# Multi-Class Connectionist AI

We have used “IRIS_TrainData.csv” containing 150 instances and 4 features for training.

Have used learnrate for learning rate, epoch for number of iterations, weights1 for input layer and hidden layer, weights 2 for hidden layer and output layer.

Then Calculating value of Sigmoid : $y = \frac{1}{1 + e^{-x}}$.

Also derivative of Sigmoid : x(1-x)

Followed by cost.

Then iteration 1 moving forward from input layer to output layer, calculating using dot function with weights for both input and hidden layer.

Then Back propagation with derivative of cost function through hidden layer and back to input layer.

Update weights after that complete cycle/iteration.

Same for training it will repeat for the number of iterations/epochs mentioned (2000 in our case)

For adding data to the list, we have used values 3 bifurcated values between 0 to 1 namely for each of the species.

Then used Scikit learn library Standardscalar to Standardize/Normalize data.

We have used IRIS_testData.csv for test data. Which does not have the respective value of type of flower associated with each instance.

It will read each sepal/Petal Length/width for each of the instance up to the last instance i.e length of test data and then predict. We have printed a matrix type formation for labelling the output that will range from 0 to 1.

Based on the Output value and iris species type will be labelled to the instance whether setosa, versicolor and virginica.

| sepal_length | sepal_width | petal_length | petal_width | Output | IrisType |
|---|---|---|---|---|---|
| 5.5 | 2.4 | 3.7 | 1.0 | [0.62054862] | versicolor |
| 5.8 | 2.7 | 3.9 | 1.2 | [0.57838754] | versicolor |
| 5.4 | 3.0 | 4.5 | 1.5 | [0.65182823] | versicolor |
| 6.0 | 3.4 | 4.5 | 1.6 | [0.57575569] | versicolor |