Zach Fontenot

Machine Learning

Dr. Kawsar

May 1, 2020

Machine Learning Pipeline: Phase III Report

1. Introduction

This report covers the third and final phase of the Machine Learning Pipeline. The Pipeline is designed to take a dataset as input and output a trained prediction model. The dataset serving as input for this report is a collection of sensor data from a user’s shoe, and the associated activity the user was performing. In the first phase of development, the data was preprocessed. Preprocessing of data included importing, cleaning, and splitting data into learning, cross-validation, and testing sets. Feature generation was also started during the first phase. The second phase’s objective was to implement logistic regression model generation. A cost function was provided to the Scipy library, which minimized a weight vector for each class of activity. Next, a cross-validation dataset was used to optimize the model. Phase three of the project is devoted to utilizing the same data, but this time creating a neural network.

1. Additional Preprocessing

Before a neural network could be created, preprocessing was done in preparation. The programmatic and manual cleaning of the first and second phase remained, but feature scaling was also recommended by Scikit in its neural network documentation. Though poor performance was expected without feature scaling, not much improvement was seen before and after applying feature scaling, with cross-validation error remaining relatively low (Figure 1).

1. Neural Network Parameter Testing

Creating an optimal neural network required the balancing of several different parameters. These included the number of hidden layers, number of nodes in each layer, number of iterations, the regularization constant, and the learning rate. Different activation functions were also available, but the sigmoid function was most common in current research. Each parameter was independently tested for. An advantage of neural networks over logistic regression became clear in this step. Optimizing neural networks was much easier due to the improved time complexity over logistic regression.

* 1. Number of Hidden Layers

The number of hidden layers was decided by consulting current research and following the lead of data science professionals. In a neural network which processes feature data as inputs, two hidden layers are all that are needed (Heaton 2017).

* 1. Number of Nodes

The number of nodes in each hidden layer was partially decided by trial and error with some guidance from current research. A test of varying amounts of nodes in each of the two layers was conducted (Figure 2). The test resulted in no noticeable trends, so one combination that performed especially well was chosen. This combination was eight nodes in the first layer and two in the second layer.

* 1. Choosing the Number of Iterations (Epochs)

Because the neural network converged rapidly, not much difference in error is seen as epochs increase (Figure 3). The model error appears to level off after about 100 iterations, so a parameter of 150 iterations was chosen.

* 1. Choosing the Regularization Constant (Lambda)

Lambda was determined purely by trial and error, following the recommendations of current research. The approach used in testing was to begin at a very low value and increase lambda by three times the current lambda. The result of this process produced an ideal lambda value of 0.1 (Figure 4).

* 1. Choosing the Step Size (Alpha)

To determine the optimal alpha value, the same approach was taken as determining the optimal lambda value. It was determined that a learning rate of 0.1 produced the best accuracy while taking the shortest amount of time to converge (Figure 5).

1. Approach

With the data processed and the optimal parameters known, neural network generation and testing could be implemented. The Scikit library was utilized, as each parameter could be entered into one class, the MLPClassifier class. After fitting the neural network to the features, the MLPClassifier’s testing tools were utilized. The “predict” function allowed for testing with an additional dataset (25% of the dataset with 75% used for training). The MLPClassifier required around ten lines of code to implement. This efficiency partly stemmed from the robust Scikit library, and partly from the modularization implemented in the second phase.

1. Results and Discussion

The optimal neural network was able to predict new activity cases with greater than ninety-five percent accuracy. This marked a slight improvement over logistic regression, but the true benefit came in the speed at which the neural network converged. The neural network scaled much better as the number of features increase, while the time for logistic regression to converge increased exponentially. Hence, the neural network approach will work better going forward, as testing new improvements will take less time.

In terms of possible improvements, the pipeline is close to being able to process any type of data. One issue faced here is feature generation and data splitting. Features are very particular to the subjects of a dataset, and data can be split differently based off of if synchronization is needed or not. The solution for now is to create a template DataObject class that is user-defined. This class is then designated for a particular dataset, requiring some amount of unique programmatic processing, but a completely generic pipeline that only accepts an entire unprocessed dataset is a deep learning problem. The easier solution is to place the responsibility of feature generation and data splitting on the user and provide a template for guidance.

Appendix

Figure 1:

Figure 2:

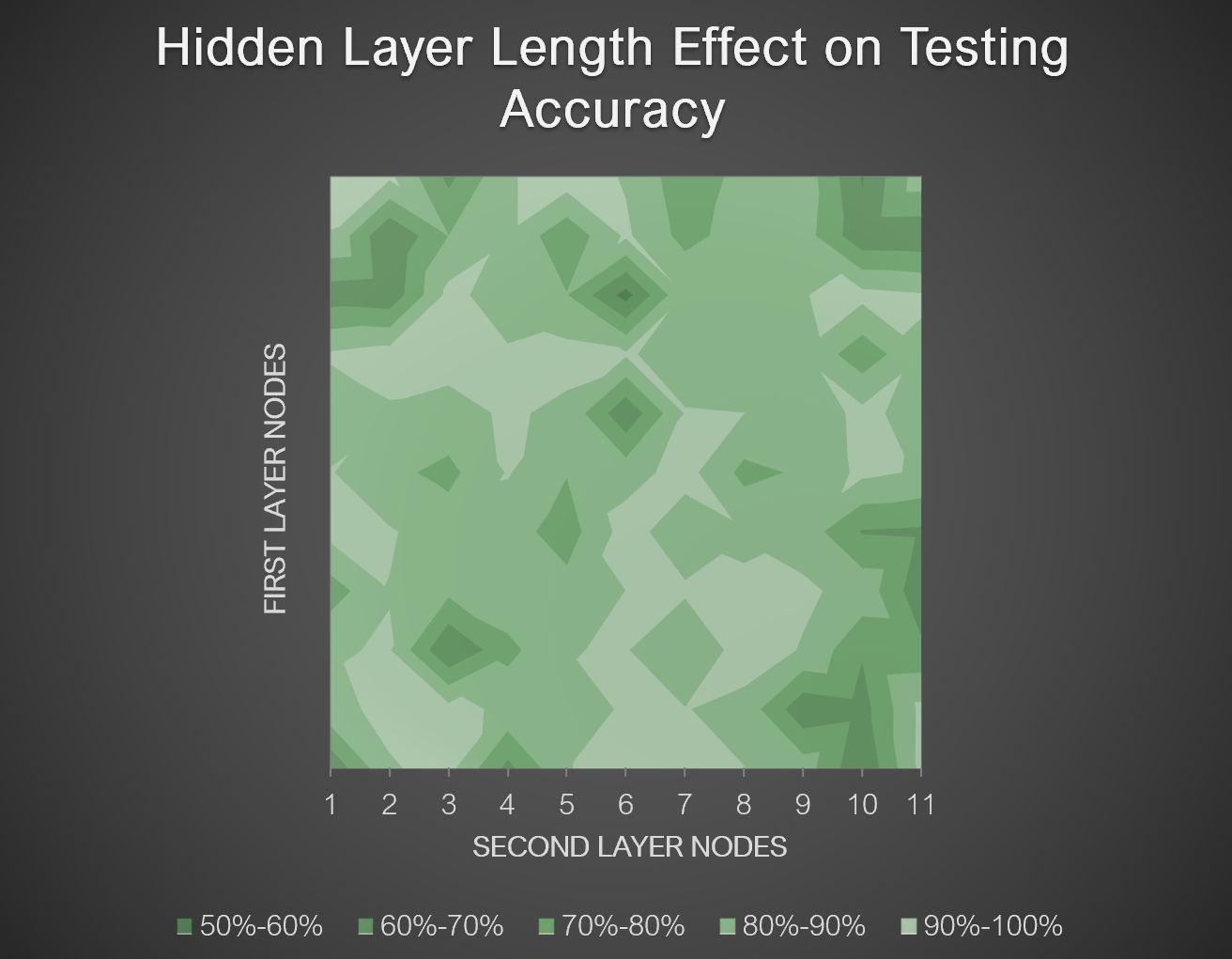


Figure 3:

Figure 4:

Figure 5:

References

Cybenko, G. (1989) “Approximations by superpositions of sigmoidal functions”, *Mathematics of Control, Signals, and Systems*, 2 (4), 303-314

Heaton, Jeff. (2017) “The Number of Hidden Layers”, Heaton Research.

Hinton, G. E.; Osindero, S.; Teh, Y. W. (2006). “A Fast Learning Algorithm for Deep Belief Nets” (PDF). Neural Computation. 18 (7): 1527–1554.