Modeling Lab, Apr 24 2023

(Due Thursday, Apr 27 at 11:59PM)

You may work in groups of up to three. All of the script files and functions should work on Octave should you want to do that.

In today's lab, we'll be working with the k-NN classifier, and the linear neural network classifier (online training and batch training), so three models total for the homework. For each classifier, you'll be given one example m-file and data set to work with, then you'll be given a new data set to run the algorithms on.

k-NN Classifier Example

The folder linked under Monday of Week 12 has several files to download. You'll also find the data file with the iris data, and two files, `knnapp1.m` and `knnapp2.m`, which will serve as template $m-$files.

Be sure to download the files and be sure that you understand what's happening in the two apps. In particular, note that with the iris data, the data set is $4x150$, and the targets are $1x150$. In the k-NN algorithm, we are assuming that the targets are integer values (not columns of the identity matrix).

In the second app, we use 5-fold cross validation to try to find a good value of k. You would need to run it once to see what's best, then change the number of k in line 22.

k-NN Classifier Homework

Use the "diabetes.mat" dataset, which contains data gathered from several hundred diabetics. There 768 data points, each containing 8 measurements (so the matrix comes to you as 768×8). The output is either "1" or "0", and this information is stored in matrix T (which comes as a 1×768 array).

Careful as you load the data and run the code- Be sure that dimensions are what the programs what. For example, you should check the `fitknn.m` file to see what the input dimensions should be.

You should look over the plot of the error and decide on the number of nearest neighbors, then run the last two lines of `knnapp2.m` using that number, and show the confusion matrix (type C and enter in the command line).

Linear Network Classifier Example

First take a look at the "TGF" classifier that we went through in class. In particular, notice that the data being input is in X and is 16×6 (meaning 6 data points in 16 dimensions). The target data is in T and is 3×6, meaning 6 points in 3 dimensions (3 because there are 3 classes).

In this example, Widrow-Hoff is used to train the network. Please pay attention to the dimensions of the inputs, the outputs, the weights W and the vector of biases b.

Line 35 shows you how to construct the output Y given data in X, and the weights and biases.

The last three lines show you how to convert the vector outputs for classification into one dimensional values to put into the confusion matrix (You shouldn't need to change those).

We only had 6 points, so we didn't separate the data. You should separate the data into training and testing sets below.

The Pseudoinverse

Training the data all at once happens in the file `TGFExample2.m`.

Linear Network Classifier Homework

We're going to work with some data gathered from breast exams. The data represents 106 patients with 9 measurements each, and the output is a classification (integers 1 to 6). The data is stored as a text file in `BreastData.m`, so you can look at the file for more information about where the data comes from.

To load the data, just type `BreastData` and a matrix X (that is 106×9 and a target matrix T (that is 6×106) is loaded. Try to be sure that the dimensions are what you need them to be for each of the algorithms.

For the lab,

1. Split the data into Training and Testing sets (70% for training, 30% for testing).
2. Do some data exploration. If you think you need to scale the data, you can use `StandardScaler.m` from the $k - NN$ example.
3. We'll build two models- one using Widrow Hoff, and one using "batch" with the pseudoinverse.
4. At the end, please note the confusion matrix output and the overall error.