Step by Step Manual



Before Starting, the data should be in following format

Let’s say we have total M =1000 data entries and 2 classes. 450 data entries belong to class 1 and 550 data entries belong to class 2. Let’s say each class is described by N=10 features, to . We can place entries of the two classes in one matrix randomly.

Input data will look like as following

.

.

.

For target data, there should be M (2 in this case) column, each column belongs to a class. Input and target data will look like

Input Data Target Data Class 1 Class2

.

.

.

Step 1: Load input and Target Data into MATLAB workspace. And then launch the GUI by running ‘DBN\_GUI.m’

Step 2: Click on “Load Workspace” button. This button will import workspace variables into MATLAB GUI environment.

Step 2.1: Select input data and click on “Import Inputs”

Step 2.2: Select Target data and click on “Import Targets”

Step 2.3: Select percentage of training data by setting the value of slider



Step 3: In step 3, we have to initialize network parameters

Step 3.1: Enter number of layers and click “Number of Layers”. If it is left blank, GUI will set Number of Layers as 3 by default.

Step 3.2: Enter number of neurons in layer 1 and click “Enter size of Layer 1”

Step 3.3: Repeat step 4.2 until all layers of the network are entered

Step 3.4: Enter number of epochs and click “Enter Number of Epochs”

Step 3.5: Enter batch size and click “Enter Batch Size”



Step 5: Hit “Train Network”. Once the network is trained



Step 6: To evaluate network on training data, hit “Evaluate on Training Data”

Step 7: To evaluate network on training data, hit “Evaluate on Test Data”

Step 5: To save output, hit “Save Predicted Classes”

In the reults panel, we can see three kind of results

**Training and Test Accuracy**

Ratio of Number of correctly classified entries to Total Number of Entries

**Region of Convergence curve**

This curve shows ratio between correctly classified and wrongly classified entries of each class

**Confusion matrix**

It shows following

%age of correctly classified entries in each class (along the diagonals)

%age of entries of a class which are placed wrongly in other classes (placed along columns)

%age of entries of a class which does not belong the class but are predicted in current class (placed along rows)