BIOS 823 Final Project: Building a Neural Network

Mairead Dillon

December 16, 2022

Abstract

A neural network was built from scratch in python. Two data sets were tested on the neural network. Using the first data set, the neural network predicted whether or not a patient has diabetes using the patient's number of prior pregnancies, glucose level in blood, blood pressure measurement, thickness of skin, insulin level in blood, body mass index, diabetes percentage, and age as predictors. The second data set predicted whether a patient's breast cancer tumor was malignant or benign using the following tumor characteristics as predictors: mean of radius, mean of texture, mean of perimeter, mean of area, mean of smoothness, mean of compactness, mean of concavity, mean of concave points, mean of symmetry, and mean of fractal dimension.

Data Background

In order to train the neural network, two data sets with binary outcomes were selected from Kaggle. The first data set originally comes from the National Institute of Diabetes and Digestive and Kidney Diseases. The data set contains 8 diagnostics measurements - number of prior pregnancies, glucose level in blood, blood pressure measurement, thickness of skin, insulin level in blood, body mass index, diabetes percentage, and age - to predict whether a patient has diabetes. The outcome variable is diabetes with 1 indicating the patient has diabetes and 0 indicating the patient does not have diabetes. All patients selected from the original database are female, 21 years old or older, and of Pima Indian heritage. The second data set originally comes from the Breast Cancer Wisconsin (Diagnostic) Dataset. The dataset contains patient IDs, patient diagnoses, and the mean, standard error, and worst value for the following tumor measurements: radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension. Only the mean values of these measurements were used as predictors in the neural network. The outcome variable, diagnosis, originally had two levels - M (malignant) and B (benign). This variable was changed to a binary variable with M = 1 and B = 0.

Data Preprocessing

The preprocessing step was very similar for the two data sets. Each data set was split into two data frames - one containing all of the predictor variables (X) and one containing only the outcome variable (y). In order to be used in the neural network, all predictor variables had to be normalized on a scale from 0 to 1. To do this, a function was created that found the maximum value and the minimum value for each variable. Then each data point was normalized by subtracting the minimum value for that variable from the data point and dividing this number by the difference between the variables maximum and minimum values. After that, the test train split function from the sklearn model selection package was used to split the data into a training set and a testing set with 80 percent of the data in the training set and 20 percent of the data in the testing set. Both the predictor data frames and the outcome vectors were split into train and test data sets. Additionally, to be used in the neural network class, all of the data frames had to be converted to numpy arrays.

Creating the Neural Network

A neural network with 2 layers was built from scratch using python version 3.10.1 and the pandas, numpy, and sklearn python packages. A python class was created with 6 functions inside of it. The neural network class accepts two inputs, the number of nodes (equivalent to the number of predictors) and the learning rate. A weight vector is created inside the class. Random values between 0.001 and .101 are assigned to the weight vector, and the vector's length is the number of nodes input value. The bias is randomly assigned from a random normal distribution. The initial values for layer 1 are calculated by finding the dot product of the input vector and the initial weight vector and then adding the bias. The sigmoid activation function is then used to calculate the second layer. The values in layer 2 are the initial predictions. The mean square error is used to predict the magnitude of the error of these predictions and gradient descent is used to determine the direction and rate needed to update the network parameters. Backpropagation is then used to update the network's weights and bias. For the diabetes data set, the neural network was run with the number of nodes in as 8 and a learning rate of 0.1. The network was then trained on the train data using the train function, and then predictions were made using the test data. The predictions function outputs probabilities between 0 and 1. For each instance, if the predictions output was greater than or equal to 0.5, the prediction was 1, the patient has diabetes. If the predictions output was less than 0.5, the prediction was 0, the patient does not have diabetes. The prediction accuracy for the test data was 80.5 percent. The confusion matrix is displayed here:

	Predicted Yes	Predicted No
Predicted Yes	90	6
Predicted No	24	34

For the breast cancer data, the neural network was run with a number of nodes input of 10 and a learning rate of 0.1. The process of training on the

train data and predicting on the test data was very similar for this data set as it was for the diabetes data. The prediction accuracy rate on the test data was 91.2 percent, and the confusion matrix is:

	Predicted Yes	Predicted No
Predicted Yes	73	0
Predicted No	10	31

Figures 1 and 2 in the appendix show graphs displaying the training error for the two data sets.

Conclusion

Both the diabetes and breast cancer data sets had high testing accuracies. Additionally, as the number of training iterations increased, the training error went down for both data sets. To improve accuracy in the future, the prediction cut off can be adjusted from 0.5 to the event rate in the data. For both data sets, the sensitivity was much higher than the specificity, so the neural network may need to be adjusted to lower the specificity. The neural network may be improved by changing the learning rate, adjusting the number of iterations it is tested on, or adding more layers.

Citations

Chauhan, A. (2022, November 9). Predict diabetes. Kaggle. Retrieved December 16, 2022, from https://www.kaggle.com/datasets/89beb558127c7ead94e5f577cc2b5a80eeca56602f8d145afcba7ce2dccadf01?resource=download

H, M. Y. (2021, December 29). Breast cancer dataset. Kaggle. Retrieved December 16, 2022, from https://www.kaggle.com/datasets/yasserh/breast-cancer-dataset

Page, D. (2022). Neural Networks and Deep Learning. Duke University.

Real Python. (2021, February 27). Python ai: How to build A neural network amp; make predictions. Real Python. Retrieved December 16, 2022, from https://realpython.com/python-ai-neural-network/

Appendix

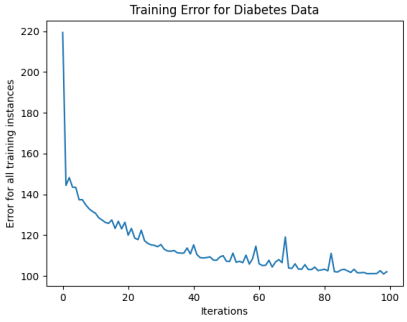

Figure 1: Training error for diabetes data. Created using the matplotlib package in python.

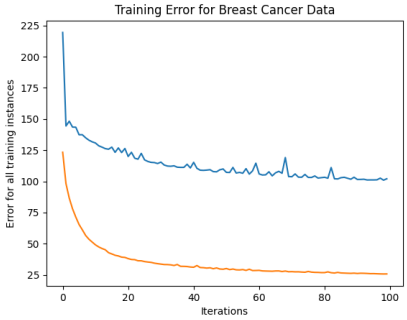

Figure 2: Training error for breast cancer data. Created using the matplotlib package in python.