Predicting Online Content Shares with Binary Classification Models

By Mohammad Azim

**Introduction:**

The aim of this project is to create a classification model to predict the shares a news link will receive based on numerous factors broadly contained in 4 categories for the provided dataset. The data was attained from the UCI Machine Learning Repository and it is recorded by the online content site, Mashable. The number of observations are 39,644 and for attributes its 61: 59 predictive attributes, 1 trivial (url), 1 variable of interest (shares). For the classification task, we removed the feature URL This is intended to simplify our task to numeric variables. Also, features related to the title of the article are present in the dataset and the title is the only nontrivial information available from the URL variable. The y-variable (shares) was transformed into a binary variable with a threshold of 1400 (the median) giving it a 50:50 ratio or distribution. The features seem to be grouped by broad categories. I identify them as:

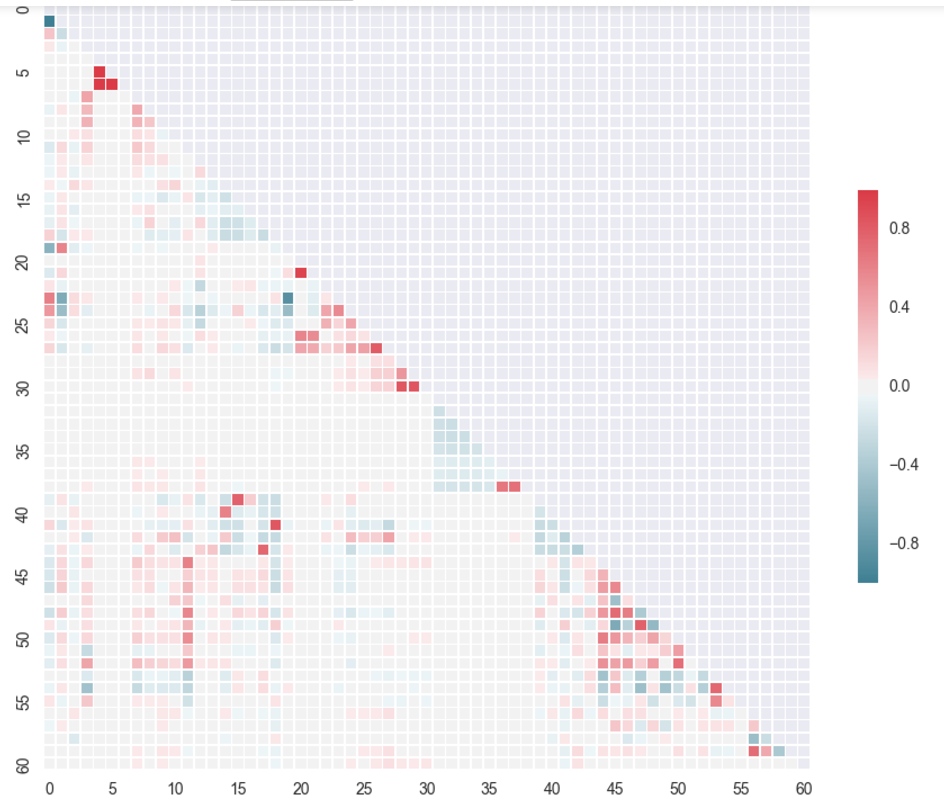
1)Content Quantified {1-12}: title, content, links, etc.

2)Category Information {[13-19],[39-43]}: which of the 5 subject channel is it, closeness to the topic

3)Access Information {20-39}: key words, Mashable links, day of week

4)Sentiment {44-59} positive negative sentiment

Figure 1



In Figure 1, a correlation matrix of the dataset is shown to get a brief understanding. For the most part, the data seems to be independent but the inner category variables do seem to have a relationship. The red and blue are mostly seen along the top corners which signify the correlation between the variables. The variable index above starts at 0, with the 0 variable being time delta (how much time has passed between the article’s publish date and the measurements). This variable is not in the 4 broad categories but might be of use. Finally, 60 is the y-variable.

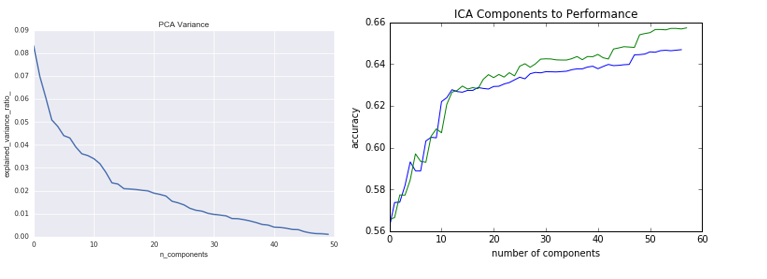
The classification problem presented is significant in its implication and being able to predict the shares can bring added value to optimize both the user experience and content creator’s efforts. The process of knowing what article you’ll like and like enough to share is ambiguous. Still if we can improve even slightly it still adds value to and sheds a little more knowledge to this process.

**Methods**:

This analysis was organized in the following steps: Feature Selection -> Model Selection -> Optimization -> Results and Evaluation. Though the process is not always chronological in practice, it is a useful organization method for reporting. In practice often the results dictated former steps such as optimization and model selection. The data set was split into training and test with a 70:30 split respectively, which left about 28K observations for our training set. Some tests were done with as little as 5K data, mainly to rapidly test parameters, and when evaluating on training, the full test set was used (which is around 12K). Even if the full 59 parameters is in use, the number of observations provides an abundance of data to train from and the 70% 30% split shouldn’t adversely affect either accuracy. However to build a better model is more significance than to make sure it classifies correctly, therefore a 70% was allocated to the training set.

In the feature selection step the two components Analysis: Principle Component Analysis (PCA) and Independent Component Analysis (ICA) were chosen. Component Analysis allows the creation of variables that are not redundant so the learning can occur more efficiently. In contrast to just removing variables, the components keep more of the information in fewer features. Choosing how many components to keep from component analysis can be an arbitrary choice. For PCA, which chooses components based on the dimensions with the greatest variance, the explained variance ratio is a metric that was used to select the best number of components. For IC, the decision is a less straight forward so the performance was used to select the number of components. A simple linear support vector machine with L2 regularization in its default Python Scikit Learn implementation was used listed as LinearSVC.

Figures 2



For the PCA the explained variance ratio was exponentially decreasing but it can be observed in figure 2 that there is inflection point 16 components, thus the first 16 PCA was saved for the next step. For ICA at first only the accuracy of the training set was considered (depicted in blue in figure 2). The initial expectation for this test was to receive accuracies that increased at first but decreased with higher dimensions, however this did not occur. It was assumed that the results could be potentially explained by over fitting, the reasoning being: the greater the number of dimension; the easier it is to linearly separate. However, the shocking results of this test showed that the accuracy on the test set (depicted in green) set was about the same with 20 features but it kept on doing better with more components. The only evidence-based choice for feature selection was to take the first 58 components which is just 1 less than the original feature set. Feature reduction is sometime taken as an absolute, but with a large enough data set more features might add more value, especially for a linear separator.

In the model selection step, we already have a 65% accuracy for the Linear SVM, and optimizations will be done to it to attempt to outperform the original. Naïve Bayes will also be used since it has a reputation as a benchmark for this type of dataset. This data set is not a pure text mining data set, but it does have the classical features that many text mining data sets have, such as tokenization and sentiment analysis. It was said by Dr. Yanjun Li (a text mining researcher), that Naïve Bayes is often the benchmark in text mining, and SVM’s usually perform the best. Therefore, these are good models to implement.

Questions that comes to mind for implementing SVM are:

1. How does the SVC do linear versus Radial Basis Kernel (RBK) for both feature sets?

Advantage of a linear separator is its smoothness, which is less prone to overfitting. However, it might have issues separating overlapping data belonging to different classes. As for RBK, it can use infinite dimensions to fit the data set. This might make it prone to over fit for outliers observations.

1. How does it do with different slack variable strengths?

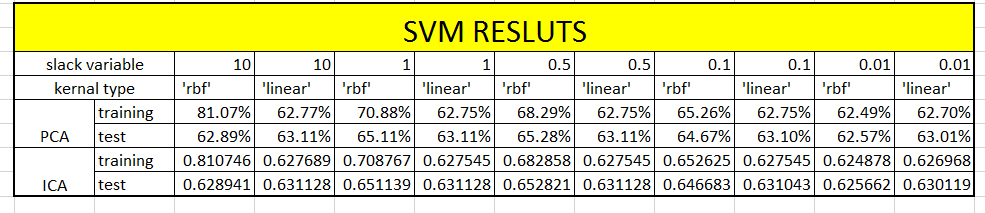
As for the linear separator, it is expected that the slack change will do much less to change it than it will for the RBK. For the slack variables for both models, we will try 5 different values [10, 1, .5, .1, .01]. With a higher slack variable, the regularization penalty is lower. So RBK will likely over fit it the slack is high, but it can also underfit if it’s too low.

The second model implementation is the Naïve Bayes classifier, specifically the Gaussian Discriminant Analysis Naïve Bayes. This implementation was created using Python’s scientific libraries numpy and scipy for it’s use of arrays and mathematical functions. The prior probability was tuned to generate different results, the 3 priors used was, count based, Laplace smoothing which add 1 to the numerator and adds k the number of y variable to the denominator, and explicit specification of 50% 50% probability set. The 50% 50% explicate priors is justified by the binarization process of the data, which explicitly split using the median as the threshold. Therefore, it was clearly intended to be 50% 50%. For the most part I except vary little change in performance with the three priors, but it is there to tune and observe. Since this is a generative model it has few parameters that can be tuned.

If SVM performs well on one data set over the other it might not be the case for Gaussian Naïve Bayes. Support Vector Machine and Gaussian Naïve Bayes’ performance are not agnostic to the data. In fact the assumptions in Gaussian Naïve Bayes makes it best suited for Gaussian distributed independent features. SVC’s main concern is with separation in the support vector point. Two classifiers might perform different on the two feature sets (PCA and ICA). So it will be tested on both

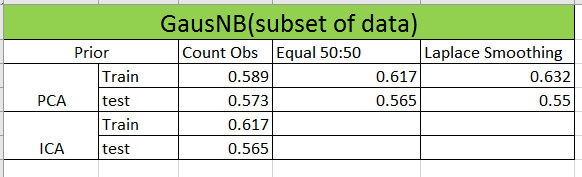
**Results:**

Table 1:



The linearSVC implementation of SMV used in PCA seems to defer in the SVC implementation and this is due to hyper parameter such as set iteration. However, the RBK kernel with SVC implementation has more regularized results, specifically the .5 slack rbf (rbf is how scikit learn refers to RBK). The trainset should definably do better than the test since that is what it is trained on. It is likely the with enough experiments to over for to test set (or p-hack). Since the accuracy on the test set is miniscule it would be wise to choose the ref kernel SVM. Also it can be obverse that rbf tends to over fit with larger slack variables. Below .5, its performance is adversely effected.

Table 2



When modeling the GaussianNB classifier, a more efficient approach was used to search through the possible options. Initially the parameters we’re searched using a small subset of the data to reduce runtime, and after the first result ICA was eliminated since it performed worse than PCA. Then PCA with normal the normal prior option was selected as the optimal model. The performance of the classifier on the full train set yielded 62% accuracy. Though this is an improvement over 57%, still the SVM classifier outperformed the benchmark Naïve Bayes, as hypothesized.

**Conclusion:**

In conclusion, after searching though the models and hyper parameters, the best model was found and selected. Our selection was the Support Vector Machine with rbk kernels and L2 regularization strength of .5 using 58 components selected by ICA. In the process, it was demonstrated that Support Vector Machine tends to outperform Naive Bayes, adding another example to the common notion. The classification problem presented vary modestly accuracy, however with 65% of the 12,000-unobserved data having been classified correctly, it shows that learning has occurred. As stated previously the classification problem presented is significant in Its implication, and ambiguous in its process, and our modest learning still adds more knowledge to this process.