"
## [31] "global_sentiment_polarity"      "global_rate_positive_words"
## [33] "min_positive_polarity"          "min_negative_polarity"
## [35] "title_subjectivity"             "title_sentiment_polarity"
## [37] "abs_title_subjectivity"         "abs_title_sentiment_polarity"
```

RMSE table

	cv_rmse	tst_rmse
KNN(scaling predictors)	7818.303	8055.442
KNN(unscaling predictors)	7864.232	8120.532
Random Forest	7754.571	8007.693
Elastic Net	7751.912	12027.657
Lasso	7956.099	10703.153
Boosting	7556.813	8009.536

From the table we can see that random forest has achieved the lowest RMSE and lasso got the nearest small RMSE. Meantime, Lasso would automatically selected significant variables. In general, Lasso performed pretty good on test data and would help us find out influences of online news popularity. Therefore we chose Lasso as our final model.

Discussion

```
## 60 x 1 sparse Matrix of class "dgCMatrix"
##                                           1
## (Intercept)                   -2.956614e+03
## timedelta                      3.618280e+00
## n_tokens_title                 6.997062e+01
## n_tokens_content               .
## n_unique_tokens                9.471245e+02
## n_non_stop_words               .
## n_non_stop_unique_tokens       .
## num_hrefs                      2.347496e+01
## num_self_hrefs                -4.201945e+01
## num_imgs                       1.316716e+01
## num_videos                     8.247697e+00
## average_token_length          -3.725385e+02
## num_keywords                   4.088449e+01
## data_channel_is_lifestyle1    -3.385696e+02
## data_channel_is_entertainment1 -7.405836e+02
## data_channel_is_bus1           3.236802e+01
## data_channel_is_socmed1        .
## data_channel_is_tech1          .
## data_channel_is_world1         .
## kw_min_min                     .
## kw_max_min                     .
## kw_avg_min                    -1.801200e-01
## kw_min_max                    -1.657082e-03
## kw_max_max                     9.279310e-04
## kw_avg_max                     3.219443e-04
## kw_min_avg                    -3.126290e-01
## kw_max_avg                    -1.887554e-01
## kw_avg_avg                     1.596654e+00
## self_reference_min_shares      .
## self_reference_max_shares      1.340621e-03
## self_reference_avg_sharess     4.922019e-03
## weekday_is_monday1             4.150802e+02
## weekday_is_tuesday1             1.450514e+02
## weekday_is_wednesday1           .           
## weekday_is_thursday1           -2.711064e+02
## weekday_is_friday1             -2.130816e+02
## weekday_is_saturday1            .           
## weekday_is_sunday1              4.065275e+02
## is_weekend1                     .           
## LDA_00                          .           
## LDA_01                          .           
## LDA_02                         -3.266644e+02
## LDA_03                          .           
## LDA_04                          1.426994e+01
## global_subjectivity             1.785406e+03
## global_sentiment_polarity      -6.442840e+02
## global_rate_positive_words     -5.147263e+03
## global_rate_negative_words      .           
## rate_positive_words             .           
## rate_negative_words             .           
## avg_positive_polarity           .           
## min_positive_polarity          -1.031567e+02
## max_positive_polarity           .           
## avg_negative_polarity           .           
## min_negative_polarity          -2.357291e+02
## max_negative_polarity           .           
## title_subjectivity              4.339057e+02
## title_sentiment_polarity        5.416645e+02
## abs_title_subjectivity          6.676634e+02
## abs_title_sentiment_polarity    2.370051e+02
```

As we can see, Lasso has already picked the significant variables for us and shrinked the other insignificant variables' coefficients to be zero. And next we will discuss these variables' influence on the shares in details.

Timedelta shows time information between the day the news published and the day we acquired the data. It ranges from 12 to 378 days. According to the result, we found that if an article was published longer, its shares tend to increase. This satisfied our common sense. The longer an article was published, the more audiences would read it. More reading definitely would increase shares.

There are five variables related to **LDA topics** , which measure the closeness of articles to five different LDA topics. The authors applied Latent dirichlet allocation (LDA) algorithm to all Mashable texts (known before publication) in order to identify the five top relevant topics and then measure the closeness of current article to such topics. Unfortunately, we still cannot figure out the what these five LDA topics are after looking through the authors' original paper. However we can still tell that the closeness to certain topic could influence the shares anyway. Here as you can see, if the news is more close to LDA_03 topic, it could significantly gain more shares.

About the features in the **words category**, we can see that number of words in the title, article and average word length can both have influence on the shares while rate of non-stop words, unique non-stop words and unique words have little to do with the shares. And we think this may because the readers are more focus on the contents itself instead of the writing style.

When it comes to the **data channel category**, we find that certain data channels like technology has positive effect on the shares while lifestyle, entertainment and bus channels have negative effect on the shares. And the world and social media channel have no effect on the shares compared to the others. This suggests that the editors should put more emphasis on the specific topic like technology instead of lifestyle, entertainment and bus if they want more shares.

About the **keywords category**, it is obvious that keywords can have a great effect on the shares of articles since most variables in this category have a non-zero coefficient, which suggests that the authors should appropriately increase amount of keywords in their articles to receive more attention and shares.

The links embedded in the articles can also have something to do with the shares. As we can see, increasing the number of links would help the articles gain more attention. At the same time, the number of images and videos have little influence on the shares itself and the reason we guess may still be that the readers are more focus on the contents itself.

Besides the number of references, **popular referenced articles** would play an important role in affecting shares, like self_reference_max_shares which means maximum shares of referenced referenced articles in Mashable, and self_reference_avg_shares which means average shares of referenced articles in Mashable. From the above we found if an article's referenced article has more shares, this article tends to be shared more. Referenced articles' shares have positive effect on original articles. This is because articles with more shares always contain hot topics or the public interested content and referring these articles implies an article is also related to topics which would grab the public's attention. As a result, this article's shares tend to increase.

Publishing time is another factor which would affect shares. Our result indicates articles published on Sunday usually acquire more shares, compared with other weekdays. This is consistent with our common sense. People always have more leisure time on Sunday to search online news and discuss interesting topics and hot topics with families and friends. Sharing is good way for them to discuss and communicate. Therefore, being published on Sunday would increase an article's shares. Besides, publishing on Thursday and Friday would have obvious decrease on shares, maybe due to busy time on working and planning weekends.

Sometimes, **subjectivity and positive words** would also affect sharing. Global_subjectivity represents text subjectivity. From the above result, we can see if an article contains more subjective content and discussion, its shares tends to increase, which means audiences would like to read those articles with subjective thinking and opinion. Generally, compared with objective information, subjective thinking would attract people more and arise hot discussion, so more subjective articles would have more shares. Global_rate_positive_words means rate of positive words in the content. The result above implies if an article's have more positive words, its shares tend to decrease. Maybe high rate of positive words would make people feel bored with its statement.

Title is an article's core. Appealing titles would always attract people at first glance and drive them to read the whole article. Our result indicates subjective title would help an article acquire more shares, which means people are willing to read articles with self-thinking. Meantime, the result said polar title would help an article obtain more shares. Maybe this is because polar titles usually generate hot discussion among audiences and audiences tend to share these articles to communicate with others. Therefore, our research indicates if authors expect their articles acquire more shares, they are supposed to create a subjective and polar title to attract more attention.

Conclusion

In this project, our objective is to figure out the key factors affecting the online news popularity and then give some suggestions to writers and online news websites. During the model selection, we compared five methods including K Nearest Neighbors, Random Forest, Boosting, Lasso and Elastic Net. According to their RMSE, we finally chose the Lasso model with RMSE as 7901. Fortunately, we can also get a meaningful interpretation by Lasso. In the Lasso model, there were 31 variables left. From our final result, we give following suggestions:

1. As for the words category, more words always come with more shares no matter in the context or in the title. So we suggest writers write as more information as they can.
2. Articles related to technique tend to be shared more, so if authors expect more shares, following latest technical development would attract more audiences.

3. Keywords, useful links and references all have positive effects on news popularity, so we suggest writers use keywords more often to make the article at hot topics and use more useful links and references to convince people.
4. Publishing time should be taken into account when trying to acquire more shares. Sunday is a good choice for gain popularity.
5. Subjective and polar titles and contexts always attract more shares. Writes are supposed to include some self-thinking and opinions.

Reference

1. UCI Machine Learning Repository: Online News Popularity Data Set. (2017). Archive.ics.uci.edu. Retrieved 19 December 2017, from http://archive.ics.uci.edu/ml/datasets/Online+News+Popularity
2. Fernandes, K., Vinagre, P., & Cortez, P. (2015). A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News. Progress In Artificial Intelligence, 535-546. http://dx.doi.org/10.1007/978-3-319-23485-4_53