CS577 -Assignment 1

# Task1

**Name : Sourav Yadav Spring 2020**

**AID: A20450418**

--------------------------------------------------------------------------------------------------------------------------------------

**1.Problem Statement :**

Classification of iris flowers from sepal and petal dimensions. This is three class classification problem which we are going to solve by implementing out own coded neural network.

**2.Proposed Solution :**

We will design a neural network in Python without Keras and then train the network on train and validation data. Finally we will evaluate performance of the network on test data.

**3.Implementation details :**

* We first load the data the Irish Flower data from the below link. Attribute Information is also given.

<https://archive.ics.uci.edu/ml/datasets/Iris>

Attribute Information:

1. sepal length in cm  
2. sepal width in cm  
3. petal length in cm  
4. petal width in cm  
5. class:  
-- Iris Setosa  
-- Iris Versicolour  
-- Iris Virginica

* Then we are going to One Hot Encode the labels of the dataset.
* We split the Iris data in Train , Test and validation Set using train\_test\_split() method in 8:2 ratio.
* Then we normalized the features using below formula.
* We designed neural network with 4 nodes in the input layers , 5 nodes in two hidden layers followed by an output node with Softmax function which will return predicated probabilities of the input data.
* Activation used in the hidden layers is Sigmoid and Activation used in the output node is Softmax.
* After designing our network , we implement forward pass in which input are fed to the network and outputs are calculated at all the nodes.
* After designing forward pass , we design backward pass in which calculate gradients of the required node using the output of the respective nodes.
* The gradients calculated in the backward pass are used to update the weights in the Gradient descent algorithms.
* We used Categorical Cross Entropy as our loss function whose derivates we need with respect to weights to update weights in Stochastic Gradient Descent.
* Now we train our network with train data and observed its performance . Results are discussed in below section.

**4. Results and discussion:**

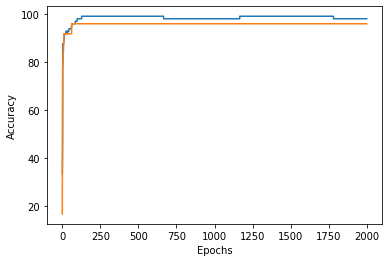
* Initial hyperparameters used during training.

Learning Rate : 0.01

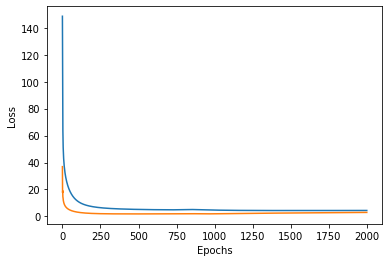
Epochs : 2000

* After training the network with above hyperparameters, we got below results on train and validation sets.

Accuracy Graph : Training Accuracy and Validation Accuracy



Loss Graph : Training Loss and Validation Loss.



* From the above graph we see that our models validation loss starts to increase after 300 epochs and hence we choose our final number of epochs as 300.
* Finally we evaluate our network on test data and got below results.

