MINST Kaggle Digit Recognizer: Contrasting the Random Forest and MLP Neural Network

Andrew Osborne
amo004@uark.edu

Josh Price
jdp024@uark.edu

April Walker
adw027@uark.edu

University of Arkansas,
Fayetteville, AR, 72701, USA

Abstract

In order to tackle the classic MINST dataset, our team implemented a random forest (RF) and a multi-layer perception (MLP) classifier using the `sklearn` Python library. As expected from a high-dimensional problem such as digit recognition, our RF preformed sub-optimally. Contrasting the accuracy between our training data and testing data suggest severe overfitting. Our final Kaggle submission reached a correct prediction rate of 96.66%. In order to optimize our MLP NN, we explored the use of stochastic and Adam gradient descent, however found the later consistently outperformed the former. In order to balance accuracy vs. complexity, our cross validation method suggests the use of 1 hidden layer and 256 nodes. Our final Kaggle submission resulted in a correct prediction rate of 97.90%, suggesting our NN is well-trained. While our MLP preformed considerably better than our RF, the training took approximately 60 times as long, implying the RF may be a desirable option when speed is of utmost importance.

1 The MINST Dataset

The MNIST digit database is a classic starting point for individuals wanting to learn the basics of computer vision and get hands on experience with machine learning algorithms. The dataset is composed of 70,000 28x28 pixel grey-scale images of handwritten numbers between zero and nine. The pre-flattened Kaggle dataset provides you with 42,000 training examples and 28,000 testing examples. Each 28x28 pixel image is represented by a 784 length vector with each element containing some integer between 0 and 255 representing the lightness or darkness of that pixel. Additionally, the training dataset provides the correct labels for each image. These images can be reshaped and rendered from the provided dataset as shown in Figure 1.

2 Random Forest

A Random Forest (RF) is a common starting algorithm for those new to machine learning. This is in part due to it's easy of use, but also due to it's flexibility. While in this assignment it is used for classification, it can be also be used in regression problems. Each tree is formed from a bootstrap sample (random sampling with replacement) then grown very similarly to a decision tree. Rather than searching for the very best split feature, it chooses the best of a random subset. Unlike the standard RF algorithm, `sklearn`'s `RandomForestClassifier` combines the probabilistic prediction of each tree, rather than picking a classification based on a majority vote.

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

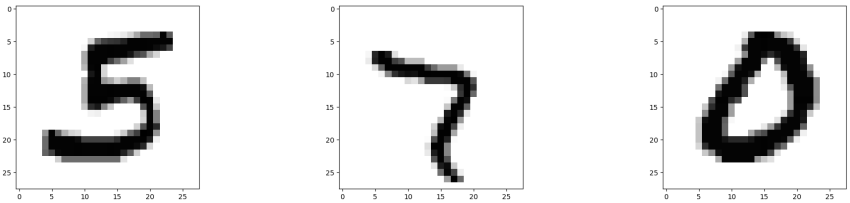

Figure 1: Image Renderings from the MNIST Dataset

These models are generally very fast to train, but depending on the complexity, can suffer from slow run-time performance. Generally speaking, overfitting can be negated by simple adding more trees to your model. This approach eventually results in diminishing returns, and for high-dimensional problems such as digit recognition, is especially impractical.

2.1 Implementation

The 784 length vector of input data is normalized to range from 0 to 1 using `sklearn`'s `preprocessing.minmax_scale`. In order to improve our model's predictive power, we contrasted the results from our hyperparameters `n_estimators` (number of trees) and `min_samples_split` (minimum number of leafs required to split a node). Specifically we allowed 10, 50, 100, and 300 trees to be developed and 2, 4, 8, and 16 as the minimum number of leafs. The later can be thought of as a measure of complexity, with lower minimum resulting in higher complexity. Once the code was prepared, training took approximately 45 minutes.

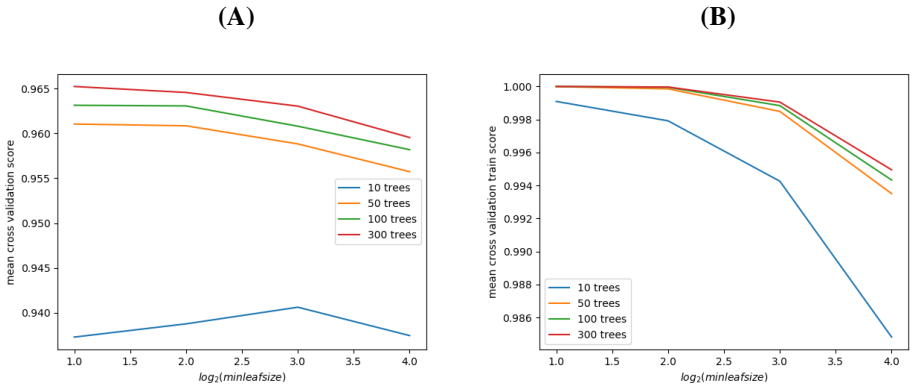

Figure 2: (A,B) Cross Validation of Hyperparameters for (A) training data and (B) testing data by contrasting mean cross validation score and $\log_2(minleafsize)$ giving the minimum leaf threshold as a measure of complexity. The scores for varying numbers of trees are shown.

3 Random Forest

3.1 Cross-Validation and Results

Training and cross-validation was done using `model_selection.GridSearchCV`. In order to visualize the results, Figure 2 contrasts the prediction rates of the training and test datasets. The $\log_2(minleafsize)$ which inversely measures the complexity suggests the more complex models preform outstandingly against the training data but have little impact on the testing data. Adding trees has a similar positive affect on both datasets, however, also decreases exponentially as more trees are added. These results suggest that our model suffers from extreme levels of overfitting. As expected, adding trees does improve our models predictive power but isn't the most practical for our dataset. Our Kaggle submission gave a prediction rate of 96.6%

4 Multi-Layer Perceptron Classifier

This will be much more interesting to talk about

4.1 Contrasting Gradient Descent Algorithms

Because it will probably be good to talk about why Adam's so good, basically because it used the components of a lot of other good methods.

4.2 Implementation

4.3 Cross-Validation and Results

5 Conclusions

References

References follow the acknowledgments. Use unnumbered first-level heading for the references. Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce the font size to `small` (9 point) when listing the references. **Remember that you can use more than eight pages as long as the additional pages contain *only* cited references.**

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.