DS 502 - Project Report

Group 4

Jannik Haas

Sitanshu Sinha

Zane Weissman

Project Description

We have a dataset of articles published by Mashable, and we want to predict the popularity of a given article as measured by the number of shares the article receives on social media. With the rapid growth of online news services and social media, it would be beneficial if we could determine the readers patterns. In particular, many so-called “viral news” sites like Mashable rely heavily on social media shares for web traffic and readership. An understanding of the kinds of articles that are more likely to “go viral” and generate lots of traffic could provide useful information to the media workers (authors, advertisers, etc.).

Data Preprocessing

We received the data from UCI Machine learning repository and it contains 59 extracted features about the articles. These features range from simple counts of words, images, and references, to more complex features such as sentiment analysis, and subjectivity. The goal field for our project will be the number of shares of the article. This is a fairly good representation of the success of the article and can help publishers and advertisers to know whether an article will be popular or not before it is published, as well as give authors an idea of what features of articles are important to increase the number of shares.
We first cleaned the data by omitting empty values and removing nonsensical values. These values included: We removed n_tokens_content <= 0. This meant that the article had no words. We also removed the variable n_non_stop_words. The value of this variable ranges from 0.999999917 to 1. The range of the shares variable is from 0 to 843,300 which is very large and the distribution is very right skewed, with only a very small number of articles having very large numbers of shares. We decided to use the log transformation of the shares as our target since this will give us a more normal distribution of values. All the numerical fields were then standardized. We then split the data into 70% training and 30% testing. Outliers were removed from the testing set to prevent our model from being largely impacted by them. We then built several regression models to predict the number of shares of the articles.

Linear regression

Using R’s built in linear regression model we first fit a model to all the features of the dataset. We then selected all the features with p-values less than 0.01 to reduce the number of dimensions. This produced a model with 21 features and an overall adjusted R^2 value of 0.0993 and a test MSE of 0.8976. The relevant features are as follows:

“n_tokens_content", "num_self_hrefs", "data_channel_is_entertainment", "data_channel_is_bus", "data_channel_is_world", "kw_min_min", "kw_max_min", "kw_avg_min", "kw_avg_max", "kw_min_avg", "kw_max_avg", "kw_avg_avg", "weekday_is_monday", "weekday_is_tuesday", " weekday_is_wednesday", "weekday_is_thursday", "weekday_is_friday", "LDA_00", "global_subjectivity", "title_sentiment_polarity", "abs_title_subjectivity”

As you can see the results were very poor and the model performed only slightly better than just guessing. Since we are using the log of shares and all the data is standardized, the MSE is simply how many standard deviations our predictions are away from the actual value on average.

Subset Selection

Since the manual feature selection performed poorly, our next approach was to use best, forward, and backward subset selection to extract relevant features. The two selection criterion we looked at were test validation error and bayesian information criterion (BIC). For all the selections, the validation error selected a model with a lot more features than the BIC selections and the difference in validation errors were insignificant, therefore we used BIC as our selection criteria to produce simpler, more interpretable models. The results of the different selections were as follows:

	Validation Error Selection	BIC Selection
Best Subset Selection	Number of features: 52 Validation MSE: 0.8974 Adjusted R^2: 0.1014	Number of features: 35 Validation Error: 0.9005 Adjusted R^2: 0.1010
Forward Subset Selection	Number of features: 40 Validation Error: 0.8954 Adjusted R^2: 0.1048	Number of features: 21 Validation Error: 0.9052 Adjusted R^2: 0.1027
Backward Subset Selection	Number of features: 43 Validation Error: 0.8969 Adjusted R^2: 0.1048	Number of features: 26 Validation Error: 0.9039 Adjusted R^2: 0.1024

As you can see from the results above, the subset selection models performed about the same as the linear regression model with features chosen based on the p-values less than 0.01.

Principal Component Regression

Since the subset selection models performed poorly we wanted to attempt a different dimensionality reduction technique. Using the “pls” package from R we built a principal component regression (PCR) model. Instead of using the original extracted features from the dataset, the PCR model first performs Principal Component Analysis (PCA) to extract relevant principal components and then builds linear regression models based on these components. With this model we were able to get slightly better results with a cross validated MSE of 0.8083 with only 12 components. So while this creates a model with less components and variables, it performs slightly better than regular linear regression. However, due to the fact that we are using the principal components, the interpretability of the model is very low.

Ridge and Lasso Regression

We attempted Ridge and Lasso to reduce the number of variables and chose the significant variables for regression. When doing Ridge we got 43 non zero coefficients. But on close analysis we say that all the non zero coefficients were very very small.
Similarly when we did Lasso regression it made all the variables insignificant and made the coefficients zero. In both the cases we were left with just an intercept. So these techniques to reduce the dimensionality did not work.

Splines

We used the R package “splines” and its function “ns” to fit splines to this dataset. Despite the increased flexibility of splines, we still predict the log of shares rather than the absolute number of shares. We start by fitting splines for all variables; however, the mean squared error is extremely high and the model is only significant for a handful of variables:

> "n_tokens_title", "n_tokens_content", "num_hrefs", "num_self_hrefs", "average_token_length", "num_keywords", "data_channel_is_lifestyle", "data_channel_is_entertainment", "data_channel_is_bus", "data_channel_is_socmed", "data_channel_is_tech", "kw_min_min", "kw_max_min", "kw_avg_min", "kw_min_max", "kw_avg_max", "kw_min_avg", "kw_min_avg", "kw_max_avg", "kw_avg_avg", "weekday_is_monday", "weekday_is_tuesday", "weekday_is_wednesday", "weekday_is_thursday", "weekday_is_friday", "LDA_00", "LDA_01", "LDA_02", "global_subjectivity", "global_rate_positive_words", "min_positive_polarity", "title_subjectivity", "title_sentiment_polarity", "abs_title_subjectivity"

Refitting the model with only these variables, we can now measure the effects of using different numbers of knots in the splines. There is a rare bug[1] in the default computation of knot locations by quantiles which sometimes causes an error during fitting, so we compute knot points evenly spaced between the minimum and maximum values of the predictor in the test set. MSEs for evenly spaced knots were as follows:

Number of evenly spaced knots	Normalized log MSE
1	.9141
2	.9098
3	.9073
4	.9077

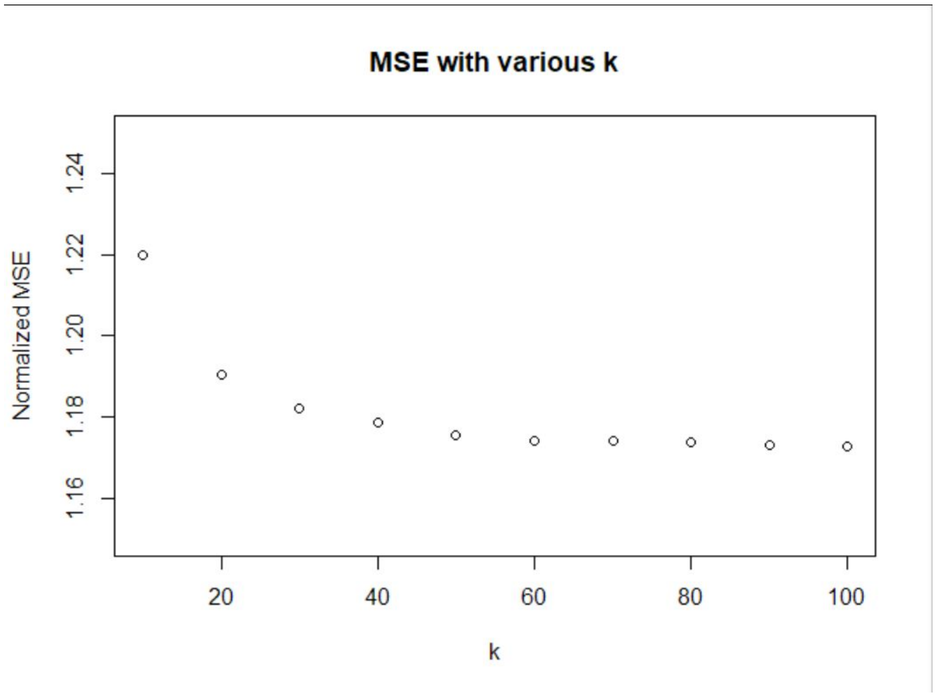
[1] See: https://stat.ethz.ch/pipermail/r-devel/2011-May/061035.html. It is hard to confirm the exact mechanisms producing this error but this explanation appears consistent with our observations of the error.

5	.9082

Because there were only five significant numeric predictors, we also attempted to pick knot points by hand based on plots of these predictors against the log of shares. We found similar MSEs with this approach, but never achieved better results than with 3 evenly spaced knots.

KNN

We used KNN as a regression model by averaging the values of the *k* nearest points rather than using them to vote on classes. We found that higher substantially higher *k* than those typically used for classification produced better results. However, the normalized MSE levels out at well above 1 standard deviation - that is, the model cannot predict shares better than guessing the mean every time. The following figure shows normalized MSE at various *k* values.



Random Forest

Random forests are another model typically used for classification that we attempted to use for regression. Much like KNN, where the results of trees are typically used to vote for a class, we instead use an average. Again, like KNN, we were unable to reach a normalized MSE lower than 1, so the model is no better than a guess. When using an extremely high number of trees (around 8,000 or more) and a sufficient number of predictor variables per branch (4 or more), the MSE appeared to level out at around 1.05. Obviously each individual tree is doing very little work at this point; the model may as well just be returning the mean every time - and if it did, it would be about as accurate.

Linear Regression on Subset of the data

We assumed that all the different models performed poorly on this regression problem due to the fact that the predictive value has such a large range of possible values (0 to 843,300). Therefore, we attempted to fit a model to only a subset of the data to see if performance increased. Articles with shares less than 5000 made up about 85% of the data so we decided to use this as our threshold and then performed a linear regression model on this subset. Similarly to our first linear regression model, we built a model using all the features and then selected features with p-values less than 0.01. This created a model with 20 features and a test MSE of 0.4100. Recalling from our earlier linear regression model with a test MSE of 0.8976, this cut our error down by over half. This model can still be very useful to a large number of users, as most of the articles will not receive shares of over 5000.

Classification

Why Classification?

Since the regression models performed poorly due to the large range of target values, we wanted to attempt to transform the problem into a classification problem. In this case we decided to predict whether an article would be popular or not. We set the threshold at 1400 shares since that evenly split the data in half of popular and not popular. All articles over 1400 would be labeled popular and those under 1400 would be labeled unpopular.

KNN

We used KNN for different K's. Got the best result of KNN with K = 10

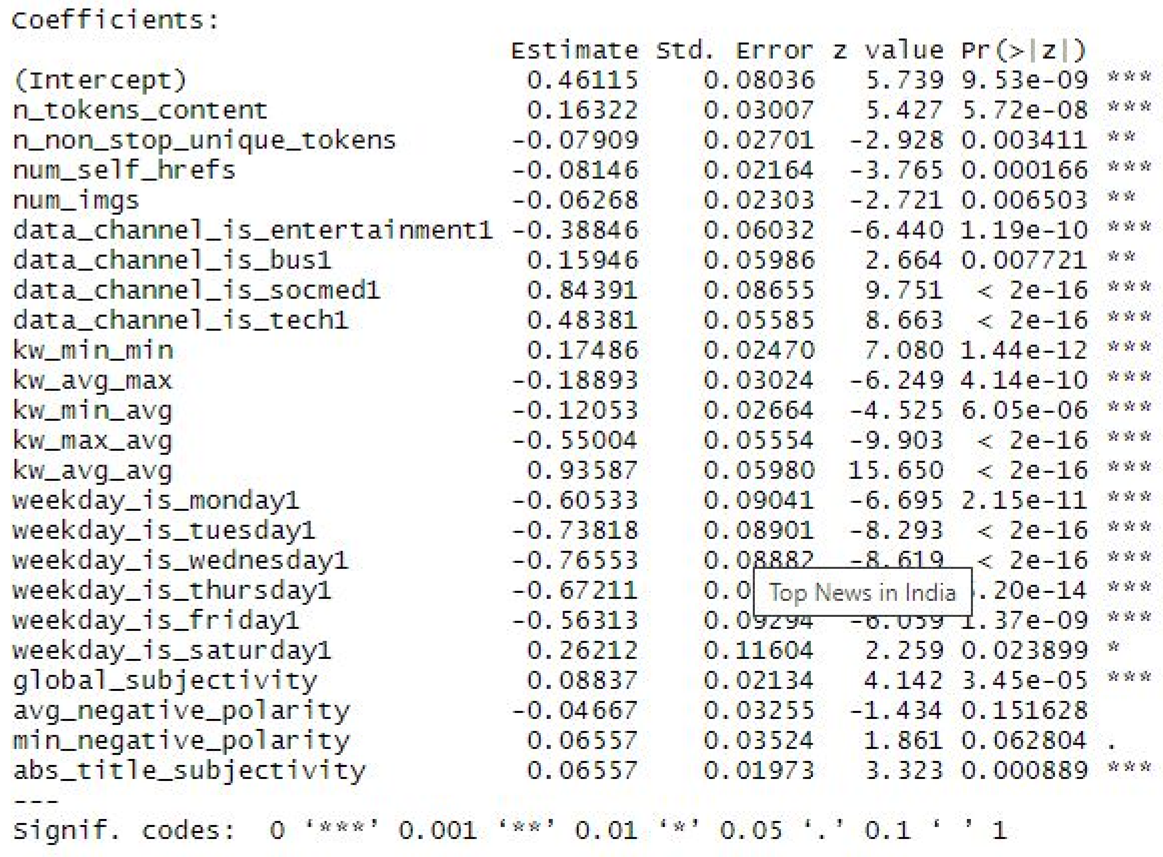
```
        Confusion Matrix
                knnFit   0      1
                   0    1649   1143
                   1    882    1291
        Accuracy :-- 59.2145
```

Logistic Regression

We ran logistic regression with all the independent variables and then checked the p value of all the independent variables. The following variables were significant in the model:

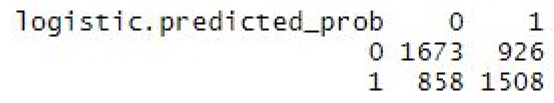
N_tokens_content,n_tokens_content,n_non_stop_unique_tokens,num_self_hrefs,num_imgs,data_channel_is_entertainment,data_channel_is_bus,data_channel_is_socmed,data_channel_is_tech,kw_min_min,kw_avg_max,kw_min_avg,kw_max_avg,kw_avg_avg,weekday_is_monday,weekday_is_tuesday,weekday_is_wednesday,weekday_is_thursday,weekday_is_friday,weekday_is_saturday,global_subjectivity,avg_negative_polarity,min_negative_polarity,abs_title_subjectivity

We built the logistic regression model again with only the significant variables:



```
Coefficients:
                               Estimate Std. Error z value Pr(>|z|)    
(Intercept)                     0.46115    0.08036   5.739 9.53e-09 ***
n_tokens_content                0.16322    0.03007   5.427 5.72e-08 ***
n_non_stop_unique_tokens       -0.07909    0.02701  -2.928 0.003411 ** 
num_self_hrefs                 -0.08146    0.02164  -3.765 0.000166 ***
num_imgs                       -0.06268    0.02303  -2.721 0.006503 ** 
data_channel_is_entertainment1 -0.38846    0.06032  -6.440 1.19e-10 ***
data_channel_is_bus1            0.15946    0.05986   2.664 0.007721 ** 
data_channel_is_socmed1         0.84391    0.08655   9.751  < 2e-16 ***
data_channel_is_tech1           0.48381    0.05585   8.663  < 2e-16 ***
kw_min_min                      0.17486    0.02470   7.080 1.44e-12 ***
kw_avg_max                     -0.18893    0.03024  -6.249 4.14e-10 ***
kw_min_avg                     -0.12053    0.02664  -4.525 6.05e-06 ***
kw_max_avg                     -0.55004    0.05554  -9.903  < 2e-16 ***
kw_avg_avg                      0.93587    0.05980  15.650  < 2e-16 ***
weekday_is_monday1             -0.60533    0.09041  -6.695 2.15e-11 ***
weekday_is_tuesday1            -0.73818    0.08901  -8.293  < 2e-16 ***
weekday_is_wednesday1          -0.76553    0.08882  -8.619  < 2e-16 ***
weekday_is_thursday1           -0.67211    0.0 Top News in India .20e-14 ***
weekday_is_friday1             -0.56313    0.09294  -6.059 1.37e-09 ***
weekday_is_saturday1            0.26212    0.11604   2.259 0.023899 *  
global_subjectivity             0.08837    0.02134   4.142 3.45e-05 ***
avg_negative_polarity          -0.04667    0.03255  -1.434 0.151628    
min_negative_polarity           0.06557    0.03524   1.861 0.062804 .  
abs_title_subjectivity          0.06557    0.01973   3.323 0.000889 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Confusion Matrix :--

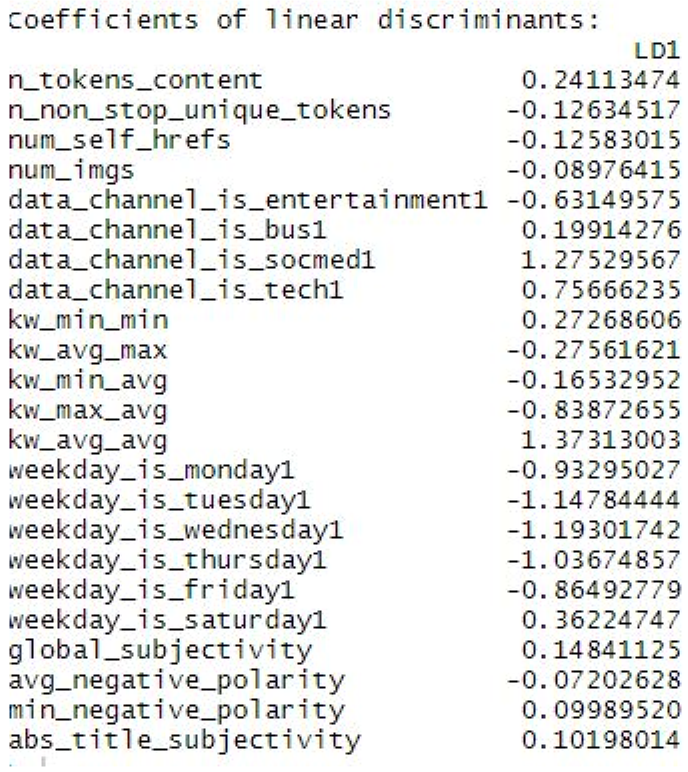


```
logistic.predicted_prob    0    1
                      0 1673  926
                      1  858 1508
```

Accuracy :-- 64.06848

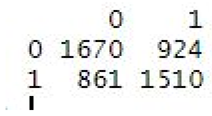
LDA

We built an LDA model with only the variables which were significant in the above model, and got the following coefficients



```
Coefficients of linear discriminants:
                                         LD1
n_tokens_content                  0.24113474
n_non_stop_unique_tokens         -0.12634517
num_self_hrefs                   -0.12583015
num_imgs                         -0.08976415
data_channel_is_entertainment1   -0.63149575
data_channel_is_bus1              0.19914276
data_channel_is_socmed1           1.27529567
data_channel_is_tech1             0.75666235
kw_min_min                        0.27268606
kw_avg_max                       -0.27561621
kw_min_avg                       -0.16532952
kw_max_avg                       -0.83872655
kw_avg_avg                        1.37313003
weekday_is_monday1               -0.93295027
weekday_is_tuesday1              -1.14784444
weekday_is_wednesday1            -1.19301742
weekday_is_thursday1             -1.03674857
weekday_is_friday1               -0.86492779
weekday_is_saturday1              0.36224747
global_subjectivity               0.14841125
avg_negative_polarity            -0.07202628
min_negative_polarity             0.09989520
abs_title_subjectivity            0.10198014
```

Confusion Matrix :--



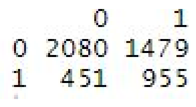
```
      0    1
 0 1670  924
 1  861 1510
```

Accuracy :-- 64.04834

QDA

We then built a QDA model with only the variables which were significant in the above model. Got the following

Confusion Matrix



```
      0    1
 0 2080 1479
 1  451  955
```

Accuracy :-- 61.1279

References

https://archive.ics.uci.edu/ml/datasets/Online+News+Popularity

http://cs229.stanford.edu/proj2015/328_report.pdf

http://faculty.marshall.usc.edu/gareth-james/ISL/