Kaggle Report-Team ABBABA OH NO ABBBAB

Nick Halliwell, Aina Lopez, Yaroslav Marchuk

March 14, 2016

Introduction

Our team consists of Nick Halliwell, Aina Lopez, and Yaroslav Marchuk. In this competition, we were given both a training and test set consisting of features of various web links from mashable.com. We were asked to predict whether the website link would fall under one of five potential categories: obscure, mediocre, popular, super sopular, and viral. We ran several algorithms to make these multi-class predictions, and found that a random forest with tuned parameters outperformed all other algorithms. Below we underline the approach we took to this competition.

Method

1. Create new variables
The first step we took was to examine the data, and determine what kind of variables we could generate from the given data, and standardize our features. Of the variables we were given, we decided that the url of the articles included useful information that could be extracted. An example of the url is the following:

http://mashable.com/2014/01/12/game-of-thrones-season-4-trailer

From the url, we took the year, day and month it was published as well as some keywords of the content of the article. Using text mining techniques, we collected all the keywords of the article titles. With this we created new variables indicating whether a specific word appears in the title of the article. Given that we have many keywords, we created a threshold for variables, meaning we created variables only for keywords that have appeared in at least 150 observations.

2. Feature Selection
We used Regularized Random Forest (RRF package) to choose the best feature set.

3. Train the model
We trained different models optimizing their parameters, explained in more detail below.

4. Evaluate the model
In order to evaluate our models, we used 5-fold Cross validation. Here, we randomly subset our training data into 5 folds, training the model on 4 folds and testing the model on the last fold. This process is repeated 5 times, with each fold used once for training. To stay consistent, we used 5-fold Cross validation on all algorithms used in competition.

Models

- **Random Forest:**
  We decided to implement a regularized Random Forest, as this algorithm typically perform well on large datasets. Random forests often do not overfitting, however, the computational speed of this algorithm can be slow. The regularized Random Forest performed feature selection for us, and then fed those relevant features into a Random Forest. We optimized the algorithm for the number of trees used, the minimum size of terminal nodes, and the number of variables randomly sampled for potential splits. Ultimately this was our most successful model.

- **K-nearest neighbors:**
  We implemented a k-NN algorithm, trying both a function written by one of us, and an R package installed from CRAN. In both cases, the k-NN performed poorly. In terms of predictions, it's accuracy was lower than most of the other algorithms, and it was very expensive to compute. We ran the algorithm using the distance metric of the Euclidean norm, and optimzed for different values of k, yet the algorithm could not outperform our Random Forest.

- **SVM**
  In addition to k-NN and Random Forest algorithms, we implemented a Support Vector Machine using a polynomial kernel. We optimized weights for the 5 different classes, giving higher weights to classes 4 and 5. The svm was outperformed by the Random Forest, and given the large time complexity of the algorithm, we turned to other methods to use in competition.

- **Boosting**
  Lastly, we tried boosting methods. Initially, we tried gradient Boosting (XGBoost), however, the amount of parameters to optimize was very large and did not observed important changes while tuning them. The accuracy did not outperformed Random Forest hence we decided to discard it.
  We also tried Generalized Boosted Regression Modeling (gbm) using h2o to speed up computations. In this case, we selected variables in a different way, using the variance importance attribute of gbm. The results were good and very similar to random forest.

Choosing Parameters

In order to optimize the parameters of each algortihm, our approach was to create a wide range of reasonable values, with large gaps between them. We then cross-validated all the models and chose the best. In addition, we created a new range of values arround the optimal value found, but with less spaced intervals. We cross-validated again and choose the optimal value. We repeated this procedure as many times as needed.

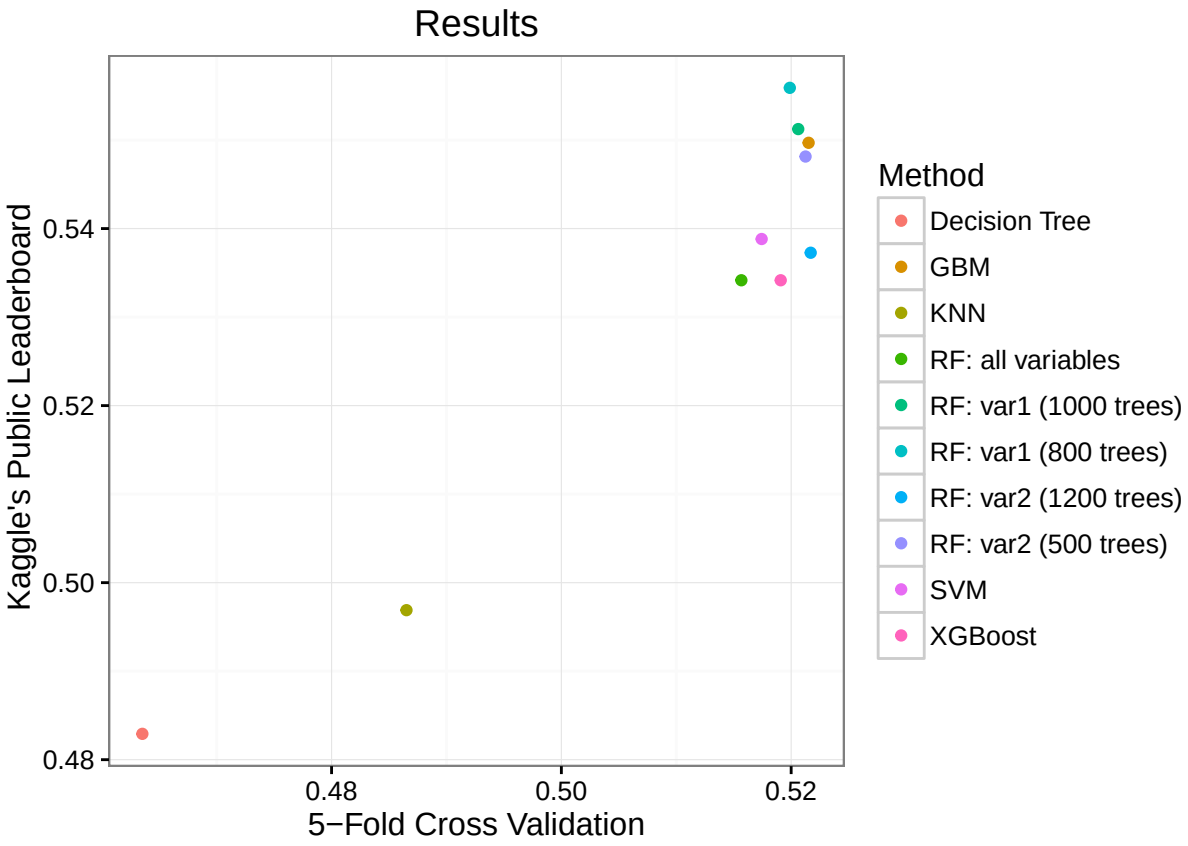
On average, only a couple of rounds of this method were needed. However, with more complex models, this method became very inefficient. We observed that selecting each parameter separetely is not enough. Instead in order to optimize the parameters, we had to select them together. This is one of the main reasons why we decided against use Support Vector Machines.

Results

We have plot some of the model accuracies (Kaggle public Leaderboard vs our 5-fold Cross Validation). The models that we are displaying are listed below:

- RF: is the random forest. var1 and var 2 use different feature sets selected via RFF using different parameters.
- GBM
- XGBoost
- KNN
- SVM
- Decision Tree

Plotted below are a small sample of all the models that we implemented, but represent the best accuracies achieved by each algortihm family. Notice that, on average, we have tried to optimize Random Forest results, because this algorithm performed the best.



Conclusions and Classifier

As we can see in the previous figure, the best accuracy in Kaggle's public Leaderboard is not necessarily the same as the best accuracy in our 5-fold CV. For our final submission, we decided to choose the two models with best balance of scores in Kaggle and in our CV. Those models were:_____.

Since we have unbalanced data, we have observed the confusion matrices. Most of the models were able to classify with good accuracy classes 1 and 2, but classes 3, 4 and 5 were very difficult to predict. For our models, the connfusion matrices are (vertical is actual and accross is predicted):

- GBM:

.	1	2	3	4	5	Error
1	5418	3942	118	0	0	0.4284
2	2502	11061	201	0	0	0.1964
3	822	3944	946	0	0	0.8344
4	129	681	86	103	0	0.8969
5	3	26	5	0	13	0.7234

- Random Forest:
- DISCUSS Advantages and limitations of our classifier

Additionally, we have performed all the cross validations with the same seed, but

Sadly, we haven't had time to implement all of our ideas (supersampling, optimizing the seed, choose Boosting and SVM parameters more carefully, ...).