CSE 176: Final Project - Evaluating Algorithms for Learning Decision Forests
Part 1 - Binary Classification Using Logistic Regression and Random Forests
Mashaallah Moradi, Jason Holmes, Jason Petersen, Apsara Fite
April 2023

I. Introduction

Classifying data is important to learning more about the dataset as a whole and to make predictions about hypothetical data that will arise. There are many different classification methods to apply to any given dataset; here we will be exploring and comparing the Logistic Regression and Random Forest classification models on a binary dataset.

II. The Dataset

The dataset given (MNISTmini) consists of images of handwritten digits 0-9 and their label, with the objective being to classify the numbers correctly.

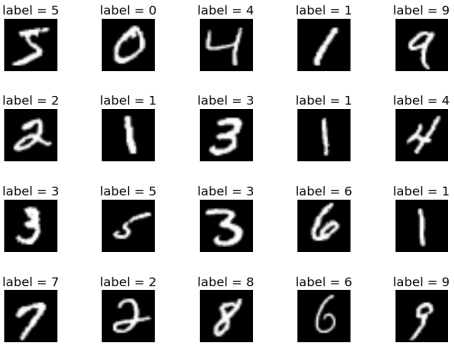

Fig. 1: Image representation of the MNISTmini dataset

For the binary classification task, we were assigned numbers 3 and 8. The dataset already consists of a training set and a testing set, but we decided to also create a validation set. It was split from the testing set at a ratio of 30%. The training set is used for training the model, the testing set is used for optimising the parameters, and the validation set is for verifying the performance of the model.

III. Logistic Regression

Tuning Hyperparameters with Cross Validation

Tuning all hyperparameters can take a significant amount of time, so it is wise to select the most important hyperparameters for tuning to increase efficiency. We are given the regularisation hyperparameter *penalty*=ℓ_2 and the algorithm *liblinear*, and have chosen to

tune the parameters *tol and C. Tol* corresponds to the tolerance for classification, and *C* is the inverse of the regularisation strength.

We set the possible tolerance levels to values between *1e-5* and 0.99, and the range of C values to between 0.1 and 100. However, we also had to increase the number of max_iters to 4000 in order for the solver to converge with higher C values. Using GridSearchCV to find the best hyperparameters via cross-validation, we found that the best hyperparameters were around 4.26 for the C value and around 0.25 for the tolerance.

Fitting the Data

Finally, we fit the tuned logistic regression model to the training set. The trained model had a 95.58% accuracy score and a 96.08% precision score for the validation set.

IV. Random Forest Classification

Tuning Hyperparameters with Cross Validation

Again, we chose the most important hyperparameters to tune for efficiency. Given that *max_features, max_depth* and *criterion* should be the default settings, the most important hyperparameter to tune is *n_estimators,* or the number of trees in the classification forest.

We specified the number of trees to be within the range 20-300. Using GridSearchCV to find the best hyperparameters via cross-validation, we found that the best number of trees to use is around 183.

Fitting the Data

Finally, we fit the tuned logistic regression model to the training set. The trained model had a 98.00% accuracy score and a 98.45% precision score for the validation set.

V. Conclusion

Comparing the accuracy and precision scores for both models, the Random Forest classifier performed better overall for this dataset. However, tuning the hyperparameters and training the Logistic Regression model took significantly less time.