Alexander Olden

Milestone Report: Identifying Handwritten Digits with Neural Networks

1. Introduction to the Problem

Image detection is a good entry point for learning about the use of neural networks, which I aim to learn more about in this project. Because image data is typically harder to work with than relational, or flat data is, more sophisticated forms of analysis are required. That said, massive computational power is not necessarily needed, as neural networks can perform the desired analysis. Simply put, the problem to be solved in my project is image detection (of handwritten digits, in this case) by using the branch of machine learning that deals with image detection.

Clients in this project cover a broad range of interests. Any organization that could benefit from automated recognition of handwritten digits. A few examples would be medical facilities (recognizing entries on forms completed by hand) and banks (for recognizing checks or other forms).

2. Data Attributes

Relevant data has been collected by the U.S. National Institute of Standards and Technology. The dataset I will use is a well-known subset called MNIST, which contains 70,000 images of handwritten digits (0-9). The data has been previously split into a matrix of 60,000 examples and a vector of 10,000 digits to serve as an outcome variable, of sorts. I accessed the dataset through the fetch_mldata method of the sklearn.datasets package in Python. The digits are normalized in terms of size as well.

The dataset comprises sparse matrices that load as digits when read into Python. As a result, there are no missing observations, and minimal data wrangling is needed beyond splitting the matrix of training observations and vector into training and test components. I also standardized the data using standard scaling – zero mean and one-unit variance – since I planned to use the data with three different algorithms. The scaling helps ensure stability in the various computations that will be needed.

3. Preliminary Exploration and Initial Findings

To begin analysis, I created a logistic regression to predict digits in the test data. The model yielded an accuracy score of 91.73% for training data and 88.71% for test data. In looking a classification report that captures precision and recall scores (as well as accuracy by digit), I found values of 89% percent for both in the training day. For test data, accuracy was actually a bit higher: while overall precision and recall scores were the same as they were for training data, accuracy for individual digits was slightly higher in several cases.

I next considered a random forest, which showed overall improvement from the logistic regression. Though some parameter tuning may still be needed, this model initially had an accuracy score of 94.64% on test data. Its classification report showed 95% for precision and recall.

From here, I will construct a multi-layer-perceptron neural network to predict digits in the MNIST dataset and compare its performance with the logistic-regression and random-forest models.