CS5691 Pattern Recognition and Machine Learning

Jan – May 2021

Assignment – 3

Report

Group 18:

BS17B033 – Shreya Nema

ME17B065 – Sahil Ansari

ME17B158 – Omkar Nath

Table of Contents

1 Dataset 1(a) – Linearly Separable 2D Data

1.1 Problem Statement

We are given a 2-dimensional artificial data, which is linearly separable for static pattern classification. For this, the dataset specifically allotted to Group 18 is used.

Based upon this, the following classifiers need to be built.

1. Perceptron for every pair of classes
2. Multilayer feedforward neural network (MLFFNN) with a single hidden layer for all classes
3. Linear SVM classifier for every pair of classes

1.2 Classification Accuracies Obtained

Upon implementing the various classifiers, the results obtained are as follows:

1.2.1 Perceptron for every pair of Classes

The perceptron classification accuracies for the various pairs of classes are as given in the tables below.

1.2.1.1 Class 0 vs. Class 1

Table 1: Perceptron Classification Accuracies - Class 0 vs. Class 1

Alpha	Train Accuracy	Validation Accuracy
0.01	100%	100%
0.001	100%	100%
0.0001	100%	100%

1.2.1.2 Class 0 vs. Class 2

Table 2: Perceptron Classification Accuracies - Class 0 vs. Class 2

Alpha	Train Accuracy	Validation Accuracy
0.01	100%	100%
0.001	100%	100%
0.0001	100%	100%

1.2.1.3 Class 0 vs. Class 3

Table 3: Perceptron Classification Accuracies - Class 0 vs. Class 3

Alpha	Train Accuracy	Validation Accuracy
0.01	100%	100%
0.001	100%	100%
0.0001	100%	100%

1.2.1.4 Class 1 vs. Class 2

Table 4: Perceptron Classification Accuracies - Class 1 vs. Class 2

Alpha	Train Accuracy	Validation Accuracy
0.01	100%	100%
0.001	100%	100%
0.0001	100%	100%

1.2.1.5 Class 1 vs. Class 3

Table 5: Perceptron Classification Accuracies - Class 1 vs. Class 3

Alpha	Train Accuracy	Validation Accuracy
0.01	100%	100%
0.001	100%	100%
0.0001	100%	100%

1.2.1.6 Class 2 vs. Class 3

Table 6: Perceptron Classification Accuracies - Class 2 vs. Class 3

Alpha	Train Accuracy	Validation Accuracy
0.01	100%	100%
0.001	100%	100%
0.0001	100%	100%

1.2.2 Multilayer Feedforward Neural Network for all classes

Classification accuracies for a MLFFNN model with one hidden layer are given below.

Table 7: MLFFNN with one hidden layer Classification Accuracies

Neurons in Hidden Layer	Train Accuracy	Validation Accuracy
10	100%	100%
15	100%	100%
20	100%	100%

1.2.3 Linear SVM Classifier for every pair of classes

The linear SVM classification accuracies for the various pairs of classes are as given in the tables below.

1.2.3.1 Class 0 vs. Class 1

Table 8: Linear SVM Classification Accuracies - Class 0 vs. Class 1

C	Train Accuracy	Validation Accuracy
0.1	100%	100%
0.5	100%	100%
1.0	100%	100%

1.2.3.2 Class 0 vs. Class 2

Table 9: Linear SVM Classification Accuracies - Class 0 vs. Class 2

C	Train Accuracy	Validation Accuracy
0.1	100%	100%
0.5	100%	100%
1.0	100%	100%

1.2.3.3 Class 0 vs. Class 3

Table 10: Linear SVM Classification Accuracies - Class 0 vs. Class 3

C	Train Accuracy	Validation Accuracy
0.1	100%	100%
0.5	100%	100%
1.0	100%	100%

1.2.3.4 Class 1 vs. Class 2

Table 11: Linear SVM Classification Accuracies - Class 1 vs. Class 2

C	Train Accuracy	Validation Accuracy
0.1	100%	100%
0.5	100%	100%
1.0	100%	100%

1.2.3.5 Class 1 vs. Class 3

Table 12: Linear SVM Classification Accuracies - Class 1 vs. Class 3

C	Train Accuracy	Validation Accuracy
0.1	100%	100%
0.5	100%	100%
1.0	100%	100%

1.2.3.6 Class 2 vs. Class 3

Table 13: Linear SVM Classification Accuracies - Class 2 vs. Class 3

C	Train Accuracy	Validation Accuracy
0.1	100%	100%
0.5	100%	100%
1.0	100%	100%

1.3 Best Model, and Test accuracy

The best model amongst the models available, as well as its performance on the test dataset is as given below.

1.3.1 Perceptron for every pair of Classes

The best model and test accuracy for the various pairs of classes are given below.

1.3.1.1 Class 0 vs. Class 1

The best model for this case is alpha = **0.0001**

This gives a test accuracy of **100%**

1.3.1.2 Class 0 vs. Class 2

The best model for this case is alpha = **0.0001**

This gives a test accuracy of **100%**

1.3.1.3 Class 0 vs. Class 3

The best model for this case is alpha = **0.0001**

This gives a test accuracy of **100%**

1.3.1.4 Class 1 vs. Class 2

The best model for this case is alpha = **0.0001**

This gives a test accuracy of **100%**

1.3.1.5 Class 1 vs. Class 3

The best model for this case is alpha = **0.0001**

This gives a test accuracy of **100%**

1.3.1.6 Class 2 vs. Class 3

The best model for this case is alpha = **0.0001**

This gives a test accuracy of **100%**

1.3.2 Multilayer Feedforward Neural Network for all classes

For a MLFFNN model with one hidden layer,

The best model is one with **10 neuron**s in the hidden layer.

This gives a test accuracy of **100%.**

1.3.3 Linear SVM Classifier for every pair of classes

The best model and test accuracy for the various pairs of classes are given below.

1.3.3.1 Class 0 vs. Class 1

The best model for this case is **C = 1.0**

This gives a test accuracy of **100%**

1.3.3.2 Class 0 vs. Class 2

The best model for this case is **C = 1.0**

This gives a test accuracy of **100%**

1.3.3.3 Class 0 vs. Class 3

The best model for this case is **C = 1.0**

This gives a test accuracy of **100%**

1.3.3.4 Class 1 vs. Class 2

The best model for this case is **C = 1.0**

This gives a test accuracy of **100%**

1.3.3.5 Class 1 vs. Class 3

The best model for this case is **C = 1.0**

This gives a test accuracy of **100%**

1.3.3.6 Class 2 vs. Class 3

The best model for this case is **C = 1.0**

This gives a test accuracy of **100%**

1.4 Confusion Matrix for the best model

The confusion matrix that is obtained for the best model, as determined by the performance on the validation dataset, is given below for each of the individual cases.

1.4.1 Perceptron for every pair of Classes

The confusion matrices for the best model for each of the individual cases are given below.

1.4.1.1 Class 0 vs. Class 1

Confusion Matrices for the best model i.e., alpha = 0.0001.

Table 14: Confusion Matrix for Training Data using Perceptron Classifier - Class 0 vs. Class 1

Class	0	1
0	200	0
1	0	200

Table 15: Confusion Matrix for Validation Data using Perceptron Classifier - Class 0 vs. Class 1

Class	0	1
0	29	0
1	0	30

1.4.1.2 Class 0 vs. Class 2

Confusion Matrices for the best model i.e., alpha = 0.0001.

Table 16: Confusion Matrix for Training Data using Perceptron Classifier - Class 0 vs. Class 2

Class	0	2
0	200	0
2	0	199

Table 17: Confusion Matrix for Validation Data using Perceptron Classifier - Class 0 vs. Class 2

Class	0	2
0	29	0
2	0	30

1.4.1.3 Class 0 vs. Class 3

Confusion Matrices for the best model i.e., alpha = 0.0001

Table 18: Confusion Matrix for Training Data using Perceptron Classifier - Class 0 vs. Class 3

Class	0	3
0	200	0
3	0	200

Table 19: Confusion Matrix for Validation Data using Perceptron Classifier - Class 0 vs. Class 3

Class	0	3
0	29	0
3	0	30

1.4.1.4 Class 1 vs. Class 2

Confusion Matrices for the best model i.e., alpha = 0.0001

Table 20: Confusion Matrix for Training Data using Perceptron Classifier - Class 1 vs. Class 2

Class	1	2
1	200	0
2	0	199

Table 21: Confusion Matrix for Validation Data using Perceptron Classifier - Class 1 vs. Class 2

Class	1	2
1	30	0
2	0	30

1.4.1.5 Class 1 vs. Class 3

Confusion Matrices for the best model i.e., alpha = 0.0001

Table 22: Confusion Matrix for Training Data using Perceptron Classifier - Class 1 vs. Class 3

Class	1	3
1	200	0
3	0	200

Table 23: Confusion Matrix for Validation Data using Perceptron Classifier - Class 1 vs. Class 3

Class	1	3
1	30	0
3	0	30

1.4.1.6 Class 2 vs. Class 3

Confusion Matrices for the best model i.e., alpha = 0.0001

Table 24: Confusion Matrix for Training Data using Perceptron Classifier - Class 2 vs. Class 3

Class	2	3
2	199	0
3	0	200

Table 25: Confusion Matrix for Validation Data using Perceptron Classifier - Class 2 vs. Class 3

Class	2	3
2	30	0
3	0	30

1.4.2 Multilayer Feedforward Neural Network for all classes

The confusion matrices for the best model of the MLFFNN Classifier with 1 hidden layer i.e., with 10 hidden neurons are given below.

Table 26: Confusion Matrix for Training Data using MLFFNN Classifier

Train Data				
Class	0	1	2	3
0	200	0	0	0
1	0	200	0	0
2	0	0	199	0
3	0	0	0	200

Table 27: Confusion Matrix for Validation Data using MLFFNN Classifier

Validation Data				
Class	0	1	2	3
0	29	0	0	0
1	0	30	0	0
2	0	0	30	0
3	0	0	0	30

1.4.3 Linear SVM Classifier for every pair of classes

The confusion matrices for the best model for each of the individual cases are given below.

1.4.1.1 Class 0 vs. Class 1

Confusion Matrices for the best model i.e., C=1.0

Table 28: Confusion Matrix for Training Data using Linear SVM Classifier - Class 0 vs. Class 1

Class	0	1
0	200	0
1	0	200

Table 29: Confusion Matrix for Validation Data using Linear SVM Classifier - Class 0 vs. Class 1

Class	0	1
0	29	0
1	0	30

1.4.1.2 Class 0 vs. Class 2

Confusion Matrices for the best model i.e., C=1.0

Table 30: Confusion Matrix for Training Data using Linear SVM Classifier - Class 0 vs. Class 2

Class	0	2
0	200	0
2	0	199

Table 31: Confusion Matrix for Validation Data using Linear SVM Classifier - Class 0 vs. Class 2

Class	0	2
0	29	0
2	0	30

1.4.1.3 Class 0 vs. Class 3

Confusion Matrices for the best model i.e., C=1.0

Table 32: Confusion Matrix for Training Data using Linear SVM Classifier - Class 0 vs. Class 3

Class	0	3
0	200	0
3	0	200

Table 33: Confusion Matrix for Validation Data using Linear SVM Classifier - Class 0 vs. Class 3

Class	0	3
0	29	0
3	0	30

1.4.1.4 Class 1 vs. Class 2

Confusion Matrices for the best model i.e., C=1.0

Table 34: Confusion Matrix for Training Data using Linear SVM Classifier - Class 1 vs. Class 2

Class	1	2
1	200	0
2	0	199

Table 35: Confusion Matrix for Validation Data using Linear SVM Classifier - Class 1 vs. Class 2

Class	1	2
1	30	0
2	0	30

1.4.1.5 Class 1 vs. Class 3

Confusion Matrices for the best model i.e., C=1.0

Table 36: Confusion Matrix for Training Data using Linear SVM Classifier - Class 1 vs. Class 3

Class	1	3
1	200	0
3	0	200

Table 37: Confusion Matrix for Validation Data using Linear SVM Classifier - Class 1 vs. Class 3

Class	1	3
1	30	0
3	0	30

1.4.1.6 Class 2 vs. Class 3

Confusion Matrices for the best model i.e., C=1.0

Table 38: Confusion Matrix for Training Data using Linear SVM Classifier - Class 2 vs. Class 3

Class	2	3
2	199	0
3	0	200

Table 39: Confusion Matrix for Validation Data using Linear SVM Classifier - Class 2 vs. Class 3

Class	2	3
2	30	0
3	0	30

1.5 Decision Region Plots

Various plots to visualize the data and the classifiers are displayed here.

1.5.1 Training Data

The plot of the training data is as shown below.

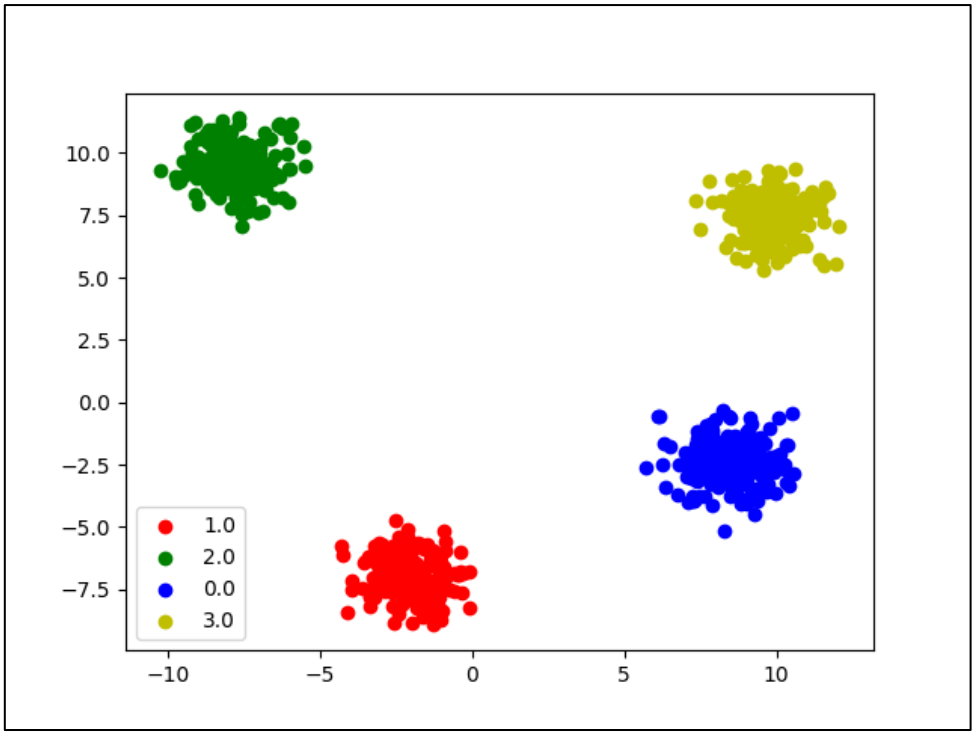

Figure 1: Training Data for Dataset 1A

1.5.2 Perceptron for every pair of Classes

The plot for the pairwise classification boundaries, for the Perceptron classifier are given below.

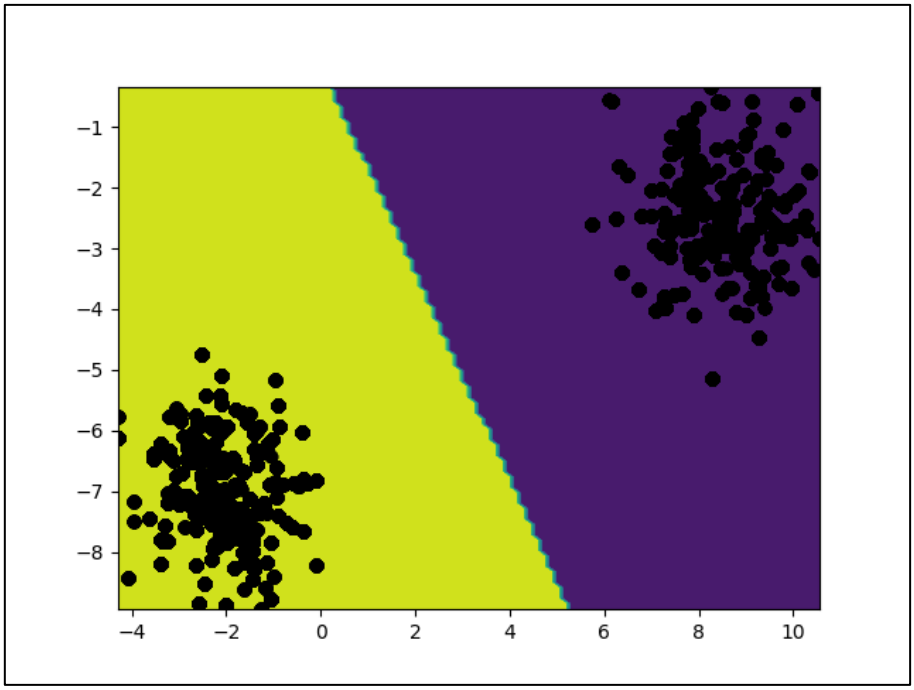

Figure 2: Perceptron Classifier - Class 0 vs. Class 1

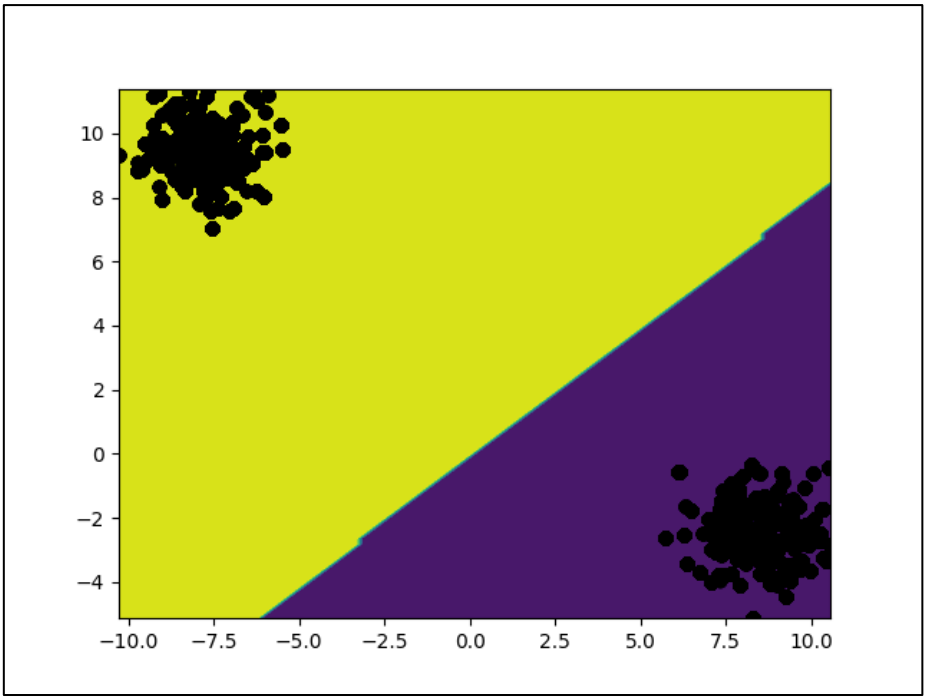

Figure 3: Perceptron Classifier - Class 0 vs. Class 2

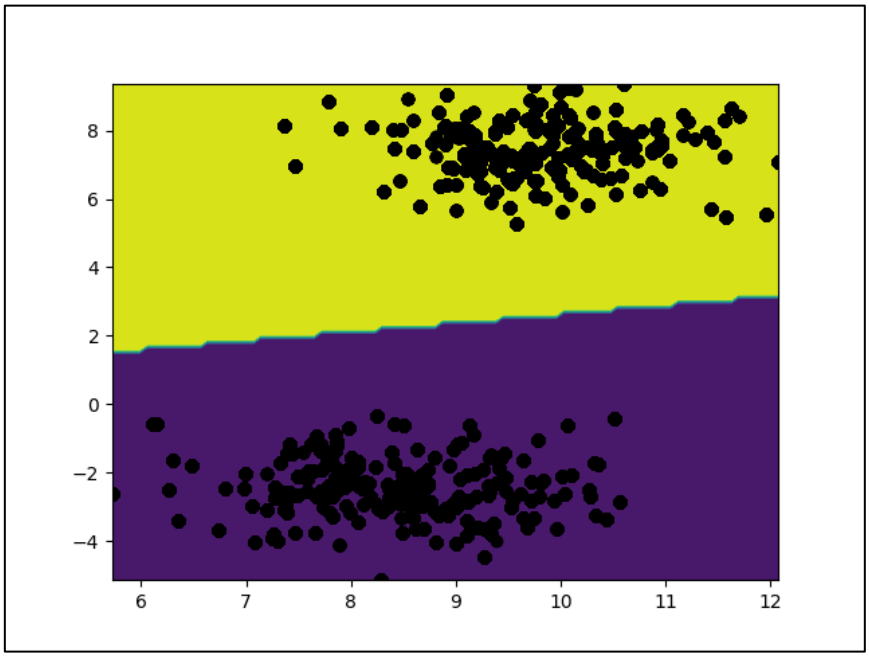

Figure 4: Perceptron Classifier - Class 0 vs. Class 3

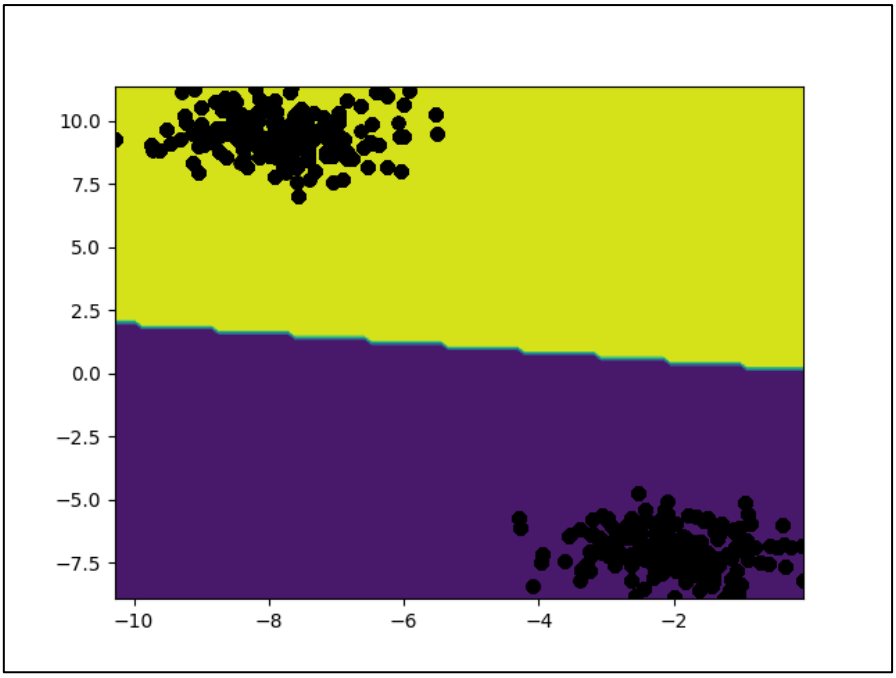

Figure 5: Perceptron Classifier - Class 1 vs. Class 2

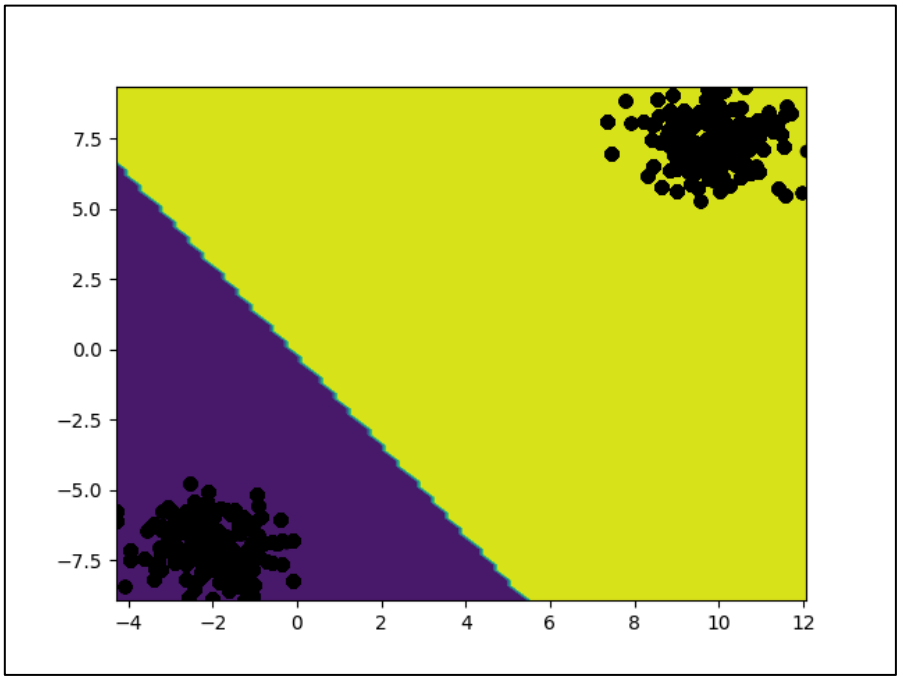

Figure 6: Perceptron Classifier - Class 1 vs. Class 3

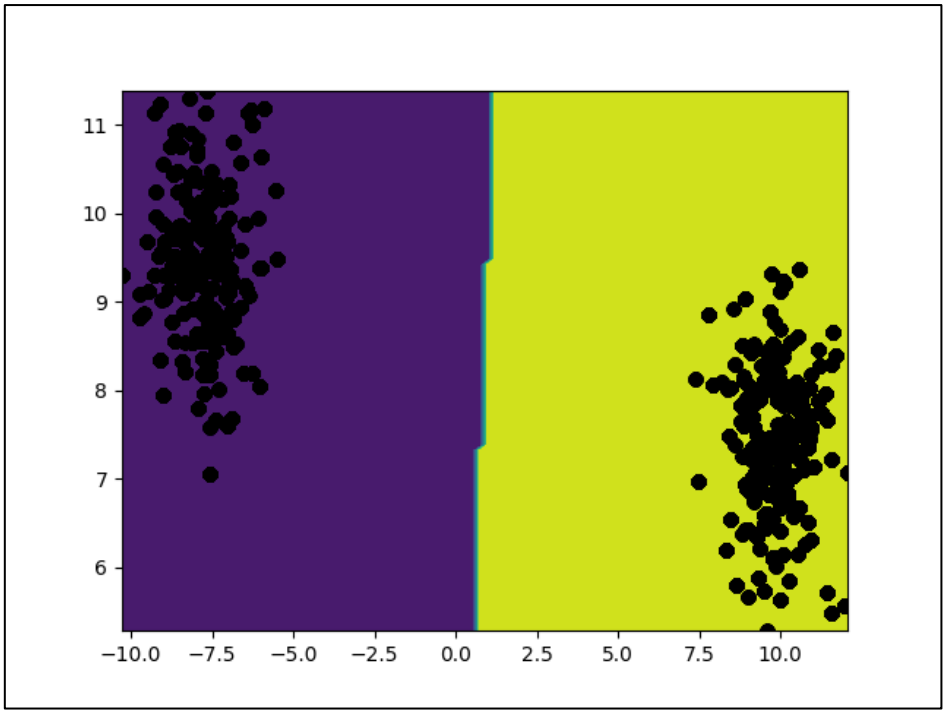

Figure 7: Perceptron Classifier - Class 2 vs. Class 3

1.5.3 Multilayer Feedforward Neural Network for all classes

The plot for the classification boundaries, for the Multilayer Feedforward Neural Network (MLFFNN) with one hidden layer is as shown in the plot below.

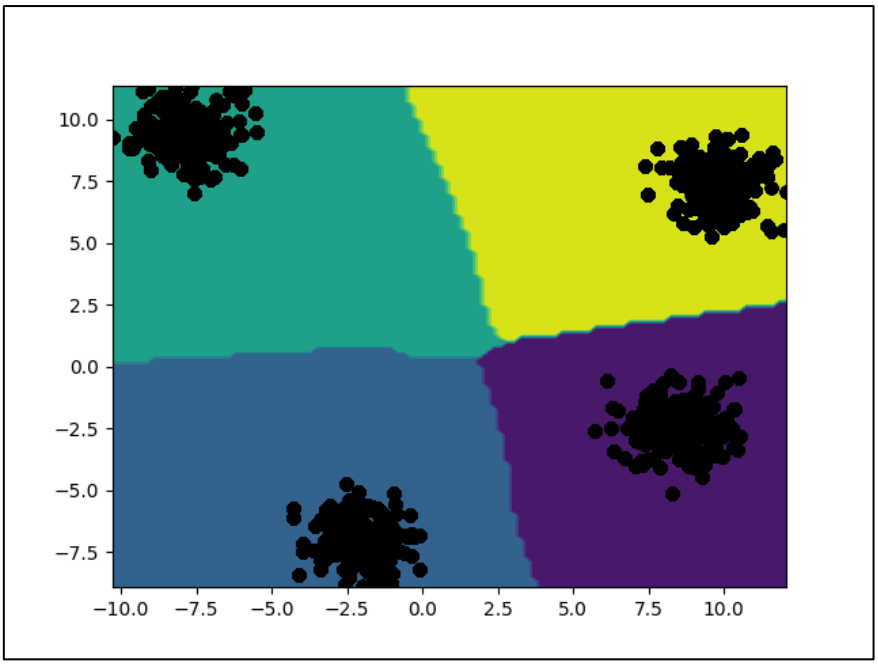

Figure 8: MLFFNN Decision Boundary

1.4.3 Linear SVM Classifier for every pair of classes

The plot for the pairwise classification boundaries, for the Linear SVM classifier are given below.

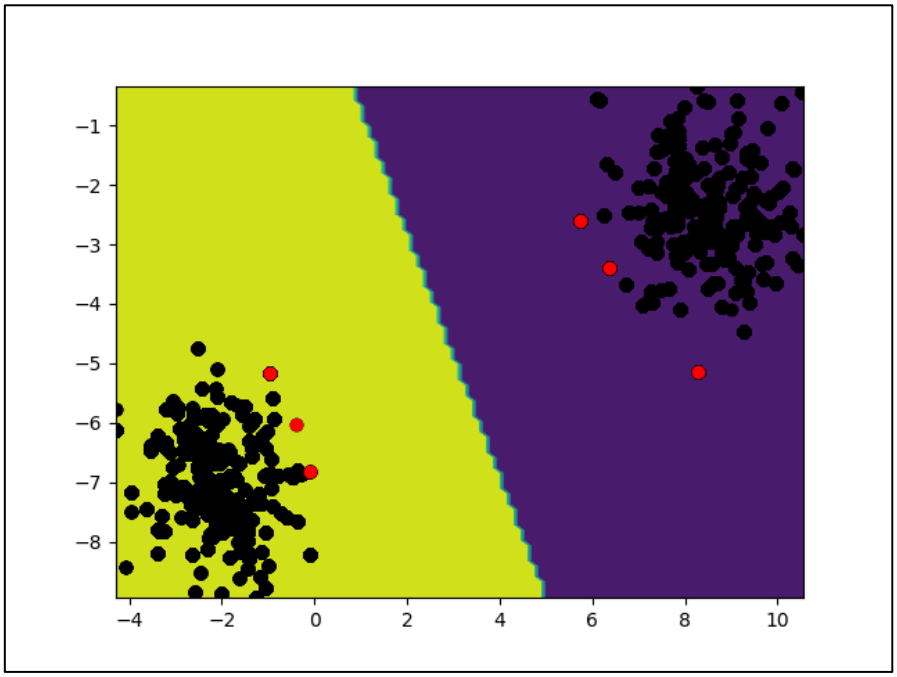

Figure 9: Linear SVM Classifier - Class 0 vs. Class 1

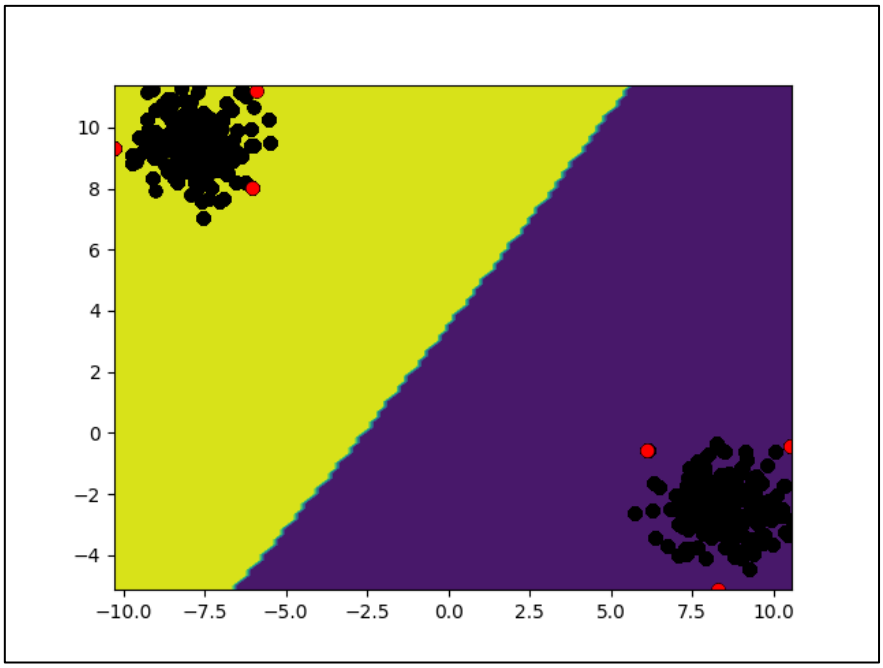

Figure 10: Linear SVM Classifier - Class 0 vs. Class 2

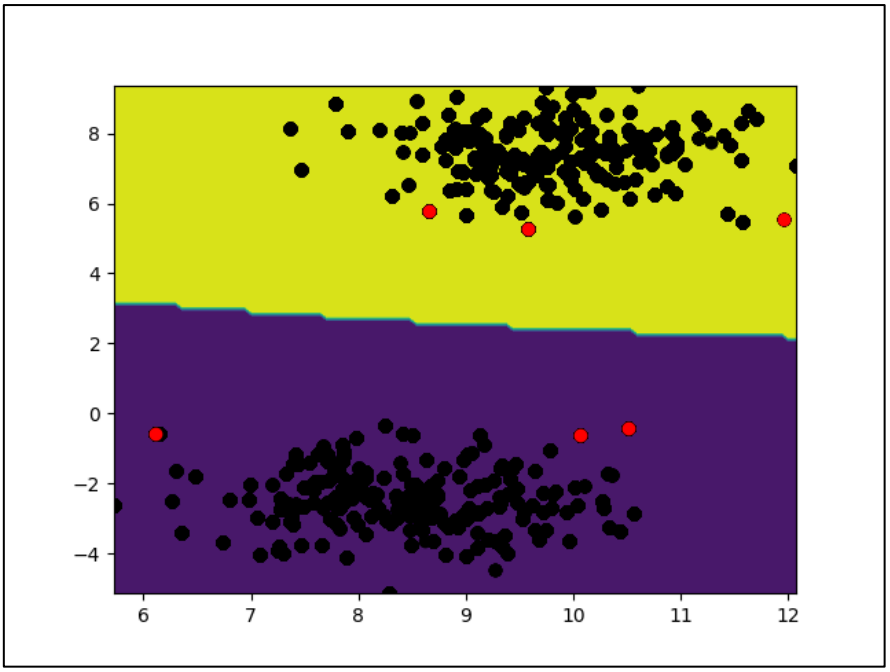

Figure 11: Linear SVM Classifier - Class 0 vs. Class 3

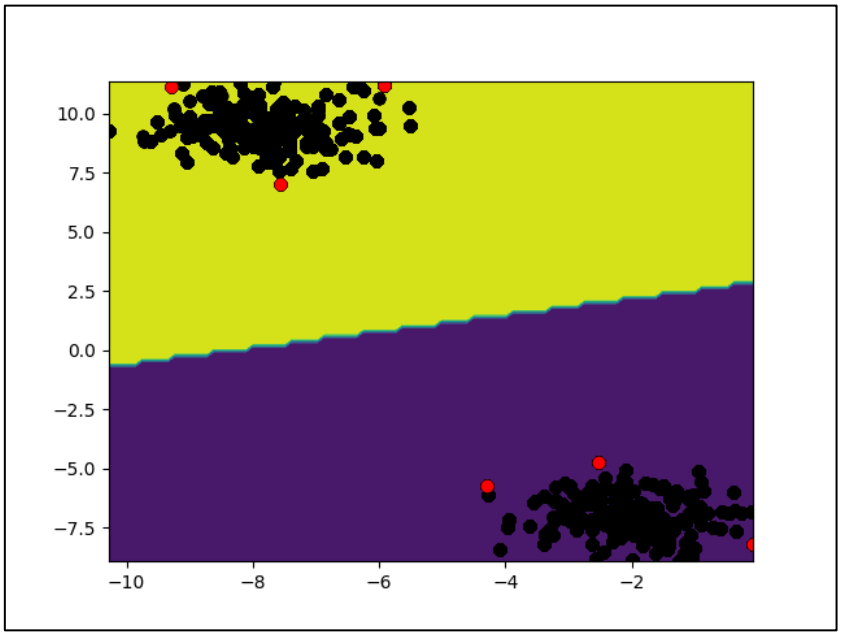

Figure 12: Linear SVM Classifier - Class 1 vs. Class 2

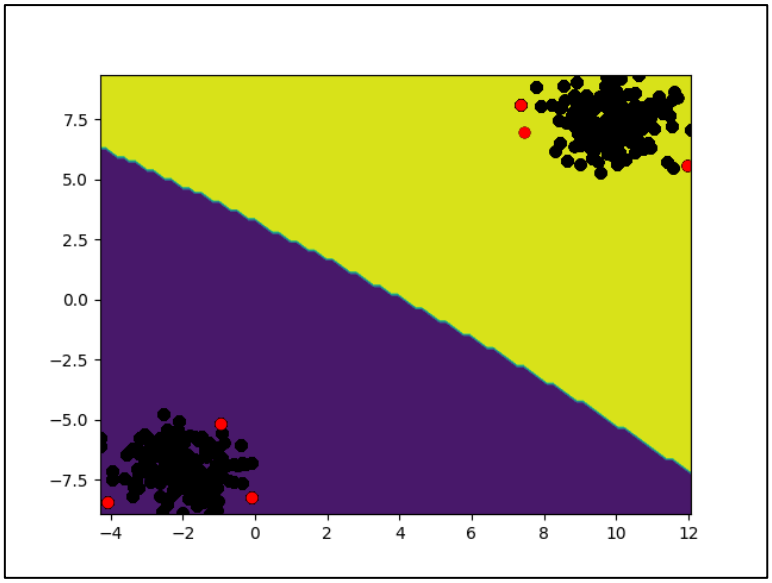

Figure 13: Linear SVM Classifier - Class 1 vs. Class 3

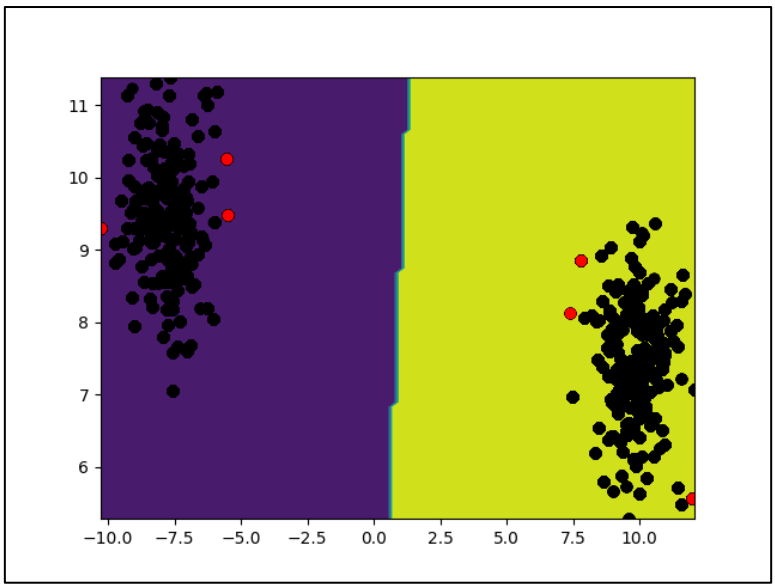

Figure 14: Linear SVM Classifier - Class 2 vs. Class 3

1.6 Observations

Some key observations after looking at the results are as follows:

- The data is already well separated. It can be seen from the pair-wise class separation graphs obtained for perceptron and SVM, that the classes are well separated. Also, for MLFFNN the four classes are perfectly separated and occupy the four corners of the graph.
- As perceptron can easily separate linearly separable data, we obtained 100% accuracy for perceptron-based classifier.
- As MLFFNN and SVM can separate both linearly and non-linearly separable data, All the three classifiers are able to perfectly model the data distribution as can be seen in the contour plots resulting in an accuracy of 100%.

2 Dataset 1(b) – Nonlinearly Separable 2D Data

2.1 Problem Statement

We are given a 2-dimensional artificial data, which is nonlinearly separable for static pattern classification. For this, the dataset specifically allotted to Group 18 is used.

Based upon this, the following classifiers need to be built.

1. MLFFNN with two hidden layers
2. Nonlinear SVM using one-against-the-rest approach:
    a. Polynomial kernel
    b. Gaussian kernel

2.2 Classification Accuracies Obtained

Upon implementing the various classifiers, the accuracies obtained are as follows.

2.2.1 Multilayer Feedforward Neural Network

Training and validation accuracies for MLFFNN Classifier with 2 hidden layers are as below.

Table 40: Training and Validation Classification Accuracies for MLFFNN Classifier

Neurons in Hidden Layer 1	Neurons in Hidden Layer 2	Activation	Train Accuracy	Validation Accuracy
20	20	relu	98.16%	95.50%
20	20	logistic	54.42%	61.80%
20	20	tanh	93.32%	91.01%
30	30	relu	98.83%	97.75%
30	30	logistic	54.09%	62.92%
30	30	tanh	97.67%	93.25%
40	40	relu	99.33%	98.88%
40	40	logistic	54.92%	61.80%
40	40	tanh	98.50%	96.63%
70	70	relu	99.67%	98.88%
70	70	logistic	55.59%	66.29%
70	70	tanh	98.83%	97.75%

2.2.2 Non-linear SVM using Polynomial Kernel

Training and validation accuracies for Non-linear SVM using Polynomial Kernel are as below.

Table 41: Training and Validation Classification Accuracies for non-linear SVM Classifier using Polynomial Kernel

Degree	Training Accuracy	Validation Accuracy
2	50.08%	52.81%
3	72.95%	74.16%
4	57.43%	56.18%
5	70.45%	73.03%
6	57.10%	57.30%

2.2.3 Non-linear SVM using Gaussian Kernel

Training and validation accuracies for Non-linear SVM using Gaussian Kernel are as below.

Table 42: Training and Validation Classification Accuracies for non-linear SVM Classifier using Gaussian Kernel

C	Training Accuracy	Validation Accuracy
0.01	65.27%	67.41%
0.1	95.49%	95.51%
1.0	98.16%	97.75%
10.0	99.16%	98.88%

2.3 Best Model, and Test accuracy

The best model amongst the models available, as well as its performance on the test dataset is as given below.

2.3.1 Multilayer Feedforward Neural Network

The best model for this case, has the parameters as:

Neurons in hidden layer 1 = **40**

Neurons in hidden layer 2 = **40**

Activation Function = **relu**

This gives a test accuracy of **98.88%**

2.3.2 Non-linear SVM using Polynomial Kernel

The best model for this case is with **degree = 3**

This gives a test accuracy of **74.16%**

2.3.3 Non-linear SVM using Gaussian Kernel

The best model for this case is with **C = 10.0**

This gives a test accuracy of **98.88%**

2.4 Confusion Matrix for the best model

The confusion matrix that is obtained for the best model, as determined by the performance on the validation dataset, is given below for each of the individual cases.

2.4.1 Multilayer Feedforward Neural Network

The confusion matrices for the best model of the MLFFNN Classifier with 2 hidden layers i.e., with (40,40) hidden neurons and relu activation functions are given below.

Table 43: Confusion Matrix for Training Data using MLFFNN Classifier with 2 hidden layers

Train Data			
Class	0	1	2
0	198	0	1
1	0	197	3
2	0	0	200

Table 44: Confusion Matrix for Validation Data using MLFFNN Classifier with 2 hidden layers

Validation Data			
Class	0	1	2
0	29	0	0
1	1	29	0
2	0	0	30

2.4.2 Non-linear SVM using Polynomial Kernel

The confusion matrices for the best model i.e., with degree = 3 are given below.

Table 45: Confusion Matrix for Training Data for Non-linear SVM using Polynomial Kernel

Train Data			
Class	0	1	2
0	196	0	3
1	82	118	0
2	75	2	123

Table 46: Confusion Matrix for Validation Data for Non-linear SVM using Polynomial Kernel

Validation Data			
Class	0	1	2
0	27	0	2
1	12	18	0
2	9	0	21

2.4.3 Non-linear SVM using Gaussian Kernel

The confusion matrices for the best model i.e., C = 10.0 are given below.

Table 47: Confusion Matrix for Training Data for Non-linear SVM using Gaussian Kernel

Train Data			
Class	0	1	2
0	198	0	1
1	3	197	0
2	1	0	199

Table 48: Confusion Matrix for Validation Data for Non-linear SVM using Gaussian Kernel

Validation Data			
Class	0	1	2
0	29	0	0
1	1	29	0
2	0	0	30

2.5 Decision Region Plots

Various plots to visualize the data and the classifiers are displayed here.

2.5.1 The Dataset

The plot of the training data is as shown below.

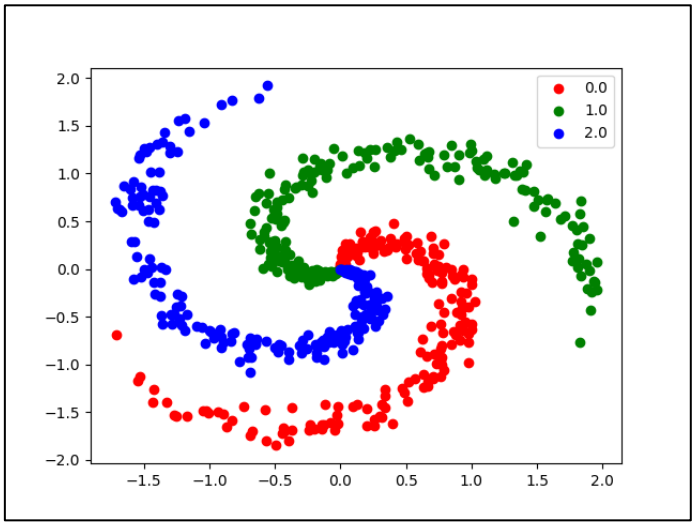

Figure 15: Plot of Dataset 1B

2.5.2 Multilayer Feedforward Neural Network – Classification Boundary

The plot for the classification boundaries, for the MLFFNN Classifier using 2 hidden layers is as shown in the plot below.

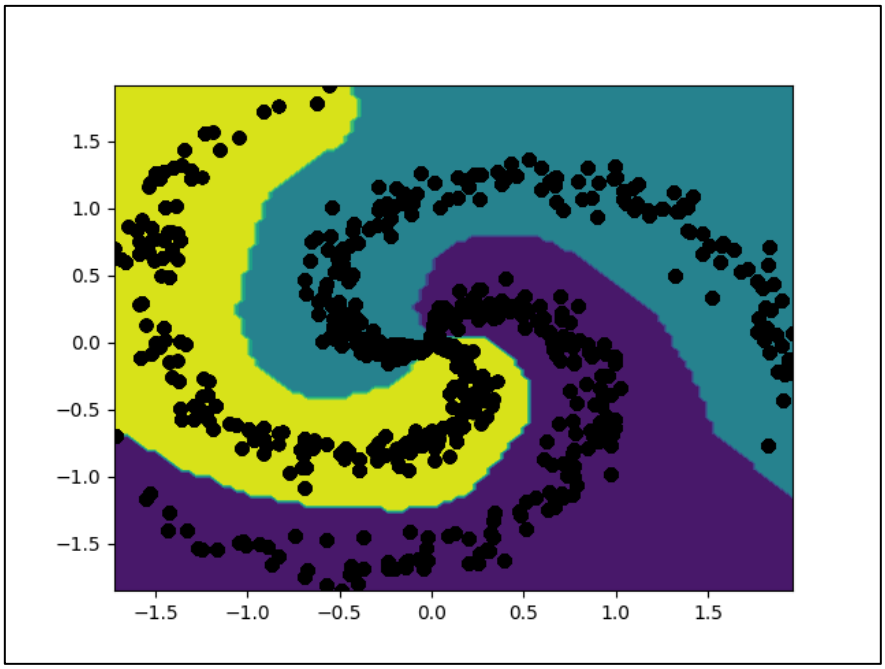

Figure 16: Decision Boundaries for MLFFNN Classifier using 2 Hidden Layers

2.5.3 Multilayer Feedforward Neural Network – Surface Plots

Plots of the surfaces of the hidden layers and the output layers of the neural network after certain epochs are given below.

2.5.3.1 Hidden Layer 1

Plots for Hidden Layer 1 are given below.

2.5.3.1.1 Node 10

Plots for the 10th node of the first hidden layer are given below.

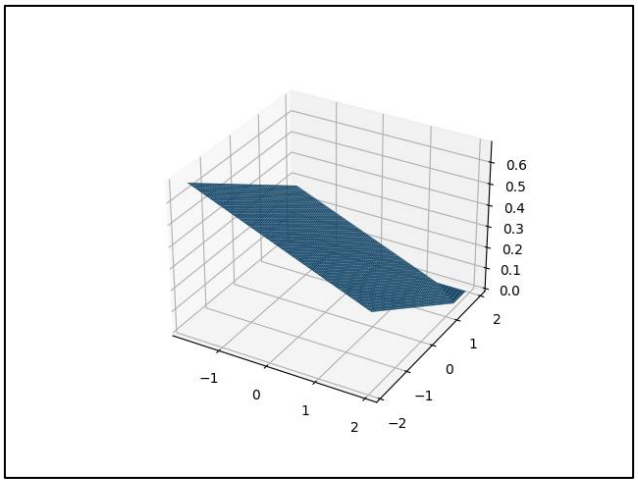

Figure 17: Hidden Layer 1 - Node 10 - Epoch 1

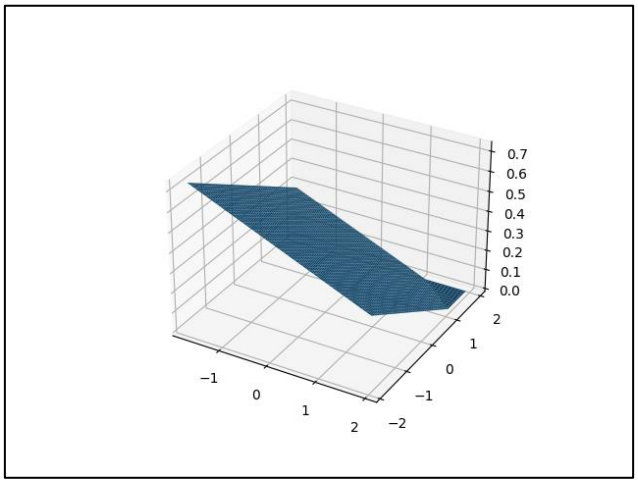

Figure 18: Hidden Layer 1 - Node 10 - Epoch 5

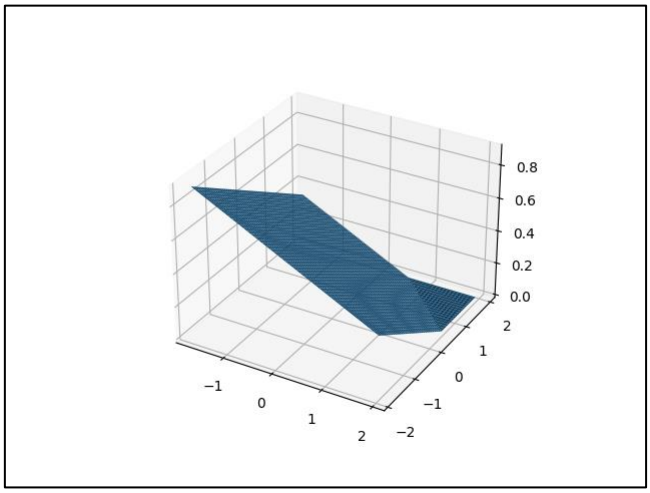

Figure 19: Hidden Layer 1 - Node 10 - Epoch 20

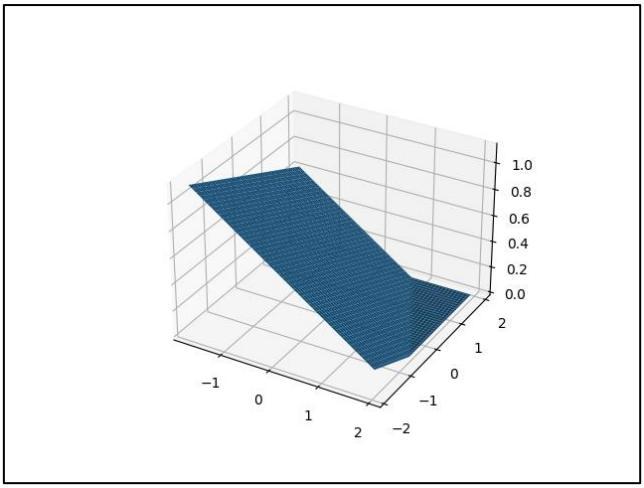

Figure 20: Hidden Layer 1 - Node 10 - Epoch 100

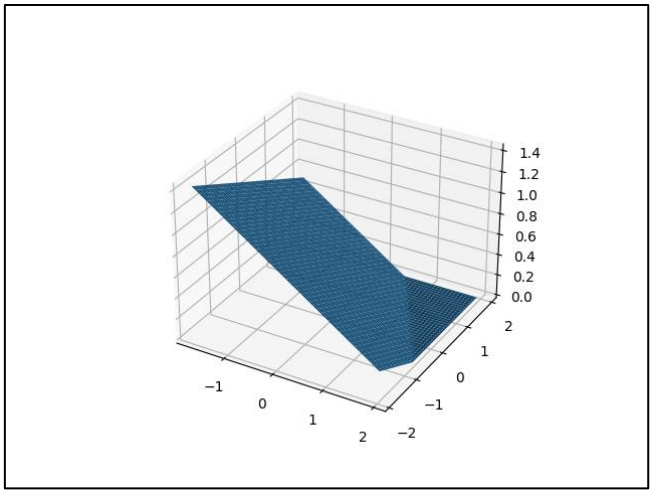

Figure 21: Hidden Layer 1 - Node 10 – Convergence

2.5.3.1.2 Node 20

Plots for the 20th node of the first hidden layer are given below.

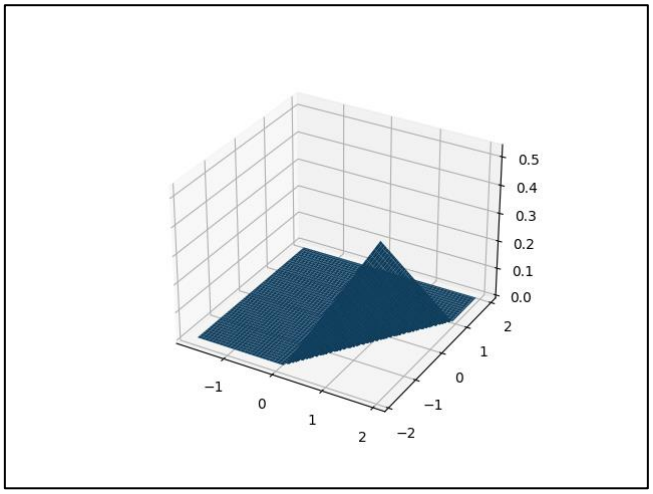

Figure 22: Hidden Layer 1 - Node 20 - Epoch 1

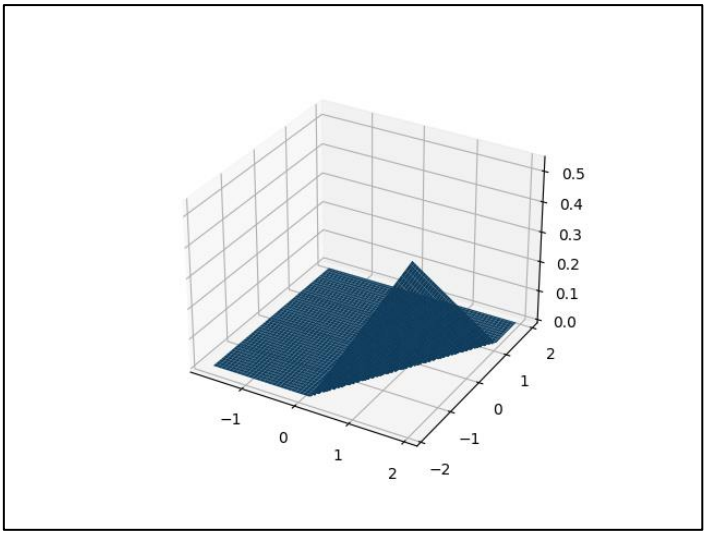

Figure 23: Hidden Layer 1 - Node 20 - Epoch 5

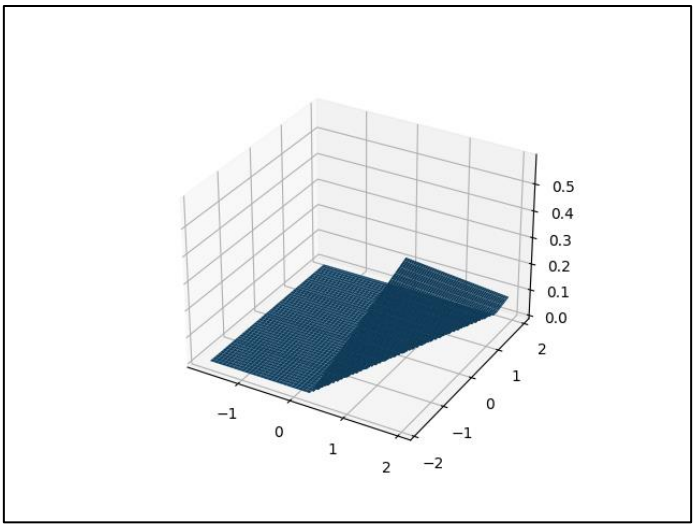

Figure 24: Hidden Layer 1 - Node 20 - Epoch 20

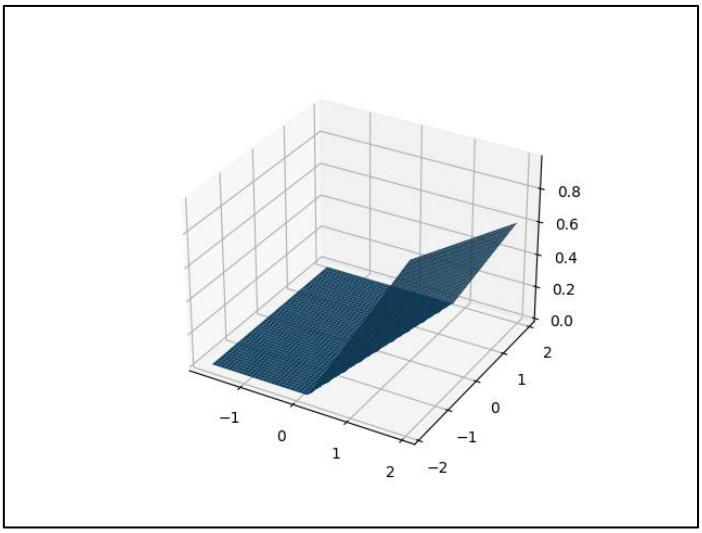

Figure 25: Hidden Layer 1 - Node 20 - Epoch 100

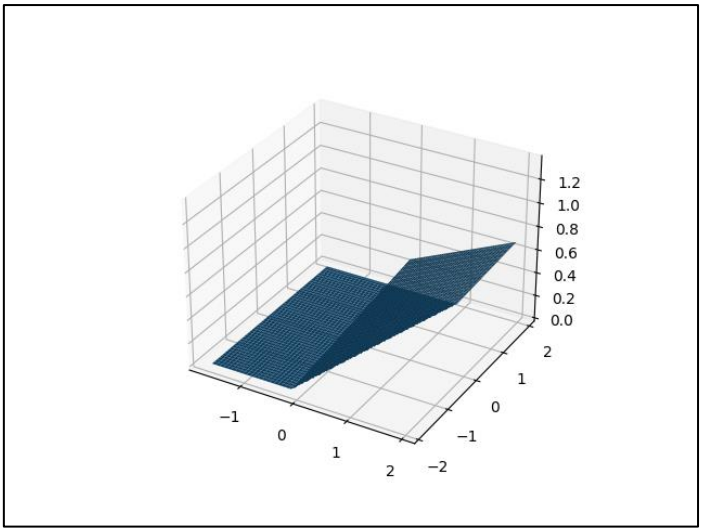

Figure 26: Hidden Layer 1 - Node 20 - Convergence

2.5.3.2 Hidden Layer 2

Plots for Hidden Layer 1 are given below.

2.5.3.2.1 Node 10

Plots for the 10th node of the second hidden layer are given below.

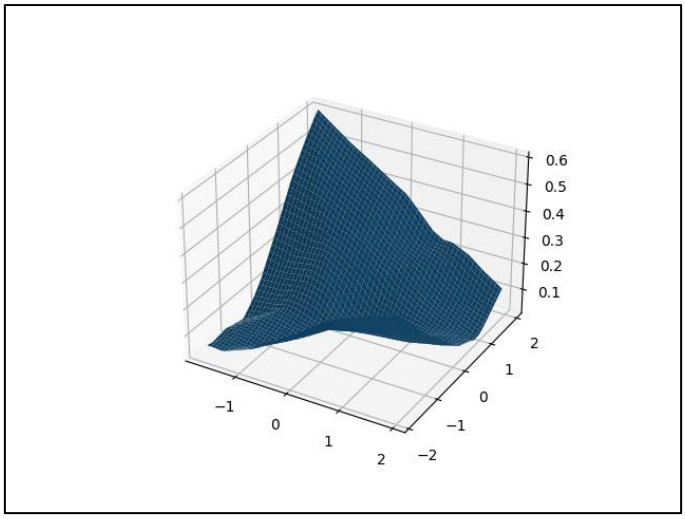

Figure 27: Hidden Layer 2 - Node 10 - Epoch 1

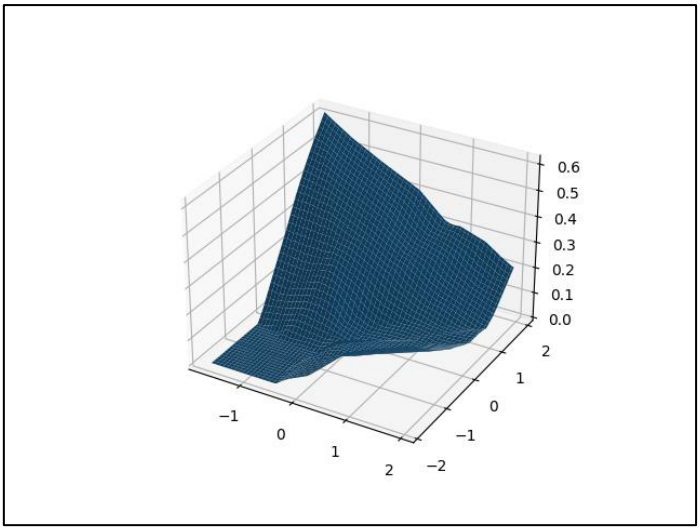

Figure 28: Hidden Layer 2 - Node 10 - Epoch 5

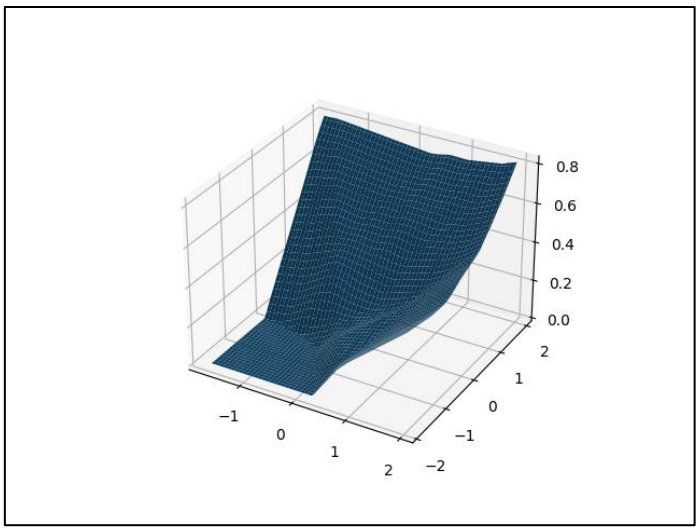

Figure 29: Hidden Layer 2 - Node 10 - Epoch 20

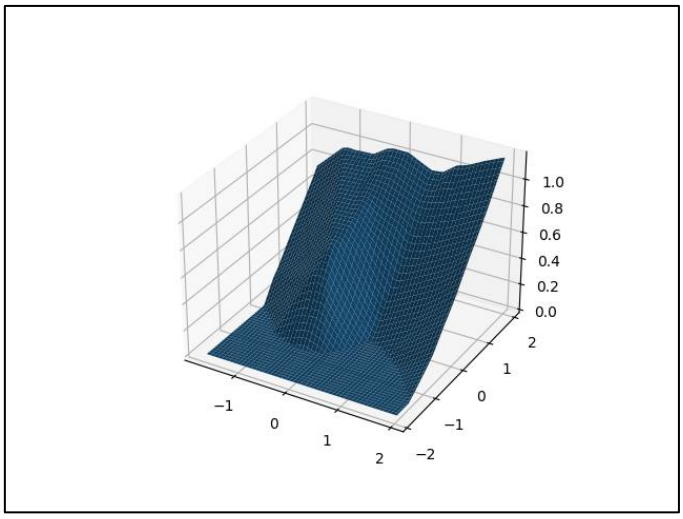

Figure 30: Hidden Layer 2 - Node 10 - Epoch 100

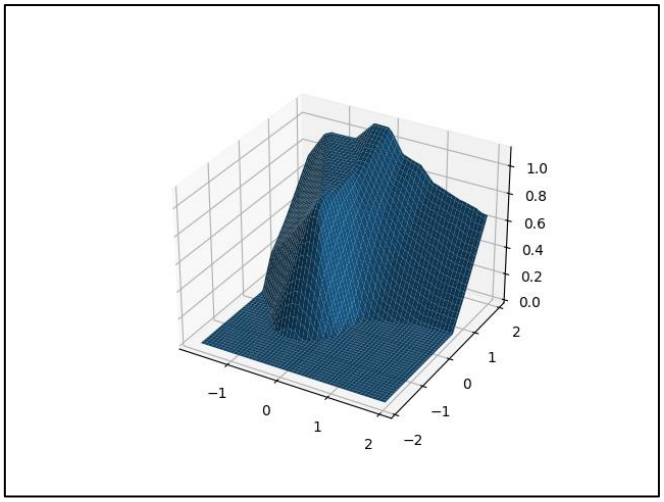

Figure 31: Hidden Layer 2 - Node 10 – Convergence

2.5.3.2.2 Node 20

Plots for the 20th node of the first hidden layer are given below.

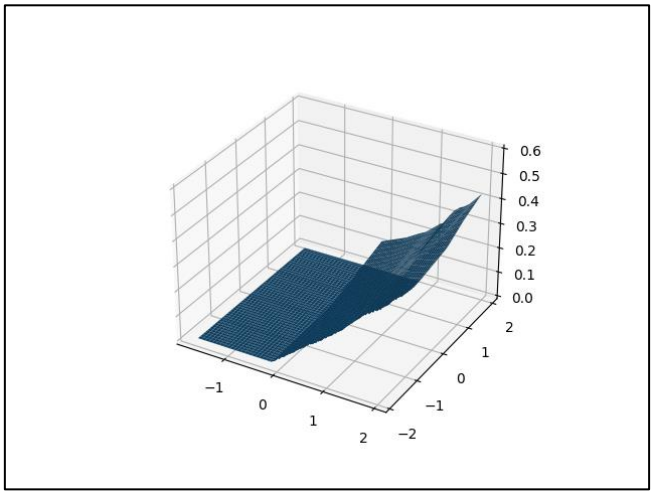

Figure 32: Hidden Layer 2 - Node 20 - Epoch 1

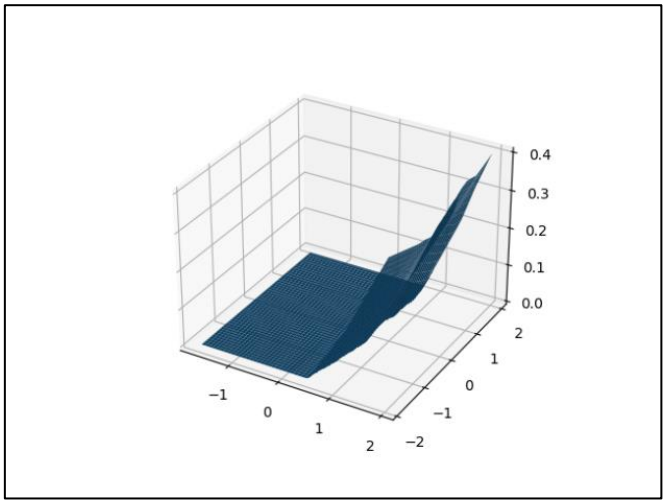

Figure 33: Hidden Layer 2 - Node 20 - Epoch 5

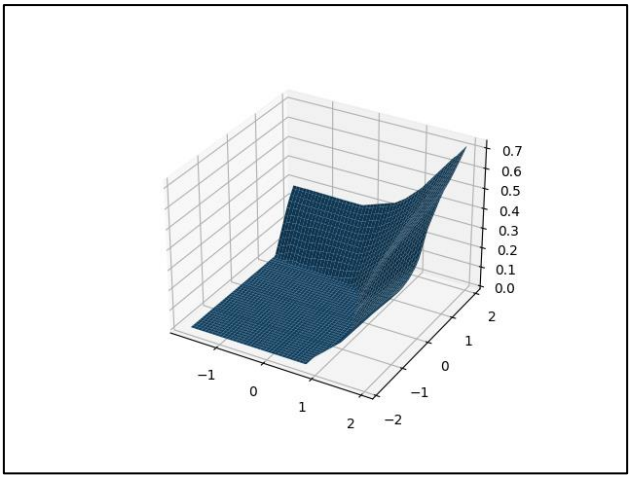

Figure 34: Hidden Layer 2 - Node 20 - Epoch 20

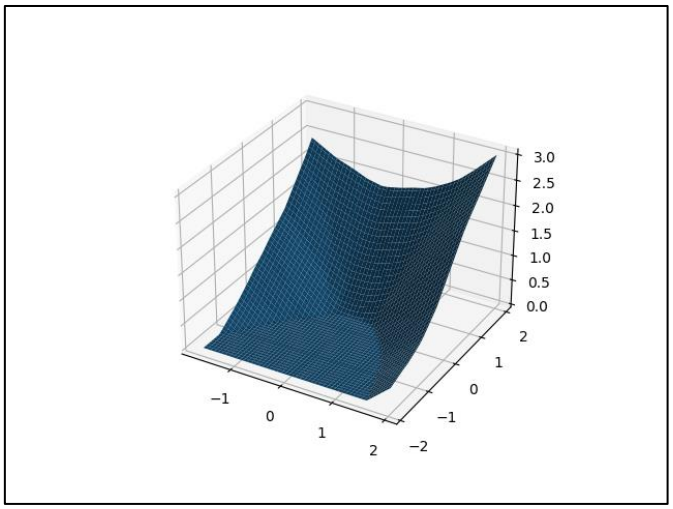

Figure 35: Hidden Layer 2 - Node 20 - Epoch 100

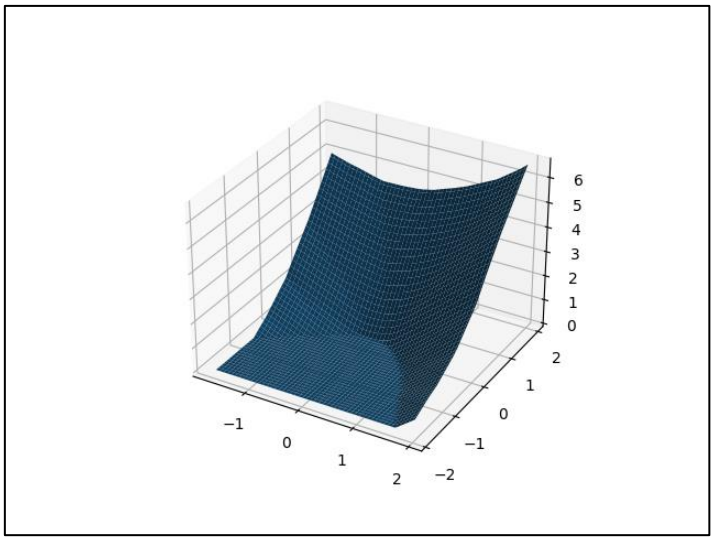

Figure 36: Hidden Layer 2 - Node 20 - Convergence

2.5.3.3 Output Layer

Plots for Output Layer are given below.

2.5.3.3.1 Node 1

Plots for the 1st node (corresponding to Class 0) of the output layer are given below.

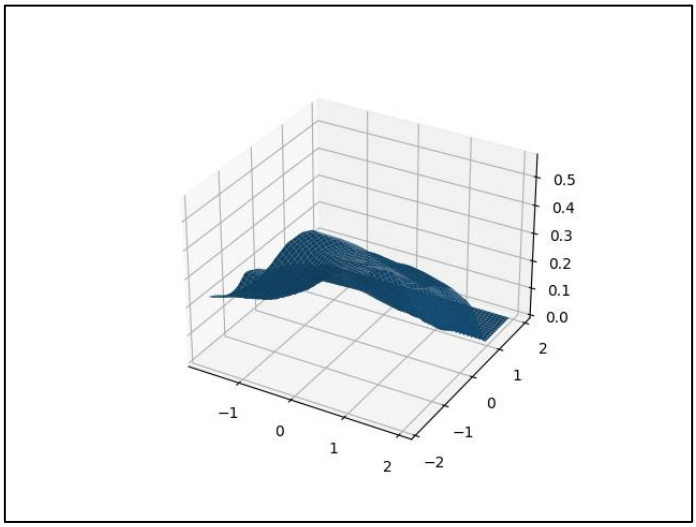

Figure 37: Output Layer - Node 1 - Epoch 1

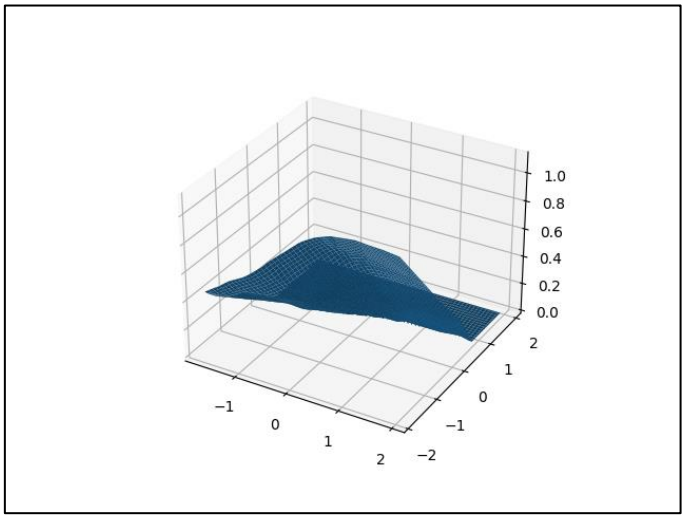

Figure 38: Output Layer - Node 1 - Epoch 5

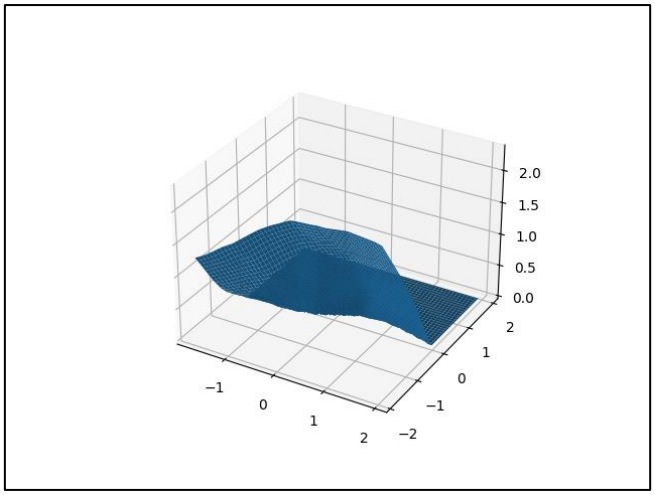

Figure 39: Output Layer - Node 1 - Epoch 20

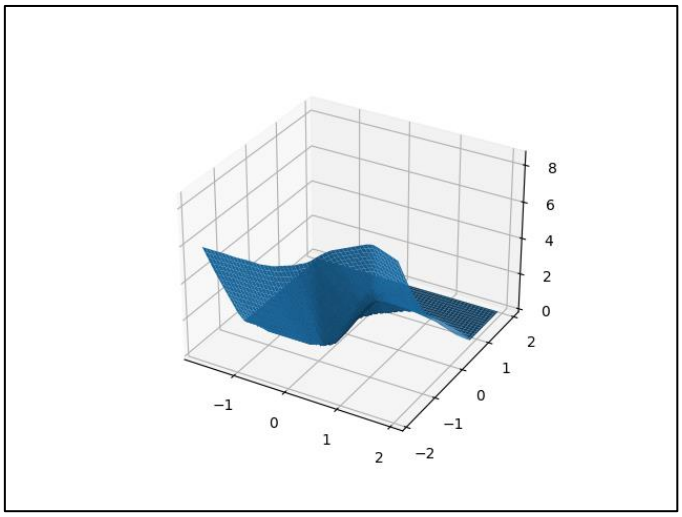

Figure 40: Output Layer - Node 1 - Epoch 100

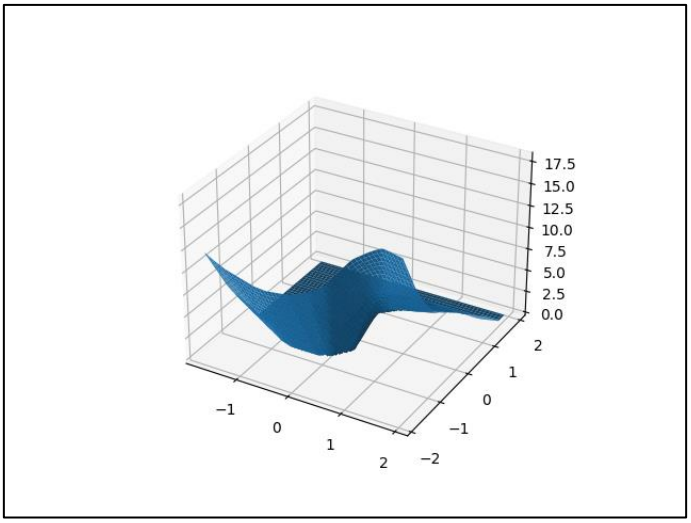

Figure 41: Output Layer - Node 1 – Convergence

2.5.3.3.2 Node 2

Plots for the 2nd node (corresponding to Class 1) of the output layer are given below.

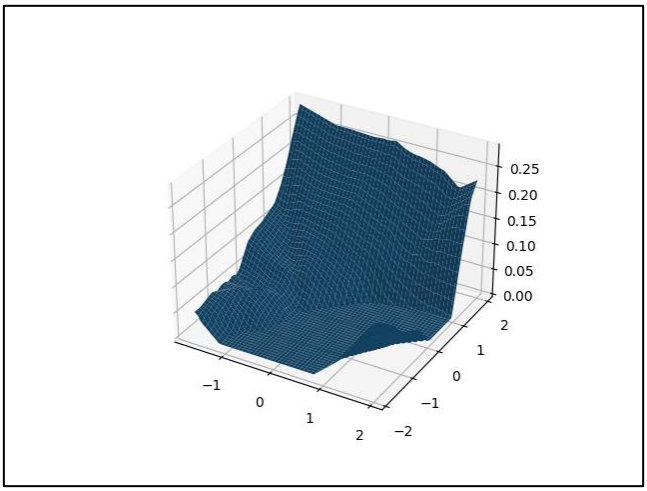

Figure 42: Output Layer - Node 2 - Epoch 1

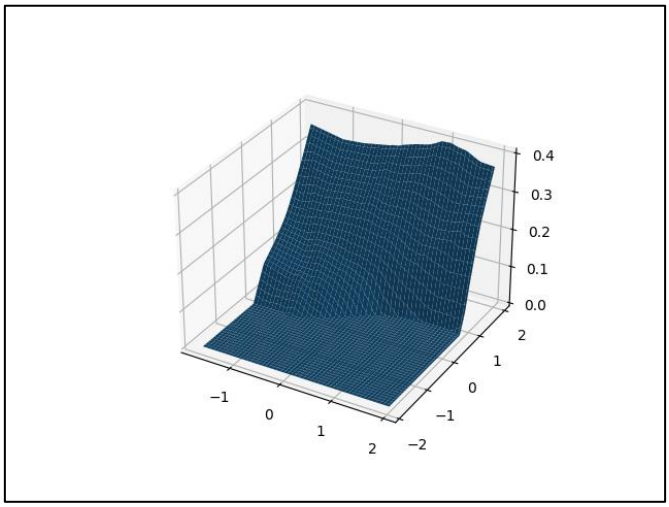

Figure 43: Output Layer - Node 2 - Epoch 5

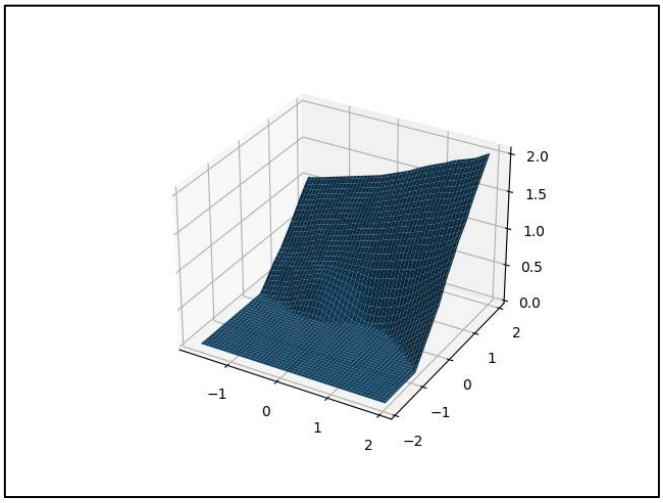

Figure 44: Output Layer - Node 2 - Epoch 20

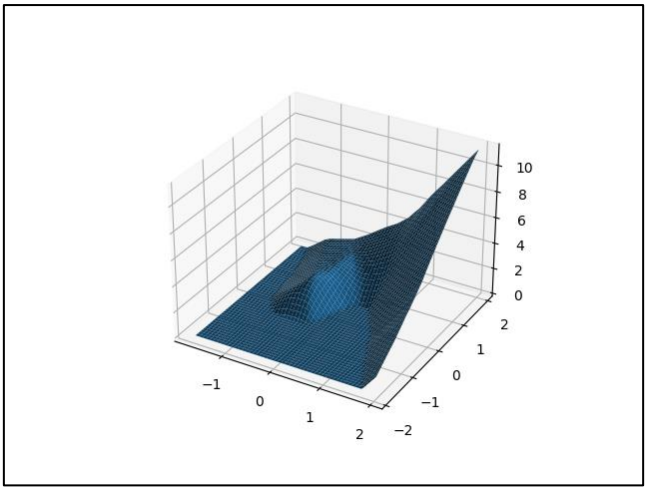

Figure 45: Output Layer - Node 2 - Epoch 100

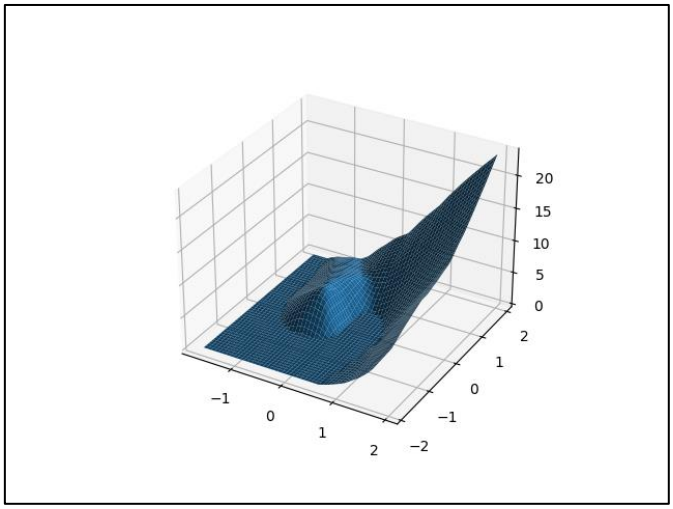

Figure 46: Output Layer - Node 2 - Convergence

2.5.3.3.3 Node 3

Plots for the 3rd node (corresponding to Class 2) of the output layer are given below.

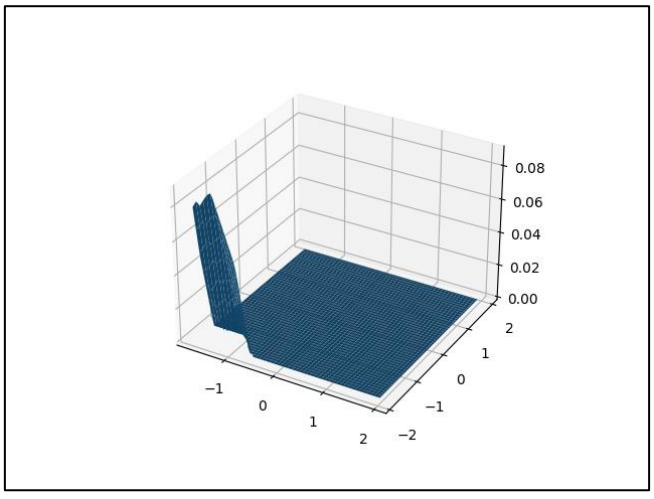

Figure 47: Output Layer - Node 3 - Epoch 1

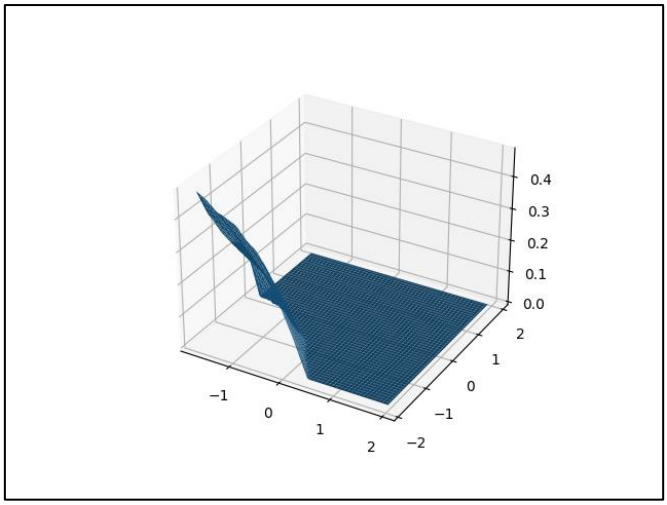

Figure 48: Output Layer - Node 3 - Epoch 5

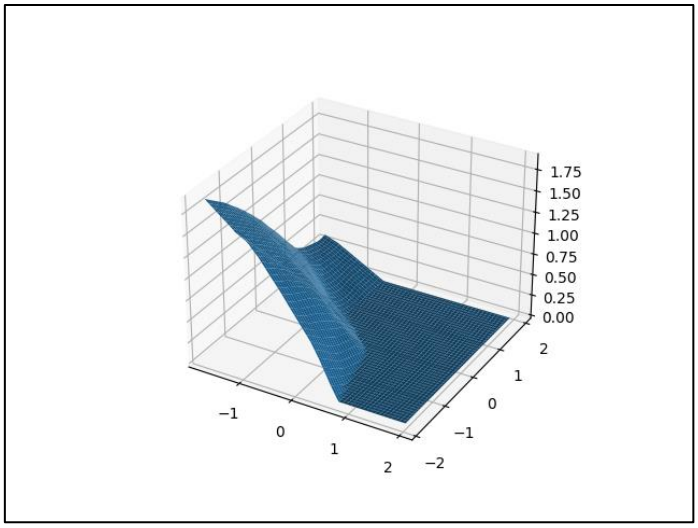

Figure 49: Output Layer - Node 3 - Epoch 20

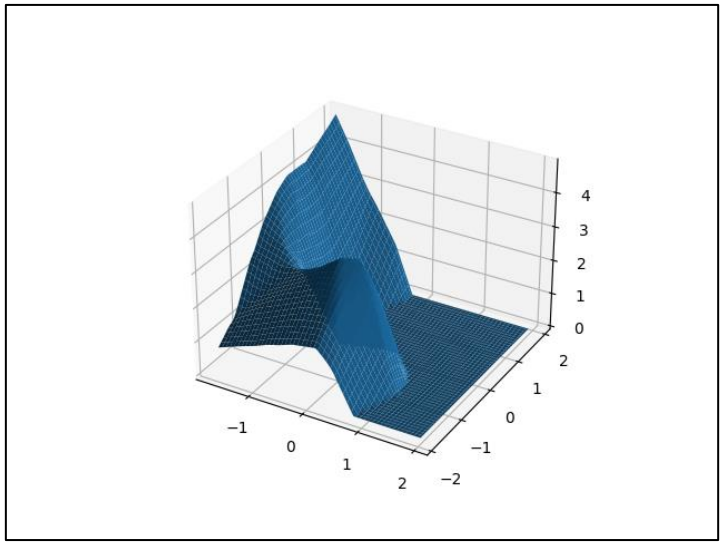

Figure 50: Output Layer - Node 3 - Epoch 100

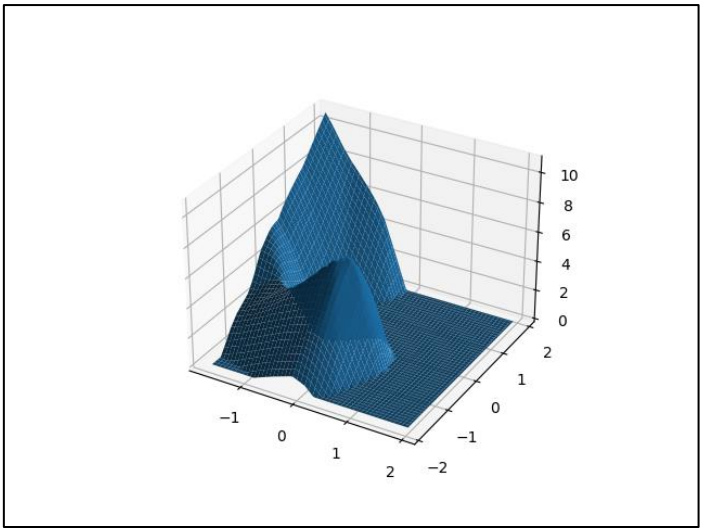

Figure 51: Output Layer - Node 3 - Convergence

2.5.4 Nonlinear SVM using one-against-the-rest approach

The plots for the classification boundaries, non-linear SVM, using one-against-the-rest approach, for different kinds of kernels are shown below.

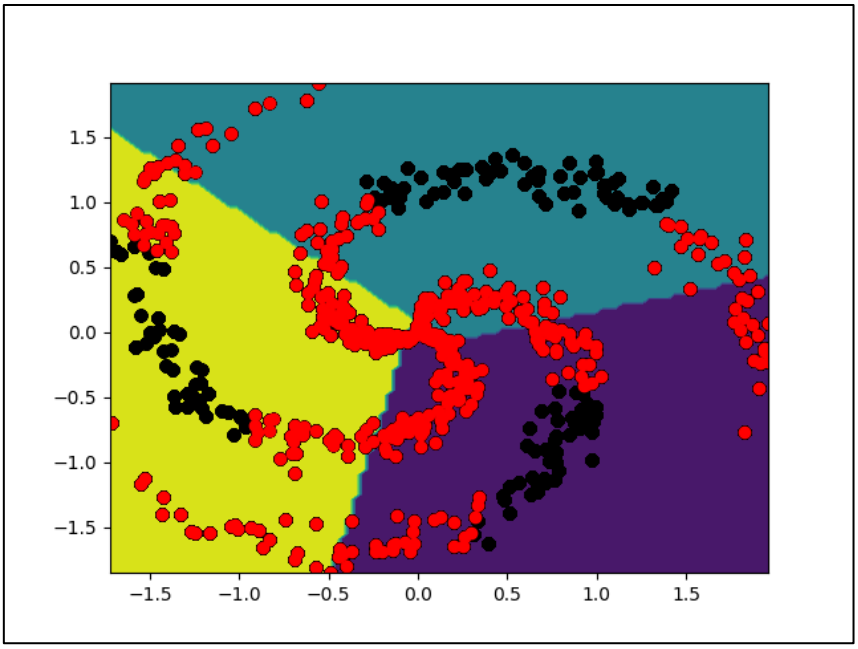

Figure 52: Decision Boundary for Non-linear SVM with Linear Kernel

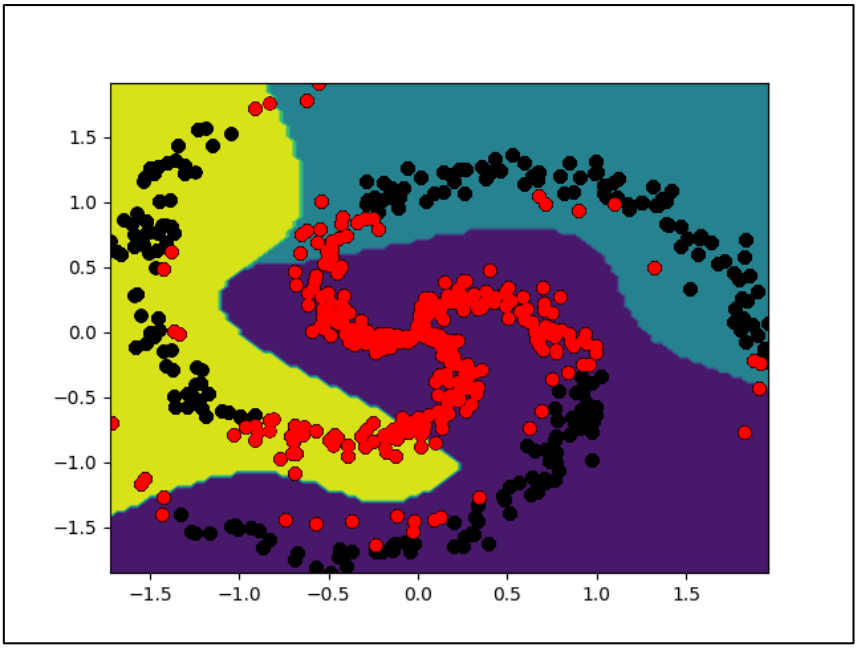

Figure 53: Decision Boundary for Non-linear SVM with Polynomial Kernel

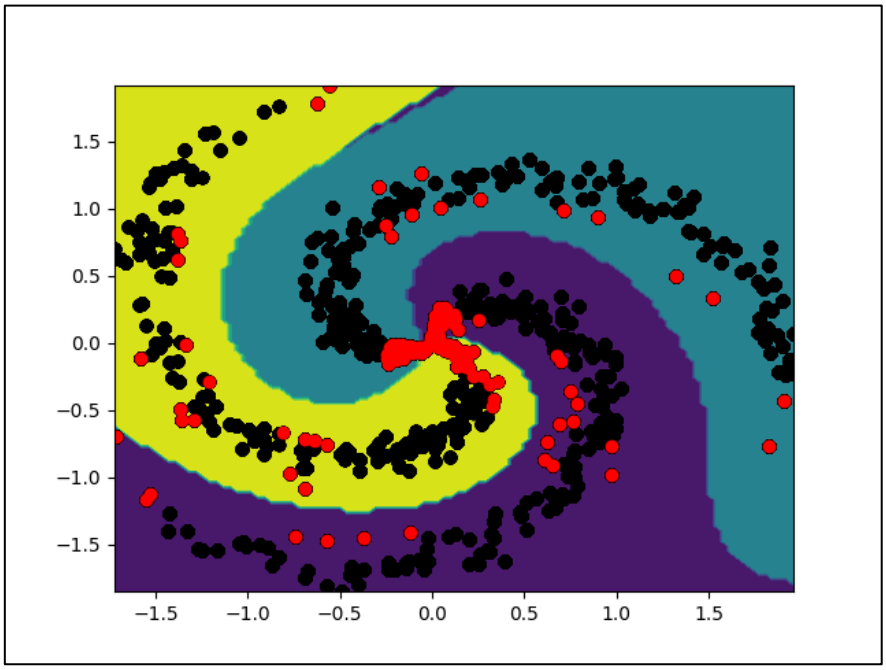

Figure 54: Decision Boundary for Non-linear SVM with RBF Kernel

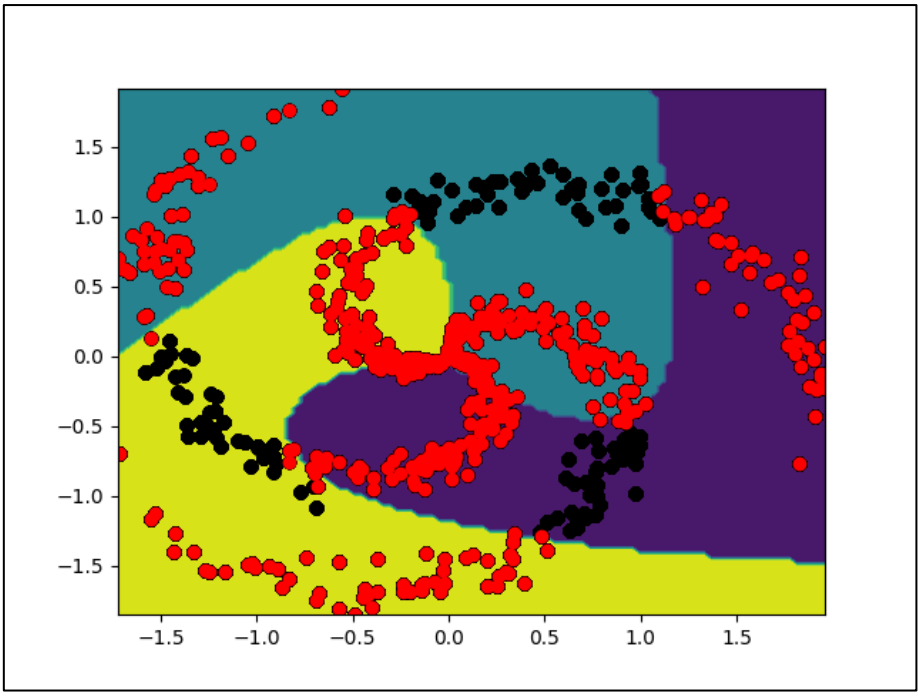

Figure 55: Decision Boundary for Non-linear SVM with Sigmoid Kernel

2.6 Observations

Some key observations after looking at the results are as follows:

2.6.1 Multilayer Feedforward Neural Network

- Multilayer Fast-forward Neural network performs better than SVM classifier giving a best-case accuracy of 98.88 % on validation dataset when using 2 hidden layers with 40 neurons each and 'relu' activation function.
- The 'logistic' activation function performs really poor than the other two function.
- The hidden layer surface plot doesn't look like output layer plots as hidden layer don't directly reflect the output layer results.
- As the number of epochs increases, each node adapts its weights to minimize the loss, as it can be seen from the surface plots for each layer where, the surfaces are gradually changing with each epoch until it converges.

2.6.2 Non-linear SVM

- In this case, the data is not that well separated as compared to 1a. The data is distributed in a spiral fashion. But data points belonging to the same class are close to each other. Hence the nonlinear SVM classifier is not able to perform that well on the given dataset.
- For SVM classifier 'Gaussian' kernel gives better performance when compared to 'polynomial' kernel as it can be seen in the contour plot, 'Gaussian' kernel giving a best-case accuracy of 98.88% and polynomial kernel giving best accuracy of 74.16%.
- The polynomial kernel results signify that high Degree doesn’t really improve classifier. However, it can be so as higher the degree more bended is the data. We can get improved results with higher degree but the time taken for classification will also increase significantly.

3 Dataset 2 – Image Data Set for Static Pattern Classification

3.1 Problem Statement

We are given a Real-world data set i.e., an Image data set for static pattern classification.

For this data, as per the mapping given to us (Group 18), we work on the classes: **coast, forest, highway, mountain, and tall building**.

Based upon this, the following classifiers need to be built.

1. MLFFNN with two hidden layers
2. Gaussian kernel based SVM using one-against-the-rest approach

3.2 Classification Accuracies Obtained

Upon implementing the various classifiers, the accuracies obtained are as follows.

3.2.1 Multilayer Feedforward Neural Network

Training and validation accuracies for MLFFNN Classifier with 2 hidden layers are as below.

Table 49: Training and Validation Accuracies for MLFFNN Classifier with 2 hidden layers

No. of neurons in hidden layer 1	No. of neurons in hidden layer 2	Solver	Activation Function	Alpha	Gamma	Train Accuracy	Validation Accuracy
100	100	lbfgs	identity	0.0001	constant	0.585324	0.465875
100	100	lbfgs	identity	0.0001	invscaling	0.585324	0.465875
100	100	lbfgs	identity	0.0001	adaptive	0.585324	0.465875
100	100	lbfgs	identity	0.001	constant	0.586177	0.465875
100	100	lbfgs	identity	0.001	invscaling	0.586177	0.465875
100	100	lbfgs	identity	0.001	adaptive	0.586177	0.465875
100	100	lbfgs	identity	0.01	constant	0.586177	0.465875
100	100	lbfgs	identity	0.01	invscaling	0.586177	0.465875
100	100	lbfgs	identity	0.01	adaptive	0.586177	0.465875
100	100	lbfgs	logistic	0.0001	constant	1	0.456973
100	100	lbfgs	logistic	0.0001	invscaling	1	0.456973
100	100	lbfgs	logistic	0.0001	adaptive	1	0.456973
100	100	lbfgs	logistic	0.001	constant	1	0.519288
100	100	lbfgs	logistic	0.001	invscaling	1	0.519288
100	100	lbfgs	logistic	0.001	adaptive	1	0.519288
100	100	lbfgs	logistic	0.01	constant	1	0.480712
100	100	lbfgs	logistic	0.01	invscaling	1	0.480712
100	100	lbfgs	logistic	0.01	adaptive	1	0.480712
100	100	lbfgs	tanh	0.0001	constant	1	0.454006
100	100	lbfgs	tanh	0.0001	invscaling	1	0.454006
100	100	lbfgs	tanh	0.0001	adaptive	1	0.454006
100	100	lbfgs	tanh	0.001	constant	1	0.501484
100	100	lbfgs	tanh	0.001	invscaling	1	0.501484
100	100	lbfgs	tanh	0.001	adaptive	1	0.501484
100	100	lbfgs	tanh	0.01	constant	1	0.492582
100	100	lbfgs	tanh	0.01	invscaling	1	0.492582
100	100	lbfgs	tanh	0.01	adaptive	1	0.492582
100	100	lbfgs	relu	0.0001	constant	1	0.498516
100	100	lbfgs	relu	0.0001	invscaling	1	0.498516
100	100	lbfgs	relu	0.0001	adaptive	1	0.498516

100	100	lbfgs	relu	0.001	constant	1	0.507418
100	100	lbfgs	relu	0.001	invscaling	1	0.507418
100	100	lbfgs	relu	0.001	adaptive	1	0.507418
100	100	lbfgs	relu	0.01	constant	1	0.474777
100	100	lbfgs	relu	0.01	invscaling	1	0.474777
100	100	lbfgs	relu	0.01	adaptive	1	0.474777
100	100	sgd	identity	0.0001	constant	0.573379	0.468843
100	100	sgd	identity	0.0001	invscaling	0.21843	0.21365
100	100	sgd	identity	0.0001	adaptive	0.572526	0.468843
100	100	sgd	identity	0.001	constant	0.573379	0.468843
100	100	sgd	identity	0.001	invscaling	0.21843	0.21365
100	100	sgd	identity	0.001	adaptive	0.572526	0.468843
100	100	sgd	identity	0.01	constant	0.572526	0.468843
100	100	sgd	identity	0.01	invscaling	0.21843	0.21365
100	100	sgd	identity	0.01	adaptive	0.572526	0.468843
100	100	sgd	logistic	0.0001	constant	0.222696	0.222552
100	100	sgd	logistic	0.0001	invscaling	0.212457	0.210682
100	100	sgd	logistic	0.0001	adaptive	0.222696	0.222552
100	100	sgd	logistic	0.001	constant	0.222696	0.222552
100	100	sgd	logistic	0.001	invscaling	0.212457	0.210682
100	100	sgd	logistic	0.001	adaptive	0.222696	0.222552
100	100	sgd	logistic	0.01	constant	0.222696	0.222552
100	100	sgd	logistic	0.01	invscaling	0.212457	0.210682
100	100	sgd	logistic	0.01	adaptive	0.222696	0.222552
100	100	sgd	tanh	0.0001	constant	0.579352	0.48368
100	100	sgd	tanh	0.0001	invscaling	0.21843	0.219585
100	100	sgd	tanh	0.0001	adaptive	0.580205	0.480712
100	100	sgd	tanh	0.001	constant	0.579352	0.48368
100	100	sgd	tanh	0.001	invscaling	0.21843	0.219585
100	100	sgd	tanh	0.001	adaptive	0.580205	0.480712
100	100	sgd	tanh	0.01	constant	0.579352	0.48368
100	100	sgd	tanh	0.01	invscaling	0.21843	0.219585
100	100	sgd	tanh	0.01	adaptive	0.579352	0.48368
100	100	sgd	relu	0.0001	constant	0.78157	0.507418
100	100	sgd	relu	0.0001	invscaling	0.222696	0.222552
100	100	sgd	relu	0.0001	adaptive	0.78157	0.504451
100	100	sgd	relu	0.001	constant	0.778157	0.501484
100	100	sgd	relu	0.001	invscaling	0.222696	0.222552
100	100	sgd	relu	0.001	adaptive	0.783276	0.501484
100	100	sgd	relu	0.01	constant	0.78157	0.507418
100	100	sgd	relu	0.01	invscaling	0.222696	0.222552
100	100	sgd	relu	0.01	adaptive	0.783276	0.510386
100	100	adam	identity	0.0001	constant	0.579352	0.468843
100	100	adam	identity	0.0001	invscaling	0.579352	0.468843
100	100	adam	identity	0.0001	adaptive	0.579352	0.468843
100	100	adam	identity	0.001	constant	0.580205	0.468843
100	100	adam	identity	0.001	invscaling	0.580205	0.468843
100	100	adam	identity	0.001	adaptive	0.580205	0.468843
100	100	adam	identity	0.01	constant	0.578498	0.468843
100	100	adam	identity	0.01	invscaling	0.578498	0.468843
100	100	adam	identity	0.01	adaptive	0.578498	0.468843
100	100	adam	logistic	0.0001	constant	0.59727	0.48368
100	100	adam	logistic	0.0001	invscaling	0.59727	0.48368
100	100	adam	logistic	0.0001	adaptive	0.59727	0.48368

100	100	adam	logistic	0.001	constant	0.58959	0.474777
100	100	adam	logistic	0.001	invscaling	0.58959	0.474777
100	100	adam	logistic	0.001	adaptive	0.58959	0.474777
100	100	adam	logistic	0.01	constant	0.56314	0.451039
100	100	adam	logistic	0.01	invscaling	0.56314	0.451039
100	100	adam	logistic	0.01	adaptive	0.56314	0.451039
100	100	adam	tanh	0.0001	constant	1	0.51632
100	100	adam	tanh	0.0001	invscaling	1	0.51632
100	100	adam	tanh	0.0001	adaptive	1	0.51632
100	100	adam	tanh	0.001	constant	1	0.51632
100	100	adam	tanh	0.001	invscaling	1	0.51632
100	100	adam	tanh	0.001	adaptive	1	0.51632
100	100	adam	tanh	0.01	constant	1	0.498516
100	100	adam	tanh	0.01	invscaling	1	0.498516
100	100	adam	tanh	0.01	adaptive	1	0.498516
100	100	adam	relu	0.0001	constant	0.999147	0.504451
100	100	adam	relu	0.0001	invscaling	0.999147	0.504451
100	100	adam	relu	0.0001	adaptive	0.999147	0.504451
100	100	adam	relu	0.001	constant	1	0.51632
100	100	adam	relu	0.001	invscaling	1	0.51632
100	100	adam	relu	0.001	adaptive	1	0.51632
100	100	adam	relu	0.01	constant	0.987201	0.519288
100	100	adam	relu	0.01	invscaling	0.987201	0.519288
100	100	adam	relu	0.01	adaptive	0.987201	0.519288
100	150	lbfgs	identity	0.0001	constant	0.585324	0.465875
100	150	lbfgs	identity	0.0001	invscaling	0.585324	0.465875
100	150	lbfgs	identity	0.0001	adaptive	0.585324	0.465875
100	150	lbfgs	identity	0.001	constant	0.585324	0.465875
100	150	lbfgs	identity	0.001	invscaling	0.585324	0.465875
100	150	lbfgs	identity	0.001	adaptive	0.585324	0.465875
100	150	lbfgs	identity	0.01	constant	0.585324	0.462908
100	150	lbfgs	identity	0.01	invscaling	0.585324	0.462908
100	150	lbfgs	identity	0.01	adaptive	0.585324	0.462908
100	150	lbfgs	logistic	0.0001	constant	1	0.459941
100	150	lbfgs	logistic	0.0001	invscaling	1	0.459941
100	150	lbfgs	logistic	0.0001	adaptive	1	0.459941
100	150	lbfgs	logistic	0.001	constant	1	0.510386
100	150	lbfgs	logistic	0.001	invscaling	1	0.510386
100	150	lbfgs	logistic	0.001	adaptive	1	0.510386
100	150	lbfgs	logistic	0.01	constant	1	0.513353
100	150	lbfgs	logistic	0.01	invscaling	1	0.513353
100	150	lbfgs	logistic	0.01	adaptive	1	0.513353
100	150	lbfgs	tanh	0.0001	constant	1	0.486647
100	150	lbfgs	tanh	0.0001	invscaling	1	0.486647
100	150	lbfgs	tanh	0.0001	adaptive	1	0.486647
100	150	lbfgs	tanh	0.001	constant	1	0.495549
100	150	lbfgs	tanh	0.001	invscaling	1	0.495549
100	150	lbfgs	tanh	0.001	adaptive	1	0.495549
100	150	lbfgs	tanh	0.01	constant	1	0.492582
100	150	lbfgs	tanh	0.01	invscaling	1	0.492582
100	150	lbfgs	tanh	0.01	adaptive	1	0.492582
100	150	lbfgs	relu	0.0001	constant	1	0.459941
100	150	lbfgs	relu	0.0001	invscaling	1	0.459941
100	150	lbfgs	relu	0.0001	adaptive	1	0.459941

100	150	lbfgs	relu	0.001	constant	1	0.486647
100	150	lbfgs	relu	0.001	invscaling	1	0.486647
100	150	lbfgs	relu	0.001	adaptive	1	0.486647
100	150	lbfgs	relu	0.01	constant	1	0.477745
100	150	lbfgs	relu	0.01	invscaling	1	0.477745
100	150	lbfgs	relu	0.01	adaptive	1	0.477745
100	150	sgd	identity	0.0001	constant	0.569966	0.468843
100	150	sgd	identity	0.0001	invscaling	0.209898	0.204748
100	150	sgd	identity	0.0001	adaptive	0.571672	0.47181
100	150	sgd	identity	0.001	constant	0.569966	0.468843
100	150	sgd	identity	0.001	invscaling	0.209898	0.204748
100	150	sgd	identity	0.001	adaptive	0.571672	0.47181
100	150	sgd	identity	0.01	constant	0.570819	0.468843
100	150	sgd	identity	0.01	invscaling	0.209898	0.204748
100	150	sgd	identity	0.01	adaptive	0.569113	0.465875
100	150	sgd	logistic	0.0001	constant	0.222696	0.222552
100	150	sgd	logistic	0.0001	invscaling	0.195392	0.195846
100	150	sgd	logistic	0.0001	adaptive	0.222696	0.222552
100	150	sgd	logistic	0.001	constant	0.222696	0.222552
100	150	sgd	logistic	0.001	invscaling	0.195392	0.195846
100	150	sgd	logistic	0.001	adaptive	0.222696	0.222552
100	150	sgd	logistic	0.01	constant	0.222696	0.222552
100	150	sgd	logistic	0.01	invscaling	0.195392	0.195846
100	150	sgd	logistic	0.01	adaptive	0.222696	0.222552
100	150	sgd	tanh	0.0001	constant	0.570819	0.47181
100	150	sgd	tanh	0.0001	invscaling	0.208191	0.20178
100	150	sgd	tanh	0.0001	adaptive	0.571672	0.47181
100	150	sgd	tanh	0.001	constant	0.570819	0.47181
100	150	sgd	tanh	0.001	invscaling	0.208191	0.20178
100	150	sgd	tanh	0.001	adaptive	0.571672	0.47181
100	150	sgd	tanh	0.01	constant	0.570819	0.47181
100	150	sgd	tanh	0.01	invscaling	0.208191	0.20178
100	150	sgd	tanh	0.01	adaptive	0.570819	0.47181
100	150	sgd	relu	0.0001	constant	0.840444	0.492582
100	150	sgd	relu	0.0001	invscaling	0.196246	0.195846
100	150	sgd	relu	0.0001	adaptive	0.84215	0.489614
100	150	sgd	relu	0.001	constant	0.841297	0.510386
100	150	sgd	relu	0.001	invscaling	0.196246	0.195846
100	150	sgd	relu	0.001	adaptive	0.843003	0.504451
100	150	sgd	relu	0.01	constant	0.827645	0.510386
100	150	sgd	relu	0.01	invscaling	0.196246	0.195846
100	150	sgd	relu	0.01	adaptive	0.826792	0.513353
100	150	adam	identity	0.0001	constant	0.580205	0.462908
100	150	adam	identity	0.0001	invscaling	0.580205	0.462908
100	150	adam	identity	0.0001	adaptive	0.580205	0.462908
100	150	adam	identity	0.001	constant	0.580205	0.462908
100	150	adam	identity	0.001	invscaling	0.580205	0.462908
100	150	adam	identity	0.001	adaptive	0.580205	0.462908
100	150	adam	identity	0.01	constant	0.579352	0.462908
100	150	adam	identity	0.01	invscaling	0.579352	0.462908
100	150	adam	identity	0.01	adaptive	0.579352	0.462908
100	150	adam	logistic	0.0001	constant	0.549488	0.424332
100	150	adam	logistic	0.0001	invscaling	0.549488	0.424332
100	150	adam	logistic	0.0001	adaptive	0.549488	0.424332

100	150	adam	logistic	0.001	constant	0.548635	0.424332
100	150	adam	logistic	0.001	invscaling	0.548635	0.424332
100	150	adam	logistic	0.001	adaptive	0.548635	0.424332
100	150	adam	logistic	0.01	constant	0.544369	0.421365
100	150	adam	logistic	0.01	invscaling	0.544369	0.421365
100	150	adam	logistic	0.01	adaptive	0.544369	0.421365
100	150	adam	tanh	0.0001	constant	1	0.540059
100	150	adam	tanh	0.0001	invscaling	1	0.540059
100	150	adam	tanh	0.0001	adaptive	1	0.540059
100	150	adam	tanh	0.001	constant	1	0.543027
100	150	adam	tanh	0.001	invscaling	1	0.543027
100	150	adam	tanh	0.001	adaptive	1	0.543027
100	150	adam	tanh	0.01	constant	1	0.504451
100	150	adam	tanh	0.01	invscaling	1	0.504451
100	150	adam	tanh	0.01	adaptive	1	0.504451
100	150	adam	relu	0.0001	constant	1	0.52819
100	150	adam	relu	0.0001	invscaling	1	0.52819
100	150	adam	relu	0.0001	adaptive	1	0.52819
100	150	adam	relu	0.001	constant	1	0.525223
100	150	adam	relu	0.001	invscaling	1	0.525223
100	150	adam	relu	0.001	adaptive	1	0.525223
100	150	adam	relu	0.01	constant	0.963311	0.513353
100	150	adam	relu	0.01	invscaling	0.963311	0.513353
100	150	adam	relu	0.01	adaptive	0.963311	0.513353
150	200	lbfgs	identity	0.0001	constant	0.584471	0.465875
150	200	lbfgs	identity	0.0001	invscaling	0.584471	0.465875
150	200	lbfgs	identity	0.0001	adaptive	0.584471	0.465875
150	200	lbfgs	identity	0.001	constant	0.586177	0.468843
150	200	lbfgs	identity	0.001	invscaling	0.586177	0.468843
150	200	lbfgs	identity	0.001	adaptive	0.586177	0.468843
150	200	lbfgs	identity	0.01	constant	0.585324	0.462908
150	200	lbfgs	identity	0.01	invscaling	0.585324	0.462908
150	200	lbfgs	identity	0.01	adaptive	0.585324	0.462908
150	200	lbfgs	logistic	0.0001	constant	1	0.51632
150	200	lbfgs	logistic	0.0001	invscaling	1	0.51632
150	200	lbfgs	logistic	0.0001	adaptive	1	0.51632
150	200	lbfgs	logistic	0.001	constant	1	0.507418
150	200	lbfgs	logistic	0.001	invscaling	1	0.507418
150	200	lbfgs	logistic	0.001	adaptive	1	0.507418
150	200	lbfgs	logistic	0.01	constant	1	0.489614
150	200	lbfgs	logistic	0.01	invscaling	1	0.489614
150	200	lbfgs	logistic	0.01	adaptive	1	0.489614
150	200	lbfgs	tanh	0.0001	constant	1	0.507418
150	200	lbfgs	tanh	0.0001	invscaling	1	0.507418
150	200	lbfgs	tanh	0.0001	adaptive	1	0.507418
150	200	lbfgs	tanh	0.001	constant	1	0.51632
150	200	lbfgs	tanh	0.001	invscaling	1	0.51632
150	200	lbfgs	tanh	0.001	adaptive	1	0.51632
150	200	lbfgs	tanh	0.01	constant	1	0.501484
150	200	lbfgs	tanh	0.01	invscaling	1	0.501484
150	200	lbfgs	tanh	0.01	adaptive	1	0.501484
150	200	lbfgs	relu	0.0001	constant	1	0.495549
150	200	lbfgs	relu	0.0001	invscaling	1	0.495549
150	200	lbfgs	relu	0.0001	adaptive	1	0.495549

150	200	lbfgs	relu	0.001	constant	1	0.48368
150	200	lbfgs	relu	0.001	invscaling	1	0.48368
150	200	lbfgs	relu	0.001	adaptive	1	0.48368
150	200	lbfgs	relu	0.01	constant	1	0.445104
150	200	lbfgs	relu	0.01	invscaling	1	0.445104
150	200	lbfgs	relu	0.01	adaptive	1	0.445104
150	200	sgd	identity	0.0001	constant	0.568259	0.48368
150	200	sgd	identity	0.0001	invscaling	0.21843	0.219585
150	200	sgd	identity	0.0001	adaptive	0.569113	0.48368
150	200	sgd	identity	0.001	constant	0.568259	0.48368
150	200	sgd	identity	0.001	invscaling	0.21843	0.219585
150	200	sgd	identity	0.001	adaptive	0.569113	0.48368
150	200	sgd	identity	0.01	constant	0.569113	0.48368
150	200	sgd	identity	0.01	invscaling	0.21843	0.219585
150	200	sgd	identity	0.01	adaptive	0.568259	0.48368
150	200	sgd	logistic	0.0001	constant	0.222696	0.222552
150	200	sgd	logistic	0.0001	invscaling	0.212457	0.210682
150	200	sgd	logistic	0.0001	adaptive	0.222696	0.222552
150	200	sgd	logistic	0.001	constant	0.222696	0.222552
150	200	sgd	logistic	0.001	invscaling	0.212457	0.210682
150	200	sgd	logistic	0.001	adaptive	0.222696	0.222552
150	200	sgd	logistic	0.01	constant	0.222696	0.222552
150	200	sgd	logistic	0.01	invscaling	0.212457	0.210682
150	200	sgd	logistic	0.01	adaptive	0.222696	0.222552
150	200	sgd	tanh	0.0001	constant	0.570819	0.480712
150	200	sgd	tanh	0.0001	invscaling	0.219283	0.225519
150	200	sgd	tanh	0.0001	adaptive	0.569113	0.480712
150	200	sgd	tanh	0.001	constant	0.570819	0.480712
150	200	sgd	tanh	0.001	invscaling	0.219283	0.225519
150	200	sgd	tanh	0.001	adaptive	0.569113	0.480712
150	200	sgd	tanh	0.01	constant	0.570819	0.480712
150	200	sgd	tanh	0.01	invscaling	0.219283	0.225519
150	200	sgd	tanh	0.01	adaptive	0.569113	0.480712
150	200	sgd	relu	0.0001	constant	0.811433	0.51632
150	200	sgd	relu	0.0001	invscaling	0.210751	0.222552
150	200	sgd	relu	0.0001	adaptive	0.809727	0.51632
150	200	sgd	relu	0.001	constant	0.81058	0.513353
150	200	sgd	relu	0.001	invscaling	0.210751	0.222552
150	200	sgd	relu	0.001	adaptive	0.813993	0.513353
150	200	sgd	relu	0.01	constant	0.793515	0.501484
150	200	sgd	relu	0.01	invscaling	0.210751	0.222552
150	200	sgd	relu	0.01	adaptive	0.792662	0.507418
150	200	adam	identity	0.0001	constant	0.583618	0.468843
150	200	adam	identity	0.0001	invscaling	0.583618	0.468843
150	200	adam	identity	0.0001	adaptive	0.583618	0.468843
150	200	adam	identity	0.001	constant	0.583618	0.468843
150	200	adam	identity	0.001	invscaling	0.583618	0.468843
150	200	adam	identity	0.001	adaptive	0.583618	0.468843
150	200	adam	identity	0.01	constant	0.583618	0.468843
150	200	adam	identity	0.01	invscaling	0.583618	0.468843
150	200	adam	identity	0.01	adaptive	0.583618	0.468843
150	200	adam	logistic	0.0001	constant	0.556314	0.445104
150	200	adam	logistic	0.0001	invscaling	0.556314	0.445104
150	200	adam	logistic	0.0001	adaptive	0.556314	0.445104

150	200	adam	logistic	0.001	constant	0.552901	0.445104
150	200	adam	logistic	0.001	invscaling	0.552901	0.445104
150	200	adam	logistic	0.001	adaptive	0.552901	0.445104
150	200	adam	logistic	0.01	constant	0.549488	0.445104
150	200	adam	logistic	0.01	invscaling	0.549488	0.445104
150	200	adam	logistic	0.01	adaptive	0.549488	0.445104
150	200	adam	tanh	0.0001	constant	0.586177	0.47181
150	200	adam	tanh	0.0001	invscaling	0.586177	0.47181
150	200	adam	tanh	0.0001	adaptive	0.586177	0.47181
150	200	adam	tanh	0.001	constant	0.586177	0.47181
150	200	adam	tanh	0.001	invscaling	0.586177	0.47181
150	200	adam	tanh	0.001	adaptive	0.586177	0.47181
150	200	adam	tanh	0.01	constant	0.587031	0.468843
150	200	adam	tanh	0.01	invscaling	0.587031	0.468843
150	200	adam	tanh	0.01	adaptive	0.587031	0.468843
150	200	adam	relu	0.0001	constant	0.999147	0.456973
150	200	adam	relu	0.0001	invscaling	0.999147	0.456973
150	200	adam	relu	0.0001	adaptive	0.999147	0.456973
150	200	adam	relu	0.001	constant	0.999147	0.474777
150	200	adam	relu	0.001	invscaling	0.999147	0.474777
150	200	adam	relu	0.001	adaptive	0.999147	0.474777
150	200	adam	relu	0.01	constant	0.976109	0.504451
150	200	adam	relu	0.01	invscaling	0.976109	0.504451
150	200	adam	relu	0.01	adaptive	0.976109	0.504451

3.2.2 Non-linear SVM using Gaussian Kernel

Training and validation accuracies for Non-linear SVM using Gaussian Kernel are as below.

Table 50: Classification Accuracies for Non-linear SVM with Gaussian Kernel

C	Gamma	Train Accuracy	Validation Accuracy
0.001	scale	0.2226962457337884	0.22255192878338279
0.001	auto	0.2226962457337884	0.22255192878338279
0.01	scale	0.2226962457337884	0.22255192878338279
0.01	auto	0.2226962457337884	0.22255192878338279
0.1	scale	0.590443686006826	0.42729970326409494
0.1	auto	0.590443686006826	0.42729970326409494
1.0	scale	0.7764505119453925	0.5400593471810089
1.0	auto	0.7764505119453925	0.5400593471810089
10.0	scale	0.9360068259385665	0.4807121661721068
10.0	auto	0.9360068259385665	0.4807121661721068
100.0	scale	0.9991467576791809	0.4421364985163205
100.0	auto	0.9991467576791809	0.4421364985163205

3.3 Best Model, and Test accuracy

The best model amongst the models available, as well as its performance on the test dataset is as given below.

3.3.1 Multilayer Feedforward Neural Network

The best model for this case is the one with the following hyperparameters.

Neurons in hidden layer 1 = **100**

Neurons in hidden layer 2 = **150**

Solver = **adam**

Activation Function = **tanh**

Alpha = **0.001**

This gives a test accuracy of **54.30%**

3.3.2 Non-linear SVM using Gaussian Kernel

The best model for this case is the one with the following hyperparameters.

C = **1.0**

Gamma = **scale**

This gives a test accuracy of **54.01%**

3.4 Confusion Matrix for the best model

The confusion matrix that is obtained for the best model, as determined by the performance on the validation dataset, is given below for each of the individual cases.

3.4.1 Multilayer Feedforward Neural Network

The confusion matrices for the best model of the MLFFNN Classifier with 2 hidden layers i.e., with (100,150) hidden neurons, adam solver, tanh activation function and alpha = 0.001 are given below.

Table 51: Confusion Matrix for Training Data using MLFFNN Classifier with 2 hidden layers

Training Data					
Class	Coast	Forest	Highway	Mountain	Tall Building
Coast	251	0	0	0	0
Forest	0	229	0	0	0
Highway	0	0	182	0	0
Mountain	0	0	0	261	0
Tall Building	0	0	0	0	249

Table 52: Confusion Matrix for Validation Data using MLFFNN Classifier with 2 hidden layers

Validation Data					
Class	Coast	Forest	Highway	Mountain	Tall Building
Coast	36	7	7	15	8
Forest	0	57	2	2	5
Highway	6	0	27	7	12
Mountain	10	13	5	32	15
Tall Building	13	4	2	21	31

3.4.2 Non-linear SVM using Gaussian Kernel

The confusion matrices for the best model i.e., with C = 1.0 and gamma as scale, are given below.

Table 53: Confusion Matrix for Training Data using Non-linear SVM using Gaussian Kernel

Training Data					
Class	Coast	Forest	Highway	Mountain	Tall Building
Coast	167	7	1	51	25
Forest	5	205	0	6	13
Highway	6	7	159	3	7
Mountain	18	9	2	194	38
Tall Building	13	9	0	42	185

Table 54: Confusion Matrix for Validation Data using Non-linear SVM using Gaussian Kernel

Validation Data					
Class	Coast	Forest	Highway	Mountain	Tall Building
Coast	33	8	4	19	9
Forest	1	59	1	1	4
Highway	4	0	28	7	13
Mountain	14	13	4	31	13
Tall Building	10	9	3	18	31

3.5 Observations

Some key observations after looking at the results are as follows:

3.5.1 Multilayer Feedforward Neural Network

- The stochastic gradient-based optimizer (adam solver) is giving best performance on both train and validation datasets. As the ‘adam’ solver works pretty well on relatively large datasets.
- The highest validation accuracy is obtained with 'adam' solver, 'tanh' activation function and regularization term (L2 penalty parameter) of 0.001.
- 'lbfgs' converges fast and performs better on small datasets. The 'lbfgs' solver on the given dataset gives higest training accuracy but lesser validation accuracy when compared to 'adam' and hence indicating overfitting.
- The 'sdg' solver gives least accuracy on both train and test datasets.
- Learning rate is only used with solver 'sdg'. 'invscaling' performs worst when compared to 'constant' and 'adaptive', which gives almost equal accuracies. As ‘invscaling’ gradually decreases the learning rate at each time step ‘t’, too small learning rate might be responsible for the model's inability to learn. while the learning rate is constant for 'adaptive' as long as the training loss is decreasing.
- The MLFFNN classifier performance decreases after normalization and standardization of given dataset.

3.5.2 Non-linear SVM using Gaussian Kernel

- The validation accuracy obtained for SVM and MLFFNN is close.
- The SVM classifier performance increases after normalization and standardization of given dataset. The validation dataset is transformed based on training dataset.
- The model performance is better at higher values of regularization parameters. The most optimal value of regularization parameter obtained is 1.0.
- The gamma by default in SVM is 1 / (n_features * X.var()). Scaling by standard deviation gives the best performance here which is sensible when the standard deviation of data is not 1.