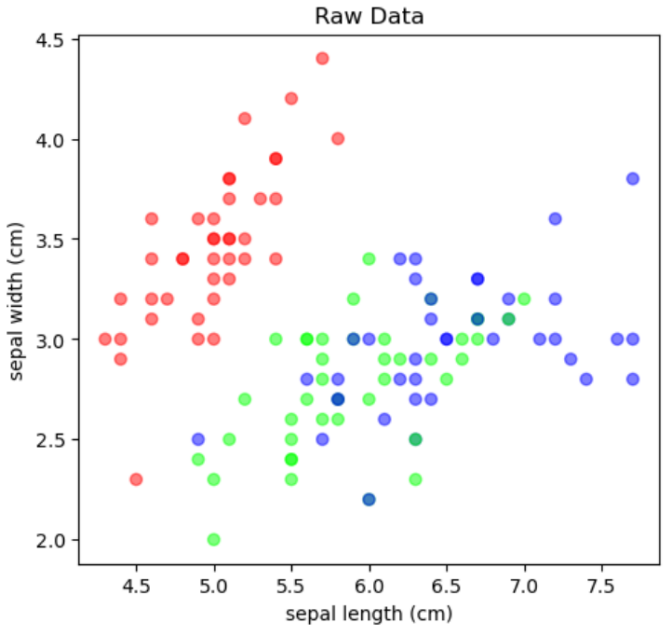
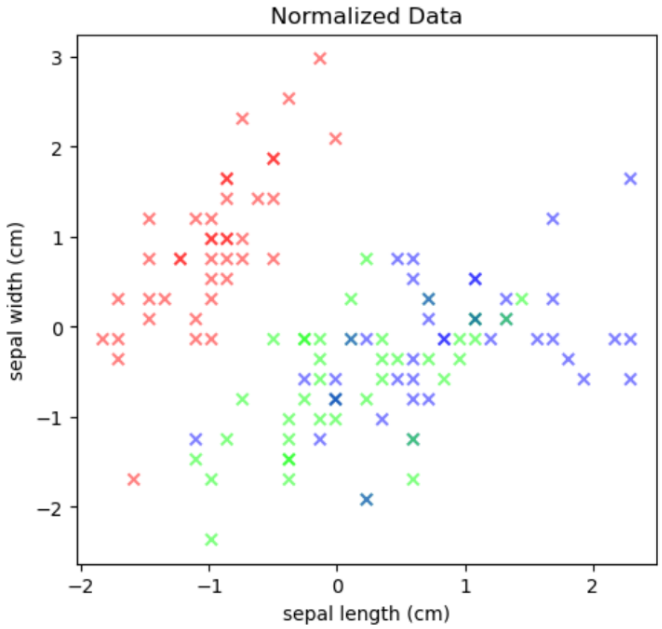
Sai Vivek Rambha (2023585403) Assignment 1 part 2(a, b) Classification & Neural Networks ECS708P

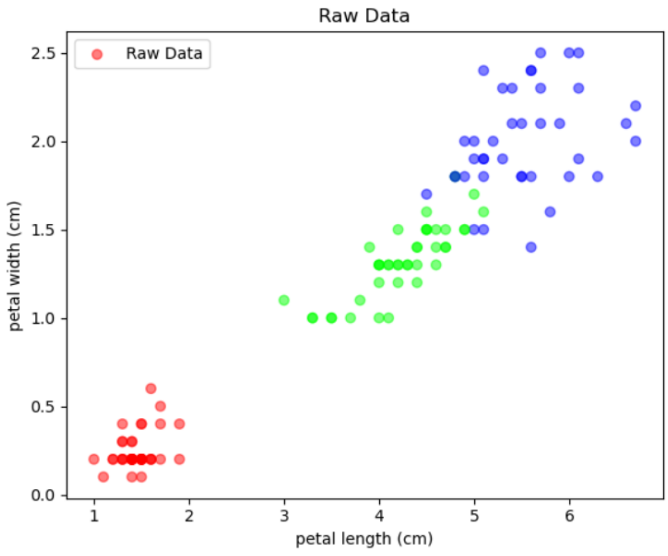
Classification:

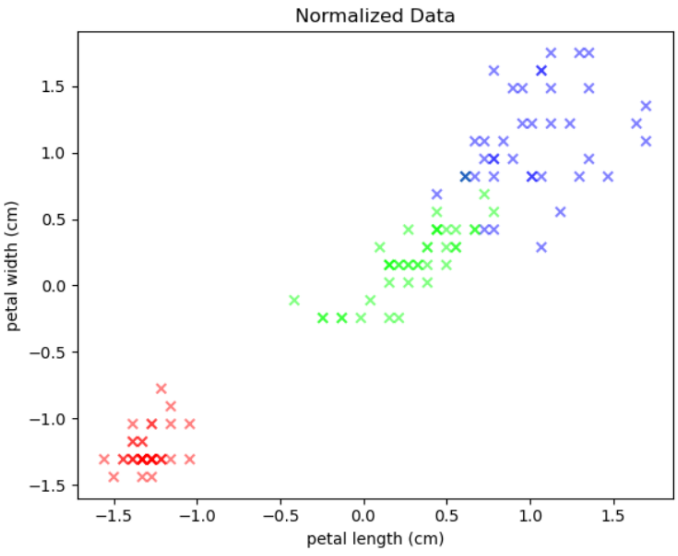
Q1. We again notice that the attributes are on different scales. Use the normalisation method from last lab, to standardize the scales of each attribute on both sets. Plot the normalized and raw training sets; what do you observe?

We use the normalisation method to bring feature values into the same range because the data is presented on different scales. The normalised data in the plot below ranges from 2 to -2 for sepal and -1.5 to 1.5 for petal, whereas the raw data ranges from 4 to 8 for sepal and 1 to 6 for petal. Data standardisation helps the model to predict accurately and improving the training time.



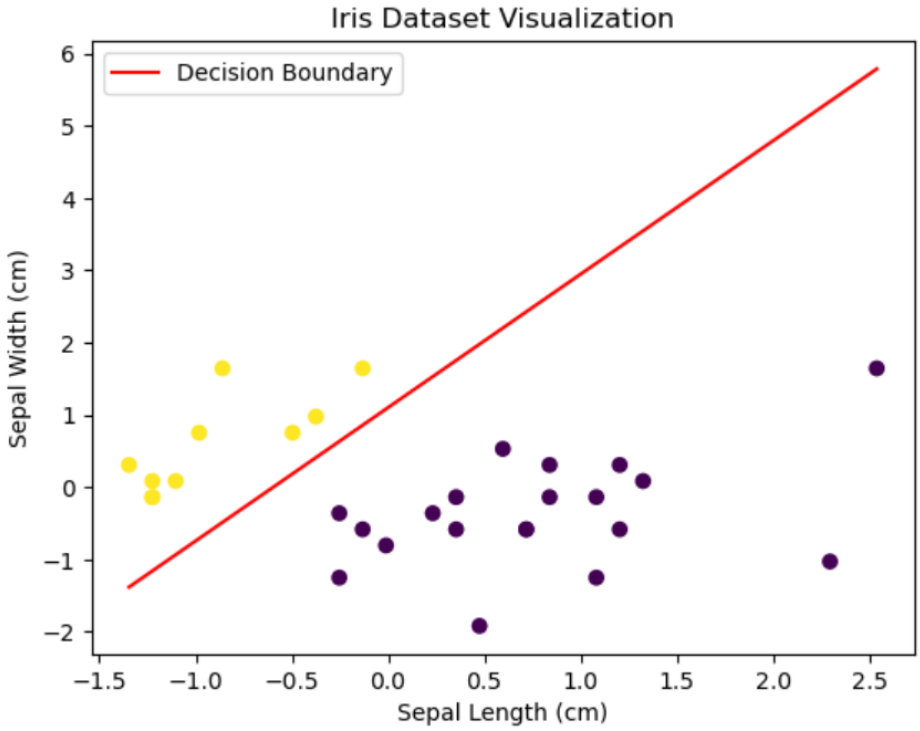


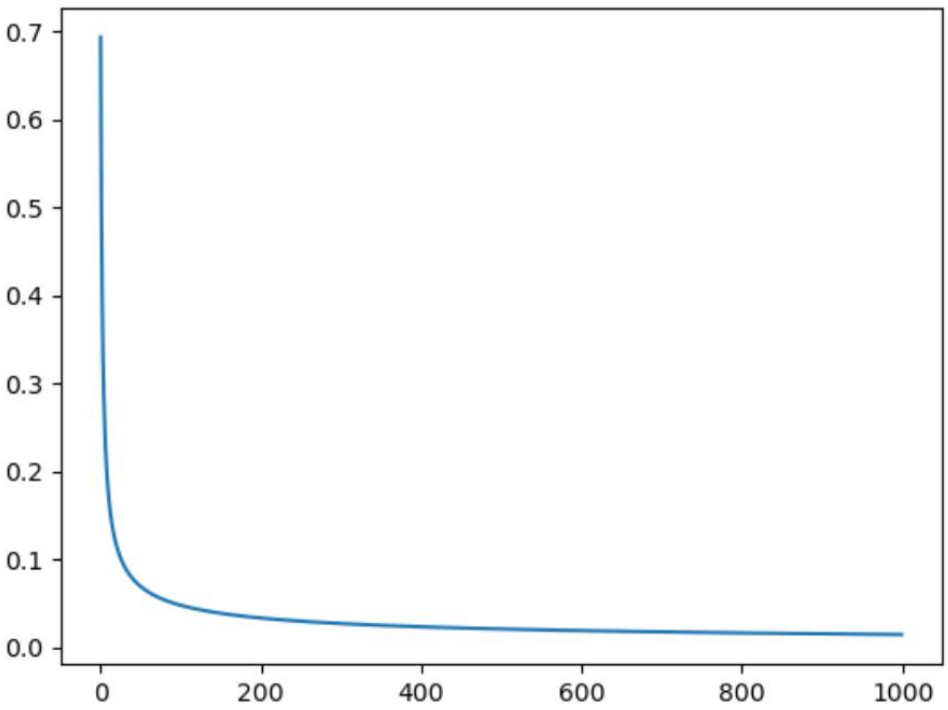




Q5: Draw the decision boundary on the test set using the learned parameters. Is this decision boundary separating the classes? Does this match our expectations?

According to the graph below, the training model accurately predicts the setosa class on the test set. The values above the decision boundary in yellow represent the setosa class based on the sepal length and width, and the values below the decision boundary represent the rest of the classes (Versicolor & Virginica). Yes, it does match our expectation as the given code considers the features sepal length and width to determine the weather its setosa or any other class and draws the decision boundary correctly based on the learned parameters. And also we observe that the cost for the setosa training model is less compared to other classes which indicates that the model accurately predicts the setosa class.






```
Parameter containing:
tensor([[-7.6483,  4.1452, -4.5710]])
Minimum cost: 0.014265164732933044
```

Q6. Using the 3 classifiers, predict the classes of the samples in the test set and show the predictions in a table. Do you observe anything interesting?

In the prediction table classification 0 is represented as Setosa, 1 is represented as Versicolor, 2 is represented as virg inica. Actual values are represented in tensor format, with tensor[1,0,0] indicating class 0, [0,1,0] indicating class 1 an d tensor[0,0,1] indicating class 2.

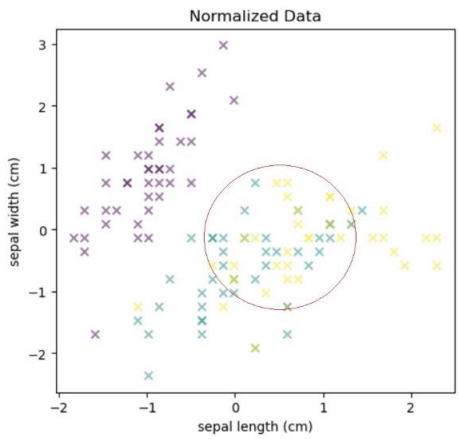
In classification 0 is repersented as Setosa, 1 is represented as Versicolor, 2 is represented as virginica

Sample	Classification	Actual values
0	1	tensor([0, 1, 0])
1	0	tensor([1, 0, 0])
2	2	tensor([0, 0, 1])
3	2	tensor([0, 1, 0])
4	1	tensor([0, 1, 0])
5	0	tensor([1, 0, 0])
6	1	tensor([0, 1, 0])
7	2	tensor([0, 0, 1])
8	1	tensor([0, 1, 0])
9	1	tensor([0, 1, 0])
10	2	tensor([0, 0, 1])
11	0	tensor([1, 0, 0])
12	0	tensor([1, 0, 0])
13	0	tensor([1, 0, 0])
14	0	tensor([1, 0, 0])
15	2	tensor([0, 1, 0])
16	2	tensor([0, 0, 1])
17	1	tensor([0, 1, 0])
18	1	tensor([0, 1, 0])
19	2	tensor([0, 0, 1])
20	0	tensor([1, 0, 0])
21	2	tensor([0, 0, 1])
22	0	tensor([1, 0, 0])
23	2	tensor([0, 0, 1])
24	2	tensor([0, 0, 1])
25	2	tensor([0, 0, 1])
26	1	tensor([0, 0, 1])
27	2	tensor([0, 0, 1])
28	0	tensor([1, 0, 0])
29	0	tensor([1, 0, 0])

Upon close observation we see that majority of the classes are predicted accurately on the test set, the model inaccu rately predicted some versicolor and virginica classes in samples 3, 15, 26. But on the other hand the setosa class is c orrectly predicted. We can see the same result based on the cost below, as setosa cost is less compared to versicolor and virginica models. This indicates that the trained model correctly predicts the setosa but not the other classes.

Training Setosa Model
Minimum train cost: 0.03267652541399002
Minimum test cost: 0.027389848604798317
Training Versicolor Model
Minimum train cost: 0.5670453310012817
Minimum test cost: 0.5886773467063904
Training Virginica Model
Minimum train cost: 0.3606893718242645
Minimum test cost: 0.29108360409736633

We can see the same thing when plotting the dataset, where data points collide with each other for versicolor and vi rginica. For instance, when plotting data for sepal length and width.



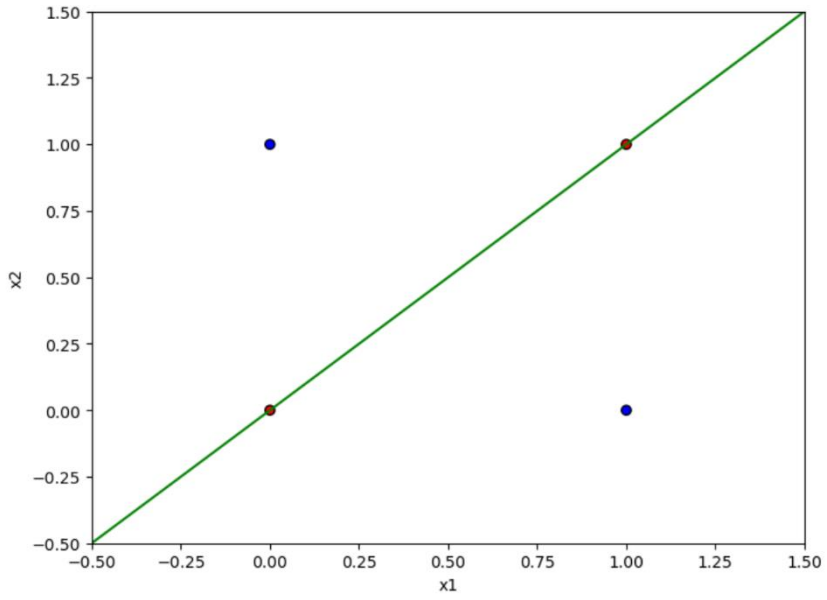
Q7. Calculate the accuracy of the classifier on the test set, by comparing the predicted values against the ground truth. Use a softmax for the classifier outputs.

The accuracy on the test set is approximately 90%. This means the model is correctly predicting the type of iris flower 90% of the time. In the code we are calculating the model outputs for each class on the test set. Then these outputs a re passed through SoftMax function to convert them into probabilities making sure each instance equals to one. The n we are taking the highest probability of each instance and comparing them to the actual class values from the test s et. Giving us a accuracy on the test set by taking the mean correct predictions.

```
Accuracy on the test set: 89.99999761581421 %
```

Q8. Looking at the datapoints below, can we draw a decision boundary using Logistic Regression? Why? What are the specific issues or logistic regression with regards to XOR?

No, we can't draw a decision boundary using Logistic regression as its linear, In Logistic regression classifies the data points using a straight line. In XOR data points [0,0] and [1,1] which are in red, we cant draw a decision boundary as it would divide rest of the datapoints in blue as separate class. So Linear regression cannot classify the data points with a single linear decision boundary. Refer the below image, here we added a hypothetical line passing from data points [0,0] which clearly shows the data points in blue are in different classes making the XOR problem not solvable by Logistic Regression.



Neural Networks:

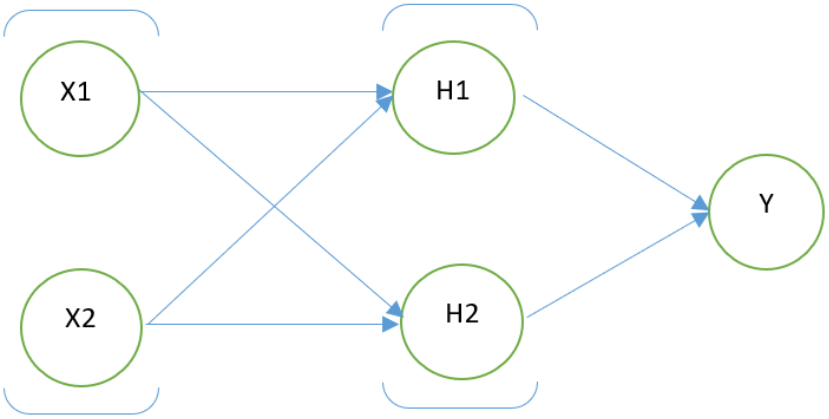
Q1. Why is it important to use a random set of initial weights rather than initializing all weights as zero in a Neural Network?

Initializing weights to zero in a neural network serves no purpose. The neurons of all layers perform the same calculation and produce the same output making it useless. Imagine initializing the weights are 0, in this case hidden unit will receive exactly the same signal for each neuron and prediction will be poor.

When we initialize the weights to random, the symmetry breaks and gives much better accuracy. In this case, each neuron is not performing the same as previous one making the prediction better. The weights are initialised in such a way that the previous layer is considered and set to random weight to new neuron based on the size of the previous layer, resulting in a controlled initialisation that is faster and more efficient.

Q2. How does a NN solve the XOR problem?

The main issue with XOR problem is it has a data points where both x,y are same give 0 and where they are different gives 1. With this we cant solve XOR problem with logistic regression cause we cant linearly separate them, so we use neural networks to solve this issue. The neural network has 2 input layers, 1 hidden layers with 2 neurons and 1 neuron in output layer.



Here X1 and x2 are data points with are two binary variables with 0 or 1, H1 and H2 are hidden layers, each neuron have wights and a bias term. The weights sum and bias are passed through non linear activation function like sigmoid. The weights determine the importance of each neuron H1 and H2 and the weights are calculated using the formula $H1=f(W11 \cdot X1+W12 \cdot X2+b1)$, $H2=f(W21 \cdot X1+W22 \cdot X2+b2)$ where Wij are the wights applied to the input connections. The Output Y combines the weights from H1 and H2 and it also has its own weights and bias and uses sigmoid function for binary classification. The output will be in a range of 0 – 1. The network trained on the XOR with appropriate weights will output 0 for inputs (0,0) and (1,1), and for points (0,1) and (1,0) the output will be near to 1.

Q3. Explain the performance of the different networks on the training and test sets. How does it compare to the logistic regression example? Make sure that the data you are refering to is clearly presented and appropriately labeled in the report.

For

```
Hidden Neurons size: 1
Minimum Training Loss: 0.5853
Minimum Test Loss: 0.5850
Train Accuracy at Best Epoch: 0.7250
Test Accuracy at Best Epoch: 0.7333
```

Hidden neuron of size 1 has lowest accuracy of around 72% and highest loss on the both train and test sets. Implying the model is underfitting the data.

```
Hidden Neurons size: 2
Minimum Training Loss: 0.4533
Minimum Test Loss: 0.4009
Train Accuracy at Best Epoch: 0.9417
Test Accuracy at Best Epoch: 0.9333
```

Hidden neuron of size 2 there is a significant improvement on both training set and test sets compared to neuron wit h size 1 and also has the highest accuracy on the train set about 94%. The model indicates that it's a better fit than ne uron case 1.

```
Hidden Neurons size: 4
Minimum Training Loss: 0.4710
Minimum Test Loss: 0.4537
Train Accuracy at Best Epoch: 0.6833
Test Accuracy at Best Epoch: 0.7000
```

Hidden neuron 4 the test loss is slightly better than the 2 neurons, but the test accuracy drops implying the start of o verfitting where model learns the data.

```
Hidden Neurons size: 8
Minimum Training Loss: 0.3097
Minimum Test Loss: 0.2504
Train Accuracy at Best Epoch: 0.9000
Test Accuracy at Best Epoch: 0.9667
```

Hidden neuron 8 the accuracy shows the lowest test loss and highest test accuracy, indicating the model is balanced.

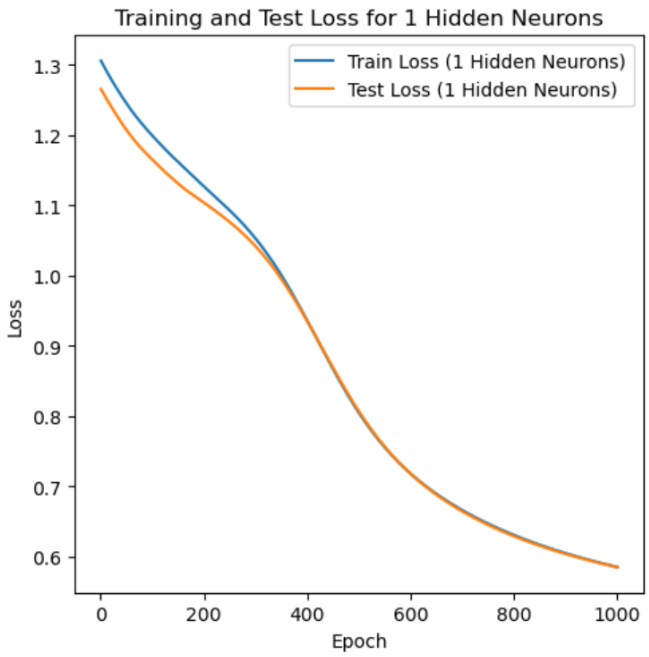
```
Hidden Neurons size: 16
Minimum Training Loss: 0.2392
Minimum Test Loss: 0.1837
Train Accuracy at Best Epoch: 0.9250
Test Accuracy at Best Epoch: 0.9667
```

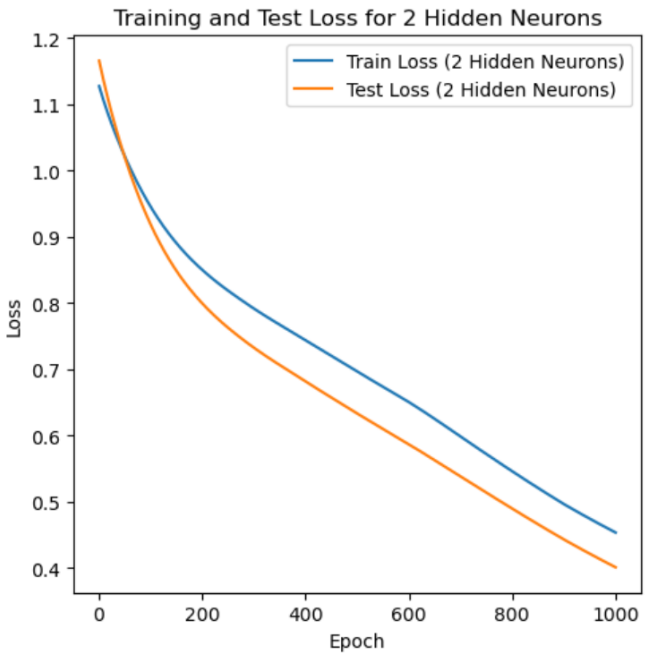
Hidden neuron 16 continues to decrease the loss, but the test loss and accuracy plateau and doesn't improve the per formance.

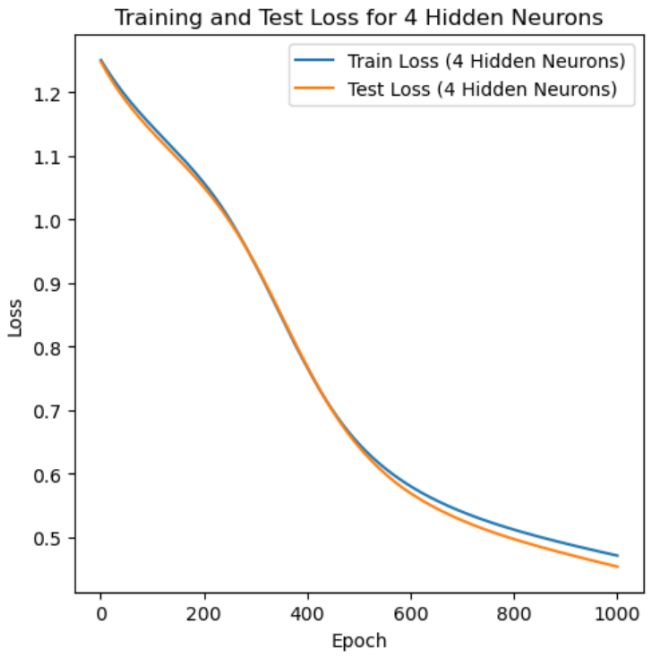
```
Hidden Neurons size: 32
Minimum Training Loss: 0.2290
Minimum Test Loss: 0.1647
Train Accuracy at Best Epoch: 0.9333
Test Accuracy at Best Epoch: 0.9667
```

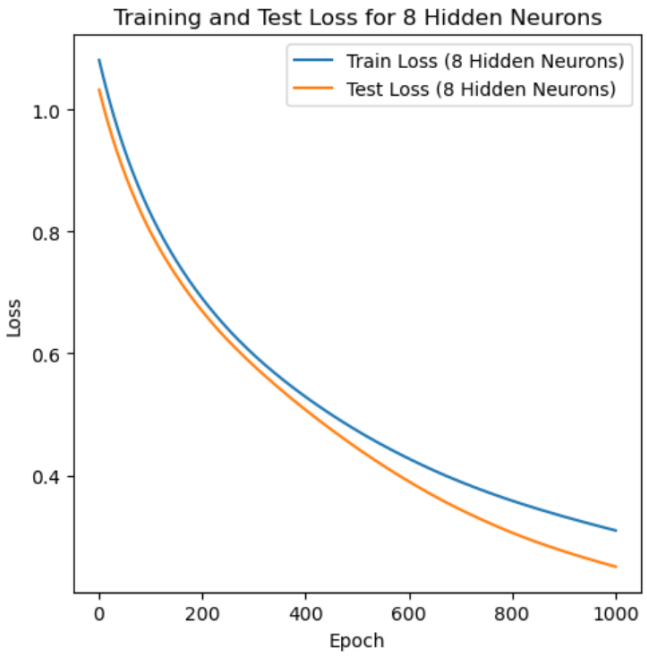
Hidden neuron 32 there is a slight improvement over 16 neuron, but no improvement in the test loss or accuracy.

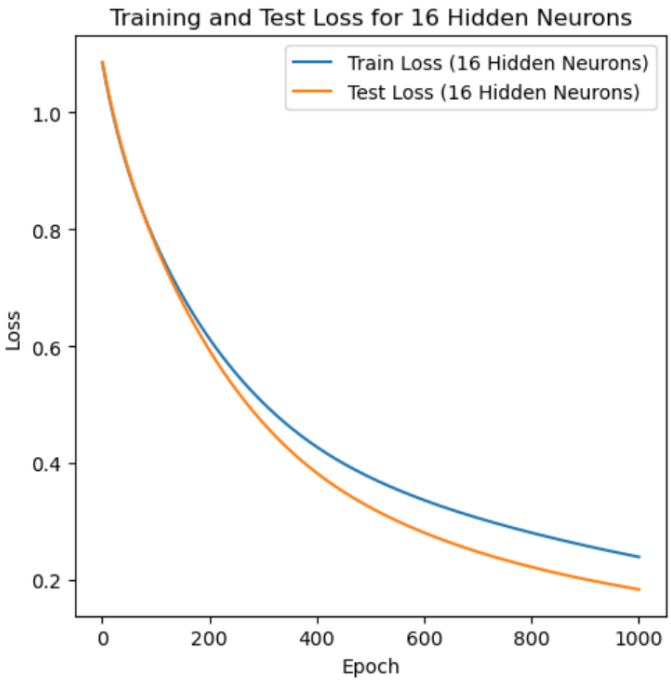
When comparing to the logistic regression, the neural networks particularly models with high hidden neurons tend to have better accuracy. Models with 8, 16 and 32 neurons achieve high accuracy above 95% whereas for logistic regres sion the accuracy on the test set compared to train set was approximately 90%. For IRIS data set, neural networks is a better model compared to classification. However, logistic regression provide good performance for setosa class.

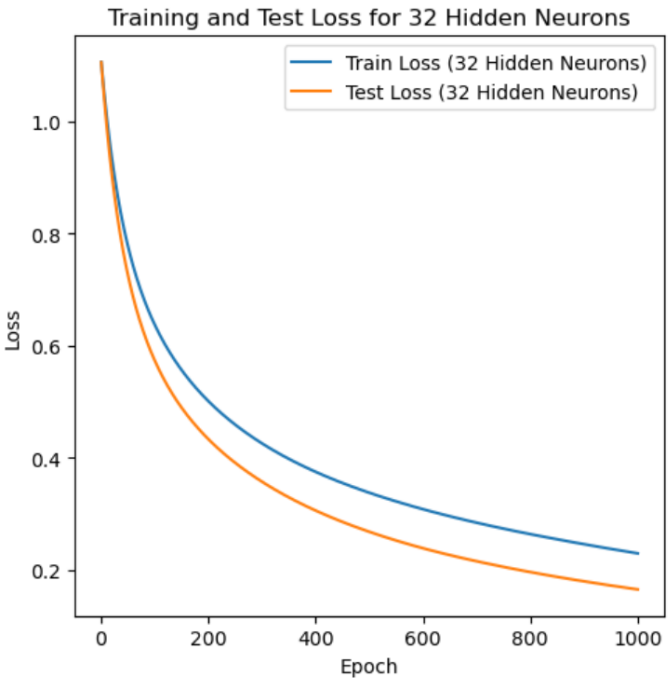

Training and Test Loss for 1 Hidden Neurons
Train Loss (1 Hidden Neurons)
Test Loss (1 Hidden Neurons)
Loss
Epoch


Training and Test Loss for 2 Hidden Neurons
Train Loss (2 Hidden Neurons)
Test Loss (2 Hidden Neurons)
Loss
Epoch


Training and Test Loss for 4 Hidden Neurons
Train Loss (4 Hidden Neurons)
Test Loss (4 Hidden Neurons)
Loss
Epoch


Training and Test Loss for 8 Hidden Neurons
Train Loss (8 Hidden Neurons)
Test Loss (8 Hidden Neurons)
Loss
Epoch


Training and Test Loss for 16 Hidden Neurons
Train Loss (16 Hidden Neurons)
Test Loss (16 Hidden Neurons)
Loss
Epoch


Training and Test Loss for 32 Hidden Neurons
Train Loss (32 Hidden Neurons)
Test Loss (32 Hidden Neurons)
Loss
Epoch