**Data Mining**

**CSE 5334**

**Assignment 2**

Nishant Chaudhary

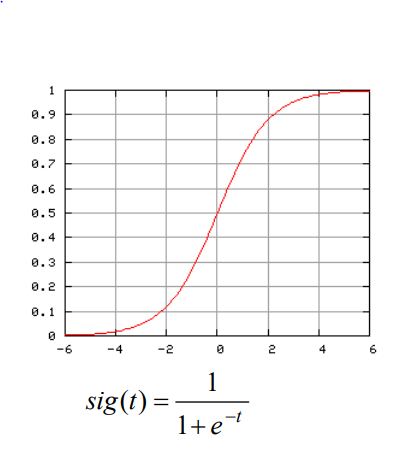
For the training data generation, I used the **numpy.random.multivariate\_normal(**mean**,**cov**[,**size**,**check\_valid**,**tol**])** function in Python. We were given two means and the covariance matrix. So I used it to generate 1000 data points using first mean and then 1000 data points using second mean. Later I also created another list in which the first 1000 elements are labelled as 0 while the last 1000 elements are labelled as 1.

In the same way I generated test data but for test data I generated only have 1000 data as compared to the 2000 data points I had for training data. For test data I have 500 points for each label while for training data I had 1000 for each of the.

For the activation function I used Sigmoid function.

A **sigmoid function** is a [mathematical function](https://en.wikipedia.org/wiki/Function_(mathematics)) having a characteristic "S"-shaped curve or **sigmoid curve**. Often, *sigmoid function* refers to the special case of the [logistic function](https://en.wikipedia.org/wiki/Logistic_function) shown in the first figure and defined by the formula

{\displaystyle S(x)={\frac {1}{1+e^{-x}}}={\frac {e^{x}}{e^{x}+1}}.}



For objective function we should use cross entropy.

the **cross entropy** between two [probability distributions](https://en.wikipedia.org/wiki/Probability_distribution) {\displaystyle p} and {\displaystyle q} over the same underlying set of events measures the average number of [bits](https://en.wikipedia.org/wiki/Bit) needed to identify an event drawn from the set, if a coding scheme is used that is optimized for an "artificial" probability distribution {\displaystyle q}, rather than the "true" distribution {\displaystyle p}.

p

q

q

p

**Problem 1 & 2:**

There are two different techniques for training a neural network: batch and online.In online training, weights and bias values are adjusted for every training item based on the difference between computed outputs and the training data target outputs. In batch training the adjustment delta values are accumulated over all training items, to give an aggregate set of deltas, and then the aggregated deltas are applied to each weight and bias.

The batch approach is:

loop maxEpochs times

for each training item

compute weights and bias deltas for curr item

accumulate the deltas

end for

adjust weights and bias deltas using accumulated deltas

end loop

Batch Versus On-line Learning

The on-line and batch modes are slightly different, although both will perform well for parabolic performance surfaces. One major difference is that the batch algorithm keeps the system weights constant while computing the error associated with each sample in the input. Since the on-line version is constantly updating its weights, its error calculation (and thus gradient estimation) uses different weights for each input sample. This means that the two algorithms visit different sets of points during adaptation. However, they both converge to the same minimum.

Note that the number of weight updates of the two methods for the same number of data presentations is very different. The on-line method (LMS) does an update each sample, while batch does an update each epoch, that is,

LMS updates = (batch updates) x (# of samples in training set).

The batch algorithm is also slightly more efficient in terms of number of computations.

I have calculated the accuracy rates for the different learning rates and tested the test data.

Reference:

https://sebastianraschka.com/Articles/2015\_singlelayer\_neurons.html