

Author classification using tree-based methods, SVMs, KNN, and neural networks

Joe Connelly, Sean Duan & David Reynolds

Introduction

The dataset analyzed in this report is a subset of data taken from a dataset that includes information about books, with each row representing a different book. The features, which are unnamed, are several objective numeric measures of a text, and the goal of this analysis is to identify the correct author of each book. There are nine different authors (i.e., response levels) and 10 features. The training set contains 10,000 books, and the test set contains 5,000 books. Cross-entropy was used to evaluate model performance.

Given that we have very little information about the predictors and that the number of predictors is significantly lower than the number of observations, we decided that a tree-based method, an SVM, a KNN classifier, and a neural network was a good range of models to consider. Linear discriminant analysis, quadratic discriminant analysis, and multinomial logistic regression were also considered, but these methods did not perform as well, and therefore they will not be discussed.

Since the data contains no missing values, no imputation was necessary. However, it was ensured that the response was correctly assigned as a factor rather than a continuous numeric variable. All other preprocessing steps are explained in the model descriptions.

Basic data analysis/description/plots

Variables 6 and 10 appeared to be highly correlated with one another, but we decided not to remove them, as there are only 10 predictors. Variables 1, 2, 5, and 7 are highly skewed, which may pose a challenge. The fact that there are no categorical predictors should help. Overall, we did not feel that any modifications needed to be made to the data before modeling.

Tree-based methods

We looked at several different types of tree-based methods. Initially, we considered a plain CART, bagging, boosting, and random forest. For each of these methods, we proceeded to do hold-out testing on a subset of the training data, using classification error rate as our criteria. After tuning all of our tree-based methods using this criteria, we selected the best tree-based method, which here was bagging.

Looking at the graphs of accuracy and node purity that we produced from the bagging method, we found that v5, v1, and v2 seemed like they were roughly the most important variables for both accuracy and node purity.

Looking at our table and graph of the outcomes, it is fairly clear that we had excellent classification ability, with a classification error of approximately 6 percent.

Support vector machines

We initially thought of using an SVM as a classification model for this project for a couple of reasons. From what we learned in class, SVC and SVMs seem to be able to fit pretty complex data, and with us knowing

really nothing about the data at all, we thought that it would be interesting to see if a more complex model would work best. Another reason we wanted to use the SVM modeling techniques was because of the usefulness in tuning the parameters. Because of how easy it is to tune the parameters, we were able to try and fit many different SVCs and SVMs.

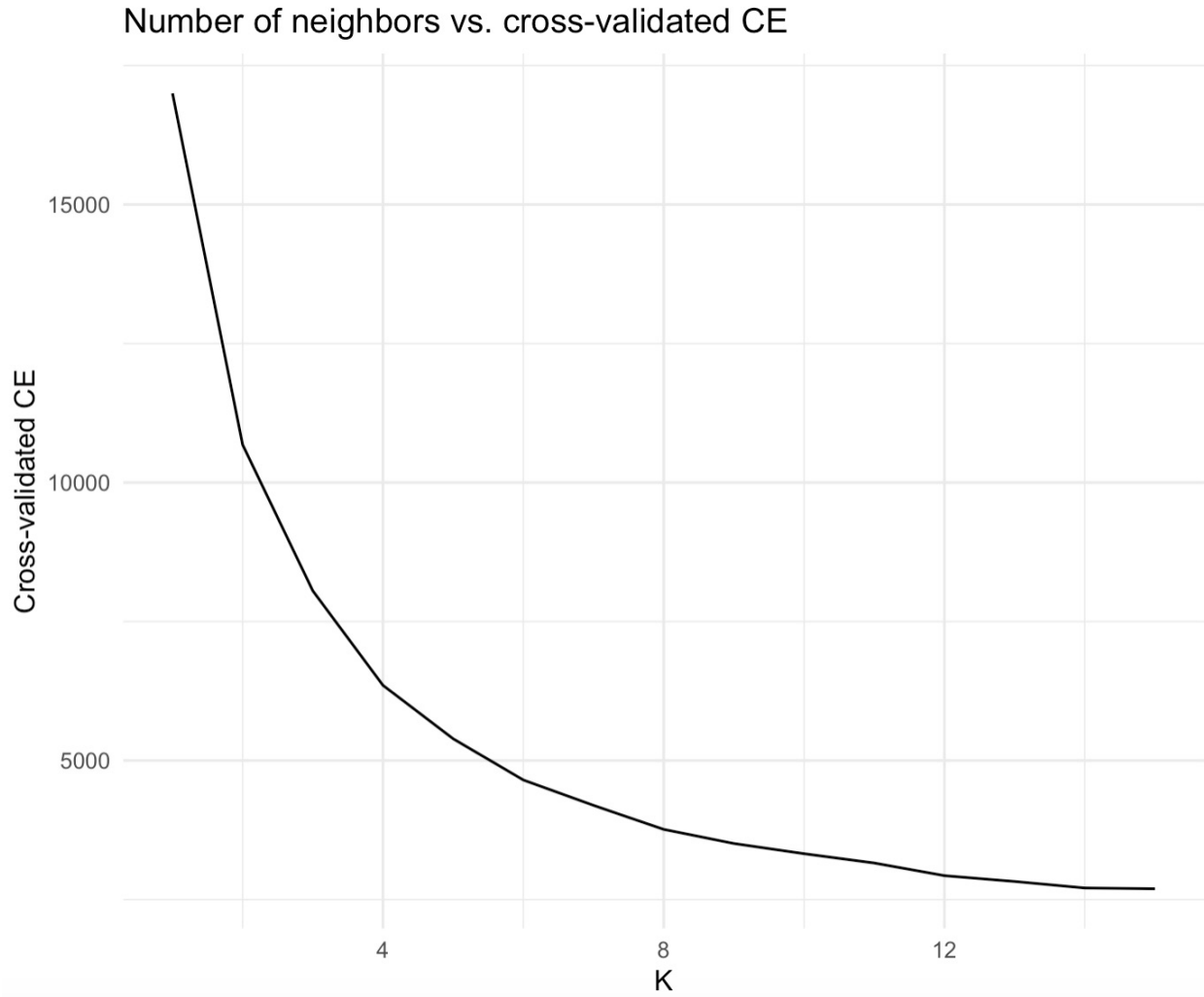
Using classification error rate as the criteria, we were able to come up with the best model in predicting the training set. Unfortunately, none of the models tested had a training error rate that was lower than 17 percent. So, our next task was to use cross entropy to tune the parameters. We used the models that were tuned based on misclassification rates as approximations and then plugged and chugged to find the best model in terms of cross-entropy. The best model was a polynomial kernel model with a cost of .0001 and degree of 3.

KNN

KNN was considered because it is very flexible and the bias/variance tradeoff can be controlled directly by varying K. The decision boundary is likely highly nonlinear since there are multiple classes in the response, so KNN is probably better to consider for this problem than LDA, QDA, or multinomial logistic regression.

Before training the KNN classifier, the predictors in the training set were centered and scaled (using “pre-Process” in caret). Five-fold cross-validated cross-entropy was used to determine the optimal value for K. In other words, the training data was split into five folds and the average cross-entropy was computed after the classifier was fit five times. This was performed for values of K ranging from 1 to 15. The five-fold cross-validated CE seemed to decrease with each increase in the number of neighbors, so 15 was chosen as the optimal value for K, with a cross-validated CE of 2,694.341. Larger values of K were not considered, as it was believed this could lead to a large decrease in variance at the cost of high bias and the cross-validated CE exhibited a clear negative trend. In addition, it was computationally difficult to compute the cross-validated CE for a large range of K, such as 1 to 50.

The (not cross-validated) training CE increased as K increased, and the 10-fold cross-validated accuracy decreased as K increased. This is likely due to small values of K leading to overfitting. The classifier with K = 15 had an accuracy of approximately 0.67 and a training cross-entropy of 6,223.914. A plot of the number of neighbors vs. cross-validated CE is below.

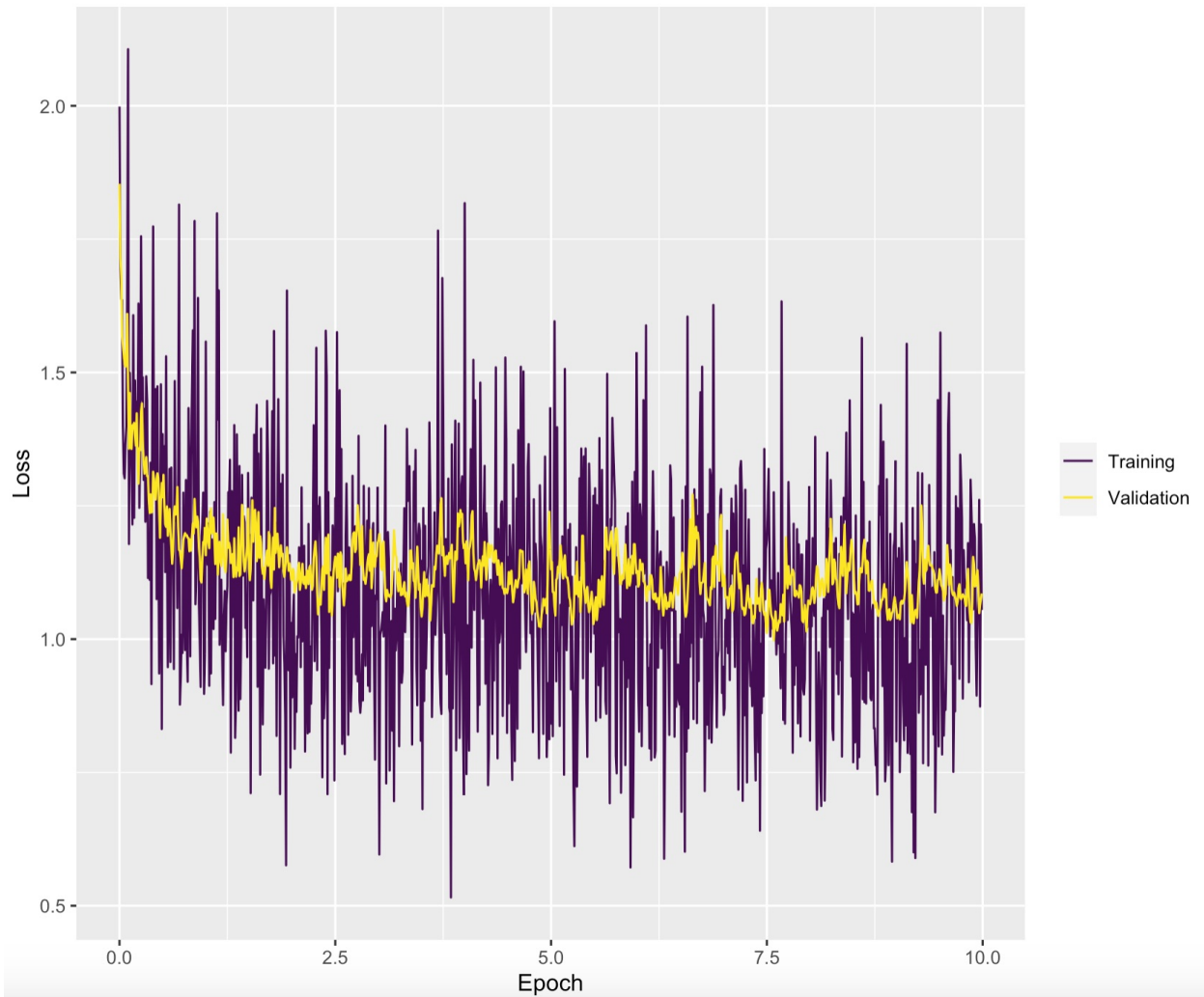


Neural network

A neural network was considered because it is a very flexible and widely used machine learning algorithm. It is very good at modeling nonlinear relationships, and it is likely that the decision boundary in this problem is not linear.

Before training the neural network, the predictors in the training and test sets were scaled to be between 0 and 1. The parameters were tuned using five-fold cross-validated cross-entropy. In other words, the training data was split into five folds and the average cross-entropy was computed after the neural network was fit five times. This was performed for a range of parameter values.

Note that it was difficult to automate the selection of parameter values in the loop (i.e., using a grid of different values), as RStudio often crashed when calculating the cross-validated cross-entropy alone. Because of this, different values for the number of epochs, learning rate, etc. were plugged into the loop and the cross-validated CEs were compared. Setting the number of hidden layers to three with nine, eight, and seven hidden variables, “optim.type” to “adam,” “n.epochs” to 10, “activ.functions” to “tanh,” the learning rate to 0.05, and the proportion of the training data used for tracing the loss on the validation set to 0.05 resulted in the lowest cross-validated CE, which was 1,996.641. This model had an accuracy of approximately 0.227 and a training cross-entropy of approximately 10,049.3. A plot of the loss during training is below.



Conclusion

Ppred1.RData is the prediction matrix for bagging, Ppred2.RData is the matrix for SVM, Pred3.RData is the matrix for KNN, and Ppred4.RData is the matrix for the neural network. All four methods performed fairly well on the training set, but it is of course difficult to evaluate performance without test labels. Cross-validated cross-entropy was used to train the KNN classifier and neural network, which may be the best method available to evaluate model performance for this particular problem. In the future, it would probably be useful to find better ways to tune parameters while maintaining computational speed. It can be difficult to train some of these models, especially neural networks, with low computing power.