Perceptron Learning Algorithm Report

Bryan Greener

2018-05-17

Implementation

This program takes data of boston housing, which consists of 14 parameters, and uses the first 13 of these parameters to estimate the value of the "medv" parameter. From there, the compare this estimate to the mean "medv" value of the dataset and categorize the result as either 0 or 1 where 1 means that the house is above average and is worth buying. The 13 input parameters are passed into a 13×1 matrix and fed through the network into the output 1×1 matrix. As the inputs are sent into the network, they are multiplied by their corresponding weights to get the result. This is represented by the following equation

$$
\begin{bmatrix} x_1 & x_2 & \cdots & x_{13} \end{bmatrix}
\begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{13} \end{bmatrix} = \begin{bmatrix} a \end{bmatrix}
$$

where x are the input parameters, w are the weights for each input parameter, and a is the activation of the output layer which is also the output estimate value of the network. The estimate is then used to update the weights as follows

$$
w_{t+1} = w_t + \eta(y - \hat{y})x
$$

where t is the time step of the network, y is the expected output value, $\hat{y}$ is the observed output value, and η is a learning rate hyperparameter typically set to 0.01.

Analysis

In my implementation of the perceptron learning algorithm I was able to a get an average of around 80% accuracy when training the network on 80% of the dataset and testing on the remaining 20%. The algorithm reached this accuracy level within around 100 epochs, during each of which the training data was shuffled. I believe that this implementation can achieve somewhere near 85% to 90% accuracy given a larger dataset. Another limiting factor to this network's success is the fact that it is only using a single input layer and a single output layer. By adding a hidden layer of 7 or 8 nodes, I think that this network could get better accuracy over time. Also we could get better results by using a gradient descent algorithm with momentum.