Indian Institute Of Technology Madras

CS5691 - Pattern Recognition and Machine Learning

Assignment 3 Report

Team 5

Jeswant Krishna K - AE17B030

Advait Walsangikar - AE17B101

Ananthu Nair S. - AE17B109

13th May 2021

Contents

1 Dataset 1a

1.1 Problem Definition

A 2-Dimensional linearly separable Artificial Dataset that is to be used for static pattern classification using Perceptrons for every pairs of classes, Multi-layer feed forward neural network and SVM classifier for each pair of classes.

1.2 Methods Used

Perceptrons

Perceptron was used in this case for supervised learning of binary classifiers and it is basically a single-layer neural network. We train a perceptron for each pair of classes and then they are used to predict the class label of the test data. The final label is the one with the highest number of votes.

Multi-layer Feed Forward Neural Network

MLFFNN is basically similar to a perceptron but has multiple layers. The inner layers are called as hidden layers and MLFFNN has one input layer and one output layer. Typically the MLFFNN layers are expected to learn more complex features as it progresses. Every layer has an activation function associated with the layer's output aiding in capturing the non-linearity in the data. Typically for classification purposes the output layer is softmaxed.

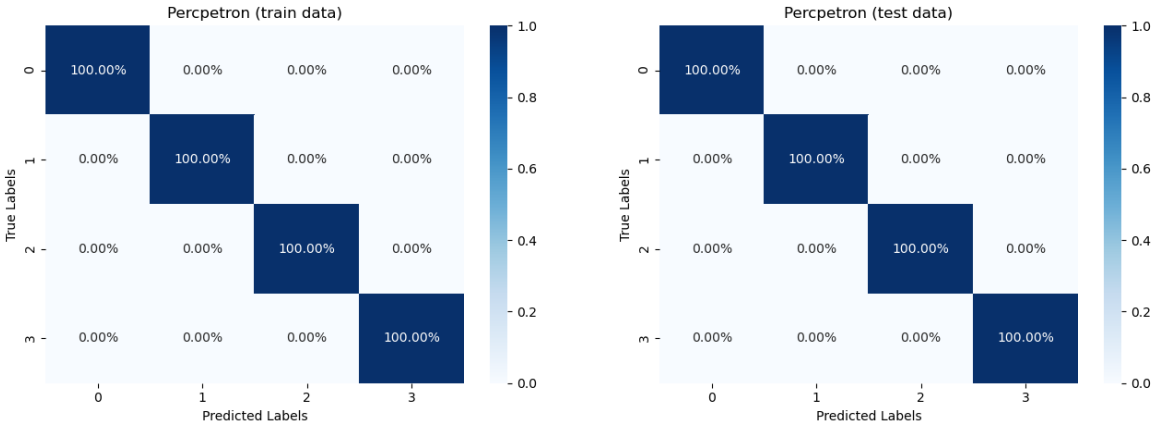
Linear SVM Classifier

The SVM is also a supervised algorithm which constructs a hyperplane or a set of hyperplanes which has largest distance to the nearest training-data point of any class (called as margin). Since its a linear SVM the hyperplane is also linear in nature. We train a SVM classifier for each pair of classes and then they are used to predict the class label of the test data. The final label is the one with the highest number of votes.

1.3 Results

Perceptron using 1 vs 1 approach

No hyperparameter tuning required for Perceptron.



(a) Training Data (b) Test Data

Figure 1: Confusion matrix for Perceptron based classification using 1 vs 1 method , for both the training data and the test data.

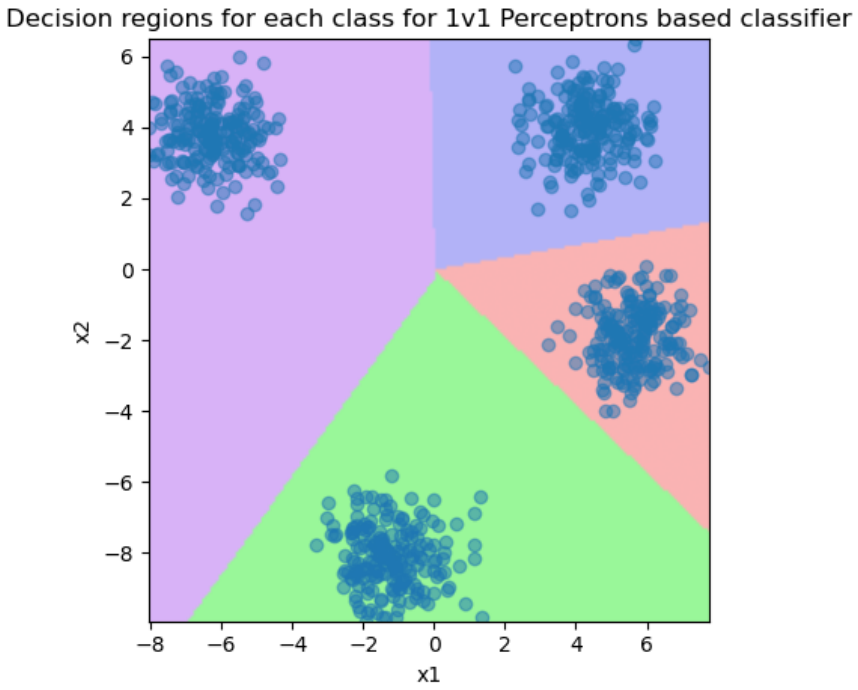


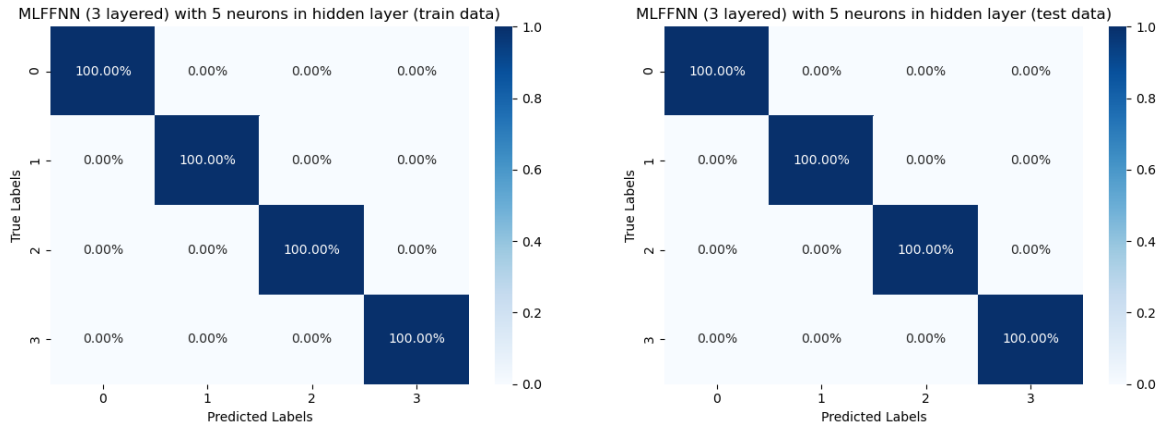
Figure 2: Decision Region plot for the perceptron classifier.

Multi-layer Feed Forward Neural Network

The Multilayer Feed Forward Neural Network with one hidden layer was run using various values of the hyperparameter, namely the number of nodes in the hidden layer.

# Nodes	Accuracies	
	Training Data	Test Data
1	50.0%	50.0%
2	78.88%	75.0%
3	95.75%	94.17%
4	99%	100%
5	100%	100%

Table 1: Table showing the Training Data and test Data accuracies for the Multi-layer Feed Forward Neural Network with various values of the number of nodes in the hidden layer



(a) Training Data

(b) Test Data

Figure 3: Confusion matrix for Multi-layer Feed Forward Neural Network with the best configuration of hyperparameters, for both the training data and the test data. Here, the number of nodes was chosen as 5, since that gave the highest accuracy among all the models.

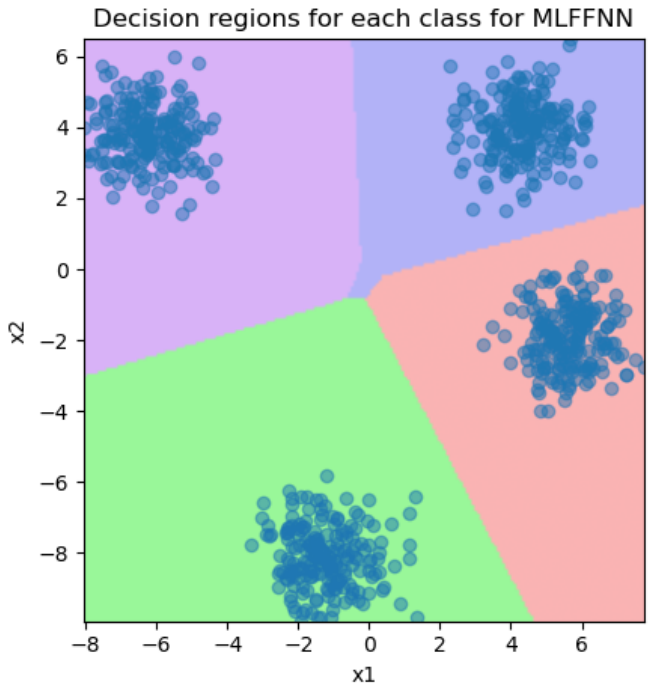
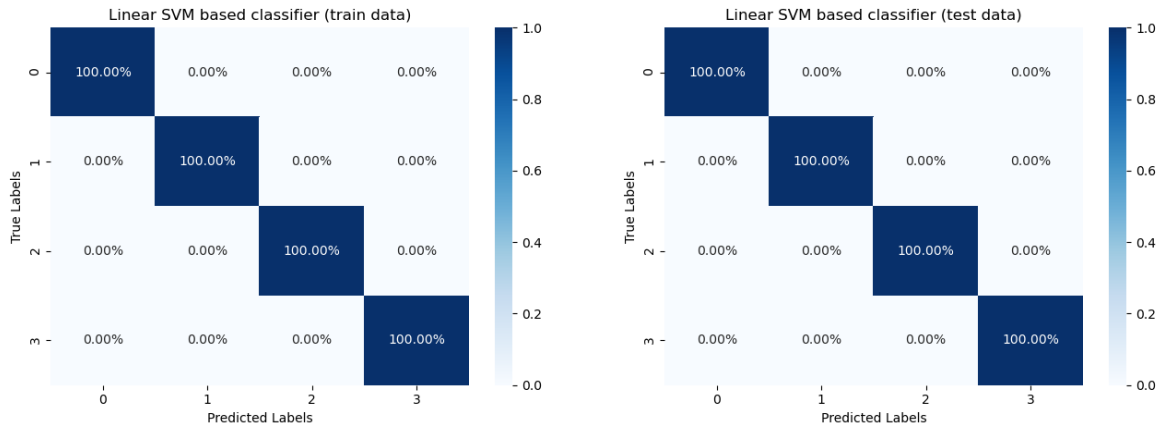


Figure 4: Decision Region plot for the best configuration of hyperparameters for MLFFNN. Here, the number of nodes was chosen as 5, since it gave the highest accuracy.

Linear SVM Classifier 1 vs 1 approach

Hyperparameter is C which controls regularization and since this particular dataset is linearly separable, hyperparameter tuning not applicable in this case.



(a) Training Data

(b) Test Data

Figure 5: Confusion matrix for Linear SVM Classifier for each pairs of classes, for both the training data and the test data.

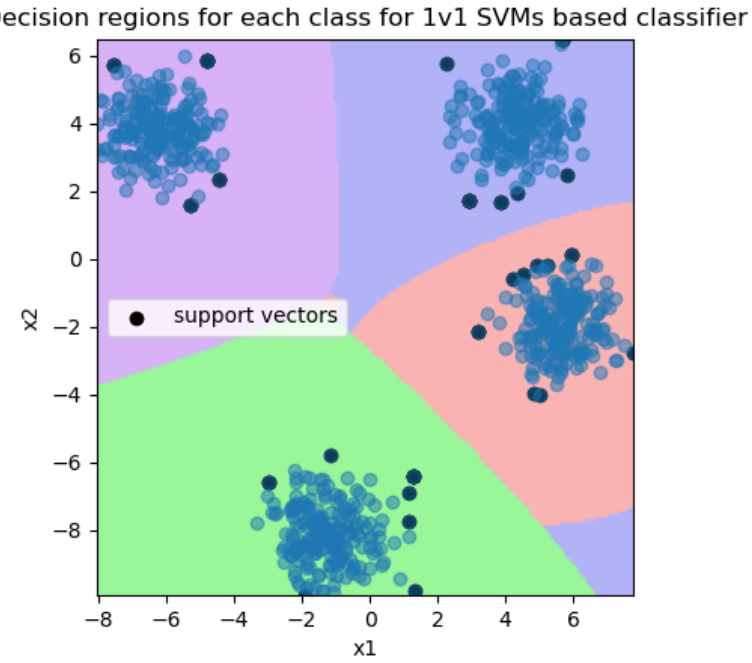


Figure 6: Decision Region plot for Linear SVM Classifier.

1.4 Inferences

1v1 Perceptrons based Classifier

- Performs really well on linearly separable datasets (100% accuracy on both train and test sets).
- The decision boundaries are straight lines.

Multi-layer Feed Forward Neural Network

- 100% classification accuracies on both train and test data.
- Non-linear decision boundaries. Look approximately linear due to well separated classes in the dataset.
- Able to perform multiclass classification without needing external methods.
- Looking at the plots for single node perceptron(Figure 2), and multi-layered one(Figure 4), the decision boundaries look better for the multilayered perceptron.

1v1 Linear SVMs based Classifier

- 100% accuracy for linearly separable dataset.
- Not expected to work well for non-linear datasets.

- Decision boundaries are non-linear due to randomness introduced due to label determination using majority vote.
- For binary classification, the boundaries are expected to be linear.

2 Dataset 1b

2.1 Problem Definition

A 2-Dimensional non-linearly separable Artificial Dataset that is to be used for static pattern classification using Multi-layer feed forward neural network and SVM classifier for each pair of classes.

2.2 Methods Used

Multi-layer Feed Forward Neural Network

MLFFNN is basically similar to a perceptron but has multiple layers. The inner layers are called as hidden layers and MLFFNN has one input layer and one output layer. Typically the MLFFNN layers are expected to learn more complex features as it progresses. Every layer has an activation function associated with the layer's output aiding in capturing the non-linearity in the data. Typically for classification purposes the output layer is softmaxed.

Polynomial SVM Classifier with 1 vs Rest approach

Before in the linear SVM the hyperplanes constructed are linear in nature and it is good for the case when the data is linearly separable. But in the cases when it is not linearly separable we can use a kernel function which maps the original space to a higher dimensional space in which the data may be linearly separable. If the nature of this kernel function is Polynomial then its a polynomial SVM classifier. We train a SVM classifier for each pair of classes and then they are used to predict the class label of the test data. The final label is the one with the highest probability of belonging to a class. The probabilities are obtained when the svm classifiers are trained with probability attribute turned to "True".

Gaussian SVM Classifier with 1 vs Rest approach

Similar to the case above the kernel here is a RBF(Radial Basis Function) kernel then its called a Gaussian SVM classifier. We train a SVM classifier for each pair of classes and then they are used to predict the class label of the test data. The label is determined the same way as above using probabilities of belongingness to a particular class.

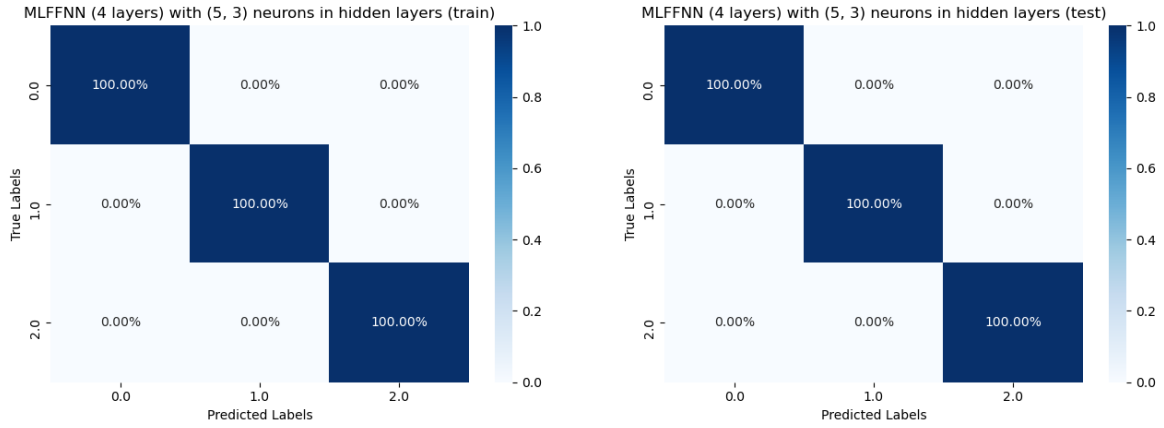
2.3 Results

Multi-layer Feed Forward Neural Network

The Multilayer Feed Forward Neural Network with two hidden layer was run using various values of the hyperparameters, namely the number of nodes in each of the hidden layer.

Number of Nodes	**Accuracies**	
	Training Data	Test Data
(2, 2)	76.83%	78.88%
(3, 3)	71.66%	64.44%
(4, 4)	99.83%	100%
(5, 3)	100%	100%

Table 2: Table showing the Training Data and test Data accuracies for the Multi-layer Feed Forward Neural Network with various values of the number of nodes in the hidden layers



(a) Training Data

(b) Test Data

Figure 7: Confusion matrix for Multi-layer Feed Forward Neural Network with the best configuration of hidden layer sizes, for both the training data and the test data. Here, the number of nodes was chosen as (5, 3), since that gave the highest accuracy among all the models.

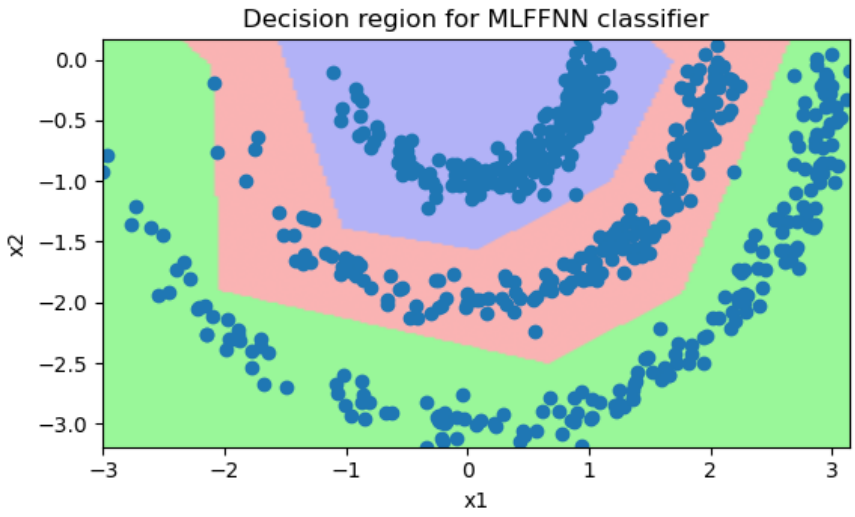


Figure 8: Decision Region plot for the best configuration of hyperparameters for MLFFNN. Here, the number of nodes was chosen as (5, 3), since it gave the highest accuracy.

The outputs of the hidden layer and the output layer layer nodes are visualized in the following graphs.

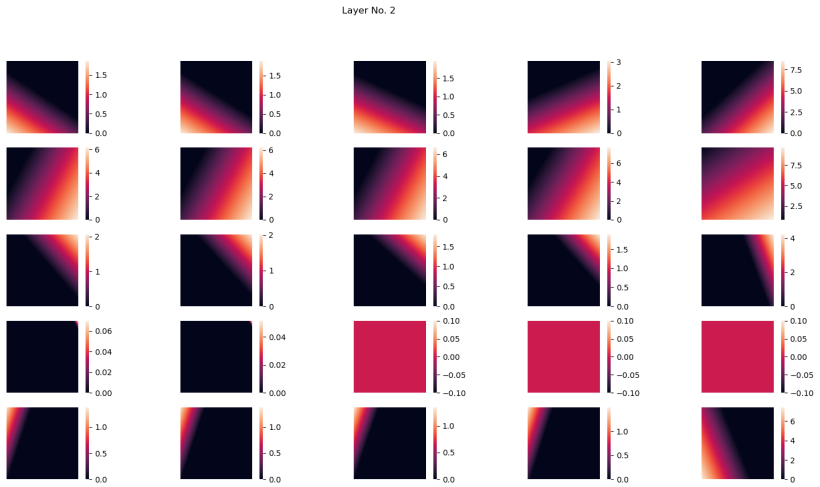


Figure 9: A plot showing the outputs of the hidden layer 1 at each of the following epochs, 1, 5, 20, 100 and after convergence

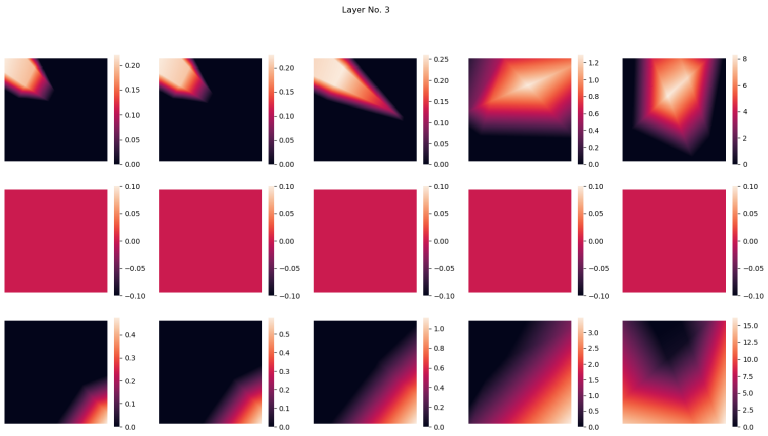


Figure 10: A plot showing the outputs of the hidden layer 2 at each of the following epochs, 1, 5, 20, 100 and after convergence

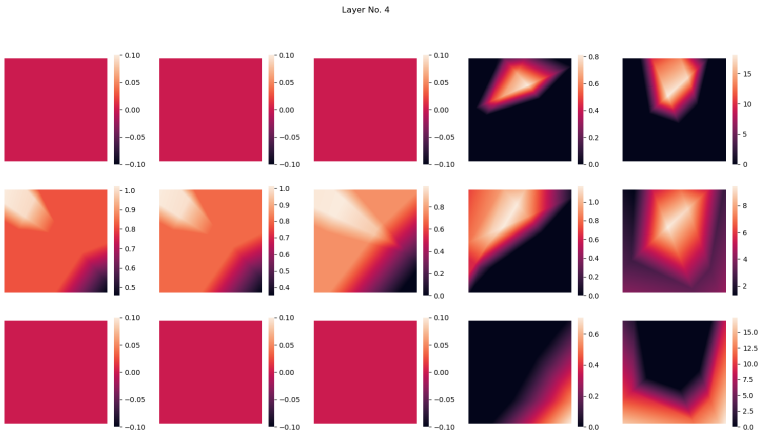


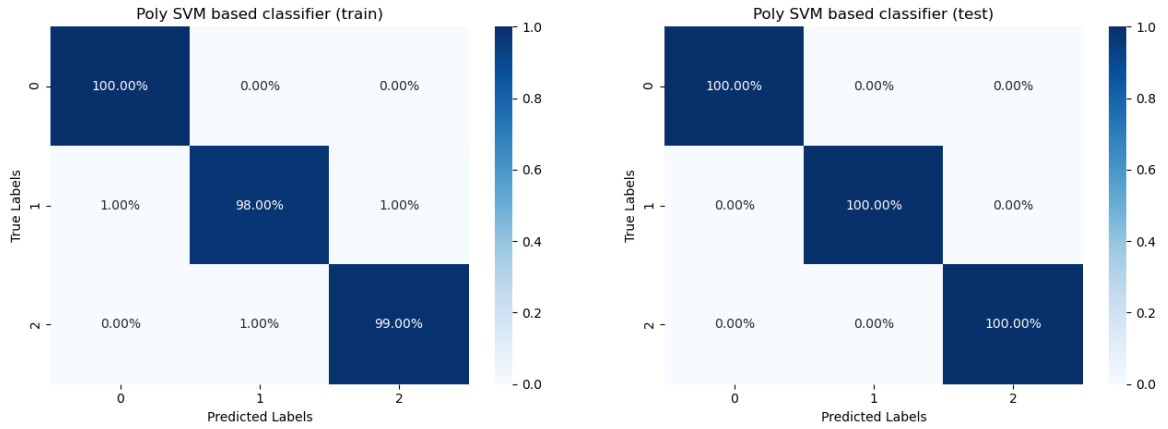
Figure 11: A plot showing the outputs of the output layer at each of the following epochs, 1, 5, 20, 100 and after convergence

Polynomial SVM Classifier

The polynomial SVM classifier for each pair of classes was used to classify the train and test dataset with various values of the hyperparameter, namely the Regularisation parameter, C (It is the inverse of regularisation).

C	Accuracies	
	Training Data	Test Data
0.001	76.66%	73.33%
0.01	88.00%	88.88%
0.1	97.50%	98.88%
1.0	97.83%	97.77%
10	98.00%	98.88%
100	98.16%	98.88%
1000	99.00%	100%

Table 3: Table showing the Training Data and test Data accuracies for the Polynomial SVM Classifier with various values of the regularization parameter.



(a) Training Data

(b) Test Data

Figure 12: Confusion matrix for Polynomial SVM Classifier for each pairs of classes, with the best configuration of the hyperparameter, for both the training data and the test data. The hyperparamter was chosen as $C = 1000$ since this configuration had the best accuracy

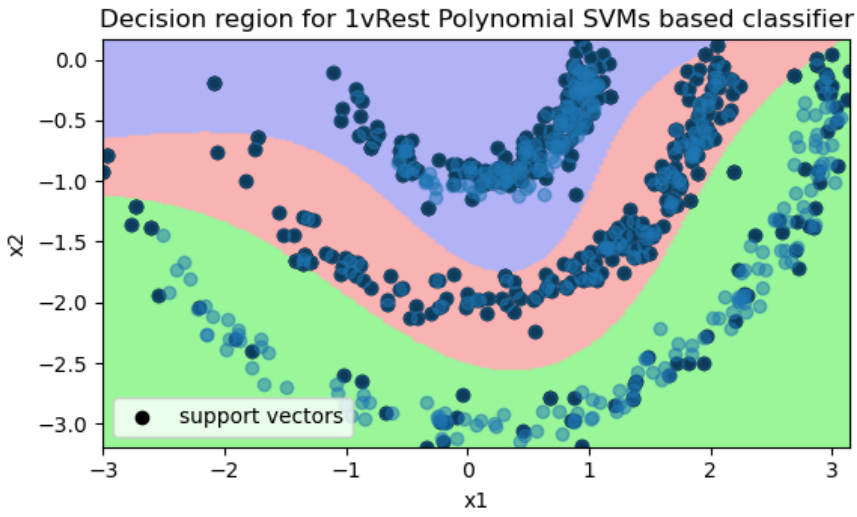


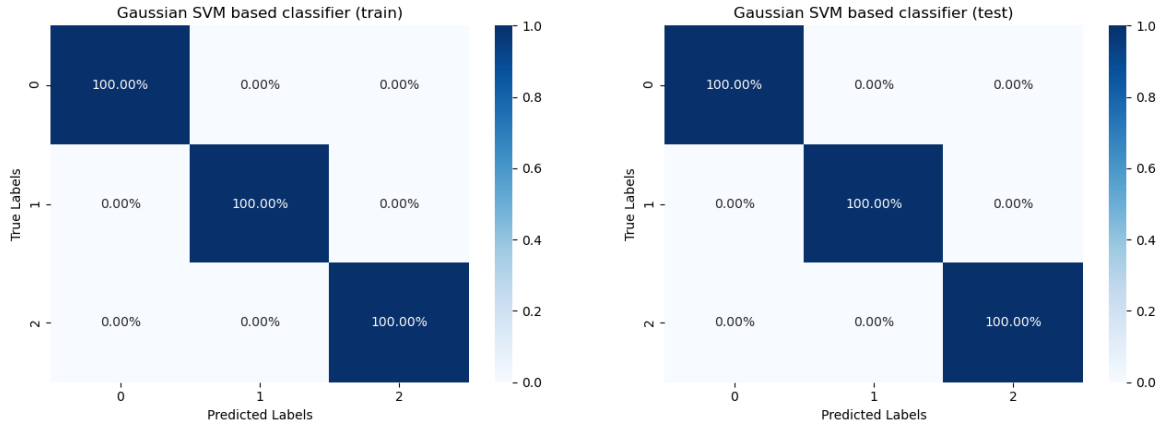
Figure 13: Decision Region plot for the best configuration of hyperparameters for polynomial SVM. Here, the hyperparameter chosen was $C = 1000$, since it gave the highest accuracy.

Gaussian SVM Classifier

The gaussian SVM classifier for each pair of classes was used to classify the train and test dataset with various values of the hyperparameter, namely the Regularisation parameter, C (It is the inverse of regularisation).

C	**Accuracies**	
	Training Data	Test Data
0.001	97.66%	100%
0.01	98.00%	100%
0.1	99.16%	100%
1.0	100%	100%
10	100%	100%

Table 4: Table showing the Training Data and test Data accuracies for the Gaussian SVM Classifier with various values of the regularization parameter.



(a) Training Data (b) Test Data

Figure 14: Confusion matrix for Gaussian SVM Classifier for each pairs of classes, with the best configuration of the hyperparameter, for both the training data and the test data. The hyperparamter was chosen as $C = 1.0$ since this configuration had the best accuracy

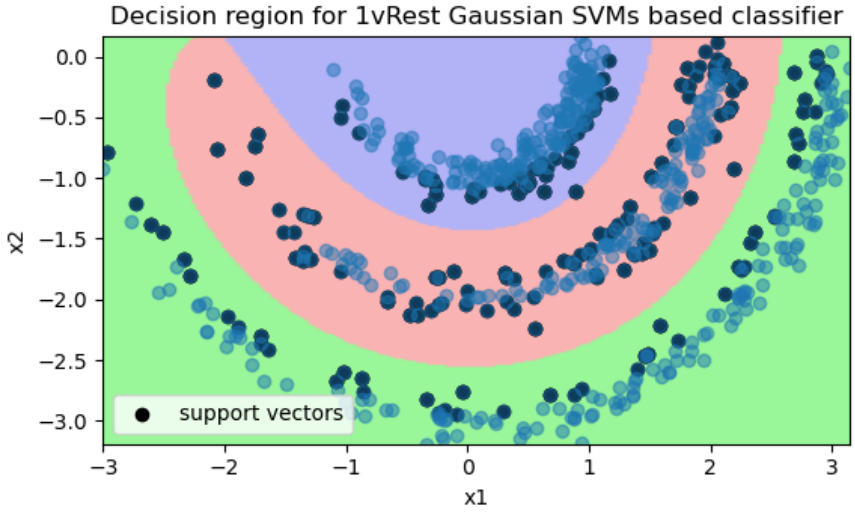


Figure 15: Decision Region plot for the best configuration of hyperparameters for Gaussian SVM. Here, the hyperparameter chosen was $C = 1.0$, since it gave the highest accuracy.

2.4 Inferences

MLFFNN

- Decision regions look like a curve made up of straight lines (not smooth due to less no. of nodes in hidden layers).
- 100% classification accuracies for test and train data for (5, 3) nodes in the hidden layers.
- relU activation for hidden layers took the least number of epochs to converge and gave the best results.
- Looking at Figures 9, 10, 11, we can see that the outputs of the individual nodes seem to be learning parts of classification for the whole model and the learnt patterns get better with each epoch towards convergence.
- The heatmap for nodes in the output layer (after convergence) show which class they prefer and thus "respond/react" to.
- The regularization parameter did not make much difference due to the dataset being separable (even though non-linear).

SVM

- Gaussian RBF based SVMs(100% train and test accuracy) performed much better than Polynomial Kernel based SVMs.
- Gaussian SVMs provide the best looking decision regions for the classes.
- Support vectors are closer to the boundary (expected since they are in a way defining the boundaries).

3 Dataset 2a

3.1 Problem Definition

A Real-World Image Dataset that is to be used for static pattern classification using Multi-layer feed forward neural network and SVM classifier for each pair of classes.

3.2 Results

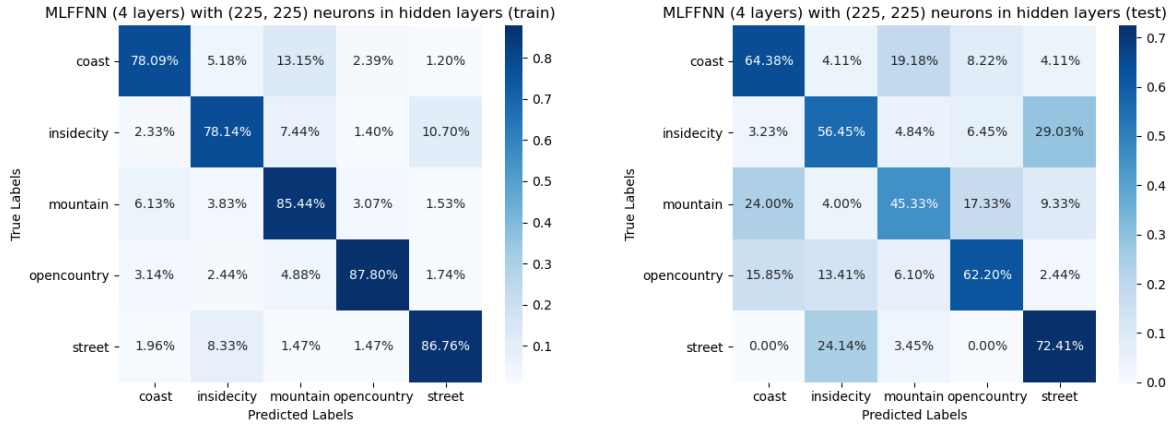
Multi-layer Feed Forward Neural Network

The Multilayer Feed Forward Neural Network with two hidden layer was run using various values of the hyperparameters, namely the number of nodes in each of the hidden layer.

Number of Nodes	**Accuracies**	
	Training Data	Test Data
(50, 50)	68.31%	56.57%
(75, 75)	70.94%	56.00%
(100, 100)	74.96%	58.57%
(125, 125)	76.68%	59.14%
(150, 150)	78.89%	58.57%
(175, 175)	80.05%	57.71%
(200, 200)	81.61%	58.57%
(225, 225)	83.42%	59.71%
(250, 250)	86.04%	57.99%
(275, 275)	86.62%	57.71%
(500, 500)	93.59%	57.71%
(1000,1000)	97.04%	57.14%

Table 5: Table showing the Training Data and test Data accuracies for the Multi-layer Feed Forward Neural Network with various values of the number of nodes in the hidden layers

From the table it can be seen that as the number of nodes are increased more than (225, 225), there is a huge increase in the train data accuracy, but also the test data accuracy falls. This can be seen as a sign of overfitting. Hence the case of number of nodes = (225, 225) is chosen as the best set of hyperparameter for this case.



(a) Training Data

(b) Test Data

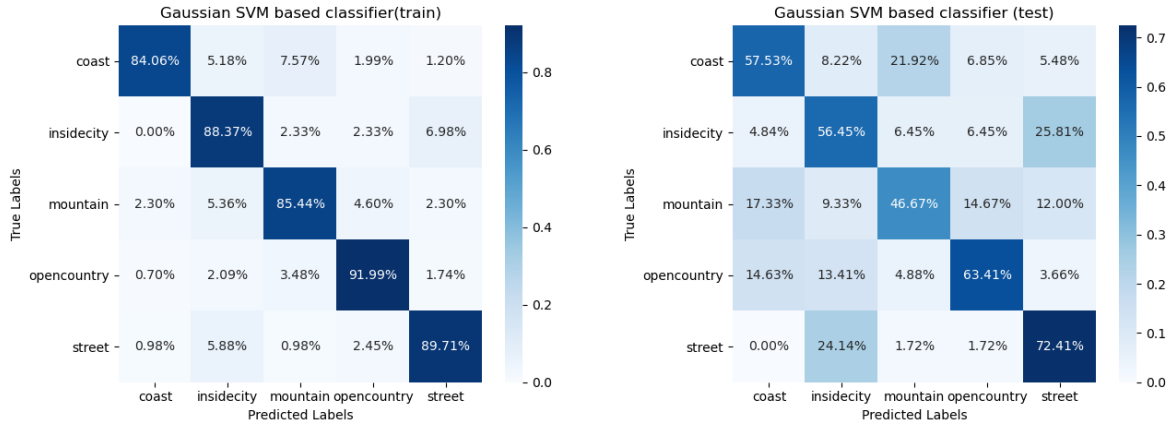
Figure 16: Confusion matrix for Multi-layer Feed Forward Neural Network with the best configuration of hyperparameters, for both the training data and the test data. Here, the number of nodes was chosen as (225, 225), since that gave the highest accuracy and also converged, among all the models.

Gaussian SVM Classifier

The gaussian SVM classifier for each pair of classes was used to classify the train and test dataset with various values of the hyperparameter, namely the Regularisation parameter, C (It is the inverse of regularisation).

Cs	**Accuracies**	
	Training Data	Test Data
0.1	71.26%	56.00%
1	76.84%	57.42%
2	82.10%	58.57%
3	85.38%	58.86%
4	87.93%	58.86%
5	89.33%	58.28%
6	90.23%	58.57%
7	91.05%	57.99%
8	91.87%	57.43%

Table 6: Table showing the Training Data and test Data accuracies for the Multi-layer Feed Forward Neural Network with various values of the number of nodes in the hidden layers



(a) Training Data

(b) Test Data

Figure 17: Confusion matrix for Gaussian SVM Classifier for each pairs of classes, with the best configuration of the hyperparameter, for both the training data and the test data. The hyperparameter was chosen as $C = 4.0$ since this configuration had the best test data accuracy in its neighbourhood

3.3 Inferences

- MLFFNN and SVM both perform similarly in terms of test and train accuracies.
- The confusion matrices suggest that that pairs such as (street, insidecity), (coast, mountain), (mountain, opencountry) are frequently confused which can be expected given how similar the features in these pairs can be.