CSE 176 - Report 1

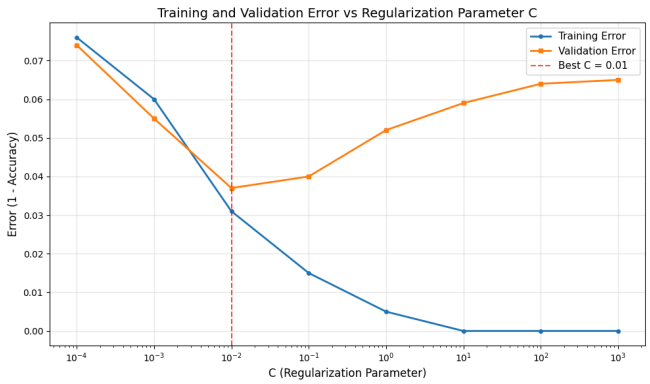
Gavin Abrigo, Ronan Tangaan, Christopher Hernandez

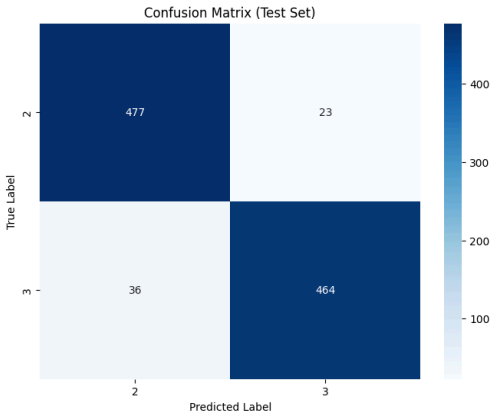
Introduction

Our digits for Part 1 were 2 and 3. To set up the models for training, we used a script to filter out the other digits from the full dataset and randomly split up the datasets into training, validation, and test sets of equal size. In order to use it, the function would take in the full data set, the digits you wanted to keep, and a seed value and it returns the 3 sets.

Logistic Regression

We used the 'liblinear' solver when training the model with l2 regularization. In order to tune our model, we decided to see what would happen when the hyperparameter C was changed. C is the inverse of regularization strength, meaning that a small C value creates stronger regularization than a large C value. We made an array of C values and trained each model with each value. After training, we would calculate the training error and the validation set error. The best model would be selected based on which C value produced the lowest of both errors. Our best C value was 0.01, which produced a training error of 3.1% and a validation error of 3.7%. After finding the best C value, we trained a model on the combination of the training and validation sets and we evaluated it on the test set made earlier. The new model produced an error of 5.9% on the test set.

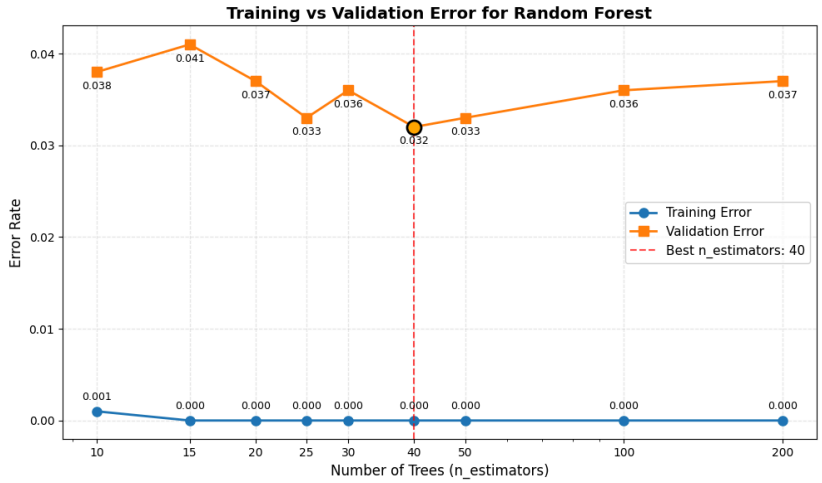




Random Forest

With the random forest, we wanted to see how the number of trees in the forest would affect the model's accuracy. Similarly to Logistic Regression, we created an array of potential tree counts and trained a model for each value. We calculated each model's error rate on the training and validation set and chose the number of trees based on the model with the lowest combined error rate. Surprisingly, our best

training error rate was 0% and our validation error was 3.2%.



After finding the best number of trees, we combined the training and validation set into a single set and trained a new model off of the combined set with the best tree count. After training, the model was evaluated on the test set. We ended up getting a test error of 4.2%