School of Mathematical and Statistical Sciences
Arizona State University

Student: Benjamin Muhlmann

Other students: Wilmer O. Martinez

Course: STP 598 (Machine Learning)

Program: Statistics MS

Instructor: Robert McCulloch

Date: Spring 2019

PREDICTING POPULARITY OF WEB ARTICLES

Wilmer O. Martinez
Department of Mathematics and Statistics
Arizona State University, Tempe
womartin@asu.edu

Benjamin Muhlmann
Department of Mathematics and Statistics
Arizona State University, Tempe
bmuhlman@asu.edu

April 27, 2019

1 Introduction

Four billion people will use the internet this year. We share and consume from a variety of apps and websites every day, liking photos, messaging friends, and getting the daily news.

Sites like Mashable.com make their money on the hub of information that is the internet, pushing content related to every imaginable topic. Web surfers visit Mashable, and the company makes their money from ads shown on every site page. The most profitable articles for the site are the ones seen the most. Many of these page views will come from shares on external social media platforms, like Twitter and Facebook. The more an article is shared, the more people will see the article, and the more money Mashable will receive.

A natural question arises: can we predict how popular an article is going to be? Or is there so much latent information in the content of an article that this task becomes impossible? In the following sections we will explore the question, attempting to classify thousands of articles as "popular" or "unpopular".

2 The Dataset

The data comes from UCI's repository of data sets, and can be found here. The data contains 61 features for 39,797 observations. Each observation is an actual article that was published on mashable.com. The response variable is the number of times each article was shared on Facebook, Twitter, Google+, LinkedIn, StumbleUpon and Pinterest.

The first step in data processing was to encode a binary response column for shares, with a "1" representing popular and a "0" representing unpopular. We manually chose >1,400 shares as a threshold for labeling articles as popular. This is because 1,400 is very close to the median number of shares, enabling us to avoid the problems that arise with imbalanced classes. This same criteria was used by [1].

The explanatory features are diverse and many, but whether they can live up to their name- "explanatory"- is to be determined. Some are self-explanatory, like "Number of words in the title" or "Number of links". Others are more mysterious, like "Closeness to LDA topic 4", but potentially more useful, as these NLP-related features can capture some of the information contained in the content of the article. A full table breaking down the features by category can be seen in the table of 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)

Figure 1: List of features by category

3 Data analysis and modeling

Data exploration and feature selection

In order to understand how each variable is related to the number of shares, we can calculate simple correlations. In Figure 2a we display the highest (absolute value) correlations with shares by category. We can see there that the correlations in general are low. Now, since we are interested in classification analysis we can check correlations between our target variable "popular" or "unpopular" and the variables.

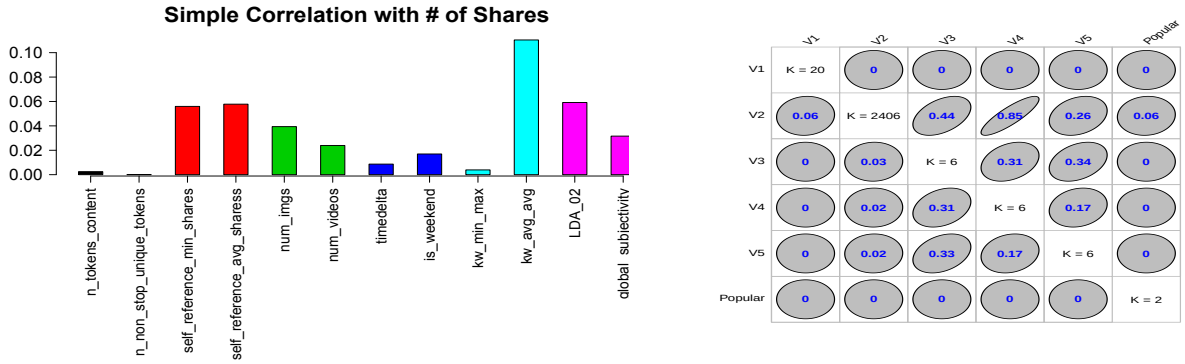
To study these correlations we use the Goodman and Kruskal's τ measure [2]. On a high level, Goodman and Kruskal's τ is a measure of association between two variables, x and y. This association measure is defined by equation (1), where $V(y)$ denotes a measure of the unconditional variability in y and $V(y|x)$ is the same measure of variability, but conditional on x, and its expectation is taken with respect to x.

$$\alpha(x, y) = \frac{V(y) - E(V(y|x))}{V(y)} \tag{1}$$

To use this measure, we first had to transform the variables which were not categorical. We created a boxplot for each numeric variable as a starting point. Basically, we create a categorical variable with six categories using the following scheme:

1. $x \leq L$
2. $L < x \leq Q1$
3. $Q1 < x \leq Q2$
4. $Q2 < x \leq Q3$
5. $Q2 < x \leq Q3$
6. $U \leq x$

where $Qi's$ are the quantiles (25, 50, 75), respectively, $L = Q1 - 1.5(Q3 - Q1)$, and $U = Q3 - 1.5(Q3 - Q1)$. A sample result of the association measure between our target variable and the first category group of features is shown in the Figure 2b (V1,..., V5 correspond to the variable categories in the table above). In this graph- although the association between the variables and the target is not so high- we can identify that there are important associations among the variables. Similarly happen with the other categories. Based on the high associations found among some of the variables we decided to explore some subgroups of features to find the best model.



(a) (b)

Figure 2: (a) Highest correlations between the features and the number of shares. (b) Goodman and Kruskal's Tau and Kendall's Tau Matrix (For the subgroup of features Words, see Figure 1)

Intro to models

We used seven classification models: Naive Bayes (NB), K-Nearest Neighbors (KNN), Boosting, Random Forest (RF), Linear Discriminant Analysis (LDA), Logistic Regression, and Deep Learning (DeepL). For each model we consider 9 subgroups which accumulate from 10% of the variables until 90%. These subgroups were chosen based on the association level according to Goodman and Kendal's τ association. In addition, we evaluate the models with the complete set of variables both without transformation and with the transformation (in some cases categorical transformation and other with the min-max scaled). In the table 1 we summarize our results. Before getting to the results we'll discuss more details on the implementation of each model.

Naive Bayes (NB)

For the NB model we use the subgroups with the categorical transformation. Also we evaluate the model with the complete set both with an without transformation. Because processing time increases as the number of variables increases, we implemented the rolling windows scheme with a training window size of 10,000 and test out-sample of size 1,000 as was done in [1]. At the end, we get 29 prediction sets each of size 1,000. The measures of evaluation (Accuracy, Recall, and AUC) were based on the the complete set of 29,000 predictions. Something interesting we found with the NB model was that when we used the complete set the Accuracy was 0.61 and AUC was close to 0.65 as was gotten in [1]. However, when we run the model with 60% of the variables, the AUC increased to 0.69. We consider this result pretty important because not only we did we improve the prediction of the NB model but also because we identified a subset of variables that is more important for the prediction of popularity. Further, it is relevant to mention that the improvement of the NB model was due to the fact that the inputs were categorical, which lends itself to classification. [3].

Boosting

For boosting, we used 10-fold CV. As with NB, we found that the best subgroup was with 60% of the vari-

ables[1]. However, the this subgroup was without transformation. This result again is pretty close to the result found in [1].

Linear Discriminant Analysis (LDA)

With this model similarly to NB the best subgroup was 60% of the variables and with the categorical transformations. Something important to mention with this model is that this model has a better performance than the Boosting model and its processing time is close to 10% of that needed for Boosting.

Logistic Regression

For all subspaces of the X variables, 10-fold cross validation was used.
We used scikit-learn's LogisticRegressionCV.
As a baseline, we min-max scaled all the predictive variables (a requirement for logistic regression) and threw them all into the model with the intention of comparing results from models fit on a subspace of the X's. The most impactful tuning parameter was the algorithm used to optimize the regression. Scikit-learn's newton-cg solver converged quickly and gave the best out-of sample performance. Newton-cg stands for the Newton Conjugate Gradient optimization algorithm. ** Explanation of algorithm **. Using all the variables gave surprisingly good out-of-sample results, averaging above 66 percent accuracy with different random seeds.

After the baseline model, we figured results would improve with a subset of variables. Wrong! We tried getting the n-most correlated variables (n $\in \{10, 20, 30, 40\}$) with the response variable (number of shares). We tried getting an 80% subset from the Kendall's Tau Matrix. We tried getting the n-best variables as determined by feature importance from an untuned extra-trees classifier. Using the same tuning parameters, we could never achieve higher than 65% accuracy indicating... **reason for full set of variables performing best**. ROC curve is below

KNN
For all subspaces of the X variables, 5-fold cross validation was used. This was preferred to 10 because computation quickly gets expensive with a high number of neighbors in 50 dimensional space
We used scikit-learn's KNeighborsClassifier and GridSearchCV to choose the best k.
Similar to the process with Logistic Regression, we started by using the entire set of X variables and just using a grid of k's from 2 to 10. 9 neighbors worked best, with an average accuracy around 61%. For KNN, using the 80% subset of variables from the Kendall's tau matrix actually increased performance slightly, boosting accuracy by about 1% on average. KNN was still the worst model by far, as the complexity of article popularity just couldn't be found reliably in neighborhoods. This might be because the variables were so sparsely distributed in high-dimensional space.

Random Forests

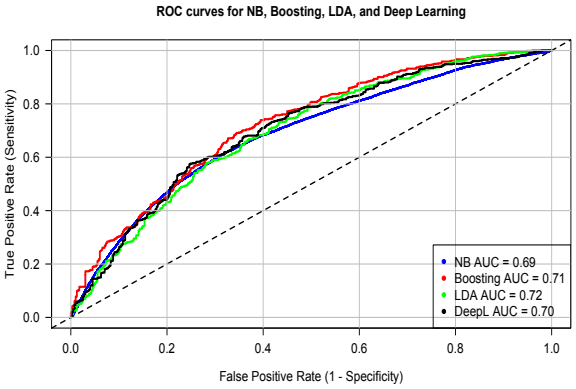
Random Forests We used 5-fold CV as computation was again expensive, especially with a tuning grid with 4x3x4=48 total parameter options.
With Random Forests we were concerned less with the subset of variables to throw into the model, as Random Forests will naturally determine which variables are actually useful for making decisions about the response variable. We used a tuning grid of max depth $\in \{2, 5, 10, 15\}$, number of trees $\in \{100, 500, 1000\}$, and min_samples_split $\in \{2, 3, 4, 5\}$. Max depth limits the depth of the tree. Number of trees determines the number of decision trees to average as part of our forest. Min_samples_split determines the minimum number of observed samples in the training set required to actually make a split at any node. Random forests performed well with best parameters $\{15, 500, 3\}$, giving average accuracy of 67.9 percent.

Random Forests was the best-performing algorithm, beating the other models with little tinkering. A comparison of the three models fit in python is below.

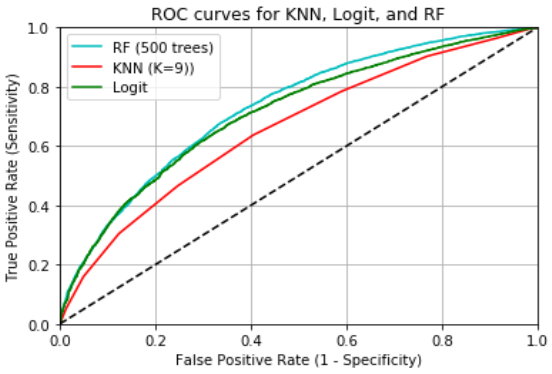
Deep Learning

For Deep Learning, we used two hidden layers with 10 nodes in each layer, 500 epochs and the activation function $Tanh$. Likewise to the previous models, we evaluate this model with each one of the define subgroups finding that the best subgroup (with 60% of the variables) had a little better performance than the model with the complete data base.

[1]This subset is the same used in NB



(a)



(b)

Figure 3: ROC Curves

Accuracy	Recall	AUC	Model
0.63	0.48	0.69	NB SG
0.64	0.54	0.69	NB T
0.66	0.59	0.71	Boosting SG
0.66	0.59	0.71	Boosting T
0.66	0.62	0.71	LDA SG
0.66	0.59	0.72	LDA T
0.61	0.33	0.70	DeepL SG
0.62	0.35	0.70	DeepL T
0.62	0.64	0.66	KNN (9N)
0.66	0.71	0.71	Logit
0.68	**0.74**	**0.73**	RF (500 Trees)

Table 1: Comparison of models

4 Conclusions

After evaluating the seven models with different percents of the explanatory variables we identify that in most of the cases 60% of the relevant variables is enough to predict the target variable with good accuracy. Also, we got to improve the accuracy of the NB model by the categorical transformation adopted in this paper. The association measure between the variables and the target was definitely crucial at the moment to find the best subset. Judging by the ROC curve, we can conclude that all the models but KNN have a similar behavior, with Random Forests performing best.

References

[1] Fernandes, Kevin and Vinagre, Pedro and Cortez, Paulo. A Proactive Intelligent Decision Support System for Predicting the Popularuty of Online News. *Springer International Publishing Switzerland*, 2015.

[2] Somers, Robert H. A Similarity Between Goodman and Kruskal's Tau and Kendall's Tau, with a Partial Interpretation of the Latter. *Journal of the American Statistical Association Vol. 57, No. 300, pp. 804-812*, 1962.

[3] Brett, Lantz. Machine Learning with R *Open Source, Panckt Publishing, BIRMINGHAM - MUMBAI*, 2013.