Andy Limber Does Kaggle

Andreas Mathew Lloyd, Francis Thomas Moynihan IV, and Feyza Seda Yilmaz

I. INTRODUCTION

Machine learning is a field of study involved in finding patterns in data without requiring a lot of initial specification of such patterns. Provided some data, a machine learning algorithm looks to predict some value (regression) or assign some label (classification) to data points. If labels for a training set are provided, this is a "supervised" problem. Typical learning algorithms for this type of problem include perceptrons, k nearest neighbours, neural networks, and decision trees and their variants, such as random forests and boosted trees. These have their associated advantages and disadvantages, such as computational intensity, resistance to outliers, interpretability, and flexibility. Neural networks, for example, are a famously "black-box" in nature.

The problem at hand is to classify popularity of articles from the website $mashable$. The original data were provided by the UCI Machine Learning Repository and includes data about the content, sentiment, and publishing details of the article, and so on. Popularity was measured by the number of shares, categorised on a scale of $1-5$, with 1 being the least popular article, and 5 the most.

This is a supervised learning problem and was primarily tackled using decision tress of different variations. The original reasoning for this was the flexibility provided, and because no obvious, initial relations were apparent.

II. THEORY

The methods discussed will be the random forest and extreme gradient boosting (XG boost) algorithms. Both of these methods rely on decision trees.

Given an $N \times M$ data set of features, a decision tree splits some set of data in half, such that (using a majority classifier) the split minimises error. This is run recursively, building up the "tree", where the "branches" are the splits, and "leaves" are the classification of that split. If allowed to run until each leaf contains only one data point, any decision tree will severely overfit the data, so several conditions are imposed to limit growth. Typical examples are limiting the depth (how many leaf nodes), and how many data points any given node must have.

To improve, several modifications can be made, some of which are implemented in their own algorithms and are discussed here.

A. Random Forest

Random forest improves classification by creating many trees and then averaging them. Each tree is created using a random sampling of both data points, as well as feature columns. Then a majority vote is taken among all trees to give the final classification. Key control parameters for a random forest are then the number of trees and the number of features that are randomly sampled.

The main advantages are introduced in this randomness. Since data points and variables are randomised for each tree, different combinations and different emphasises are considered. This reduces overfitting in general, but is also ideal when little is known about the data. For this reason, it was considered as one of the main algorithms for training.

B. XG Boost

Extreme gradient boosting is a form of boosting trees. Boosting a decision tree is a way to improve a decision tree that also uses a combination of trees. Boosting will run over a number of rounds and at each round, a weak classifier is trained on the data. Then in the next round, the classifier is re-trained and improved by taking into account the problems of the previous round classifier. The results of all classifiers are finally aggregated in some way, similar to random forest. A weak classifier is used to reduce the general error, which is related to the root of the V_c dimension.

The similarities with random forest mean that many of the advantages are shared. For this reason it was decided to experiment with both of them, as very little was known about the data set, and little initial specification was required. This is in contrast to some algorithms, such as multinomial regression, where terms need to be specified to interact and picked beforehand.

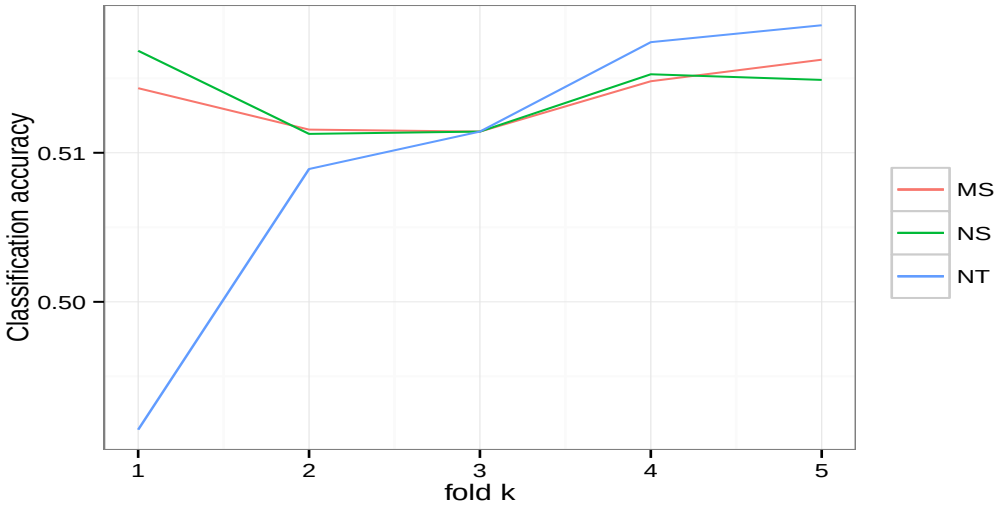

FIG. 1: The evolution of classification accuracy with changing parameters of the random forest, where NT is the number of trees, MT are the number of variables randomly sampled, and NS is a depth control parameter.

The final classifier was a combination of these two methods, where probability of predictions were used to determine the label. This will be further discussed later.

III. NUMERICAL EVALUATION AND OPTIMISATION

Accuracy of different methods was evaluated by cross validation. Typically a ten fold cross validation was used where the test set was of size 9000. Ten fold was used because accuracy was relatively unstable for a lower number of folds.

A. Algorithm parameters

To optimise the parameters, grid searches were used across different sets of values. The combination that gave the best classification accuracy over the cross validation was chosen. As an example of this, Fig.1 shows how cross validation evolves with parameters of the random forest.

It can be seen that, as parameters affect the algorithms in complex ways, classification accuracy does not evolve straightforwardly. This makes optimisation a demanding task, as a relatively dense grid of all important parameters must be searched over. This task is more difficult for the XG boost, as there are more parameters. There is also a trade off, as parameters that improve overfitting often cause longer run times.

B. Added features

To improve classification, features should be informative. New features can be generated and tested from known variables. Ideally they would represent some facet of the problem and so more easily understood. It can be seen in Fig.2 that the selected features improved the classification significantly. This is important for both individual classifiers and so helps the final combination.

The process of finding features was to think of variables that could help improve the classification, then run a cross validation to see changes. This was a relatively intensive task as different combinations had to be considered. For this purpose, it was useful to use the importance plots that the random forest algorithm produces, which indicates how good variables are for reducing error.

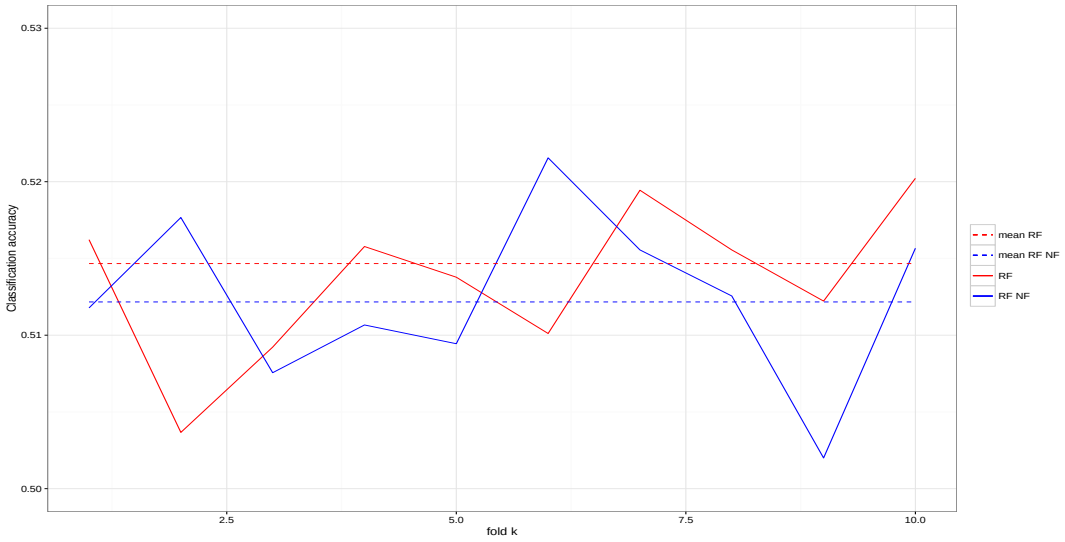

FIG. 2: The classification accuracy for a random forest (RF) with and without (NF) generated features. The means are shown as dashed lines.

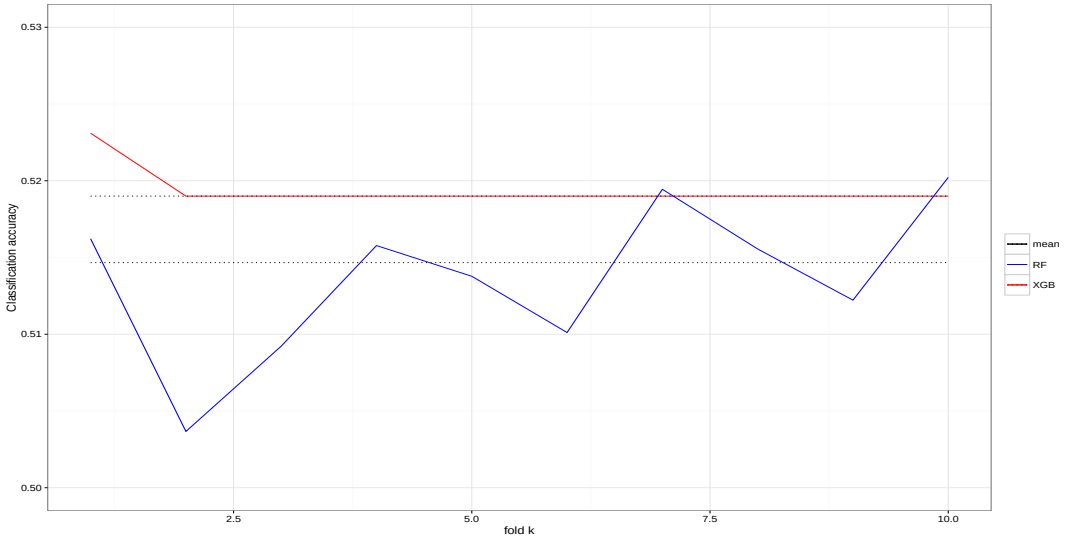

FIG. 3: The classification accuracy for the XG boost (XGB) and the random forest (RF). The means are shown as dashed lines. Both have features added.

C. Combination of models

Initially, several algorithms were tested, with random forest and XG boost achieving highest accuracy. A comparison of the two is shown in Fig.3 and the XG boost is seen to be significantly better. The improved accuracy and especially the stability means that it is attractive to use this as a classifier, giving more predictability in out of sample testing and for Kaggle submission. This likely originates from the way XG boost combines trees, in that the randomness is lower.

To combine the advantages of the two models, the final classifier was considered as a mixture of the two. This was done by selecting the labels predicted by the XG boost, except when the probability of the prediction was below a certain threshold. If they were, the random forest's prediction was used.

This was advantageous because the two different algorithms were better or worse at predicting some of the categories, as seen in Table.I. Ideally, a combined classifier is using the strongest parts of each algorithm and eliminating the weaker parts. The effect can be seen in Fig.4 as the final classifier significantly improves on the two individuals. It is true though, that the classifier has lost the stability of the XG boost classifier. This is not a major problem, however, because the poorer scores are still relatively high.

Method	Class 1 accuracy	Class 2 accuracy	Class 3 accuracy
Random forest	0.46	0.80	0.04
XG boost	0.52	0.76	0.73
Combined	0.5	0.77	0.06

TABLE I: Table summarising the accuracy for different methods on the first three classes. The final two are not shown because all methods have similarly low accuracy, and due to low frequency of these classes, they do not significantly affect the overall accuracy. Note that there are significantly more popularity 2 values than 1 or 3.

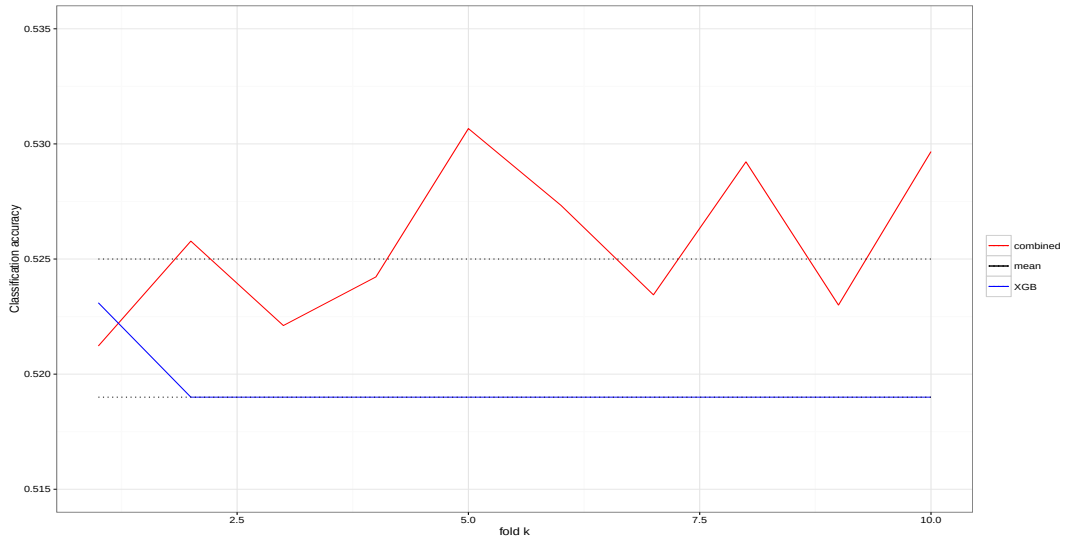

FIG. 4: The classification accuracy for the combined classifier (alKK) and the XG boost (XGB). The means are shown as dashed lines.

As a final tweak to improve the classification, rows in the training set that had popularity 4 or 5 were removed. This was done so that training focused more on the first three labels, which were far more common. The effect of this is seemingly extremely significant, as seen in Fig.5, but this is misleading because the cross validation used does not maintain a constant proportion of labels in the different folds. It was judged, however, that such a significant difference must constitute some improvement.

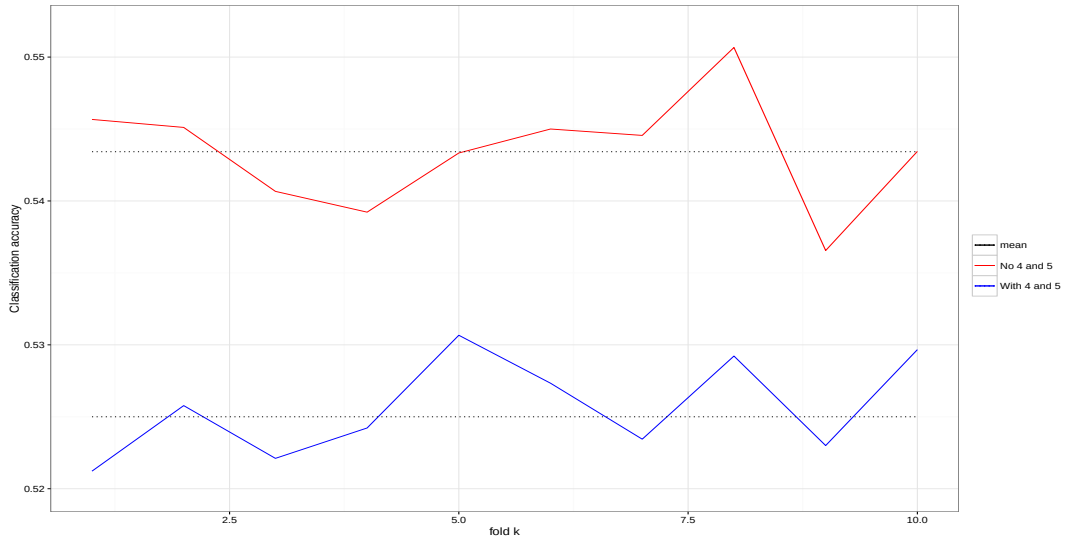

FIG. 5: The classification accuracy for the combined classifier when trained with and without the 4 and 5 popularity rows.

IV. EVALUATION

The performance of the classifier is significantly stronger after modification and optimisation. Using a high fold cross validation means that improvements can be considered true and likely to carry over onto the Kaggle leader board. All of improvements are relatively cost effective too, as the added features and parameter optimisation do not carry the calculation over into a highly demanding problem.

The main issue observed with the classification is that no class is predicted particularly well. The combination of the two classifiers improves on this, but considering the post-combination performance for individual classes is not hugely different, they were likely not the best classifiers to combine. Through further experimentation, a set of classifiers that have "expertise" for the different classes could be found. A combination of experts would significantly alleviate this problem. This is not an easy task however, and would require enumerating a large number of models with different parameters as well as features.

Feature selection and generation would also help improve classification. All of the original features were used because it was found that, despite low importance in the classifier, inclusion improved the classification. As the added features performed well, it is likely that more generated features would further improve accuracy. This was an advantage of using tree based models, in that feature selection was not a pivotal step since many rounds and trees were used. Better selection would help specifying models, however.

The removal of rows from the data can be seen as treating the 4 and 5 popularity ratings as outliers. This was effective, but an improved classifier may not benefit as much. It would be advantageous to develop some method of weighting points to keep focus on labels 1 to 3, but still be able to predict the remaining labels.