1.Problem Statement :

This is three class classification problem. Classification of iris flowers from sepal and petal dimensions.

2.Proposed Solution :

We will design a neural network in Python and then train the network on train and tune our hyperparameters using validation data. Finally we will evaluate performance of the network on test data.

3.Implementation details:

- ➢ We first loaded the data the Iris Flower data from the below link. https://archive.ics.uci.edu/ml/datasets/Iris
- ➢ Then we are going to One Hot Encode the labels of the dataset.
- ➢ We split the Iris data into Train , Test and validation Set in 80:20 ratio and we normalized the features using below formula.

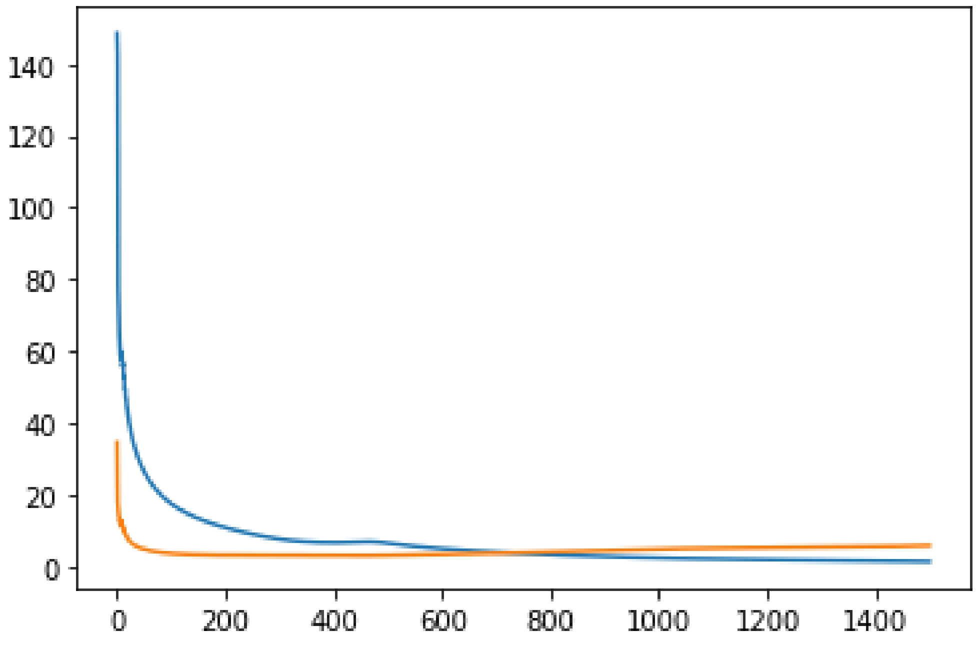
  Initial Hyperarameters :

  - Input nodes : 4
  - Output nodes : 3, activation for output layer : softmax
  - first hidden layer Hidden nodes in: 15 , activation function: sigmoid
  - second hidden layer Hidden nodes in: 15 , activation function: sigmoid
  - Loss function: categorical cross entropy
- ➢ After designing our network , we implement forward propagation in which input are fed to the network and outputs are calculated at all the nodes. After designing forward pass, we design backward propagation in which we 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.

4. Results and discussion:

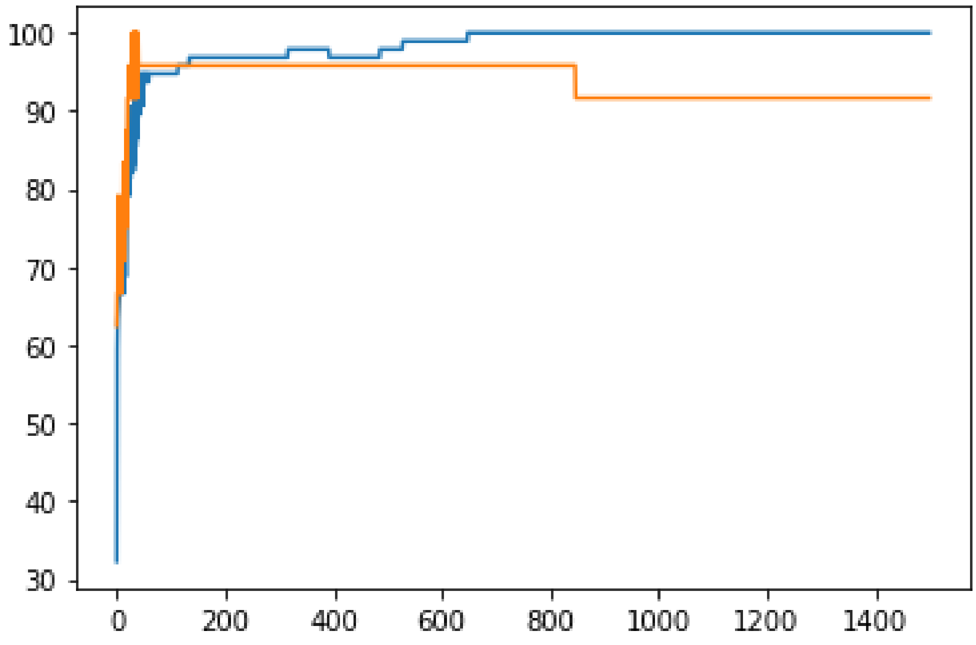
- ➢ Hyper parameter tuning : Initial hyperparameters used during training.
  Learning Rate : 0.01
  Epochs : 1500

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



Loss Graph : Training Loss and Validation Loss.

- **Orange color** denotes loss on validation set
- **Blue color** denotes loss on training set



Accuracy Graph : Training Loss and Validation Loss.

- **Orange color** denotes accuracy on validation set
- **Blue color** denotes accuracy on training set

- As you can see from above graph validation loss starts to increase after 420 epochs which indicates model starts to overfit and so we choose epoch = 420 .
- We again train our model with train data (train data + validation data) using following hyperparameter :
  - Epoch : 420
  - Output nodes : 3, activation for output layer : softmax
  - first hidden layer Hidden nodes in: 15 , activation function: sigmoid
  - second hidden layer Hidden nodes in: 15 , activation function: sigmoid
  - Loss function: categorical cross entropy.

So on train set we got following performance:

```
Accuracy 96.8503937007874
Loss 9.788982288882657
```

- Finally we evaluate performance on test data.

```
Test accuracy is 92.5
Test loss is  3.961095160307607
```

- We saved our model using pickle and loaded the same model and we see that we get the same performance.