# CS671: DEEP LEARNING

ASSIGNMENT 1

**SURAJ(B18027) [*1], PRANAV ADLINGE (B19183) [†2] and NAMAN TAYAL (B18123) [‡3]**

[1] *b18027@students.iitmandi.ac.in*
[2] *b19183@students.iitmandi.ac.in*
[3] *b18123@students.iitmandi.ac.in*

Textolivre
Linguagem e Tecnologia

**DOI:**

**Corresponding author**
**Edited by**
**Received on**
**Accepted on**
**Published on**


*Email: b18027@students.iitmandi.ac.in
†Email: b19183@students.iitmandi.ac.in
‡Email: b18123@students.iitmandi.ac.in

## Contents

# 1 AIM

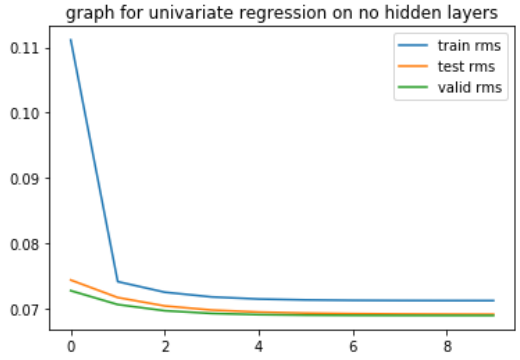
The Aim of this assignment is to implement Perceptron, MLFFNN and propagation learning algorithm, on the given datasets, from scratch, using python, and evaluate the plots of epochs vs average error, plots of MSE(Mean Squared Error), plotsof model output and target output, Comparison of performance of different models for classification of each data-sets, Scatterplot for training, test, and validation data.

# 2 REGRESSION

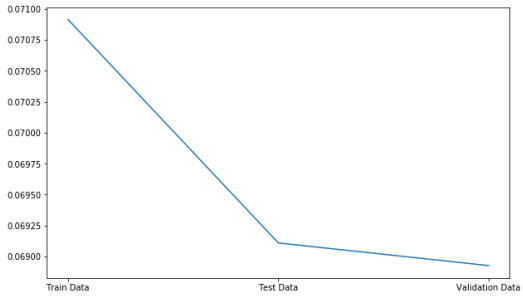
## 2.1 UNIVARIATE DATA

This data-set has one dimensional input and one output variable. The models applied on this data are perceptron with linear activation function, MLFNN with one hidden layer and MLFNN with two hidden layers.

### 2.1.1 NO HIDDEN LAYERS

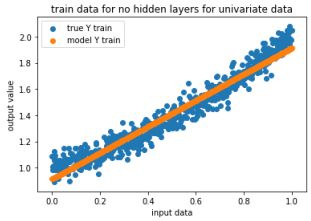


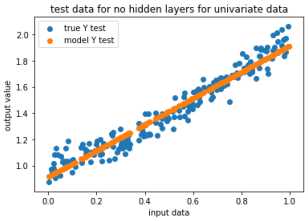

(a) RMS vs Epoch.

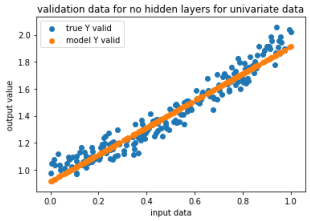



(b) RMS values for different datasets.

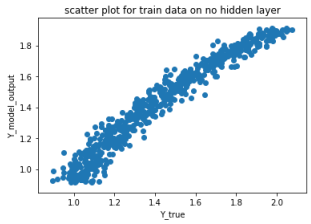
MODEL OUTPUT AND TARGET OUTPUT

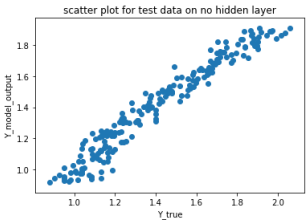


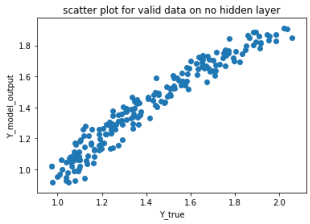




MODEL OUTPUT VS TARGET OUTPUT

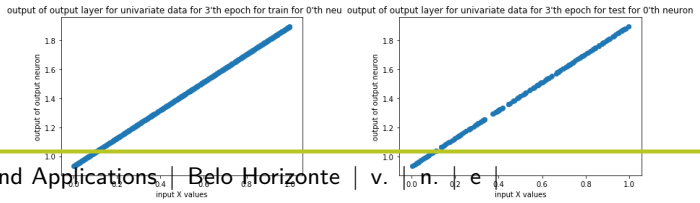


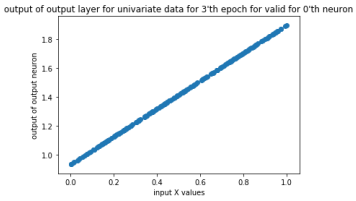



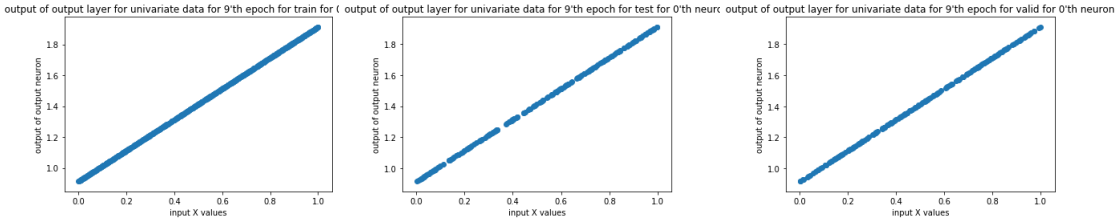

PLOT OF OUTPUT NODES AT DIFFERENT EPOCH VALUES DURING TRAINING

after 3 epochs

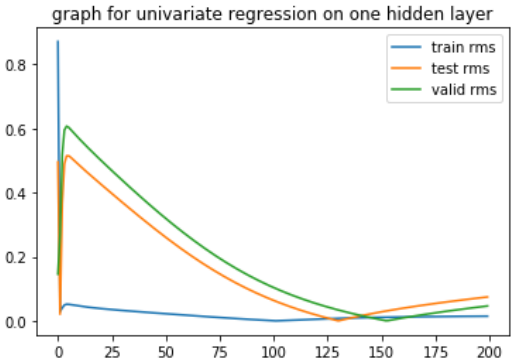




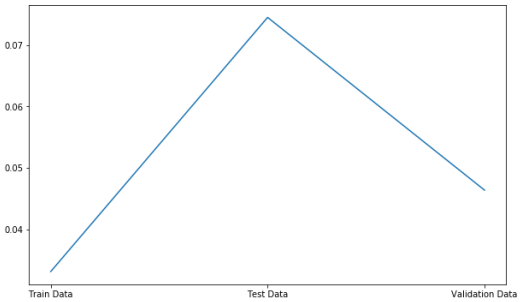

after 9 epochs




### 2.1.2 ONE HIDDEN LAYER

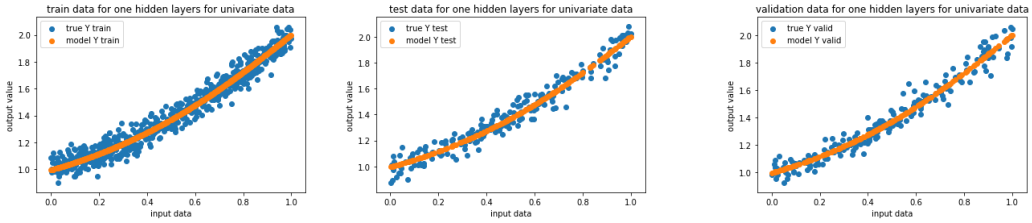



(a) RMS vs Epoch.

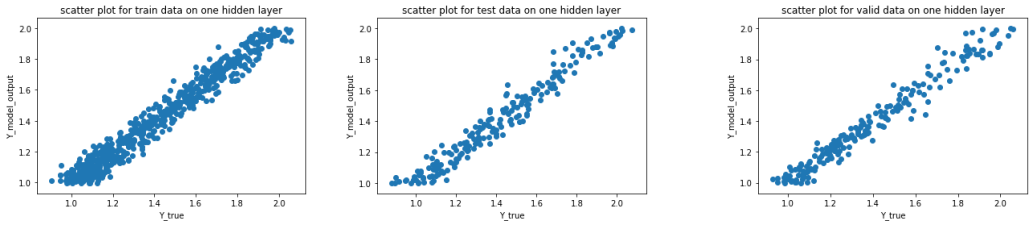



(b) RMS values for different datasets.

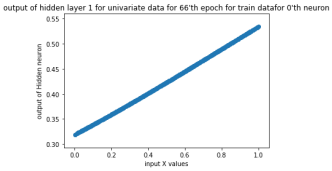
MODEL OUTPUT AND TARGET OUTPUT



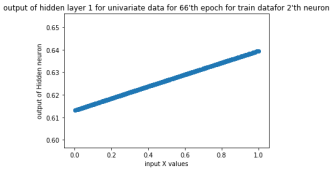

MODEL OUTPUT VS TARGET OUTPUT



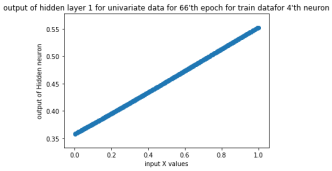

PLOT OF HIDDEN NODES AT DIFFERENT NO OF EPOCHS DURING TRAINING at epoch value 66




(a) Neuron 0

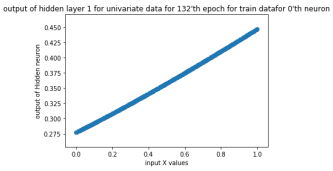



(b) Neuron 2

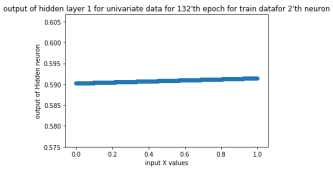



(c) Neuron 4

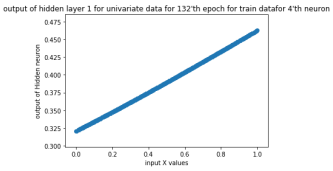
at epoch value 132




(a) Neuron 0



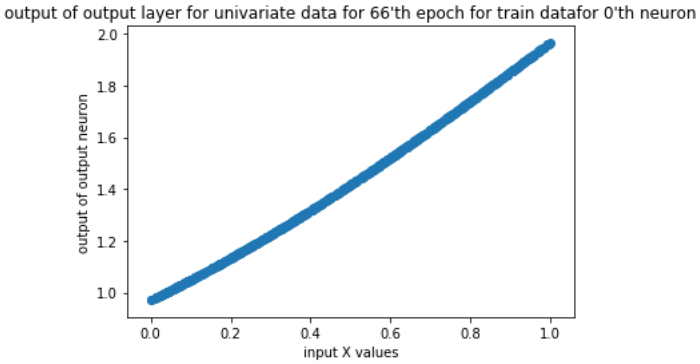

(b) Neuron 2



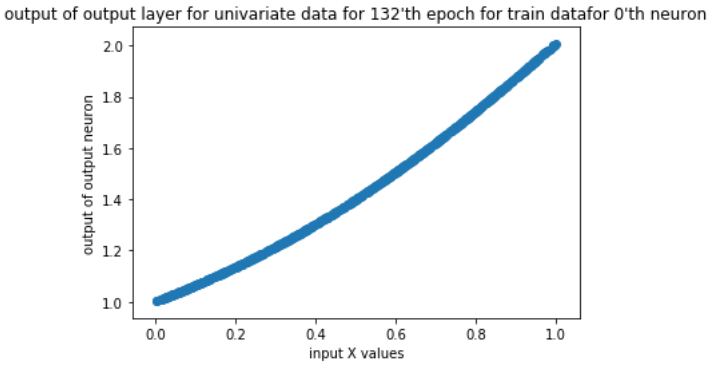

(c) Neuron 4

PLOT OF OUTPUT NODES AT DIFFERENT NO OF EPOCHS DURING TRAINING

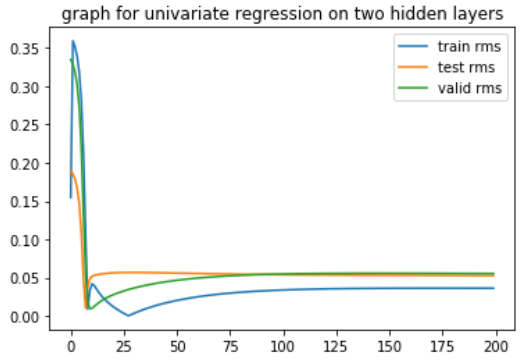
at epoch value 66



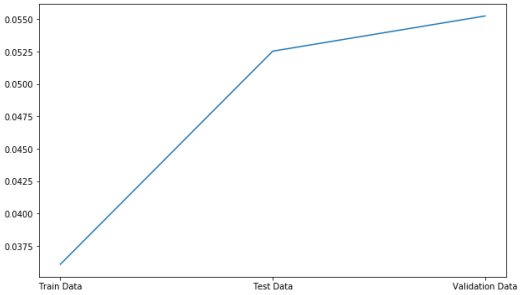

at epoch value 132



### 2.1.3 TWO HIDDEN LAYERS

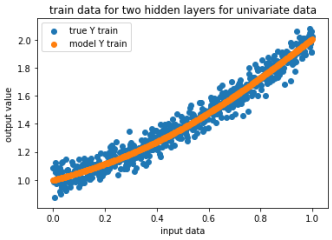



(a) RMS vs Epoch.

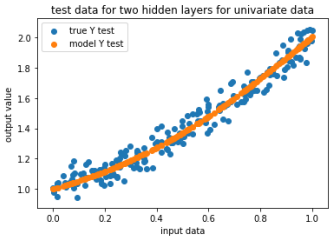



(b) RMS values for different datasets.

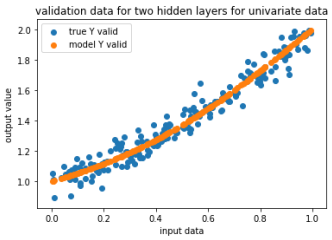
MODEL OUTPUT AND TARGET OUTPUT




(a) TRAIN DATA

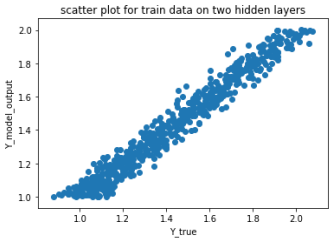



(b) TEST DATA

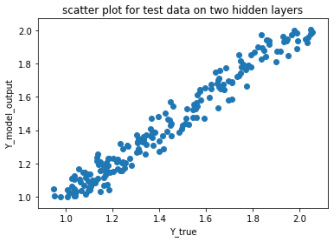



(c) VALIDATION DATA

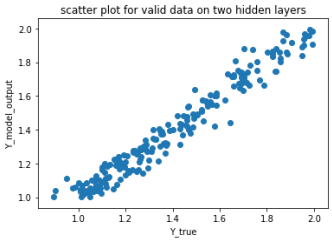
MODEL OUTPUT VS TARGET OUTPUT




(a) TRAIN DATA



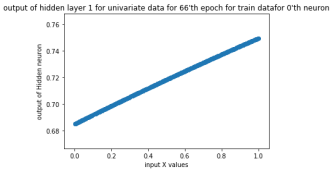

(b) TEST DATA



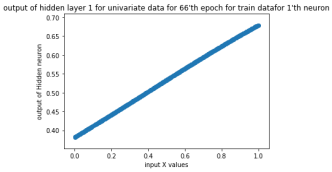

(c) VALIDATION DATA

PLOT OF HIDDEN LAYER 1 AT DIFFERENT NO OF EPOCHS DURING TRAINING

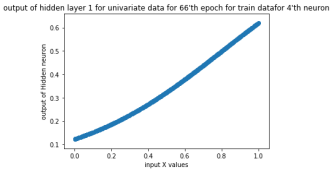
at epoch value 66




(a) Neuron 0

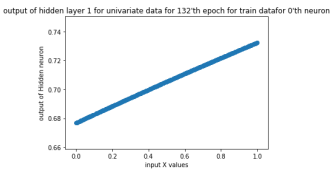



(b) Neuron 2

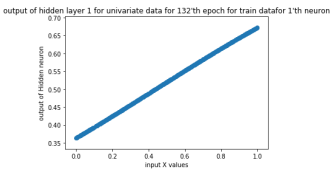



(c) Neuron 4

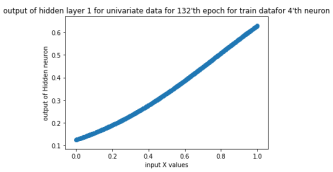
at epoch value 132




(a) Neuron 0



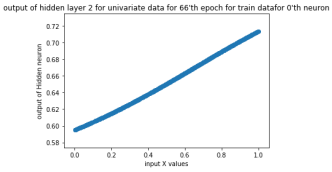

(b) Neuron 2



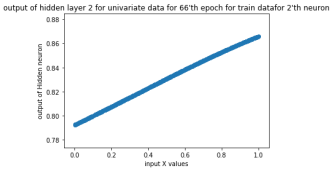

(c) Neuron 4

PLOT OF HIDDEN LAYER 2 AT DIFFERENT NO OF EPOCHS DURING TRAINING

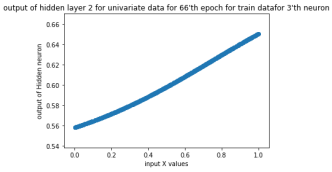
at epoch value 66




(a) Neuron 0

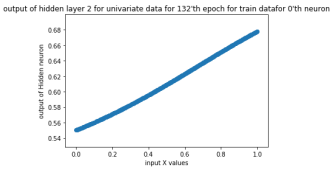



(b) Neuron 2

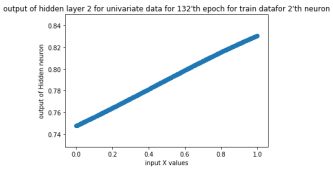



(c) Neuron 3

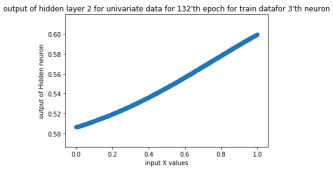
at epoch value 132




(a) Neuron 0



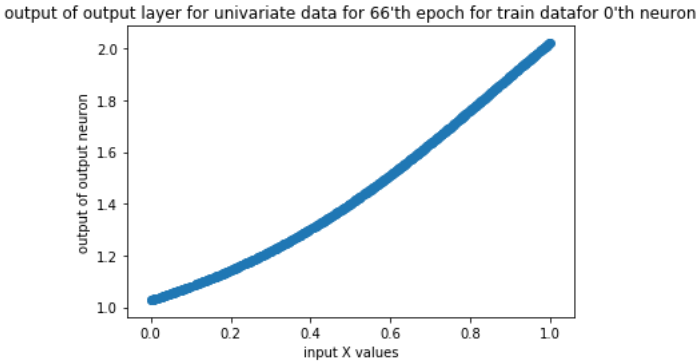

(b) Neuron 2



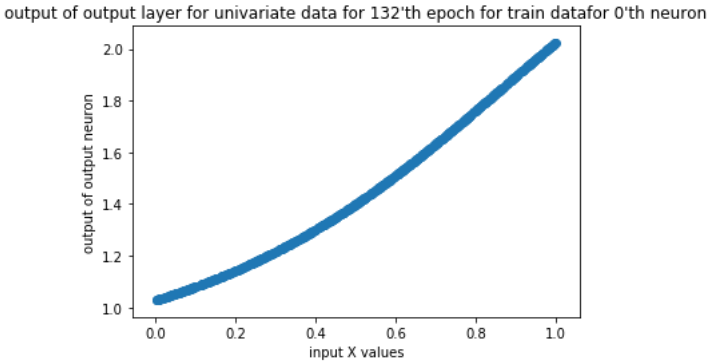

(c) Neuron 3

PLOT OF OUTPUT NEURON AT DIFFERENT NO OF EPOCHS DURING TRAINING

at epoch value 66




at epoch value 132



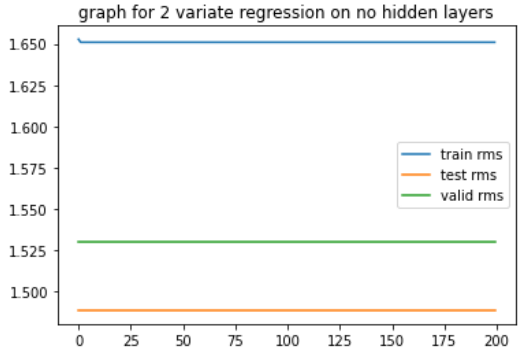

### 2.1.4 OBSERVATIONS

Since the data is univariate and associated function between input and output variable is linear so all three models, model with no hidden layers, one hidden layer, two hidden layers performs well on the data and anyone of them can be used. But in such cases, model with no hidden layers should be chosen cause it get less weights to update and hence is faster then other two. Output of hidden nodes of models is also linear and net output too is linear.

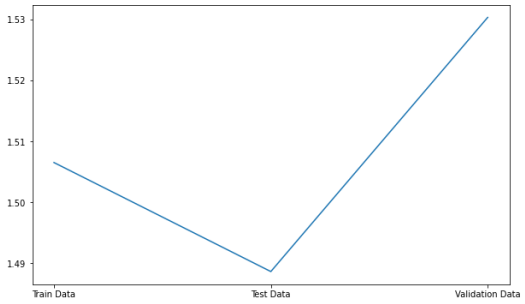
## 2.2 BIVARIATE DATA

This data-set has two dimensional input and one output variable. The models applied on this data are perceptron with linear activation function, MLFNN with one hidden layer and MLFNN with two hidden layers.

### 2.2.1 NO HIDDEN LAYERS

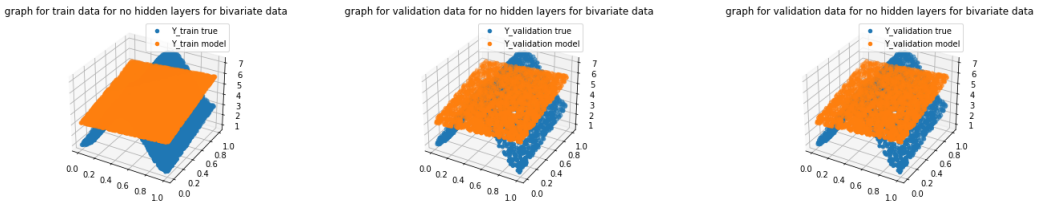



(a) RMS vs Epoch.



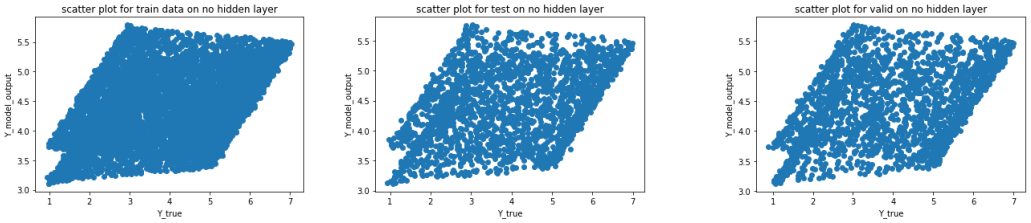

(b) RMS values for different datasets.

MODEL OUTPUT AND TARGET OUTPUT



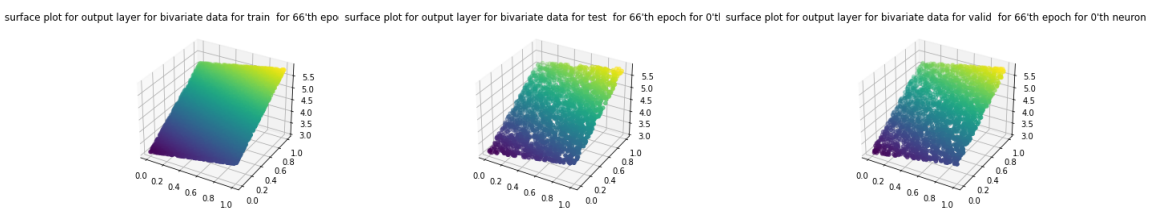

This model is not able to learn the underlying distribution of the training data as the model is linear and data cannot be apropriately modeled using a linear model. Hence a more complex model must be used.

MODEL OUTPUT VS TARGET OUTPUT

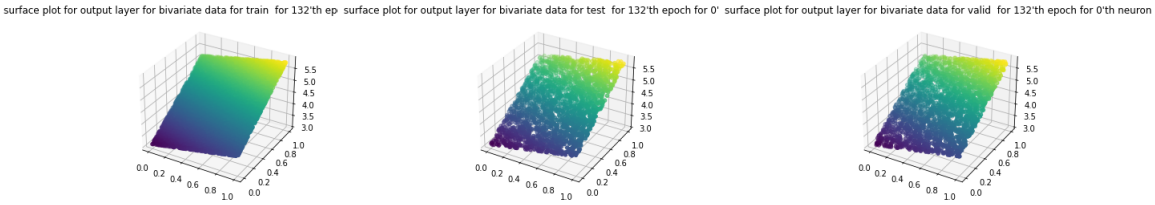



PLOT OF OUTPUT NODE AT DIFFERENT EPOCH VALUES DURING TRAINING

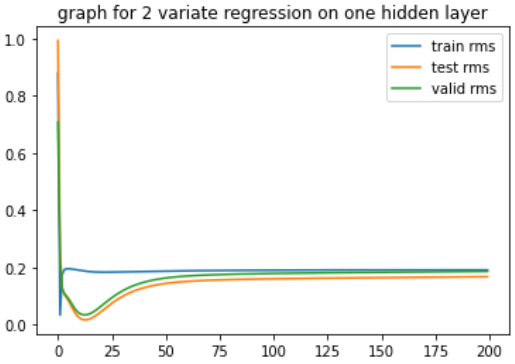
After 66 epochs



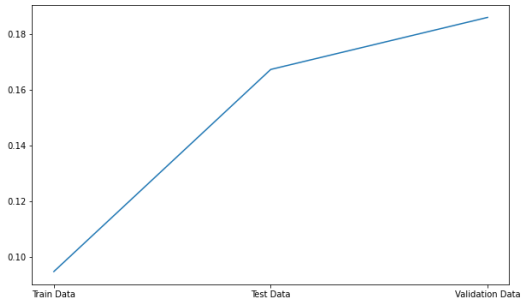

After 132 epochs




### 2.2.2 ONE HIDDEN LAYER

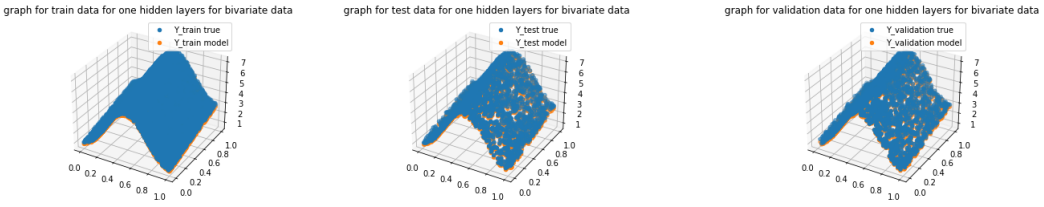



(a) RMS vs Epoch.




(b) RMS values for different datasets.

MODEL OUTPUT AND TARGET OUTPUT

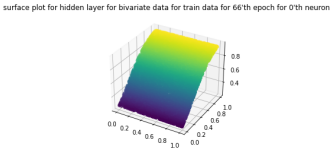



This model is performing well and able to predict the output variable very close to the true value of output variable.

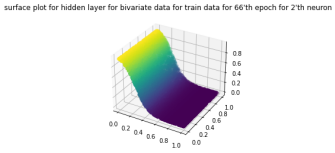
MODEL OUTPUT VS TARGET OUTPUT


PLOT OF HIDDEN NODES AT DIFFERENT NO OF EPOCHS DURING TRAINING

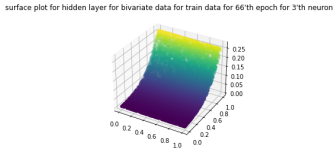
At 66th Epoch



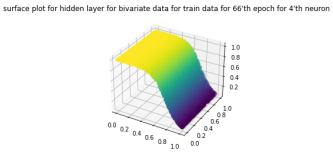

(a) Neuron 0




(b) Neuron 2

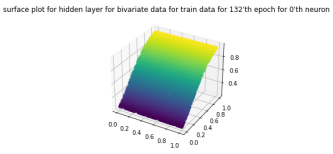



(c) Neuron 3

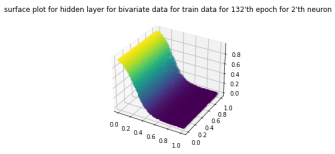



(d) Neuron 4

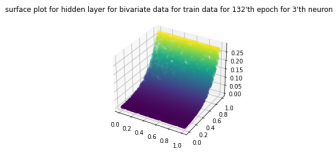
At 132th Epoch



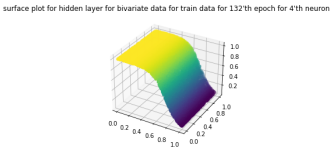

(a) Neuron 0




(b) Neuron 2



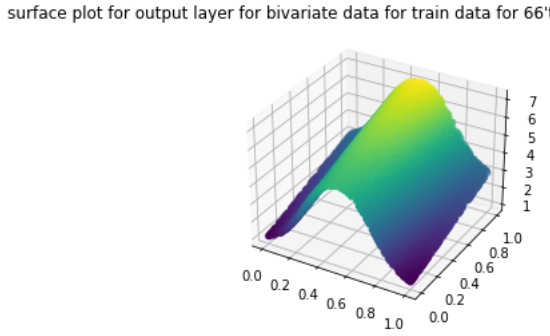

(c) Neuron 3



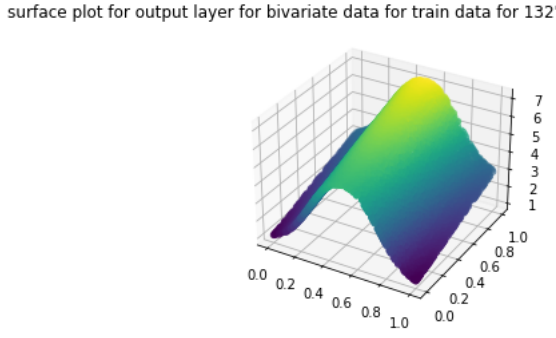

(d) Neuron 4

PLOT OF OUTPUT NODES AT DIFFERENT NO OF EPOCHS DURING TRAINING

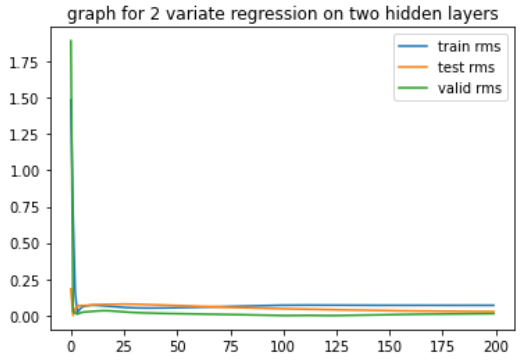
At 66th Epoch



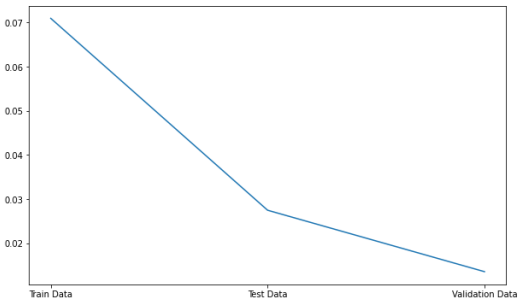

At 132th Epoch



### 2.2.3 TWO HIDDEN LAYERS

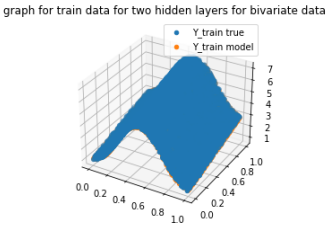



(a) RMS vs Epoch.

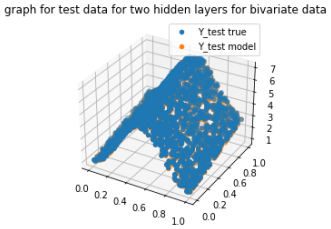



(b) RMS values for different datasets.

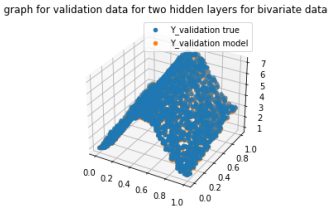
MODEL OUTPUT AND TARGET OUTPUT




(a) TRAIN DATA

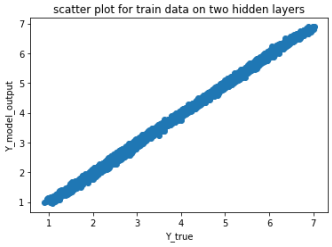



(b) TEST DATA



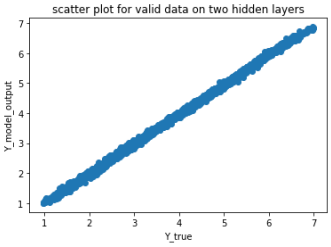

(c) VALIDATION DATA

MODEL OUTPUT VS TARGET OUTPUT




(a) TRAIN DATA

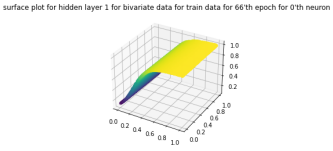

(b) TEST DATA



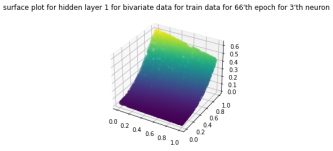

(c) VALIDATION DATA

PLOT OF HIDDEN LAYER 1 AT DIFFERENT NO OF EPOCHS DURING TRAINING

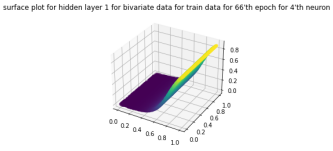
at epoch value 66




(a) Neuron 0

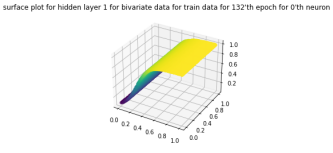



(b) Neuron 3

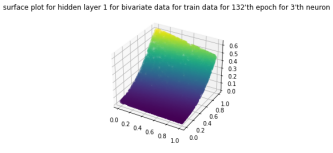



(c) Neuron 4

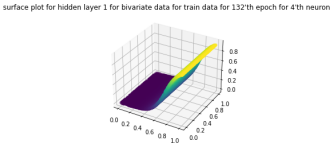
at epoch value 132




(a) Neuron 0



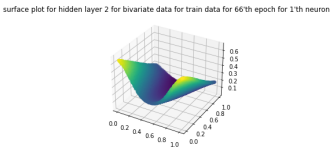

(b) Neuron 3



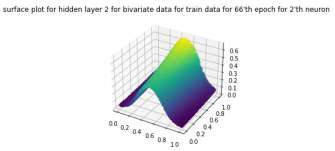

(c) Neuron 4

PLOT OF HIDDEN LAYER 2 AT DIFFERENT NO OF EPOCHS DURING TRAINING

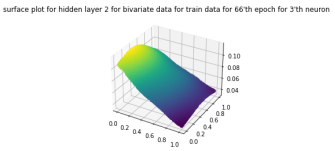
at epoch value 66




(a) Neuron 1

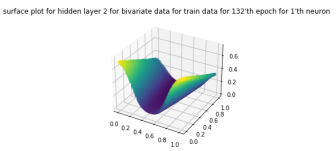



(b) Neuron 2

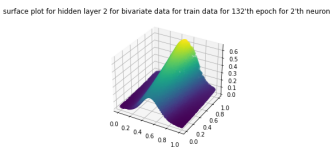



(c) Neuron 3

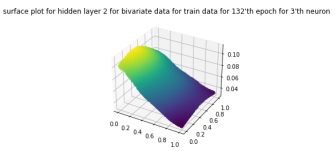
at epoch value 132




(a) Neuron 1



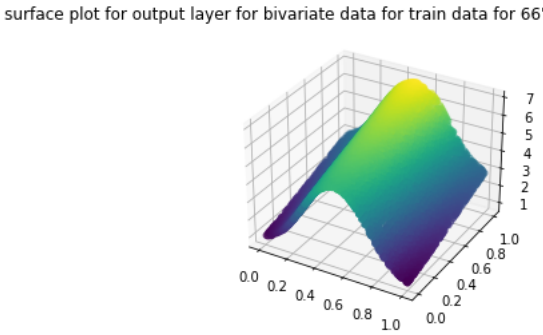

(b) Neuron 2



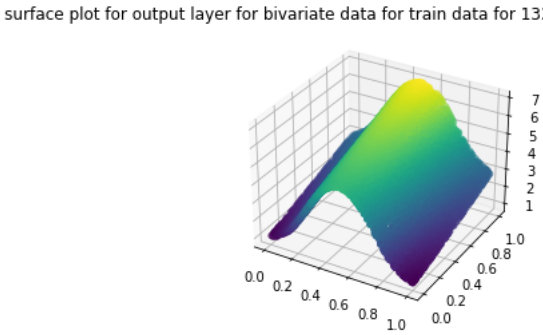

(c) Neuron 3

PLOT OF OUTPUT NEURON AT DIFFERENT NO OF EPOCHS DURING TRAINING

at epoch value 66




at epoch value 132




## 2.3 OBSERVATIONS

The associated function with given input and corresponding output is non-linear and that's why model with no hidden layer fails miserably. So we have to use models with one hidden layer or model with two hidden layers. These two models seems to work quite well on this data set. Output of hidden nodes of models too is non linear and net output of models(with one and two hidden layers) is non linear as well. Models with hidden layers work well not only on training data but validation data and test data as well hence we can say that cases of over-fitting and under-fitting are not been observed.

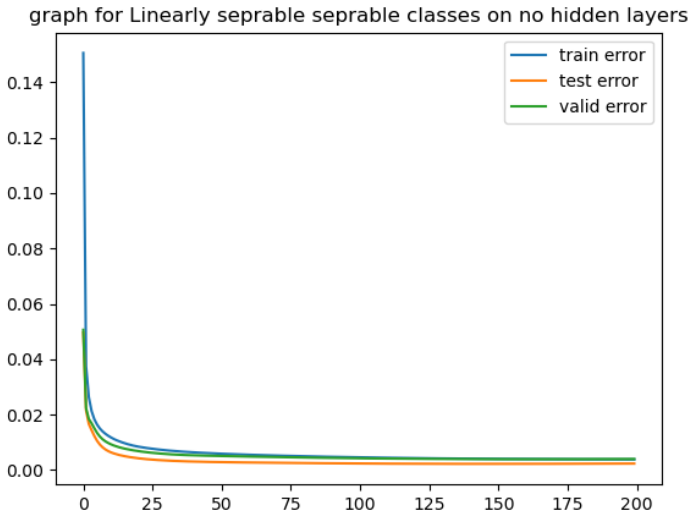
# 3 CLASSIFICATION

## 3.1 LINEARLY SEPARABLE

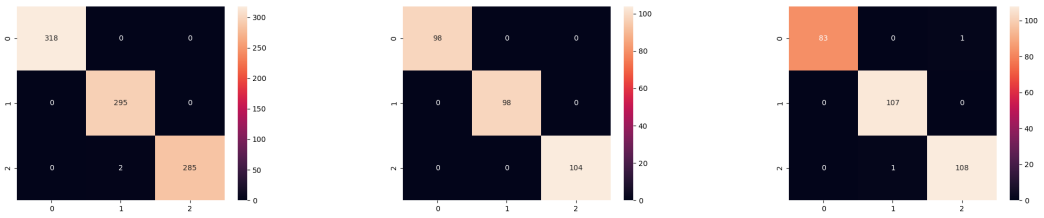
We were given a 2 - dimensional artificial data of 3 classes that is linearly separable. Each class had 500 data points. We used this data to train different models and predict the class using that model. We divided data into 20% test, 20% validation, and 60% train data.

### 3.1.1 Model with no hidden layers

We trained our model for no hidden layer for a total of 200 epochs and got the following RMSE vs Epochs Plot.



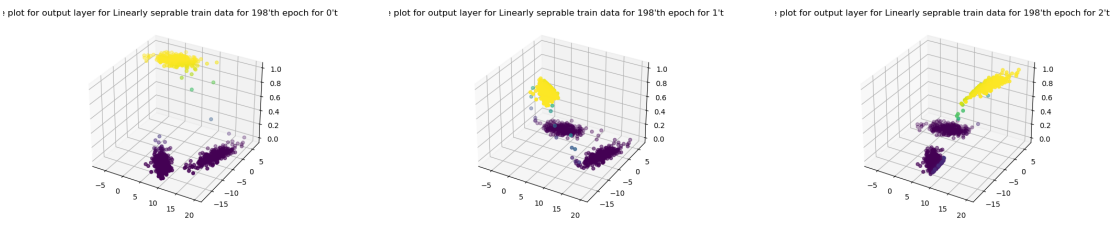

We got the following confusion matrix for train, test and validation data.




(a) Train Data (b) Test Data (c) Validation Data

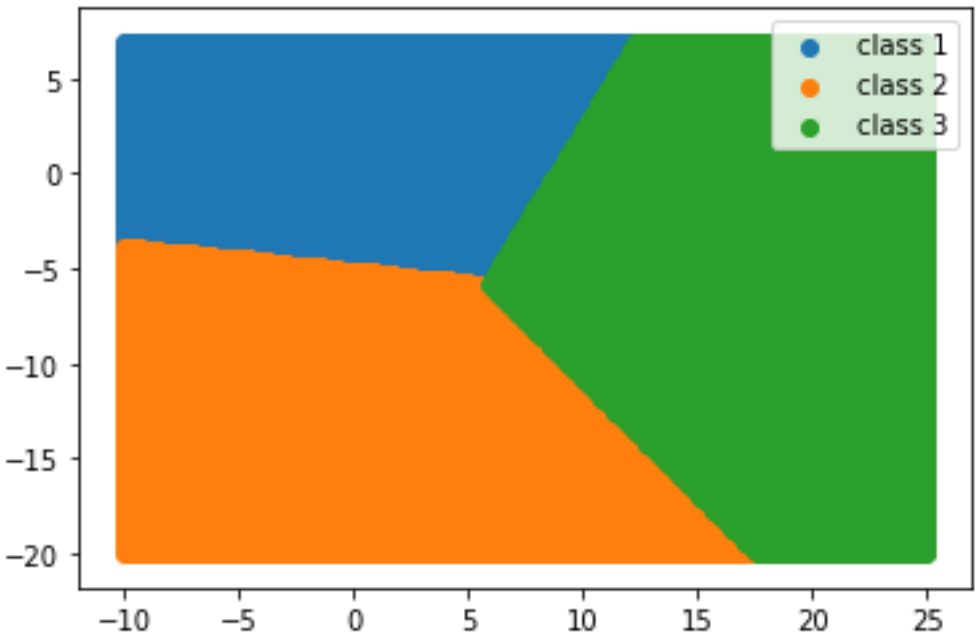
Accuracy for each data is very close to one because this is a Linearly Separable dataset so even the model with no hidden layer is pretty accurate.

PLOTS OF OUTPUT AT DIFFERENT NODES OF OUTPUT LAYER AFTER MODEL IS TRAINED




(a) Neuron 0 (b) Neuron 1 (c) Neuron 2

CLASS PREDICTED BY MODEL OF DIFFERENT POINTS

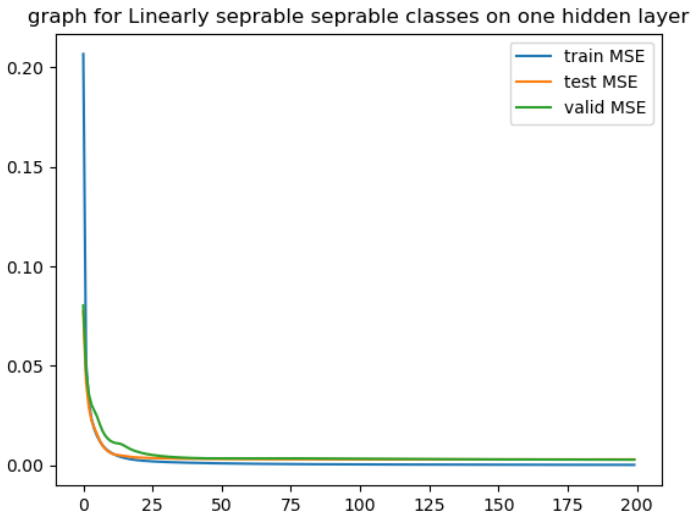



(a) TRAIN DATA

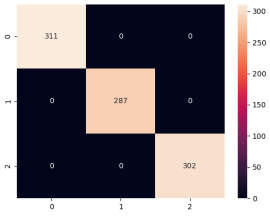
As we can see clearly from the above picture, decision region is linear.

### 3.1.2 Model with 1 hidden layer

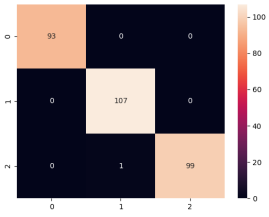
We added a hidden layer with 5 neurons to our model and then again trained it for 200 epochs. As a result we got the following RMSE vs Epochs Plot.



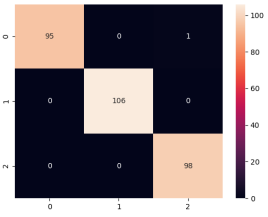

We got the following confusion matrix for train, test and validation data.




(a) Train Data

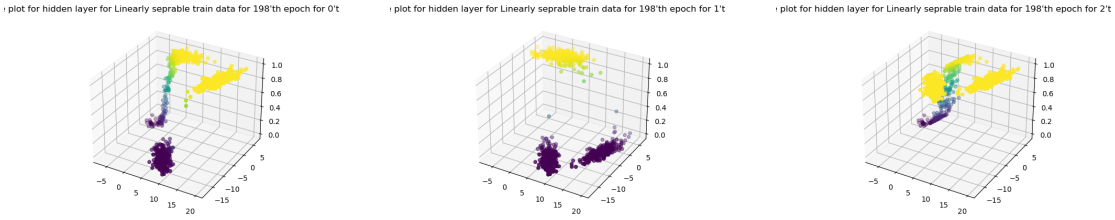



(b) Test Data

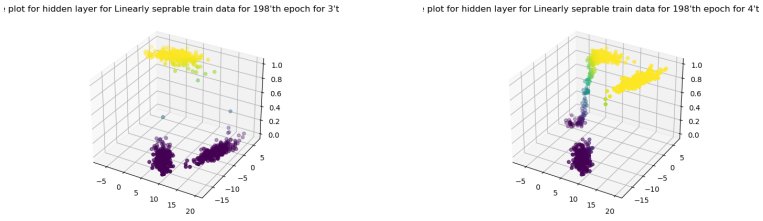



(c) Validation Data

PLOTS OF OUTPUT AT DIFFERENT NODES OF HIDDEN LAYER 1 AFTER MODEL IS TRAINED

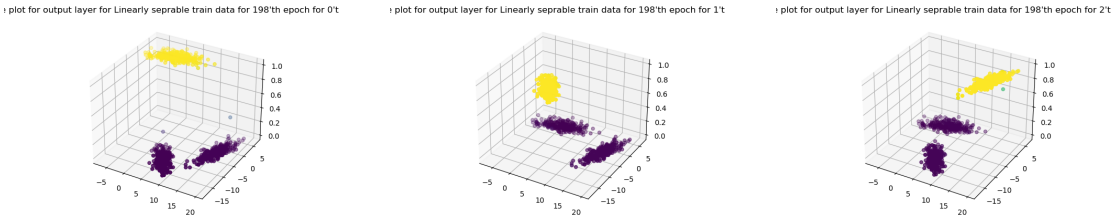



(a) Neuron 0 (b) Neuron 1 (c) Neuron 2



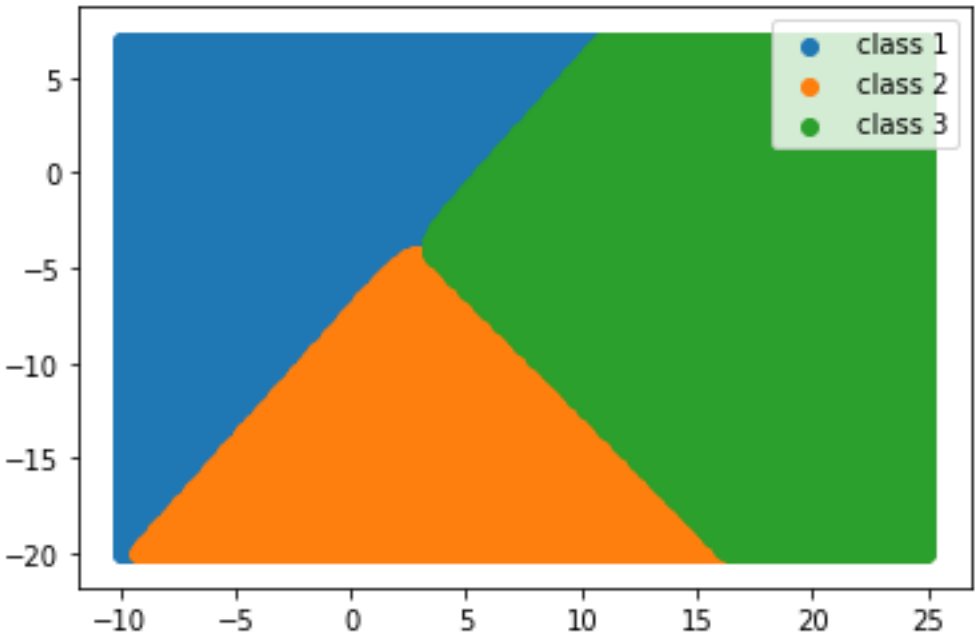

(a) Neuron 3 (b) Neuron 4

PLOTS OF OUTPUT AT DIFFERENT NODES OF OUTPUT LAYER AFTER MODEL IS TRAINED




(a) Neuron 0 (b) Neuron 1 (c) Neuron 2

Class predicted by model of different points:

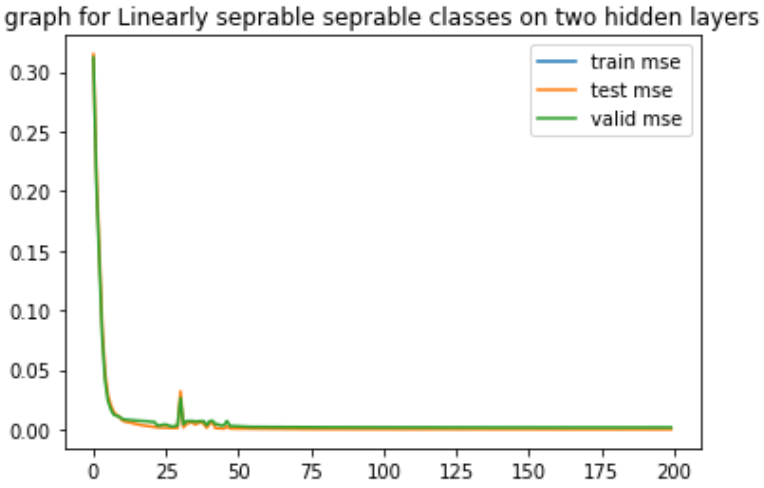



(a) TRAIN DATA

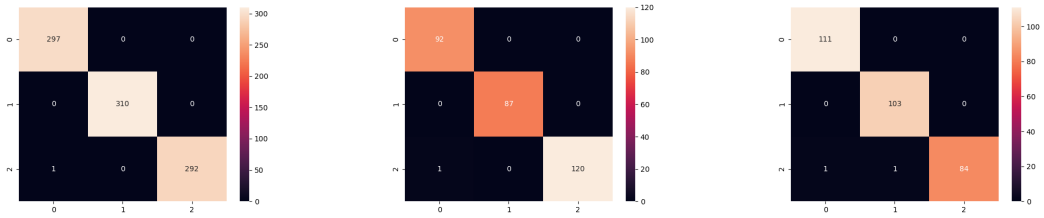
As we can see clearly from the above picture, decision region is linear.

### 3.1.3 Model with 2 hidden layer

In this model we added 2 hidden layers, with 5 neurons in layer 1 and 4 neurons in layer 2, to the model and trained it for 200 epochs. We got the following RMSE vs Epochs plot.

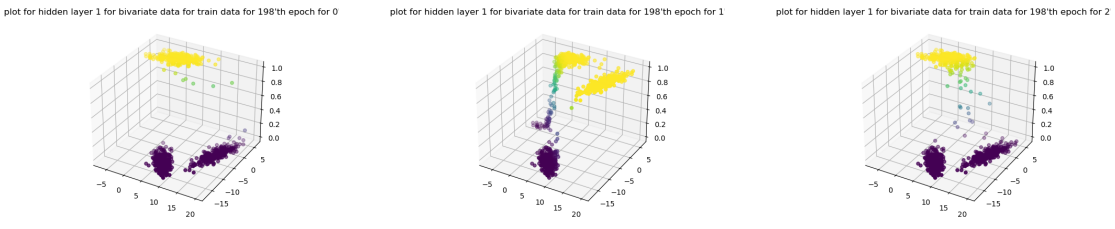



We got the following confusion matrix for train, test and validation data.

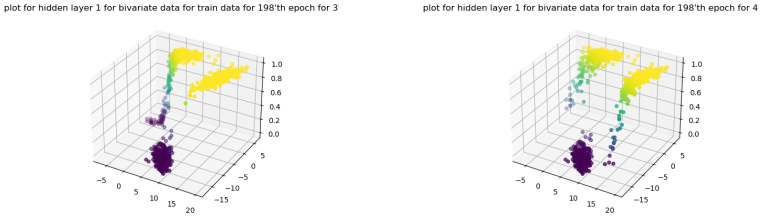


(a) Train Data (b) Test Data (c) Validation Data

PLOTS OF OUTPUT AT DIFFERENT NODES OF HIDDEN LAYER 1 AFTER MODEL IS TRAINED

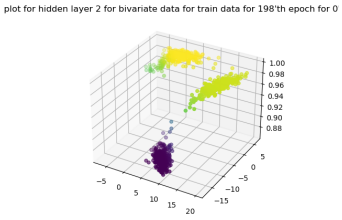



(a) Neuron 0 (b) Neuron 1 (c) Neuron 2

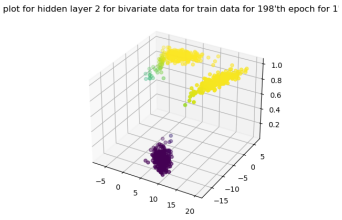



(a) Neuron 3 (b) Neuron 4

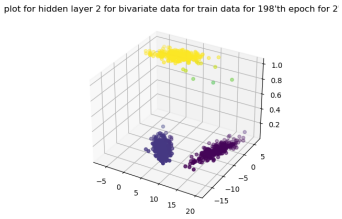
PLOTS OF OUTPUT AT DIFFERENT NODES OF HIDDEN LAYER 2 AFTER MODEL IS TRAINED



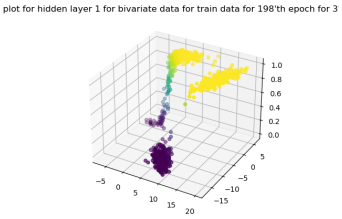

(a) Neuron 0




(b) Neuron 1

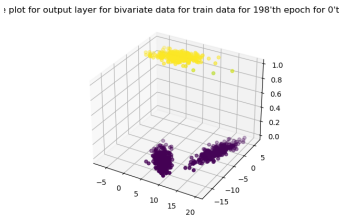



(a) Neuron 2

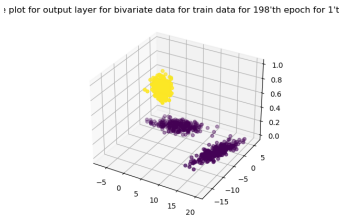



(b) Neuron 3

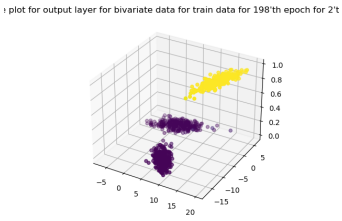
PLOTS OF OUTPUT AT DIFFERENT NODES OF OUTPUT LAYER AFTER MODEL IS TRAINED




(a) Neuron 0

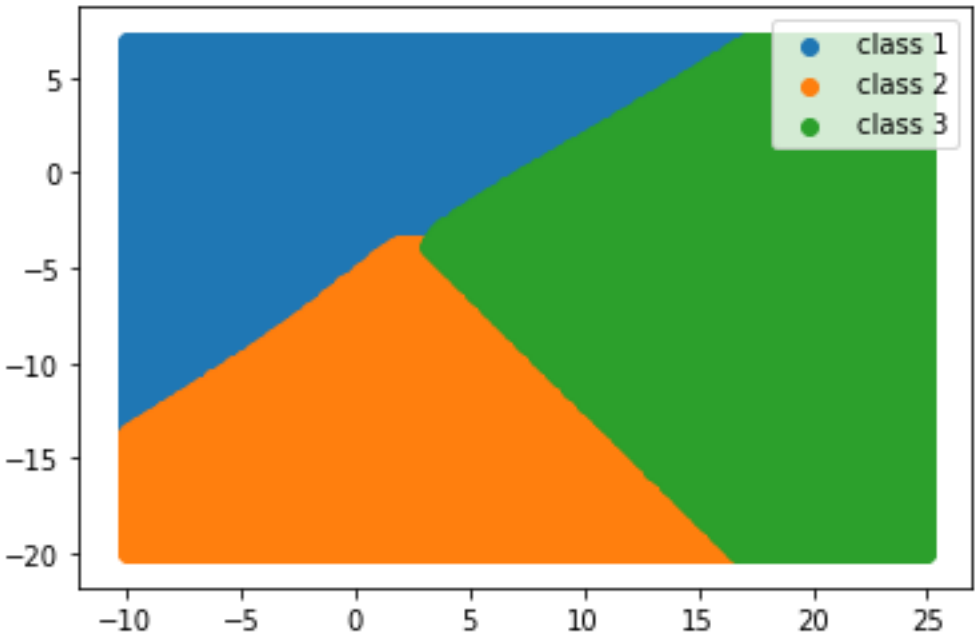



(b) Neuron 1




(c) Neuron 2

Class predicted by model of different points:




(a) TRAIN DATA

As we can see clearly from the above picture, decision region is linear.

### 3.1.4 Observations

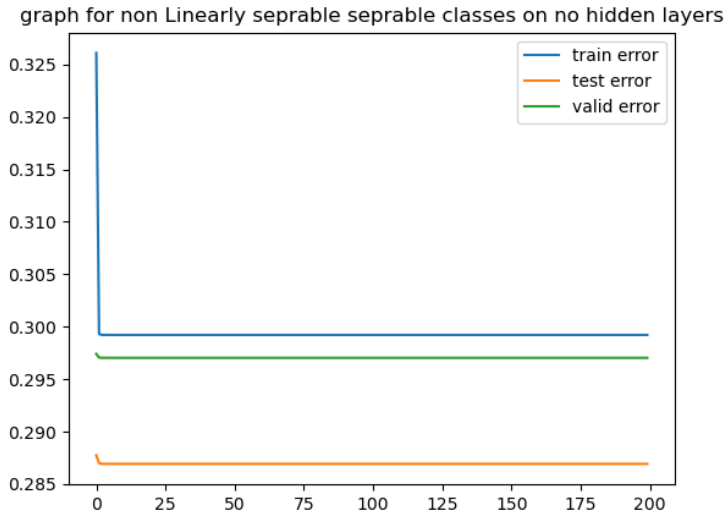
As we can see from the decision region plots and confusion matrix we are able to classify each example very accurately using the no hidden layer model. So, we should use that model as it is very simple so it will take less time to train and also it will require less training examples to train (in comparison to 1 and 2 hidden layer models). We are able to classify this data using a simple model because it is Linearly Separable and a simple no hidden layer model is enough to create a linear boundary between classes.

## 3.2 NON-LINEARLY SEPARABLE

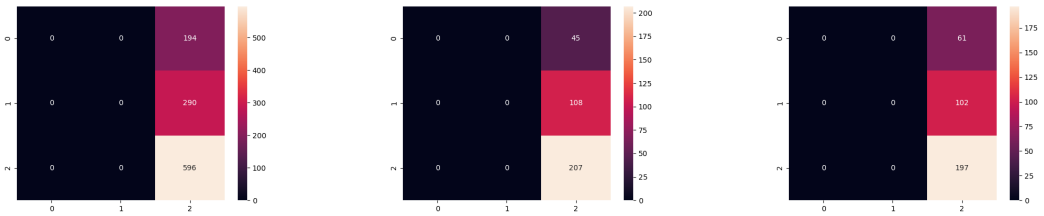
We were given a 2 dimensional artificial data of 3 classes which were non linearly separable. In the data 300 examples were from class 1, 500 examples were from class 2, and 1000 examples were from class 3.We used this data to train different models and predict the class using that model. We divided data into 20% test, 20% validation, and 60% train data.

### 3.2.1 Model with 0 hidden layers

We trained our model for no hidden layer for a total of 200 epochs and got the following RMSE vsEpochs Plot.



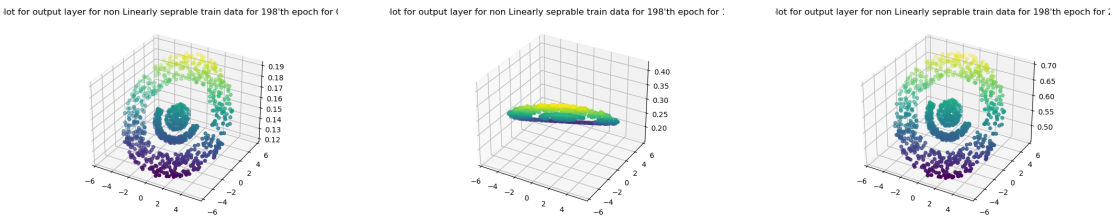

We got the following confusion matrix for train, test and validation data.




(a) Train Data (b) Test Data (c) Validation Data

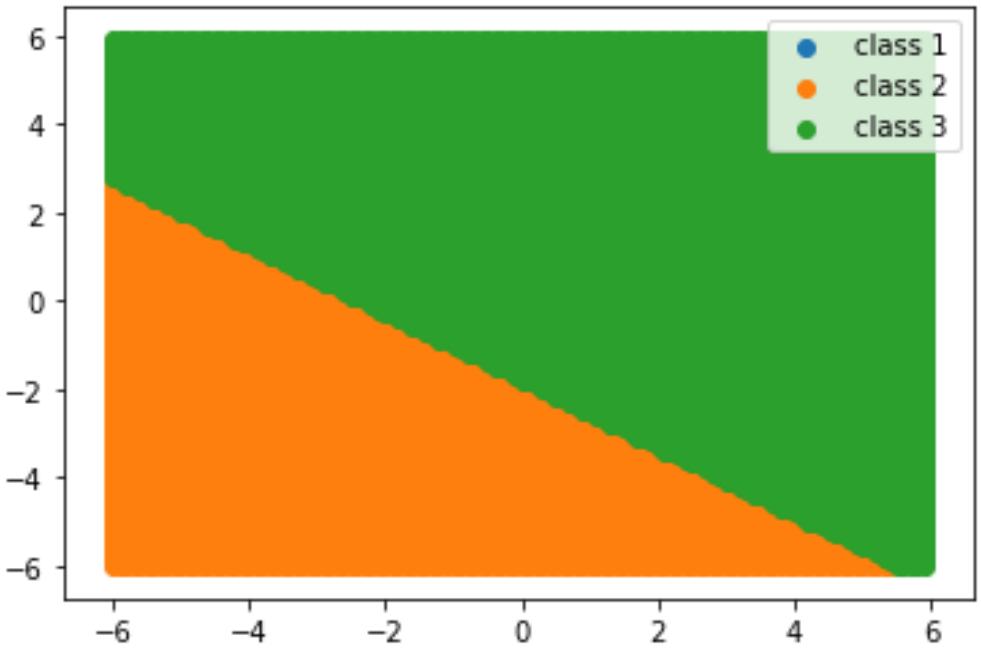
Training accuracy in this model is 0.55 which is very low because this model is not sufficient to classify Non Linearly Separable classes of data.

PLOTS OF OUTPUT AT DIFFERENT NODES OF OUTPUT LAYER AFTER MODEL IS TRAINED



(a) Neuron 0 (b) Neuron 1 (c) Neuron 2

Class predicted by model of different points:

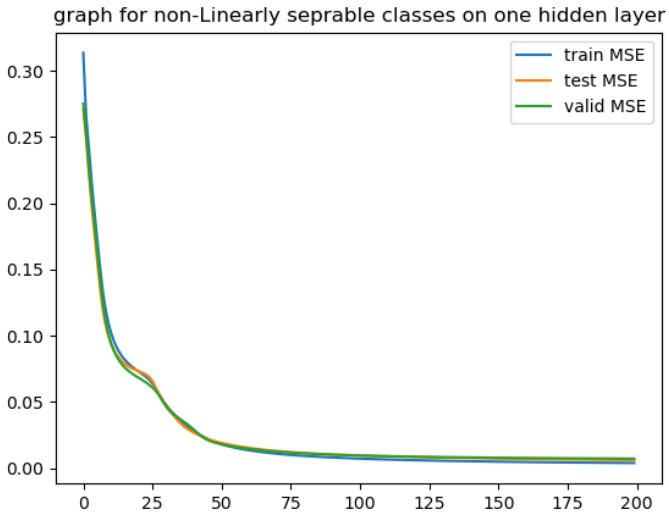



(a) TRAIN DATA

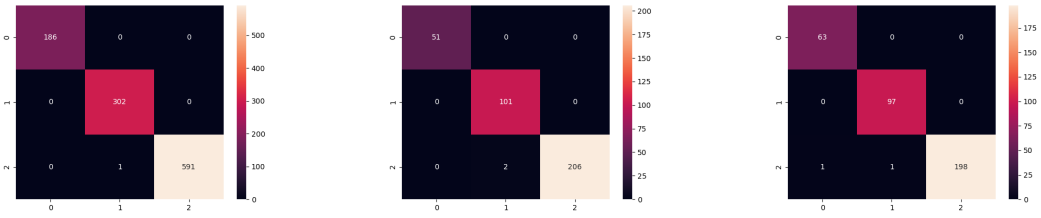
As we can see clearly from the above picture, decision region is linear.
But actual decision region is non-linear, so this model is not accurate to classify these points.

### 3.2.2 Model with 1 hidden layer

We added a hidden layer with 5 neurons to our model and then again trained it for 200 epochs. As a result we got the following RMSE vs Epochs Plot.

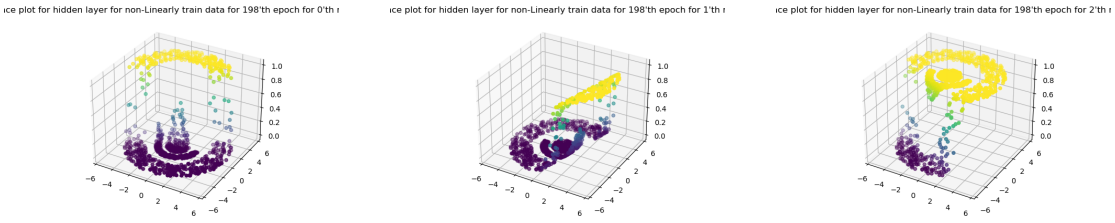



We got the following confusion matrix for train, test and validation data.

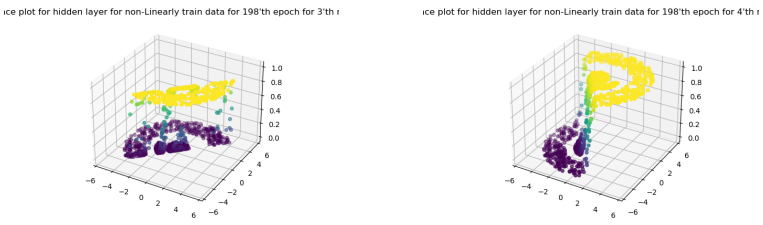



(a) Train Data (b) Test Data (c) Validation Data

PLOTS OF OUTPUT AT DIFFERENT NODES OF HIDDEN LAYER 1 AFTER MODEL IS TRAINED

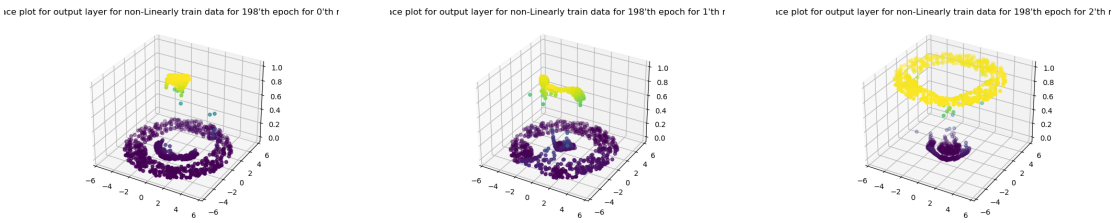



(a) Neuron 0 (b) Neuron 1 (c) Neuron 2



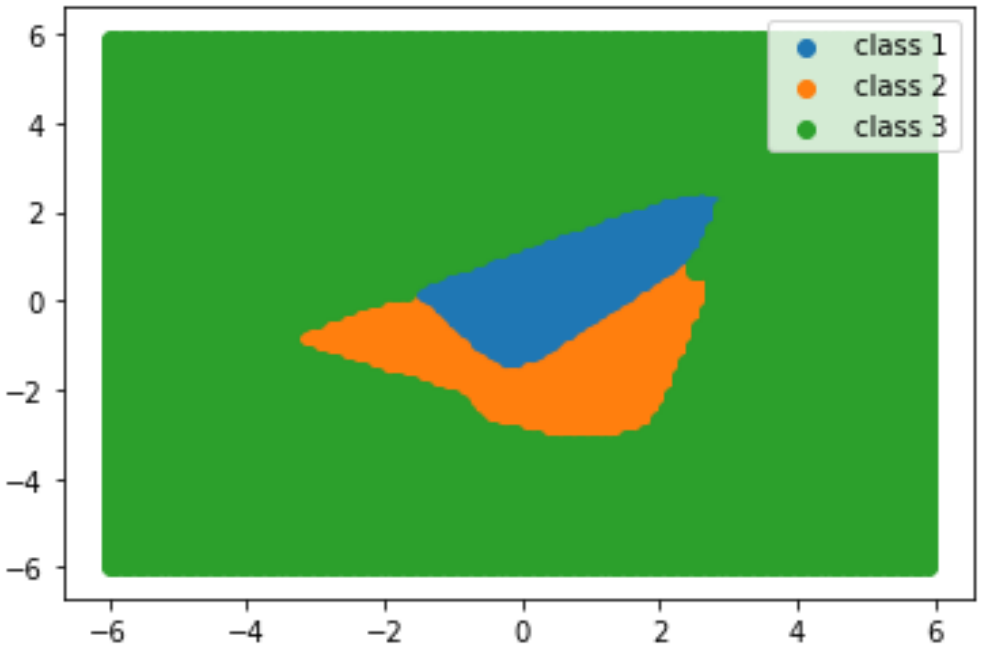

(a) Neuron 3 (b) Neuron 4

PLOTS OF OUTPUT AT DIFFERENT NODES OF OUTPUT LAYER AFTER MODEL IS TRAINED



(a) Neuron 0 (b) Neuron 1 (c) Neuron 2

Class predicted by model of different points:

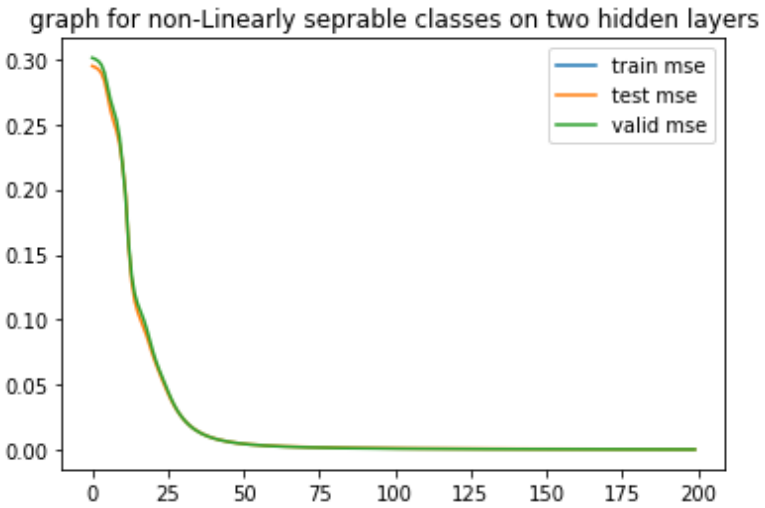



(a) TRAIN DATA

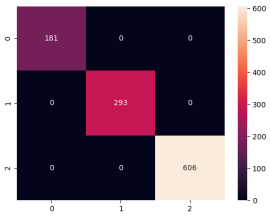
As we can see clearly from the above picture, decision region is non-linear.
And this model seems to classify these points quite accurately.

### 3.2.3 Model with 2 hidden layer

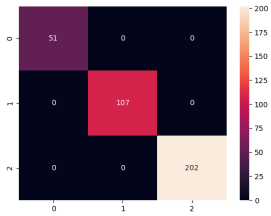
In this model we added 2 hidden layers, with 5 neurons in layer 1 and 4 neurons in layer 2, to the model and trained it for 200 epochs. We got the following RMSE vs Epochs plot.



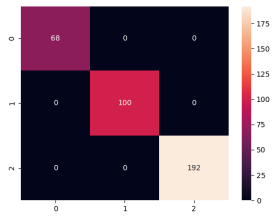
We got the following confusion matrix for train, test and validation data.




(a) Train Data



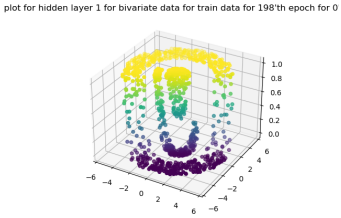

(b) Test Data



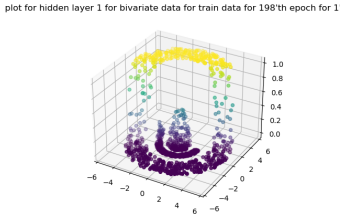

(c) Validation Data

Accuracy in this model is 1 because it is a complex neural network which can classify the non linearly separable classes of data.

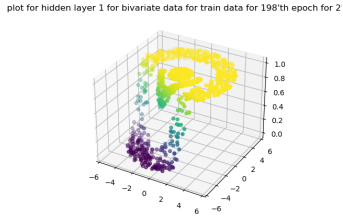
PLOTS OF OUTPUT AT DIFFERENT NODES OF HIDDEN LAYER 1 AFTER MODEL IS TRAINED



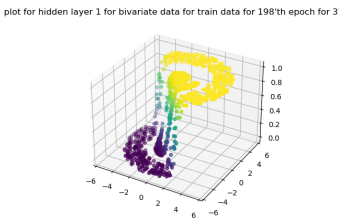

(a) Neuron 0



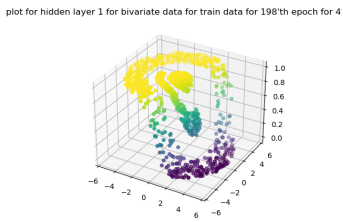

(b) Neuron 1




(c) Neuron 2

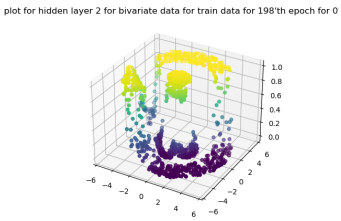



(a) Neuron 3

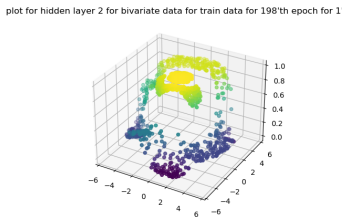



(b) Neuron 4

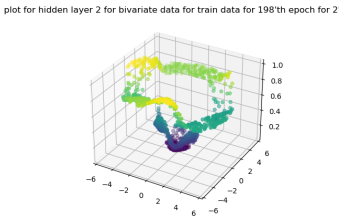
PLOTS OF OUTPUT AT DIFFERENT NODES OF HIDDEN LAYER 2 AFTER MODEL IS TRAINED



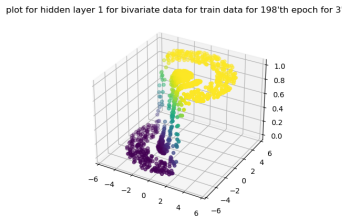

(a) Neuron 0




(b) Neuron 1

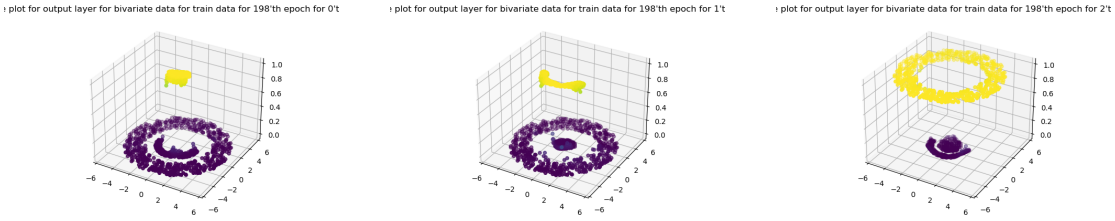



(a) Neuron 2



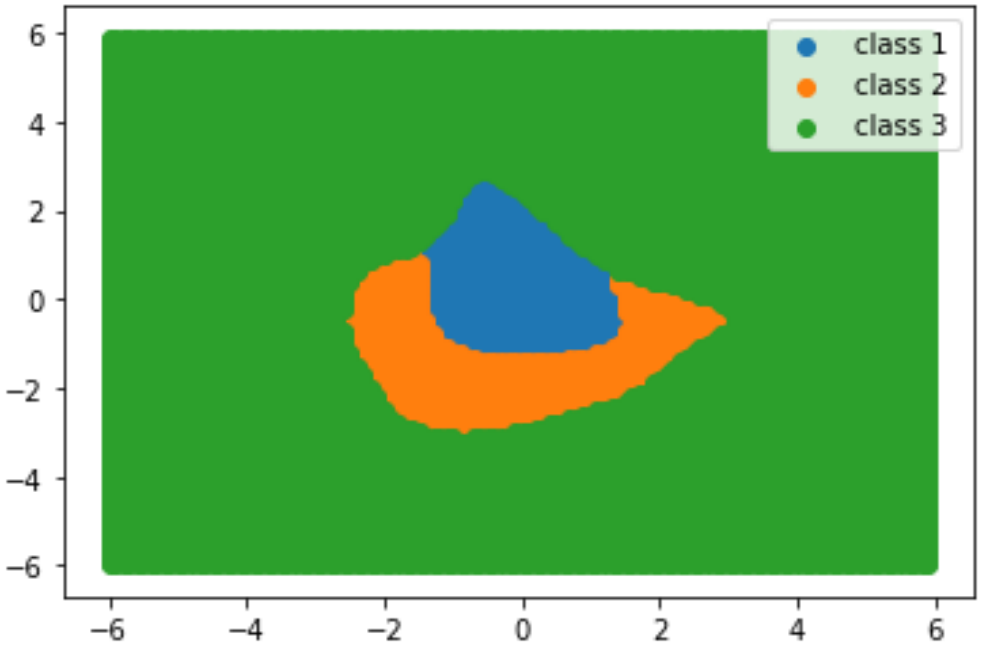

(b) Neuron 3

PLOTS OF OUTPUT AT DIFFERENT NODES OF OUTPUT LAYER AFTER MODEL IS TRAINED




(a) Neuron 0 (b) Neuron 1 (c) Neuron 2

Class predicted by model of different points:




(a) TRAIN DATA

As we can see clearly from the above picture, decision region is non-linear.
And this model seems to classify these points quite accurately.

### 3.2.4 OBSERVATIONS

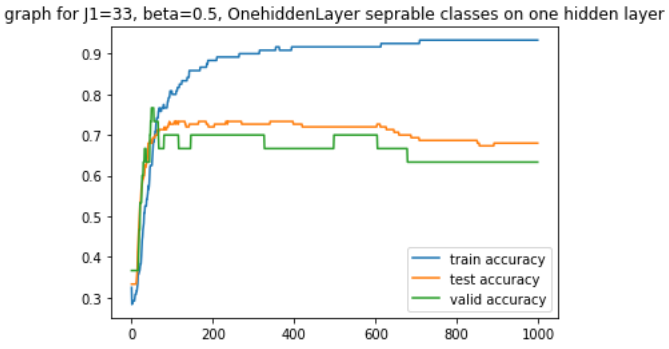
As we can see from the decision region plot and confusion matrix, model with 0 hidden layer cannot be used to classify this data because of very low accuracy. We can use both 1 hidden layer model and 2 hidden layer model for classifying this non linearly separable data, as they both have accuracy 1. Preferred model would be 1 hidden layer model as it has higher accuracy than no hidden layer model and also it is simpler than 2 hidden layer model. 1 hidden layer model will be faster to train and will require less training examples than 2 hidden layer model. We are not able classify this data using a simple no hidden layer model as this data is non linearly separable and it requires a complex model to create a non linear boundary between classes.

## 3.3 IMAGE CLASSIFICATION

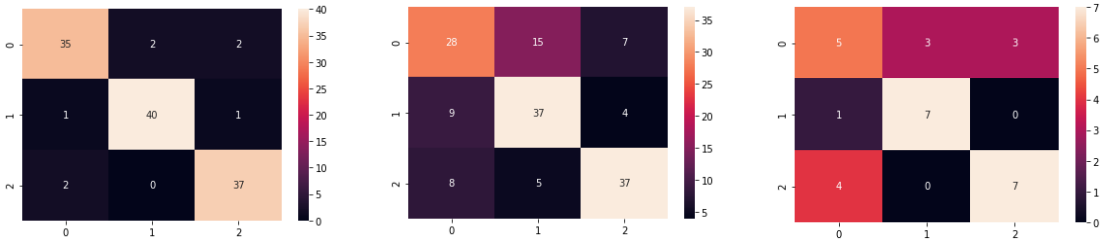
150 Train Images of 3 classes were given, 120 were taken for training data and 30 were taken in validation data. Apart from this 150 test images were given. the bag of visual words representation was obtained and data is saved in 3 csv files. These are present in data/Classification/ImageGroup23 folder attached. This is 32 dimensional data with 3 class labels for each data point.

### 3.3.1 Model with one hidden layer

This architecture had 33 neurons in the hidden layer.
RMSE vs Epoch Plot



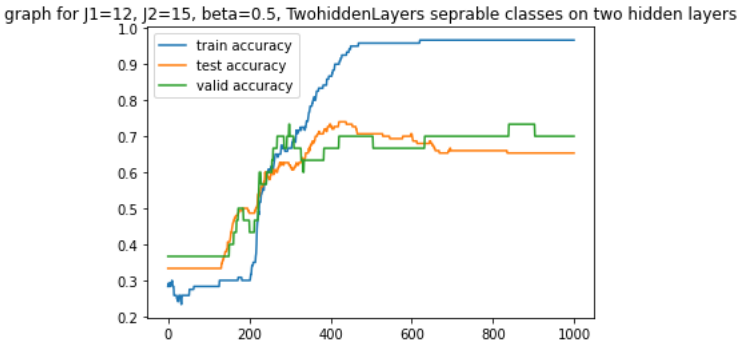

Confusion Matrix for train,test and validation data



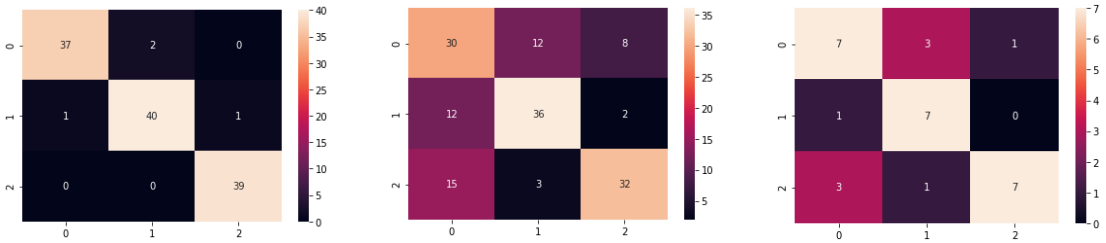
(a) Train Data (b) Test Data (c) Validation Data

### 3.3.2 Model with two hidden layer

This architecture had 18 neurons in first hidden layer and 12 neurons in second hidden layer.
RMSE vs Epoch Plot




Confusion Matrix for train,test and validation data



(a) Train Data (b) Test Data (c) Validation Data

### 3.3.3 OBSERVATINOS

Both models are performing well on training data but giving only 60-70 percent accuracy for test data. The reason to this could be because we have only 120 images in training set and hence if we

use complex models, the models will overfit on the data and memorise the data. This will lead to bad performance on unseen data.