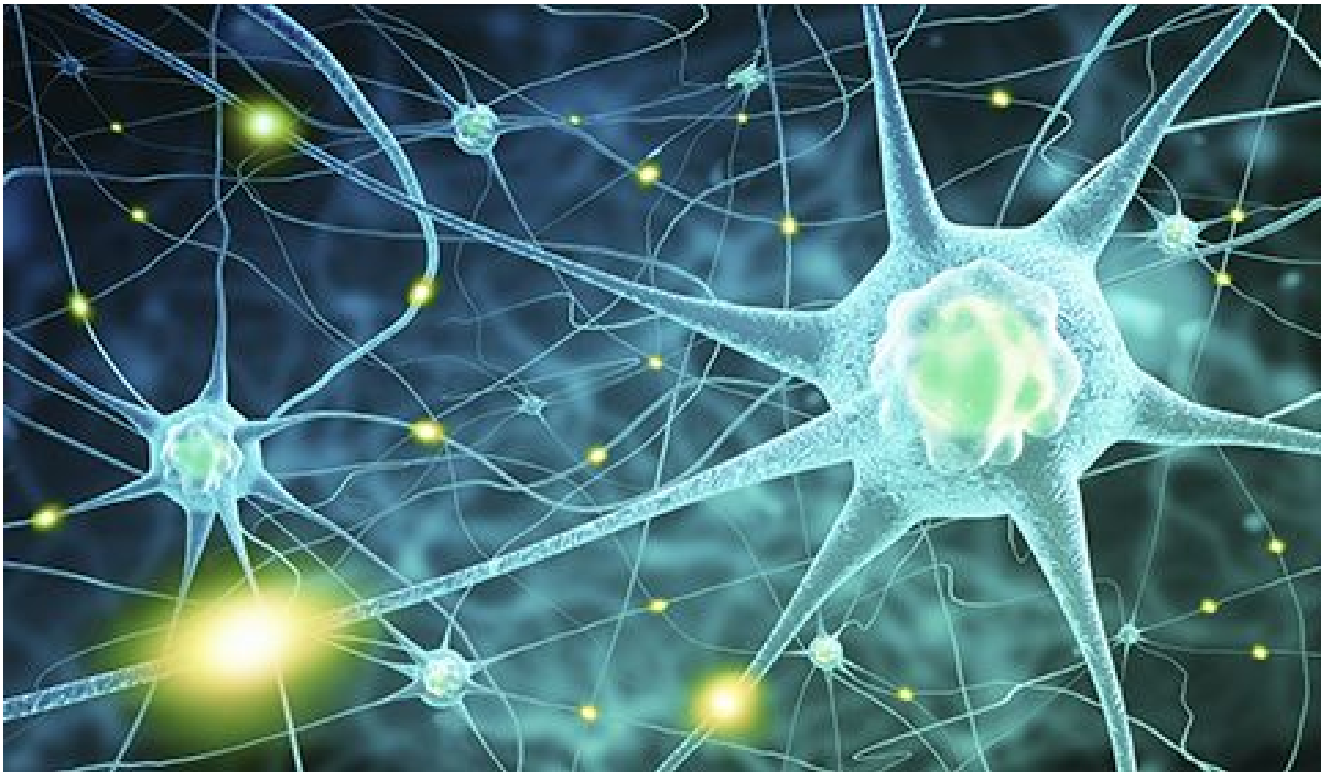
Assignment 2 Part 2

Deep Learning - CS577



Akshay R

A20442409

Task:

Problem Statement 1:

Multi-Class classification of IRIS dataset using a fully connected neural network. The network must contain three layers. Sigmoid as the activation function for the first two layers and softmax for the output layer. Loss considered is categorical cross entropy.

Problem Statement 2:

Multi-Class classification of wine dataset using a fully connected neural network. The network must contain three layers. Sigmoid as the activation function for the first two layers and softmax for the output layer. Loss considered is categorical cross entropy.

Proposed Solution:

The IRIS dataset has 150 data points distributed over 3 classes, 50 data points and labels in each class. The data must be normalized and further divided into train, validation and test data sets. The labels must be encoded into a categorical variable with three values.

A fully connected neural network must be built that takes in input in the shape of a 1-D array and provides an output out of three neurons. The model must be tweaked to obtain the best accuracy and minimal loss.

The wine dataset has 178 data points over 3 classes. Unlike the Iris data set which has 4 features, the wine data has 12 features which are the readings recorded by chemical experiments performed on different wine samples. These features must be separated from the label which is present at the first column of the dataset ulike iris where the last column is the label.

The same fully connected neural network is used to train the model with a few tweaks in the model parameters.

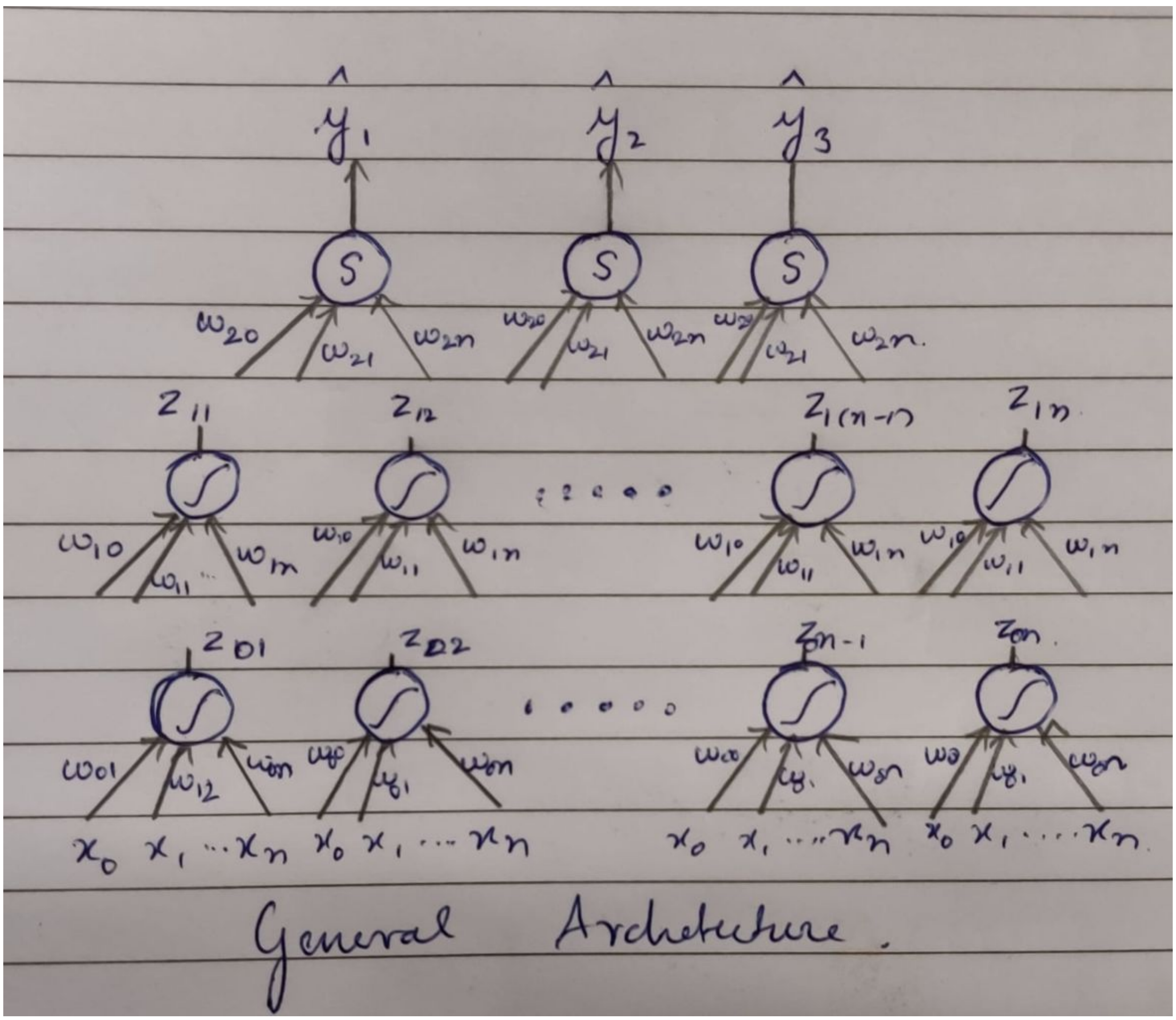
Implementation and methodology:

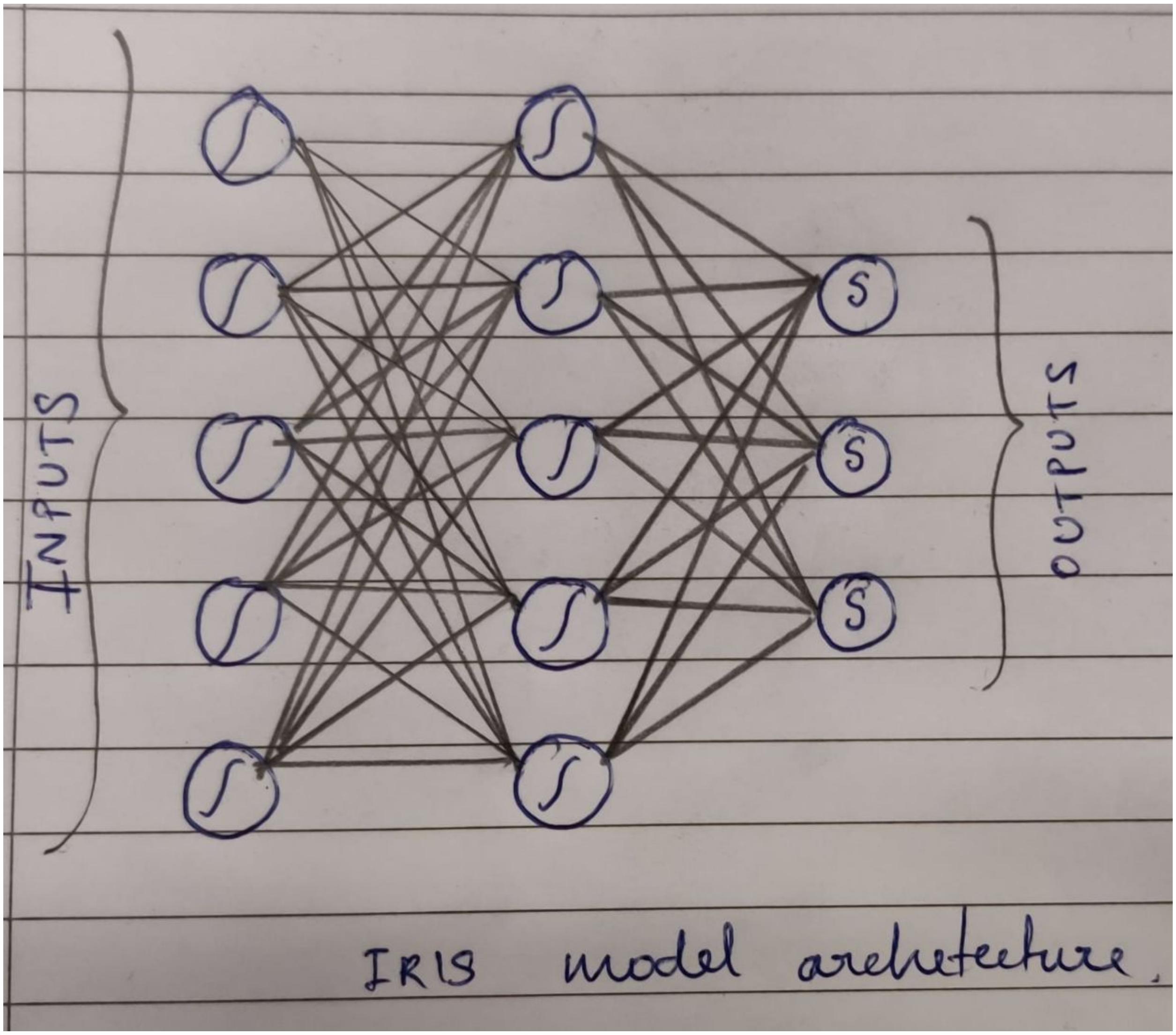
The IRIS dataset obtained is first divided into features and labels with the last column being the label and first three columns are the feature variables. The label column has names of three petal types which is one hot encoded into class 0, 1, 2 respectively based on the type of label.

Similarly the wine dataset obtained is also divided into features and labels with the first column being the label and the rest of the columns being features. The label is further one hot encoded in the same way to obtain three classes.

First the data is split into train and test data points. This is then normalized by first taking the mean and standard deviation of the train data points, then subtracting the train data by its mean value and then dividing by its standard deviation The same procedure is followed for the test data. This procedure is followed for both datasets.

The resulting data sets are further divided into train and validation datasets for training.





For the IRIS dataset the model built has an input layer with 4 neurons since we have 4 features and an output layer of 3 neurons since we have 3 classes to classify. As given in the problem statement the activation function used is sigmoid functions for the two hidden layers used. The hidden layer is chosen to have 5 neurons each (random selection).

The output layer has a softmax function as its activation. During each epoch we train this model by passing the weights which are selected randomly and the input as obtained from the features. This is the forward propagation of the network. During the backward propagation we use gradients calculated by chain rule to determine the update in the weights to reach a state where the model performs the best.

For the wine dataset we use 12 neurons for the input layer. 128 neurons for the hidden layer layer. Weights are chosen randomly as computed for the Iris dataset. Gradients and loss are calculated in a similar manner as done for the IRIS dataset.

We use categorical cross entropy as the loss function to determine loss and hence calculate the validation loss as coded in the program.

We perform a number of epochs to successfully determine the validation accuracy and validation loss achieved by the model.

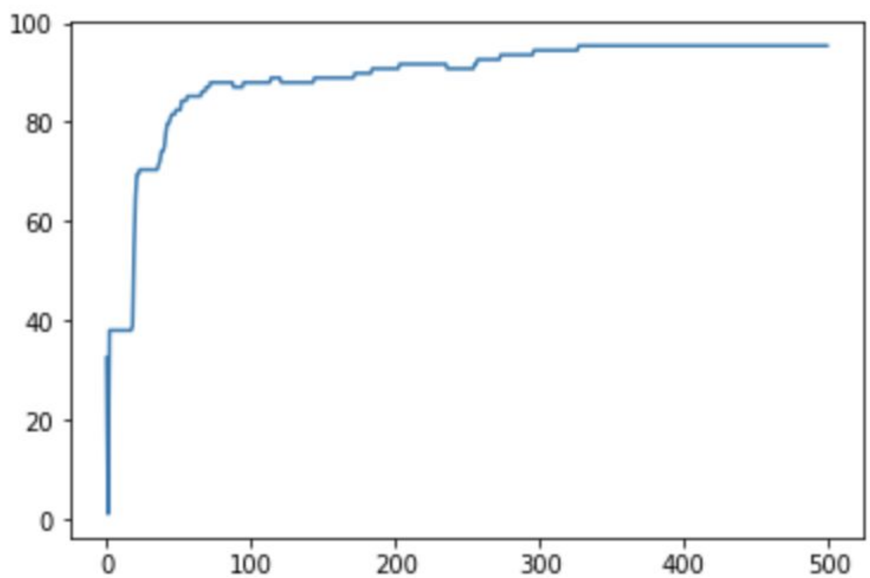
Graphs are plotted with validation accuracy vs epochs and validation loss vs epocs, the readings are recorded.

Results and discussion:

Iris dataset results:

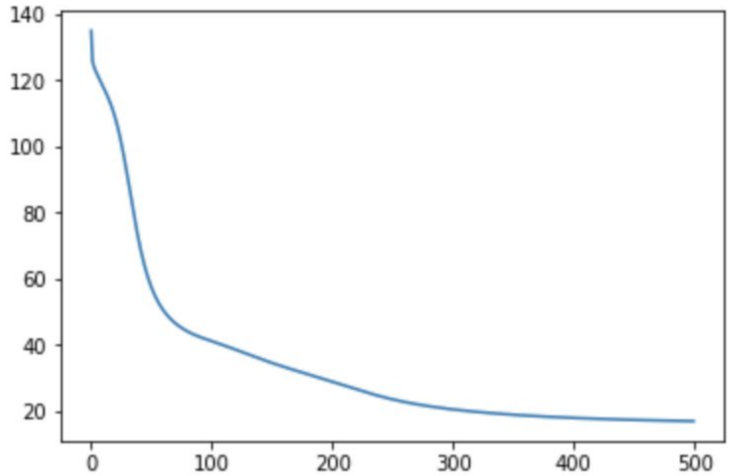
The graph of validation accuracy vs epocs looks as follows:

```
import matplotlib.pyplot as plt
plt.plot(range(epoch),accuracy)
plt.show()
```



The graph of validation loss vs epochs are as follows:

```
import matplotlib.pyplot as plt
plt.plot(range(epoch),loss_val)
plt.show()
```



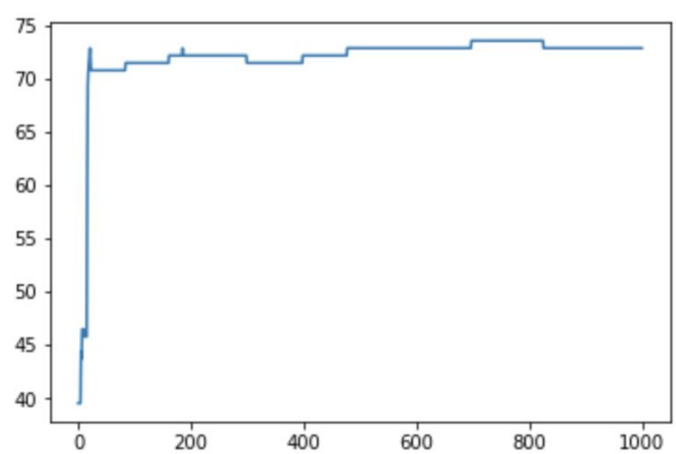
This model is then presented with the test dataset separated at the beginning of the procedure to obtain the test accuracy and loss which is found to be:

Accuracy : **93.33%**

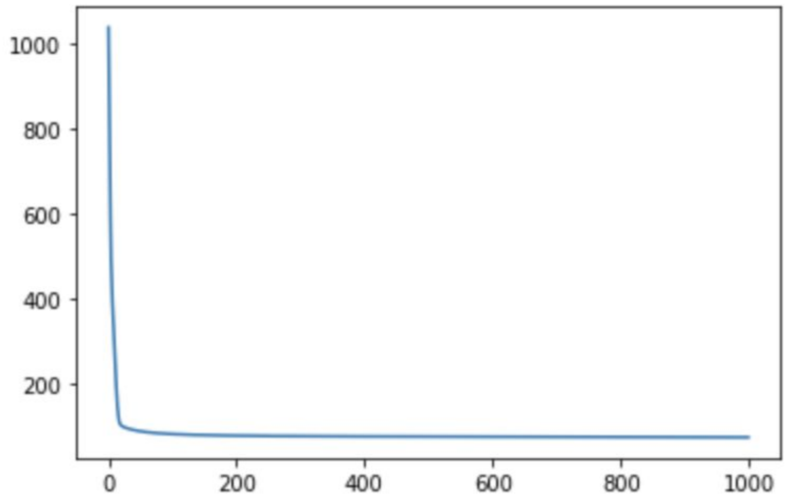
Wine dataset results:

The graph of validation accuracy vs epocs looks as follows:

```
#results
import matplotlib.pyplot as plt
plt.plot(range(epoch),accuracy)
plt.show()
```



The graph of validation loss vs epochs are as follows:



This model is then presented with the test dataset separated at the beginning of the procedure to obtain the test accuracy and loss which is found to be:

Accuracy : **83.33%**

References:

- Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names
- Neural networks and backpropagation explained in a simple way
- Understanding the Mathematics behind Gradient Descent.