Kernel Methods for Pattern Analysis
Assignment Solution 2

Rachit Garg CS14B050 | Keerthana S CS13B041 | Sphoorti K CS13B042

January 25, 2019

1 Linearly separable data set

1.1 SVM

Linear kernel

Parameter estimation: Kernel gram matrix is used to choose kernel parameters.Since linear kernel does not have any kernel parameters,the figure looks like:

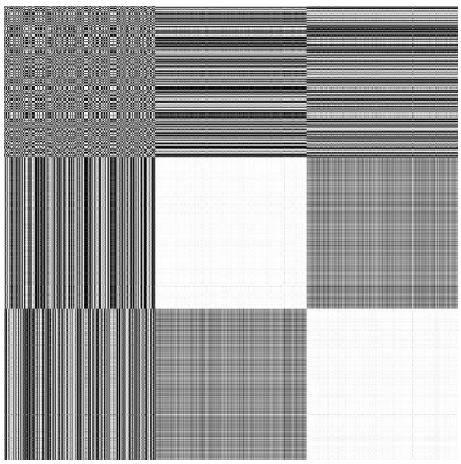

Figure 1: Grey level image of kernel gram matrix for dataset 1a training set using linear kernel

Values in kernel gram matrices are values of normalised kernel function for all pair of training examples which is measure of similarity of points.In the above figure we see that the region near diagonal has higher values since we expect examples of same class to be similar,on the other hand,region away from diagonal is grey/black since they belong to different class.

The parameters to be estimated in linear kernel is "c" which is tradeoff parameter.It is done by doing a search for "c"(in the range of $2^{-}10$ to $2^{1}0$)which leads to maximum classification accuracy.The plot for accuracy on validation set vs log(C) looks like:

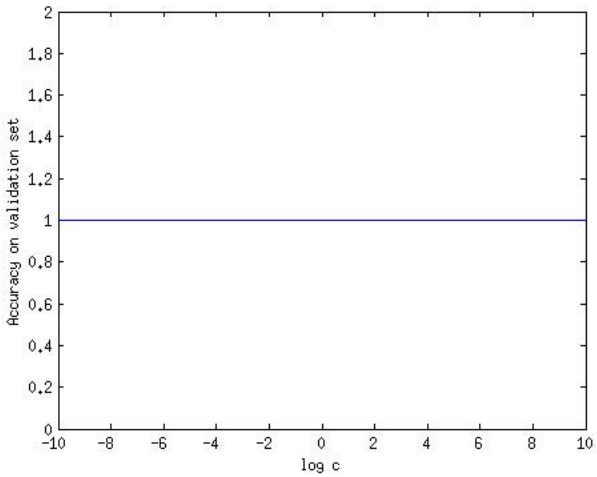

Figure 2: Figure showing accuracy for validation set for different values of c

We see that accuracy=1 for all values of c.The reason for high accuracy is because the dataset is linearly separable and linear kernel is enough to fit a hyperplane.

Decision region plot:
The decision region plot for c=1 looks like:

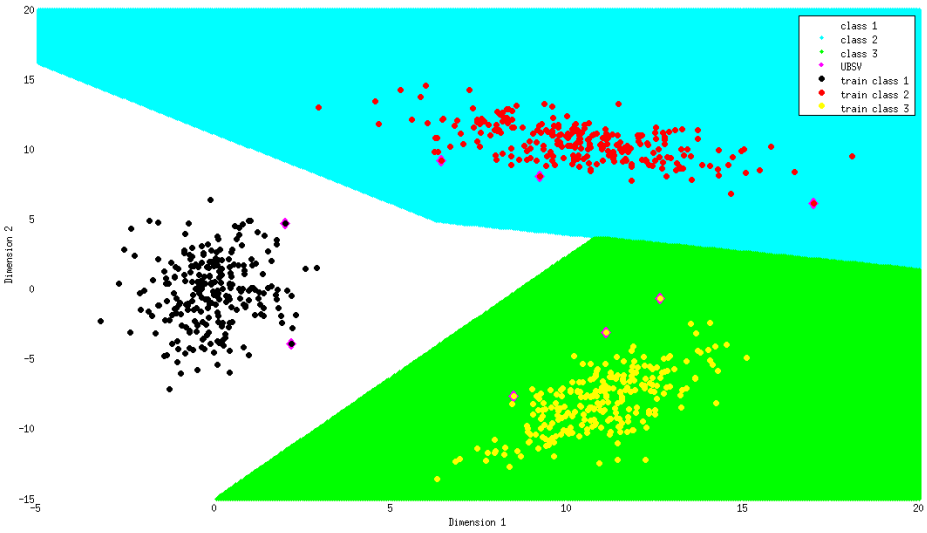

Figure 3: Figure showing decision region plot for dataset 1a using linear kernel,c=1

From the figure we see that, the decision boundaries separating each pair of classes are lines(linear),and the support vectors obtained by the training model are only unbounded i.e they lie on the margin because the points from each pair of classes are well separated.Decreasing c value may lead to more support vectors(especially bounded),because it gives more weight for margin term(maximising),which may end up having more points within margin.

Confusion matrix:
The confusion matrix for test set look like:

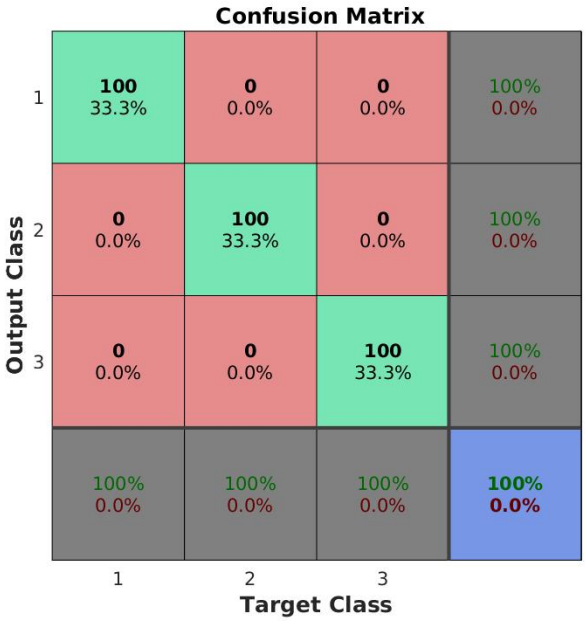

Figure 4: Figure showing confusion matrix for test set

We see that $accuracy = 100\%$ on test set.

Polynomial kernel

Parameter estimation:
The kernel parameters for polynomial function are p,a and b which are chosen using kernel gram matrix.We find that any $p >= 1$,gave kernel gram matrix as expected(block diagonal matrix),hence we choose least "p" i.e p=1 and we see that b=0 is enough(2 dimensions in kernel space) which is enough to fir bivariate data.Hence the value of "a" chosen doesn't have any effect on normalised kernel function values,hence a is chosen to be 1.

The kernel gram matrix for above configuration looks like:

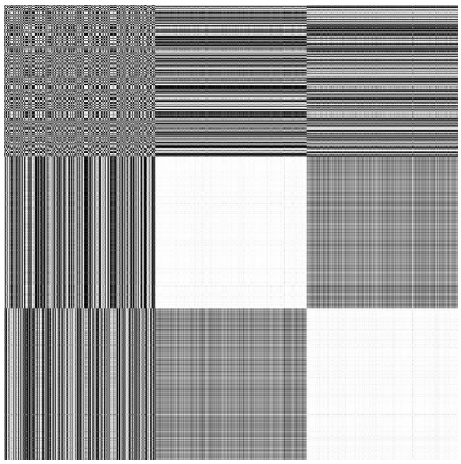

Figure 5: Figure showing grey level image of kernel gram matrix for kernel parameters a=1,b=0,p=1

We see that it is the same one as obtained in linear kernel.
For choosing c,we see accuracy on validation set which is found to be 1 for any value of "c".Hence

the below plots are plotted for c=1.
Since the model used is same the decision region plots and confusion matrix will be same
Decision region plot

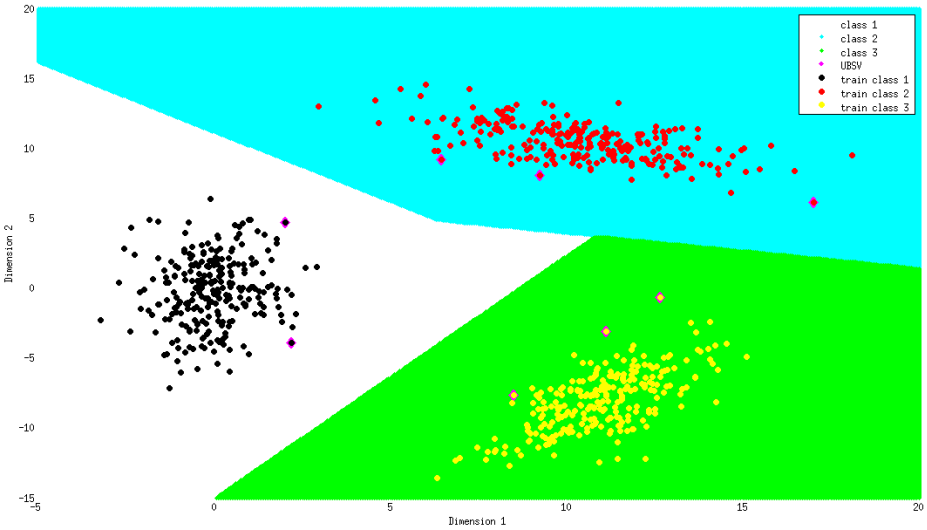

Figure 6: Figure showing decision region plot for dataset 1a polynomial kernel,c=1

Confusion matrix
The confusion matrix for dataset 1a using polynomial kernel and parameters a=1,b=0,p=1 ,c=1 looks like:

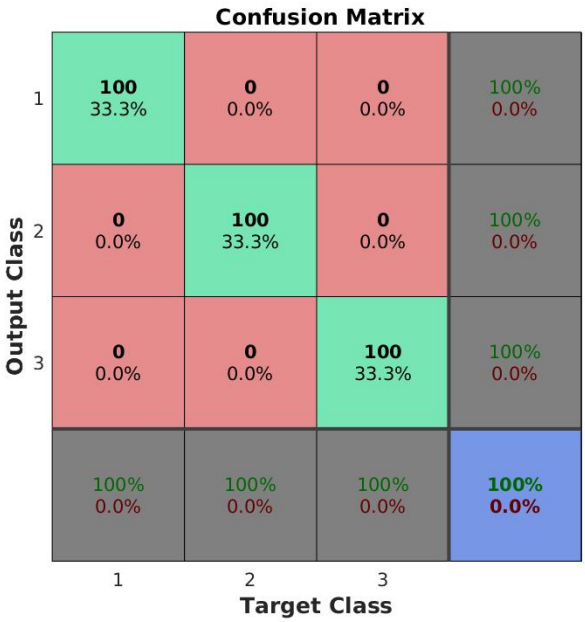

Figure 7: Figure showing confusion matrix for test set for dataset 1a polynomial kernel,c=1

We see that $accuracy = 100\%$ for test set using polynomial kernel.

Gaussian kernel

Parameter estimation:
The kernel parameter to be estimated for gaussian kernel is σ where kernel function in gaussian kernel is given by:
$K(x_m, x_n) = e^{-||x_m - x_n||^2/\sigma}$ where σ is width parameter.
The σ value is chosen based on kernel gram matrix and the value obtained is σ=100.

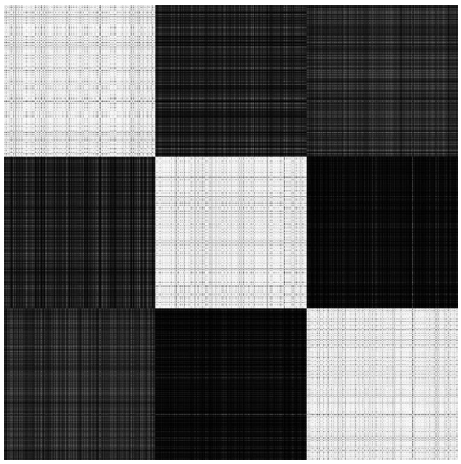

Figure 8: Grey level image of kernel gram matrix for dataset 1a training set using gaussian kernel for $\sigma = 100$

From the figure of kernel gram matrix ,we see that kernel function values are much more high for examples of same class(especially class 1) and less when examples between classes are considered.This is indicated by more darker /black areas off the block diagonal and white area in block of first class.This is because we see that class1 examples are distributed in a circular fashion where cosine similarity has wider cone hence cosine similarity values vary whereas in gaussian kernel it takes euclidean distance into account.Hence the shape of data can also influence the choice of kernel to be used.
To find the hyperparameter.error on validation set is plotted across logc.

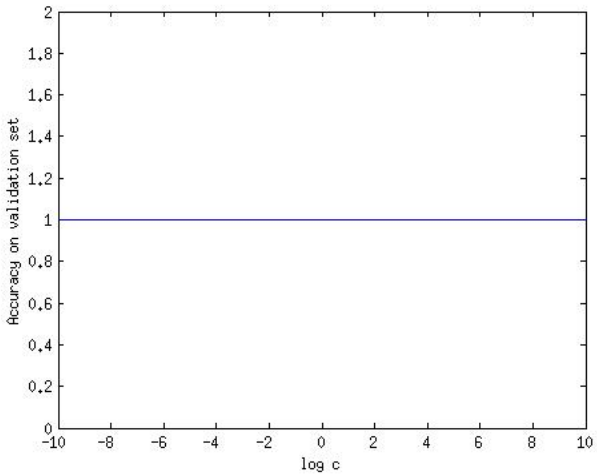

Figure 9: Figure showing accuracy for validation set for different values of c

We observe that accuracy is 100% for all values of c.Hence decision region plot is plotted for c=1.

Decision region plot:

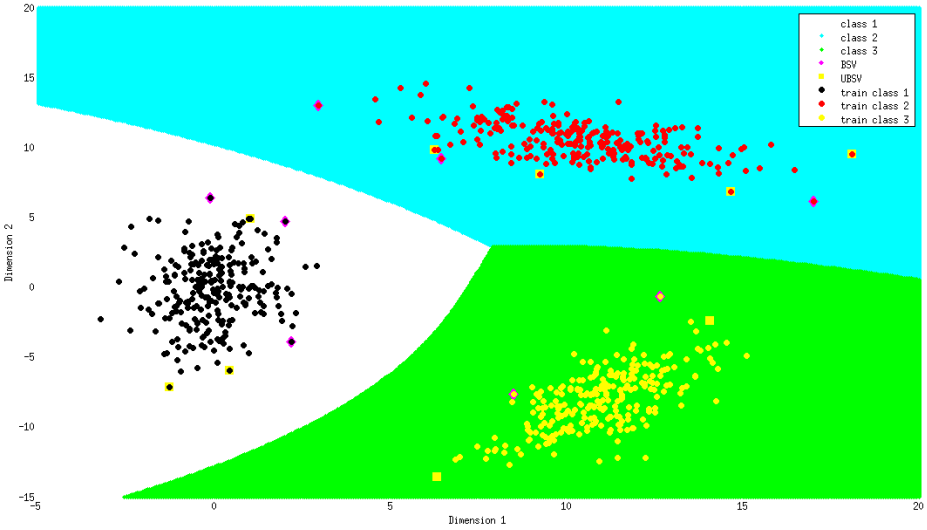

Figure 10: Figure showing decision region plot for dataset 1a gaussian kernel for $\sigma = 100$,c=1

The shape of decision boundary is non linear in case of gaussian kernel unlike linear or polynomial kernel where it was linear.We also see the difference in number of support vectors in linear and gaussian kernel for same c.

Confusion matrix:

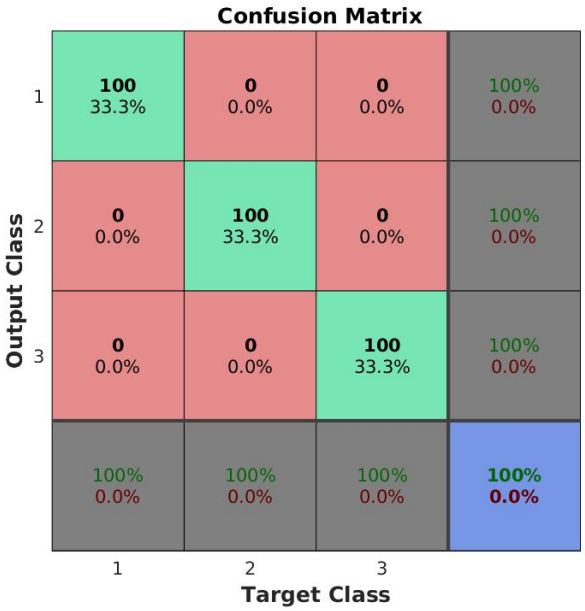

Figure 11: Figure showing confusion matrix for test set for dataset 1a gaussian kernel for $\sigma = 100$,c=1

The accuracy is observed to be 100% for test data using gaussian kernel.

1.2 Logistic Regression

In logistic regression the posterior probabilities of the class are ensured to be between 0 and 1 and the sum over all classes is 1 by using a softmax acting on a linear function. Cross entropy was used as the error function. And the python library sci-kit learn was used for this implementation.

Parameter estimation: Logistic Regression was done on the dataset with polynomial basis function. The parameter that was estimated was degree of the basis expansion. We tried varying the degree between 0 and 3, since this dataset is a linearly separable dataset, we got perfect results on the test set with degree 1. Here we present our findings.

The error on the validation set vs the degree was plot and that was used to select what is the best model.

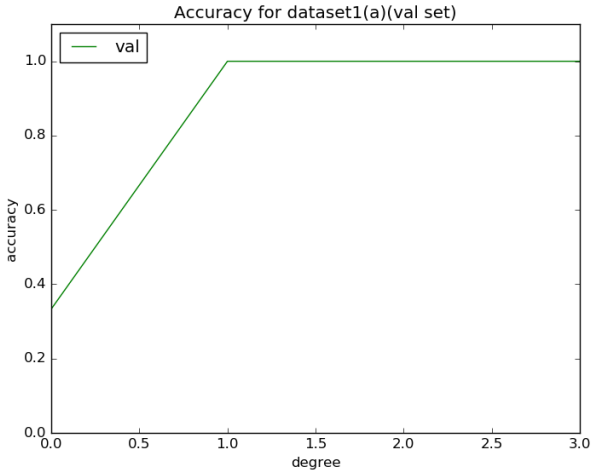

Figure 12: Figure showing accuracy for validation set for different values of degree

We see that accuracy=1 for all values of degree greater than equal to 1.The reason for high accuracy is because the dataset is linearly separable and basis expansion is not needed.

Decision region plot:
The decision region plot for degree=1 looks like:

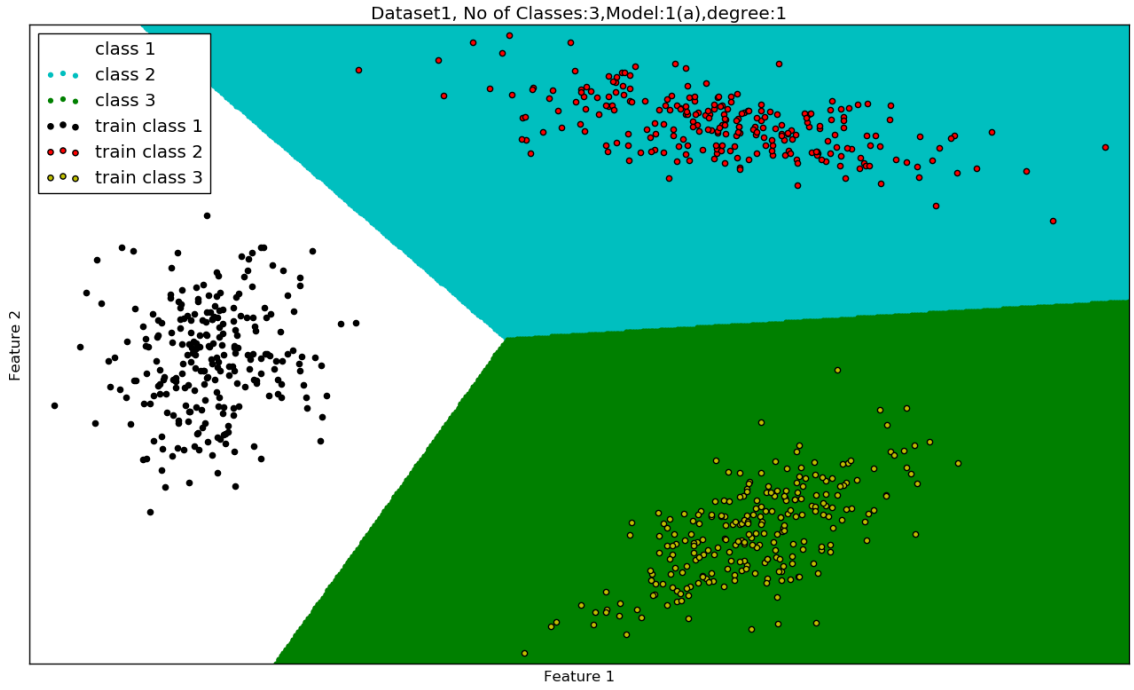

Figure 13: Figure showing decision region plot for dataset 1a using logistic regression with polynomial expansion of degree equal to 1

From the figure we see that, the decision boundaries separating each pair of classes are lines(linear),and the points in the train set are all classified correctly.

Confusion matrix:

The confusion matrix for test set look like:

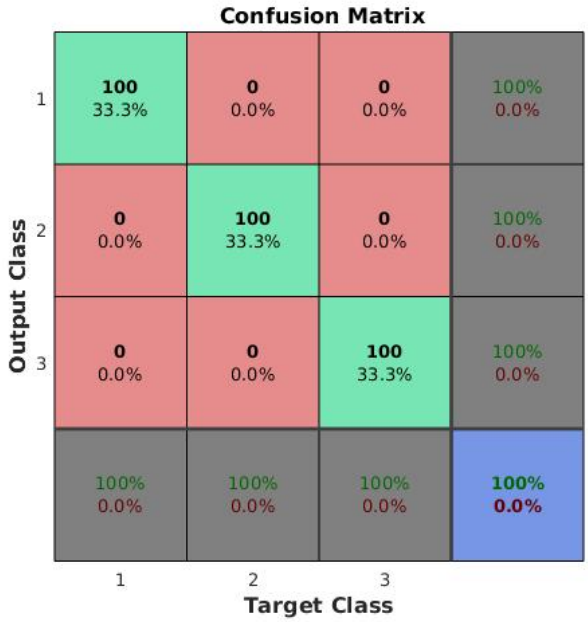

Figure 14: Figure showing confusion matrix for test set

We see that $accuracy = 100\%$ on test set.

1.3 MLFFNN with 2 hidden layers

Multilayer feed forward with 2 hidden layers was trained. The hidden layers uses tanh activation function, and the output layer uses softmax. Cross entropy error function is used.

Model selection: The no of hidden layer nodes were varied from 2 to 10, for the 2 hidden layers, and the best model according to cross validation was obtained for the two hidden layers having 3 nodes each. The neural network model is able to learn the decision boundary with less no of nodes, because the boundary is linear which is less complex.

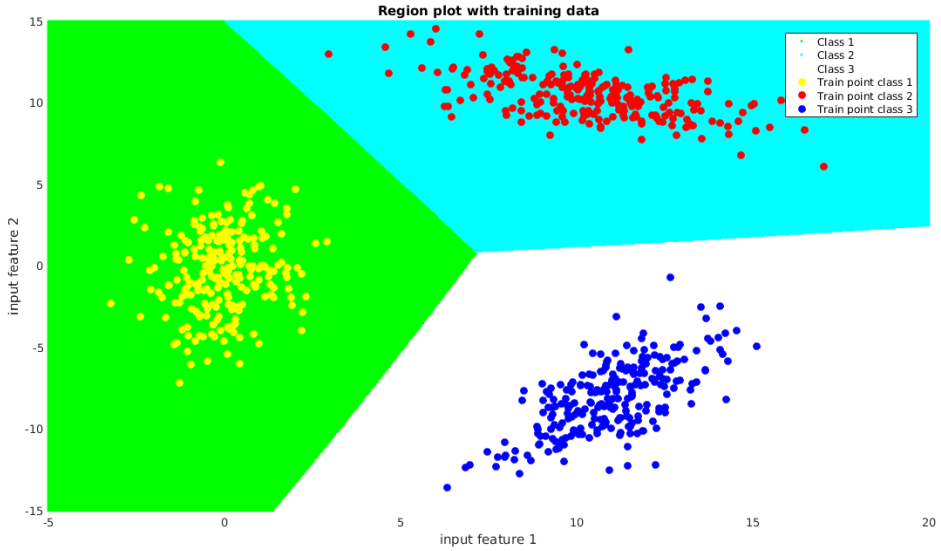

Figure 15: Region plot for dataset linearly separable data set - 1(a)

The model clearly separates the different classes as we can see from the surface plots.

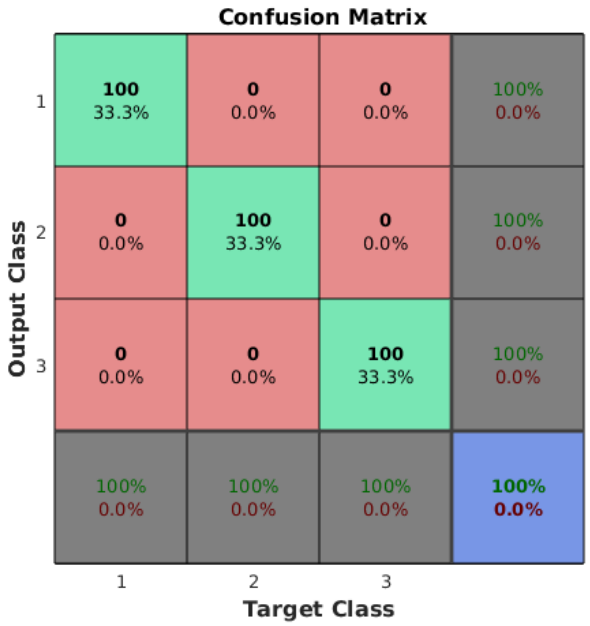

Figure 16: Confusion Matrix plot for dataset 1(a)
We get 100% accuracy for the data set, because it is linearly separable and since neural network is capable learning linear decision boundary.

2 Non-linearly separable data set

2.1 SVM

Polynomial kernel

Parameter estimation
To choose the kernel parameters we use kernel gram matrices.As we plotted kernel gram matrices for various values of "p" it was observed that it was performing well for even values of "p".The data across is actually separated by a circular surface when visualized.Hence "p" is chosen to be 2.Similarly for a=1,b=0.3 the kernel gram matrix looks best.Hence the above are chosen as kernel parameters.

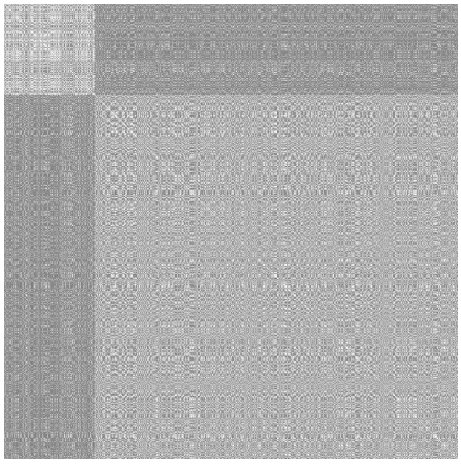

Figure 17: Figure showing grey level image of kernel gram matrix for kernel parameters a=1,b=0.3,p=2 ,dataset 1b

The plot for accuracy on validation vs *logc* looks like:

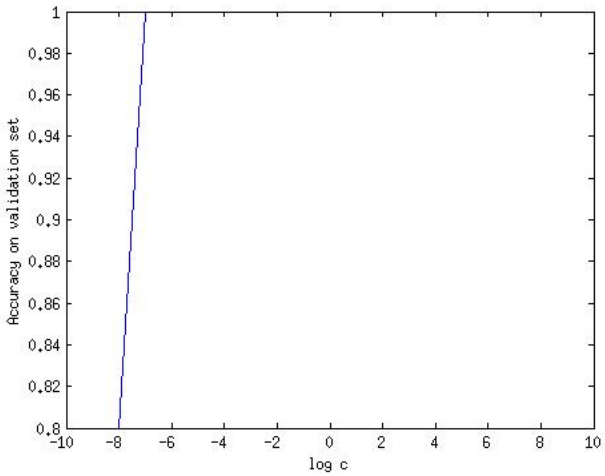

Figure 18: Figure showing accuracy for validation set for different values of c

We see that accuracy is 1 for any value of c above $2^{-}6$, lower values of "c" leads to too many support vector as margin is large.Hence we choose "c" to be 1.

Decision region plot
The decision region plot for c=1 looks like:

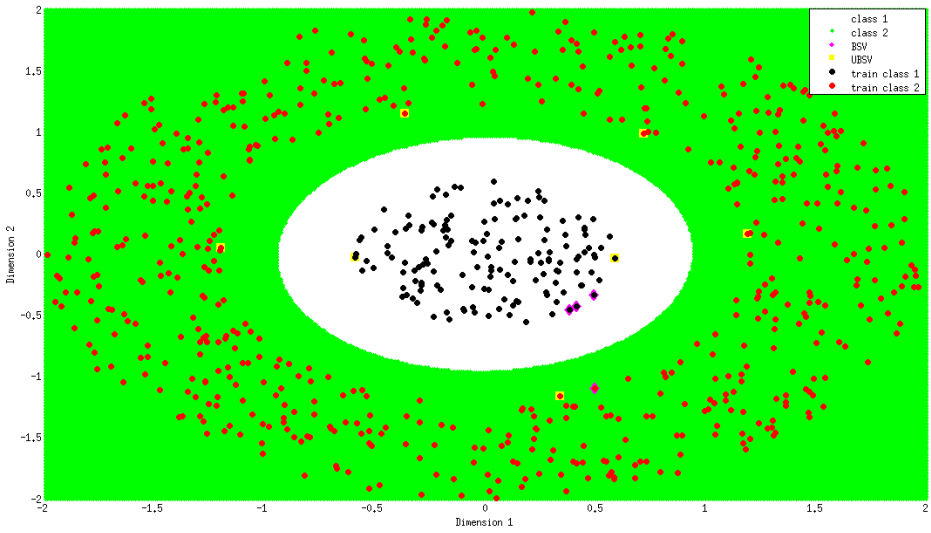

Figure 19: Figure showing decision region plot for dataset 1b using polynomial kernel,c=1

We see that unbounded support vectors are those which lie on margin.Since margin is not so large("c" is 1),we have only few points which are within margin i.e bounded support vectors.

Confusion matrix

The confusion matrix for dataset 1a using polynomial kernel and parameters a=1,b=0.3,p=2 ,c=1 looks like:

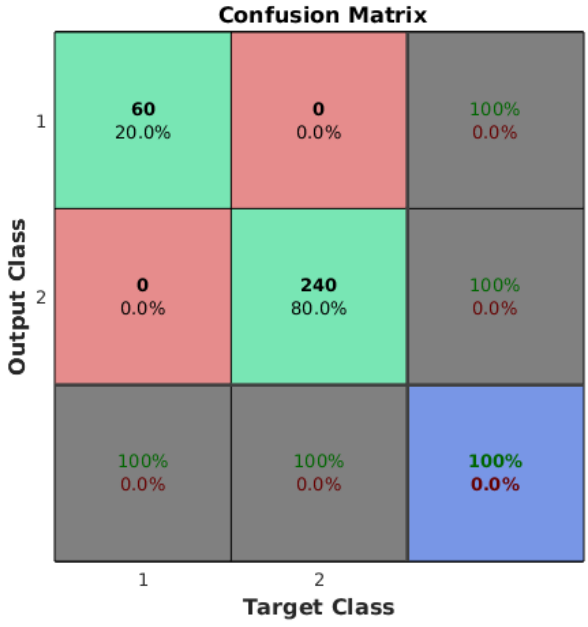

Figure 20: Figure showing confusion matrix for test set for dataset 1b polynomial kernel,a=1,b=0.3,p=2,c=1

$accuracy = 100\%$

Gaussian Kernel

Parameter estimation:
σ is chosen by kernel gram matrices.Best value of $\sigma$is found to be 3.

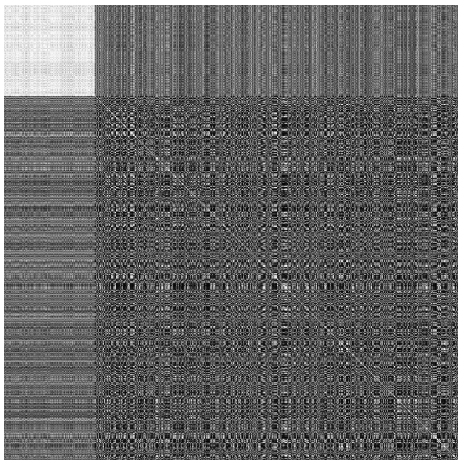

Figure 21: Grey level image of kernel gram matrix for dataset 1b training set using gaussian kernel for $\sigma = 3$

We can see the class imbalance in the above figure.

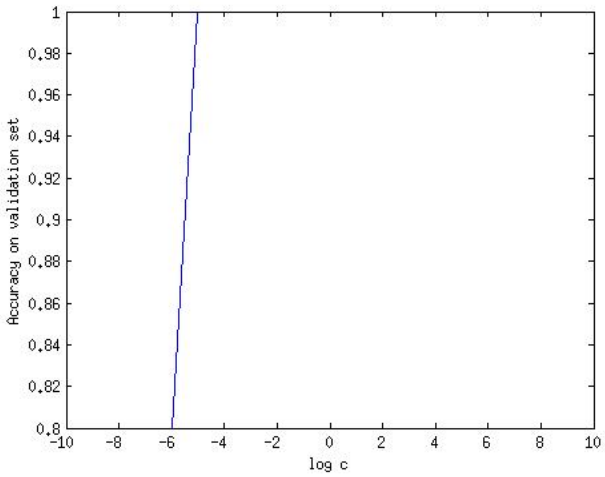

Figure 22: Figure showing accuracy for validation set for different values of c

We see that accuracy on validation set is 100% for wide range of "c".We choose a c such that number of support vectors is less."c" chosen is 1 here.

Decision region plot:

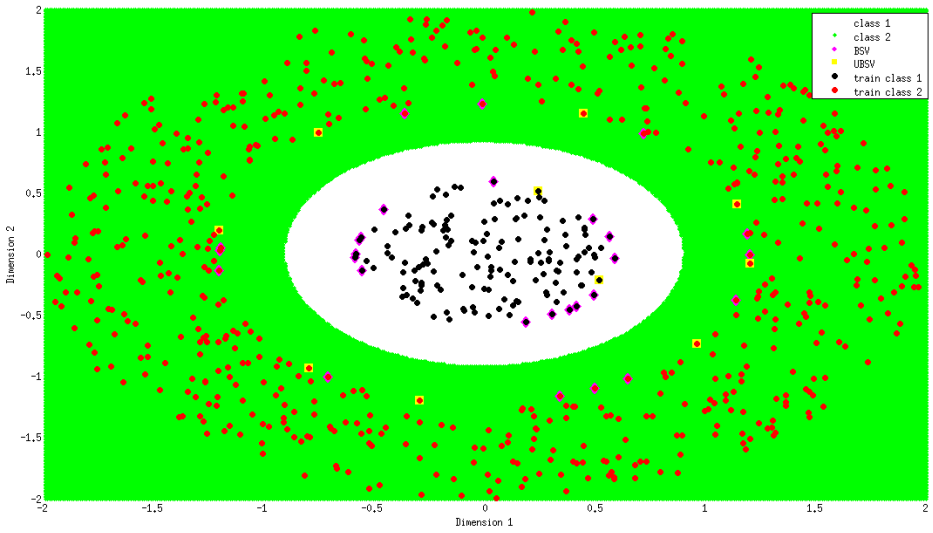

Figure 23: Figure showing decision region plot for dataset 1b gaussian kernel for $\sigma = 3$,c=1

We see that number of support vectors are more compared to that in polynomial kernel for same value of "c" both because of change in margin and also because of change in shape of decision boundary

Confusion matrix:

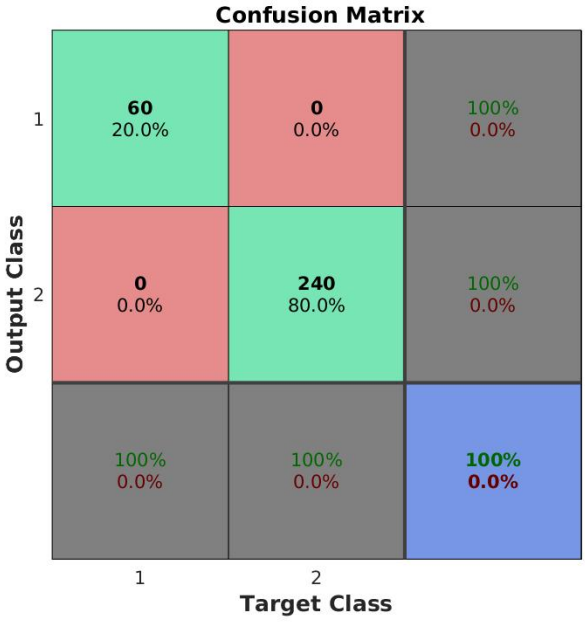

Figure 24: Figure showing confusion matrix for test set for dataset 1b gaussian kernel for $\sigma = 3$,c=1

2.2 Logistic Regression

In logistic regression the posterior probabilities of the class are ensured to be between 0 and 1 and sum over all classes is 1 by using a softmax acting on a linear function. Cross entropy was used as the error function. And the python library sci-kit learn was used for this implementation.

Parameter estimation: Logistic Regression was done on the dataset with polynomial basis function. The parameter that was estimated was degree of the basis expansion. We tried varying

the degree between 0 and 6, since this dataset is not linearly separable dataset, we got perfect results on the test set with degree greater than 1, i.e 2. Here we present our findings.

The error on the various sets vs the degree was plot. Here we notice only one line as all the three overlap. The validation set vs degree was used to select what is the best model. We notice a maximum accuracy of 1 on validation set when degree is 2. Hence model with degree 2 is chosen as the best model.

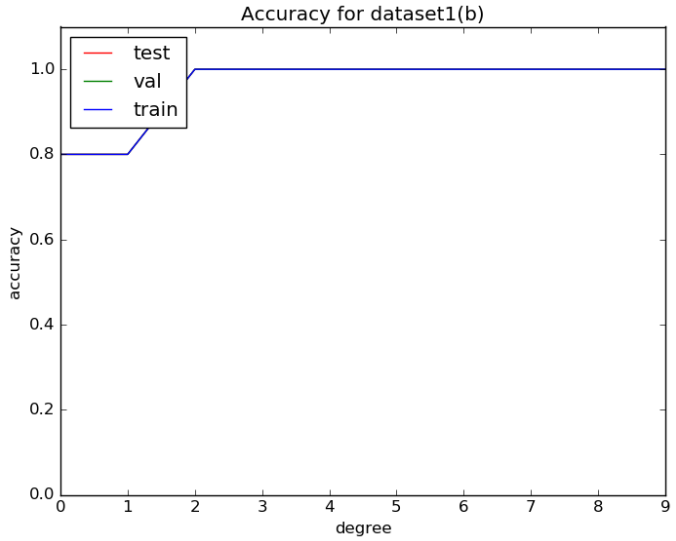

Figure 25: Figure showing accuracy for train,test and valid set for different values of degree

We see that accuracy=1 for all values of degree greater than equal to 2.

Decision region plot:
The decision region plot for degree=2 looks like:

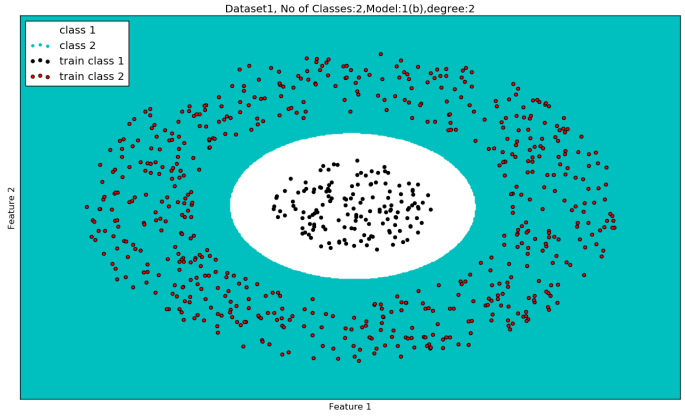

Figure 26: Figure showing decision region plot for dataset 1b using logistic regression with polynomial expansion of degree equal to 2

From the figure we see that, the decision boundaries separating each pair of classes are circular/elliptical indicating that a separation function of degree 2 is involved,and the points in the train set are all classified correctly.

Confusion matrix:
The confusion matrix for test set look like:

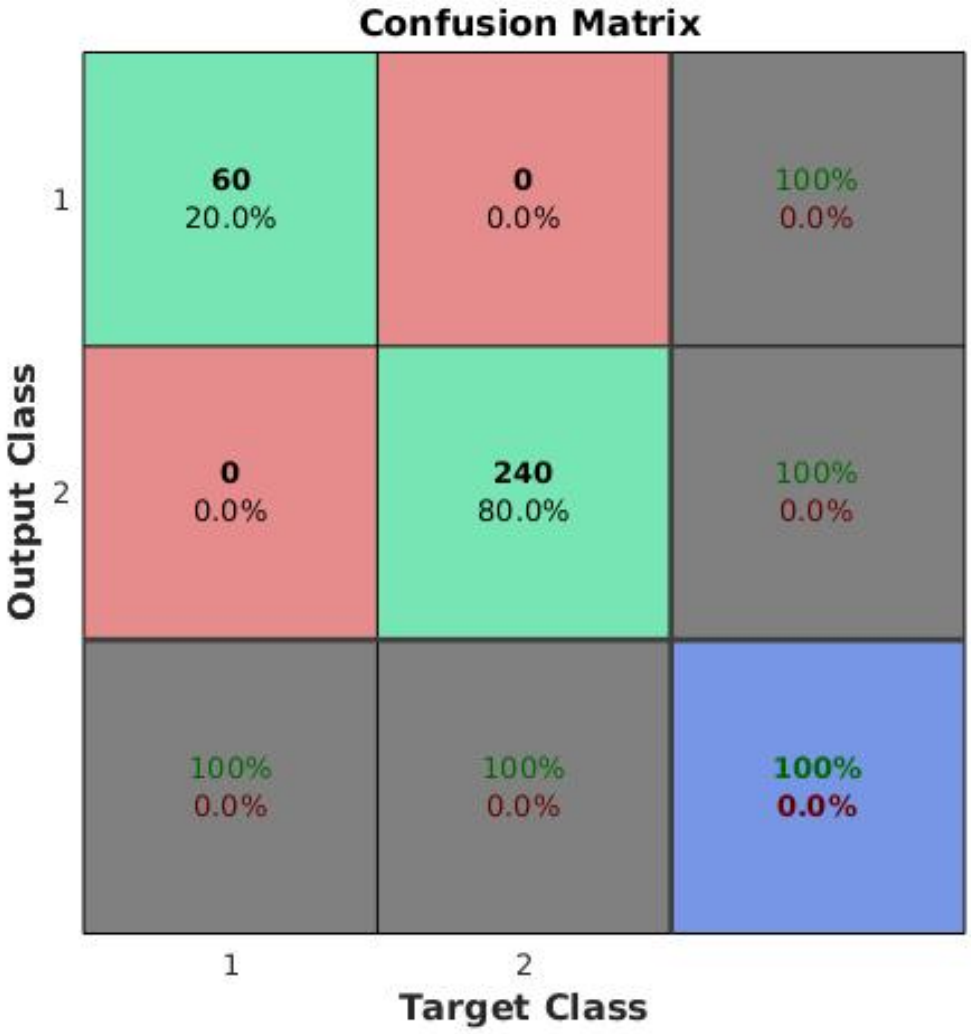

Figure 27: Figure showing confusion matrix for test set

We see that $accuracy = 100\%$ on test set.

2.3 MLFFNN for 2 hidden layers

The neural network trained consists of 2 hidden layers. The hidden layers use tanh activation function, and the output layer use softmax function. Cross entropy error function is used.

Model selection: The no of nodes in the two hidden layers was varied from 2 to 15. The best model selected using cross validation function was for 6, 4 nodes in hidden layers 1 and 2 respectively.

Decision region plot:

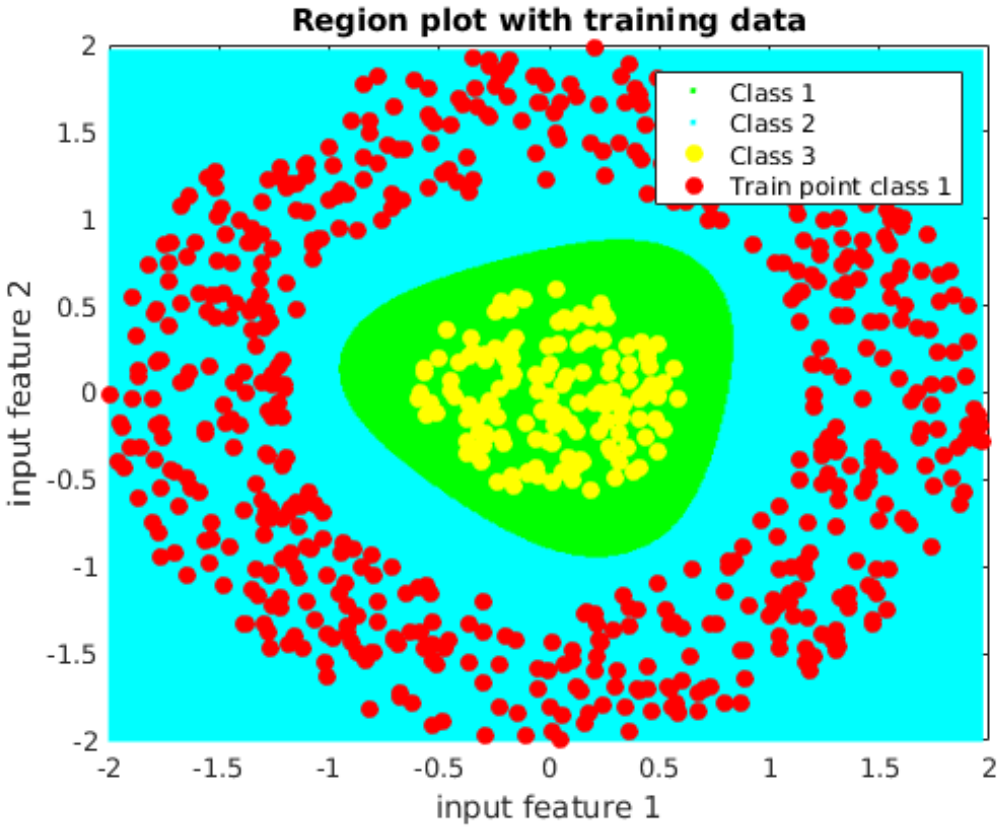

Figure 28: Decision region plot for Dataset 1(b)

As we can see from the region plot the decision boundary separates the 2 classes without any misclassification.

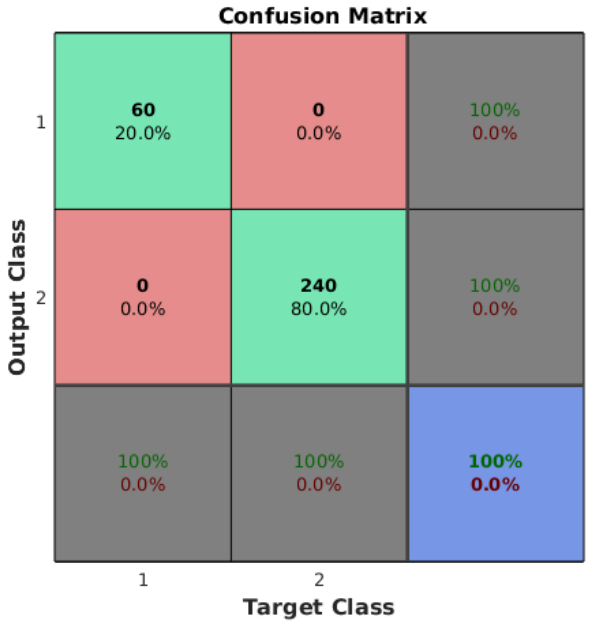

Figure 29: Confusion matrix plot for Dataset 1(b)

We get a test accuracy of 100% in this case because the dataset itself is separable and the neural network model is capable of learning non linear decision boundaries. The model leaned

here is slightly more complex (higher no of nodes in the two layers) than that learned in linearly separable dataset case. This is because the decision boundary was linear in the former case, but non linear in this case. And so requires higher complex model to learn the boundary.

Hidden Layer 2 outputs in between training:

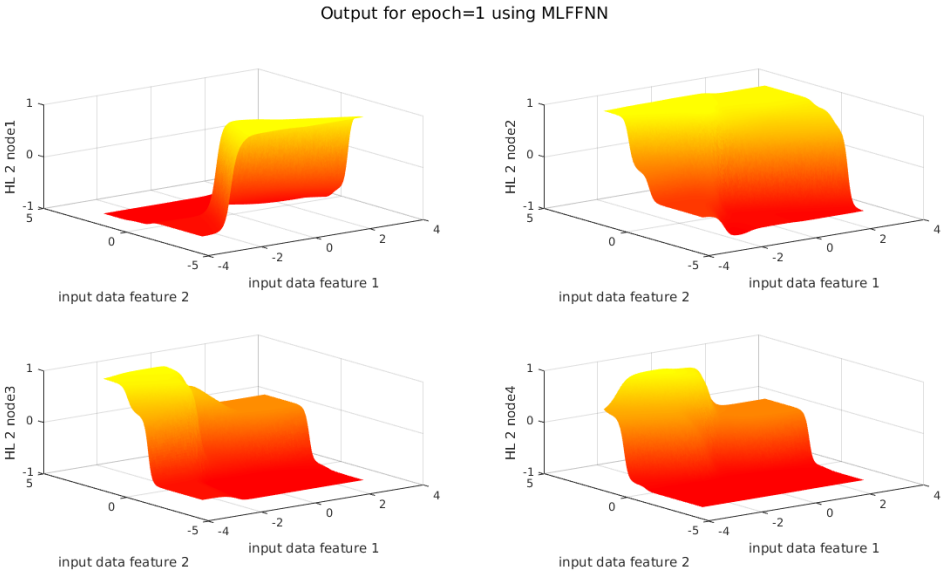

Figure 30: Hidden layer 2 outputs after first epochs

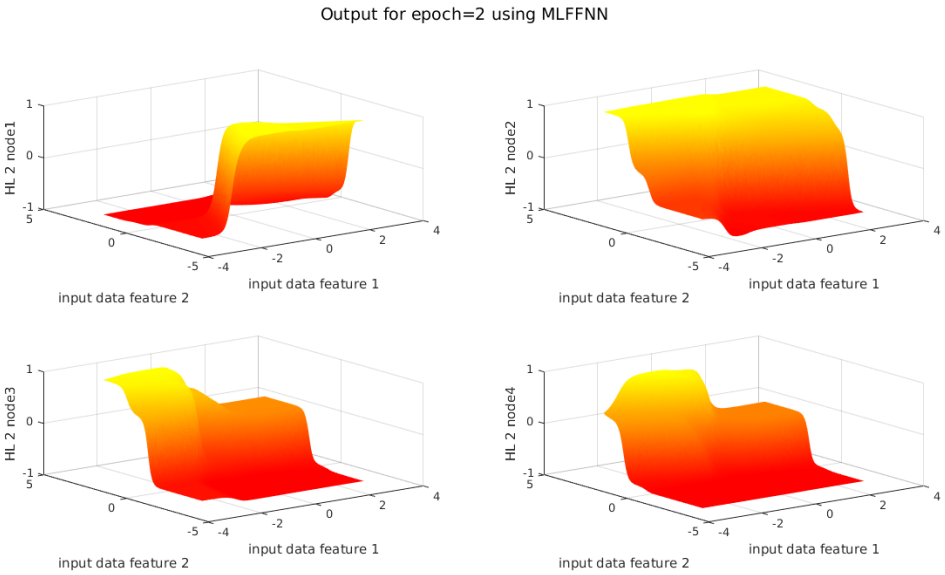

Figure 31: Hidden layer 2 outputs after 2 epochs

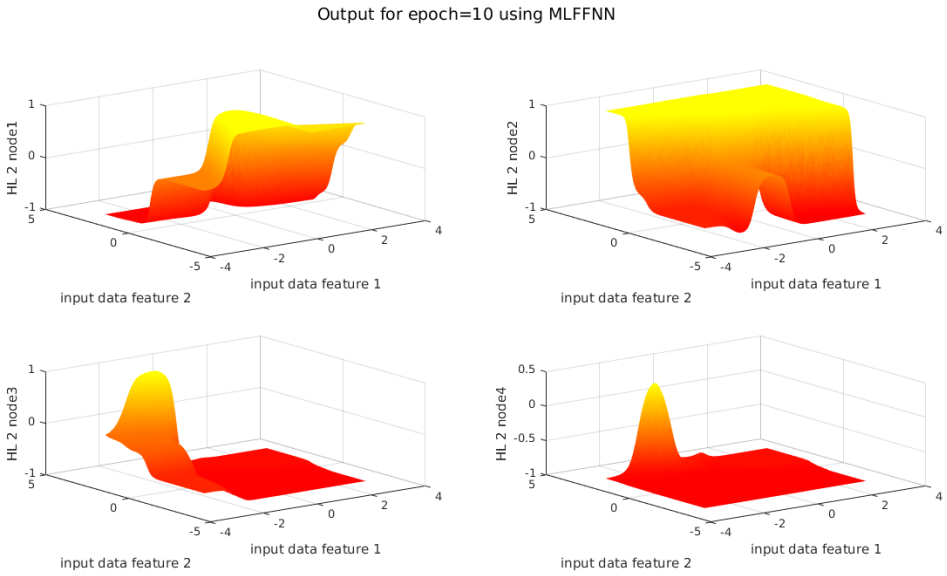

Figure 32: Hidden layer 2 outputs after 10 epochs

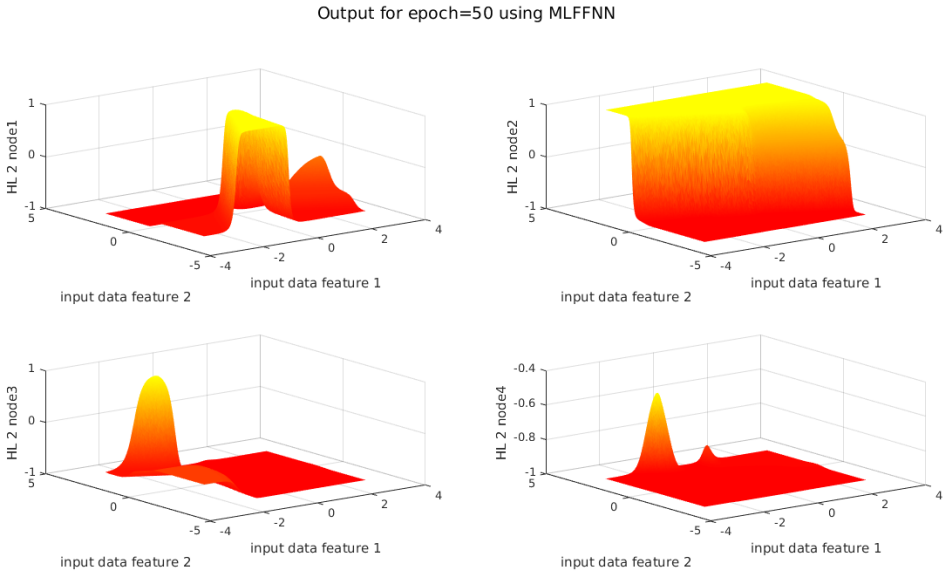

Figure 33: Hidden layer 2 outputs after 50 epochs

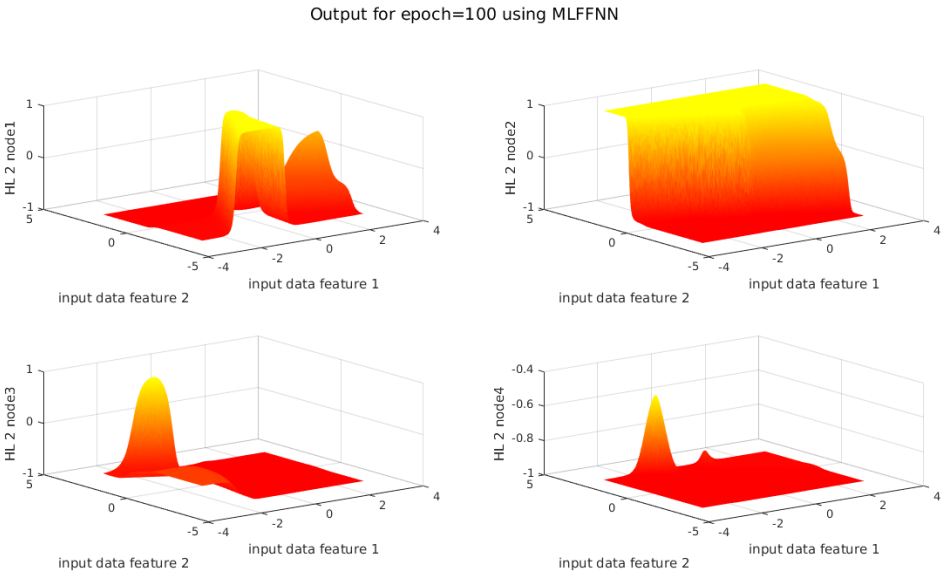

Figure 34: Hidden layer 2 outputs after 100 epochs

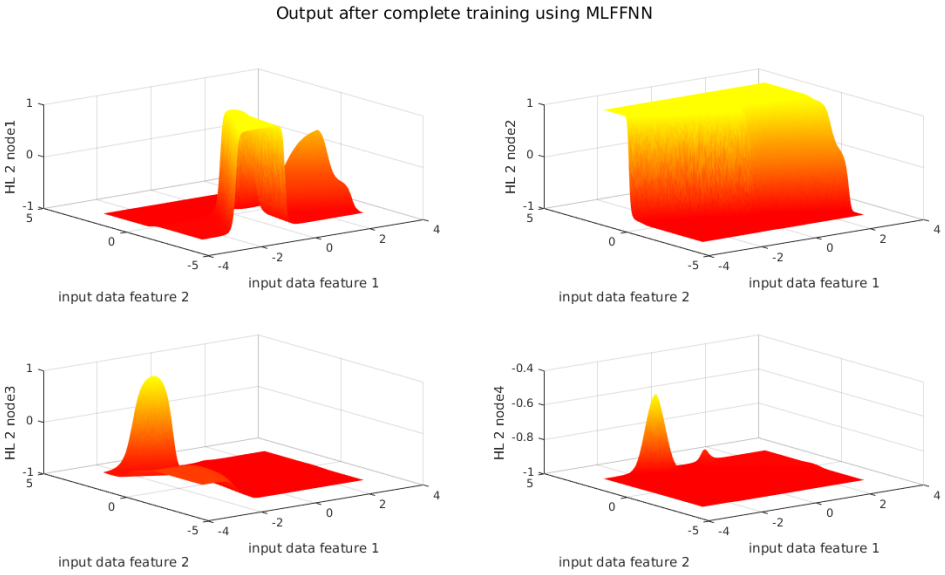

Figure 35: Second Hidden layer outputs after complete training of model

From the plots of hidden layer 2, we can see that each of the nodes slowly learn the combination of different s-shape surfaces.

Output layer plots:

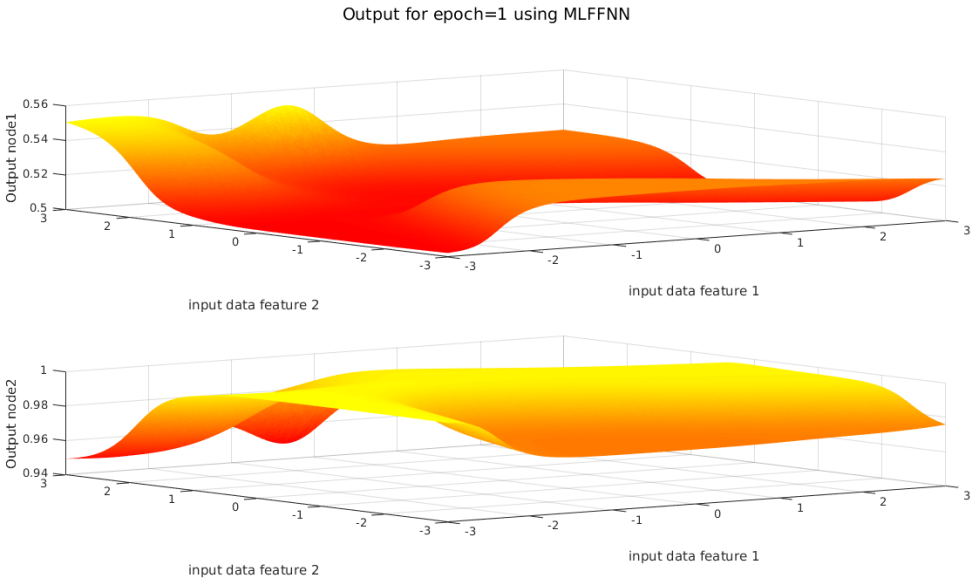

Figure 36: Output layer outputs after 1^{st} epoch

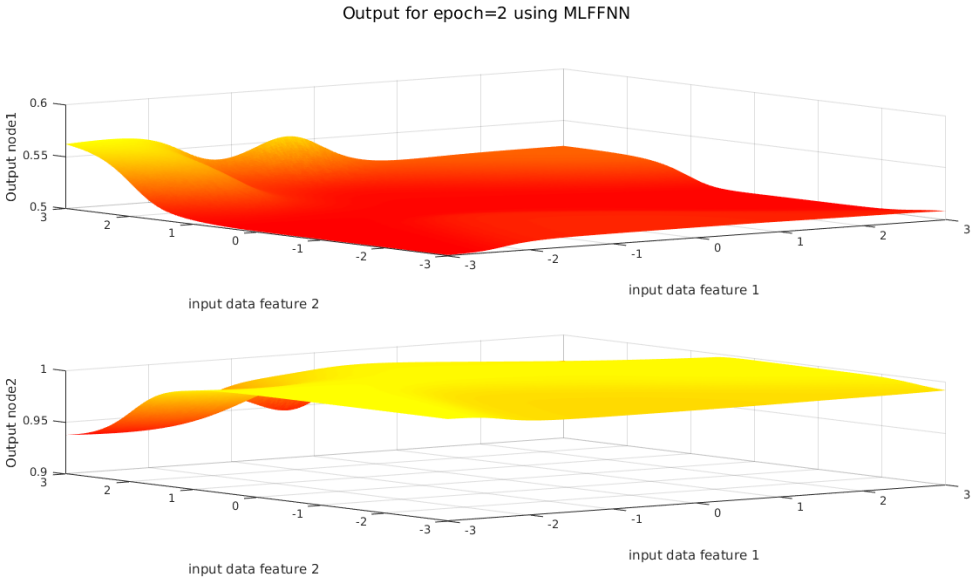

Figure 37: Output layer outputs after 2 epochs

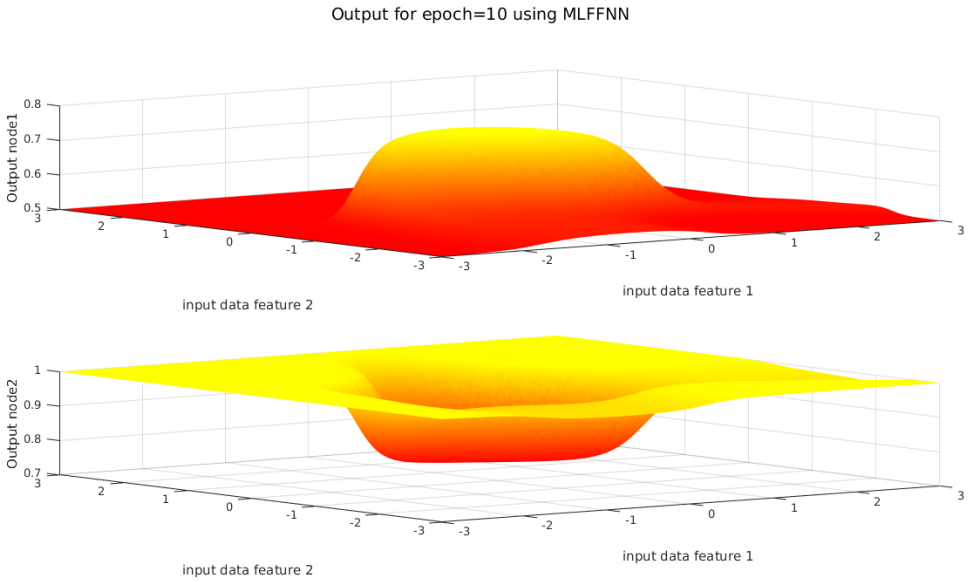

Figure 38: Output layer outputs after 10 epochs

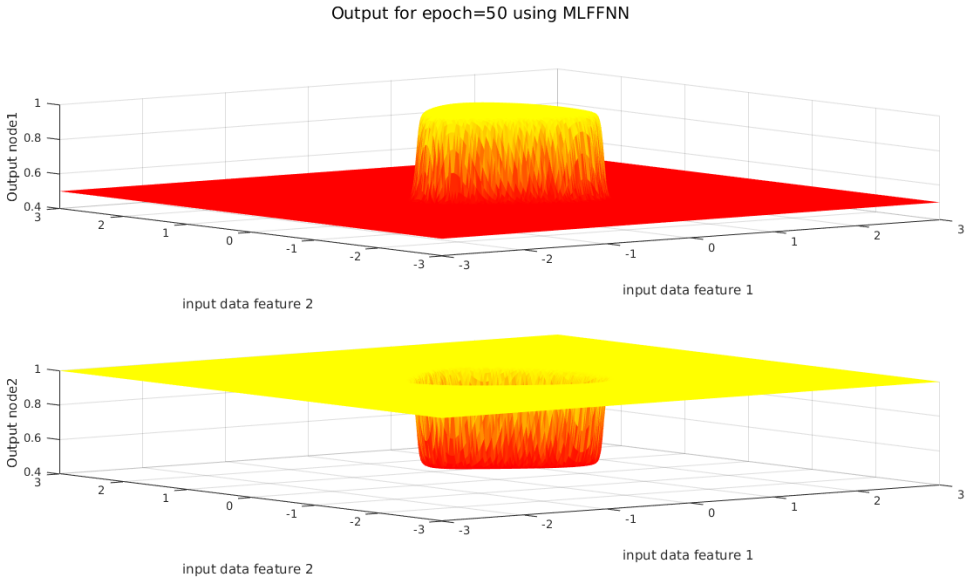

Figure 39: Output layer outputs after 50 epochs

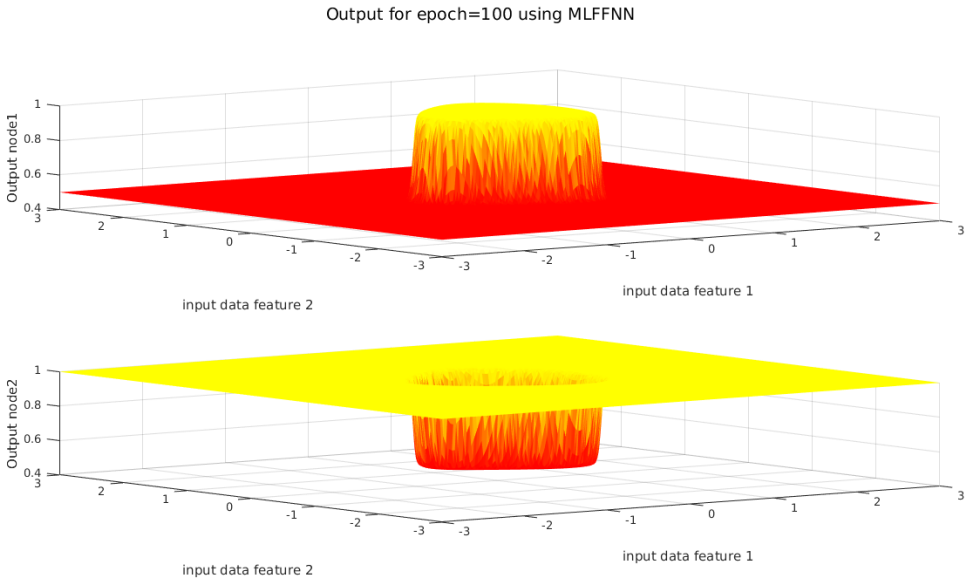

Figure 40: Output layer outputs after 100 epochs

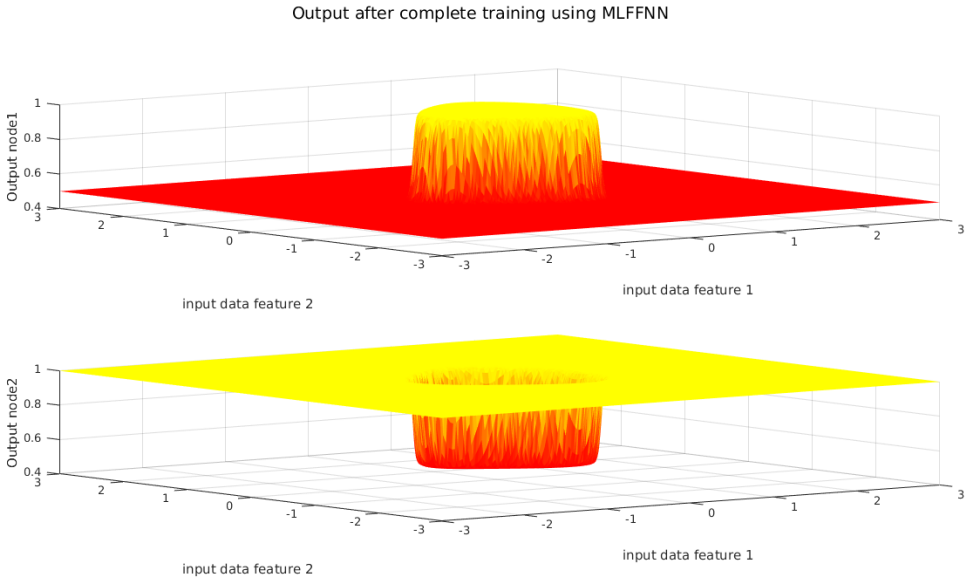

Figure 41: Output layer outputs after complete training of model

The first node outputs 1 for the points inside the circular region as observed from the plot. And the 2^{nd} node outputs 1 for points outside the circular region. The circular region where there is a transition from 0 to 1 or 1 to 0 in the two nodes depict the decision boundary.

Comparison between the models

From the decision region plots we can see that SVM, polynomial regression learns a perfect circular boundary whereas neural network model doesn't. This is because in SVM, maximal separating surface is learned. And in polynomial regression the surface is being tried to fit a polynomial whereas in MLFFNN there is no condition imposed on non linearity except that the decision boundary should separate the two classes.

3 Overlapping data set

3.1 SVM

Polynomial kernel

Parameter estimation
We see that the dataset is linearly non-separable.Hence we choose p=1,a=1,b=0 as kernel parameters.The kernel gram matrix for above configuration looks like:

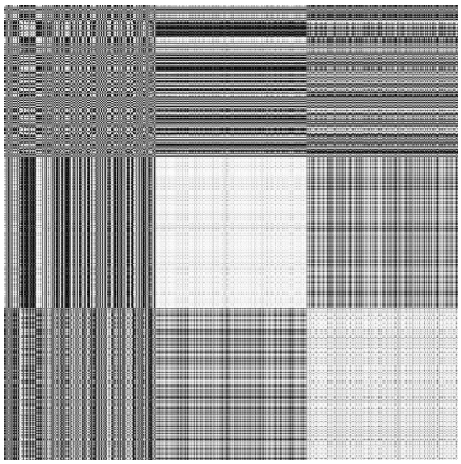

Figure 42: Figure showing grey level image of kernel gram matrix for kernel parameters a=1,b=0,p=1

Since we used the same parameters for dataset 1a also,we see that,in this case for few examples same class kernel function values are lower than 1a(grey patch in diagonal blocks),for few examples kernel function values for different classes kernel function values are higher(non diagonal blocks are less darker),because of overlapping.

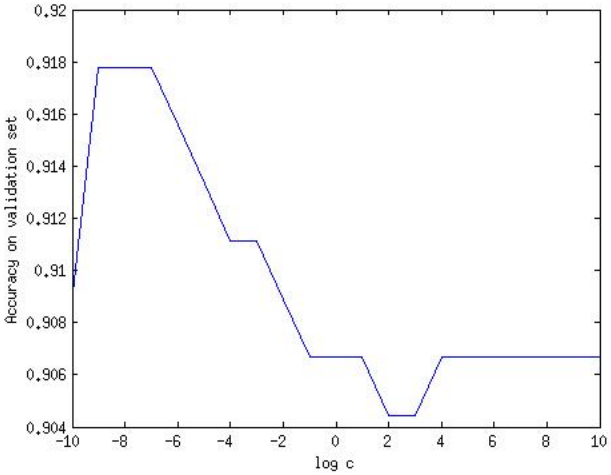

Figure 43: Figure showing accuracy for validation set for different values of c

We obtain best value of c as 0.0078.

Decision region plot:

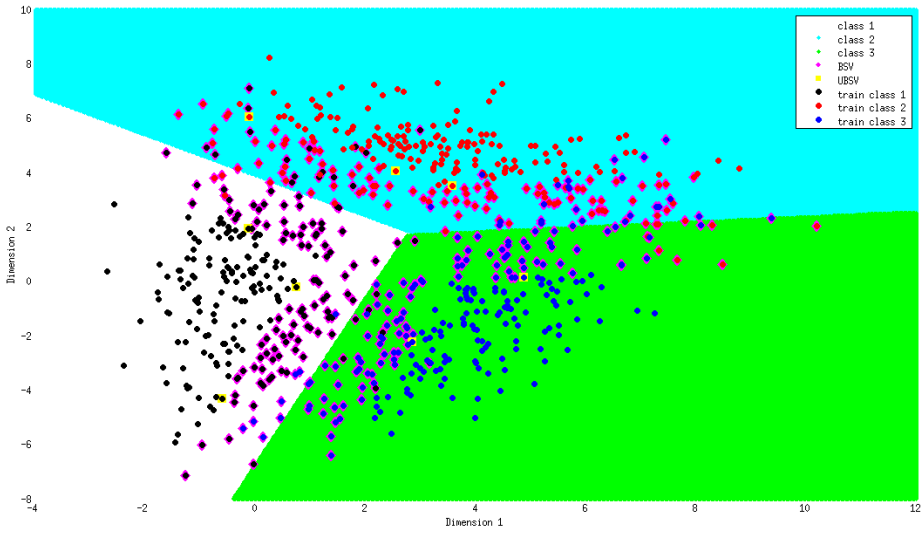

Figure 44: Figure showing decision region plot for dataset 1c using polynomial kernel,a=1 b=0 p=1 c=0.0078

Since "c" is small,we find that margin is larger,hence more points lie within the margin,number of bounded support vectors is high.

Confusion matrix

The confusion matrix for dataset 1c using polynomial kernel and parameters a=1,b=0,p=1 ,c=0.0078 looks like:

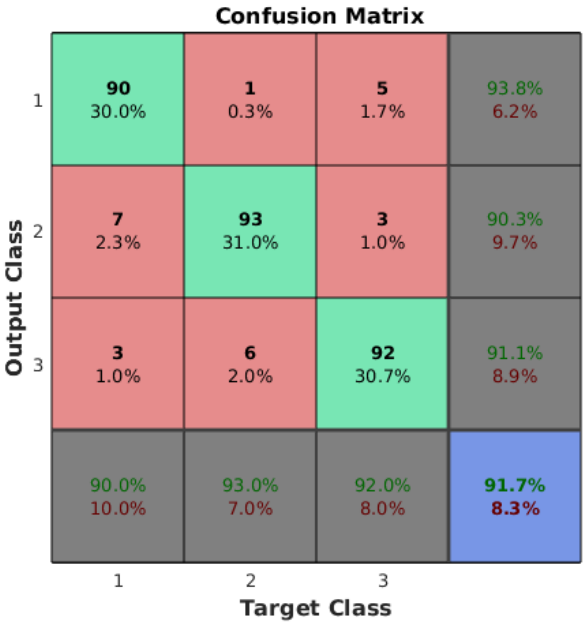

Figure 45: Figure showing confusion matrix for test set for dataset 1c polynomial kernel,c=0.0078

The accuracy in this case is observed to be 91.2% which is less compared to datasets 1a and 1b.

Gaussian kernel

Parameter estimation:
σ value chosen is 33 after observing kernel gram matrices.
The kernel gram matrix for $\sigma = 33$ is shown:

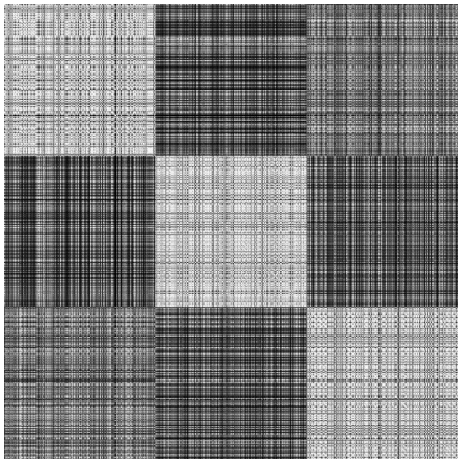

Figure 46: Grey level image of kernel gram matrix for dataset 1c training set using gaussian kernel for $\sigma = 33$

We observe that the kernel gram matrix is better representing similarity of points of same class and dissimilarity of points of different class because it depends on distribution of data,for this distribution euclidean distance works better than cosine similarity.

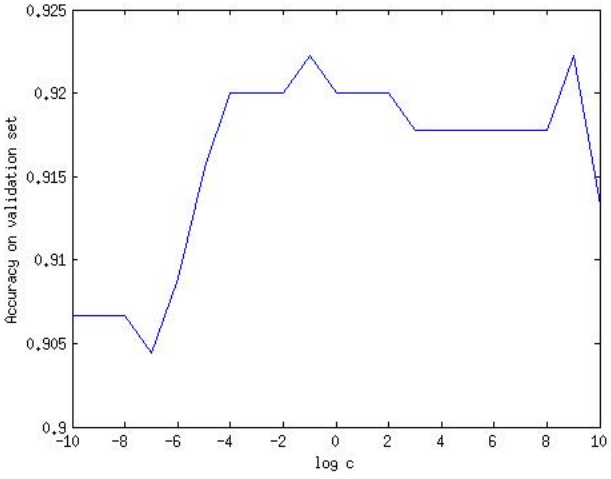

Figure 47: Figure showing accuracy for validation set for different values of c

The best value for c is observed to be 0.5.
Decision region plot:

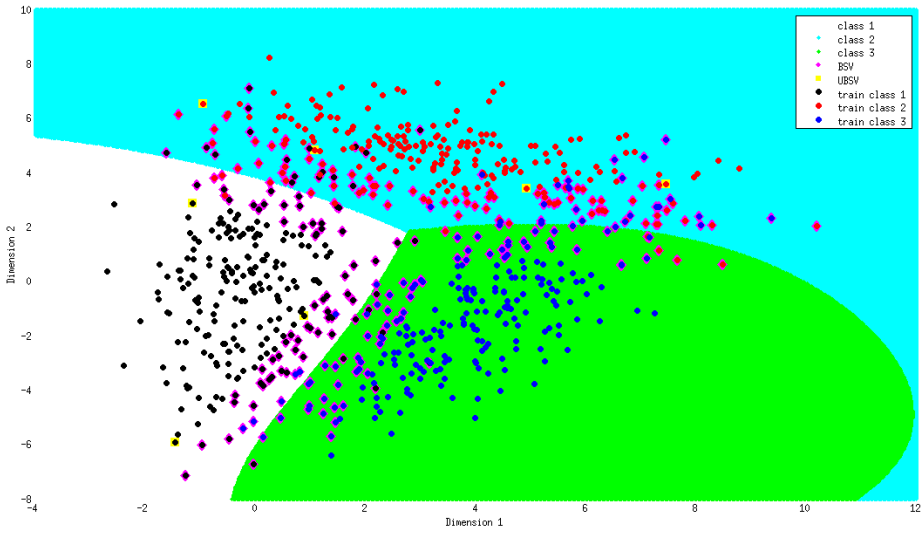

Figure 48: Figure showing decision region plot for dataset 1c gaussian kernel for $\sigma = 33$,c=1

We see that both decision surface and number of support vectors change from polynomial to gaussian kernel.Since the value of c is observed in polynomial kernel is low,the number of support vectors is high in polynomial case.

Confusion matrix:

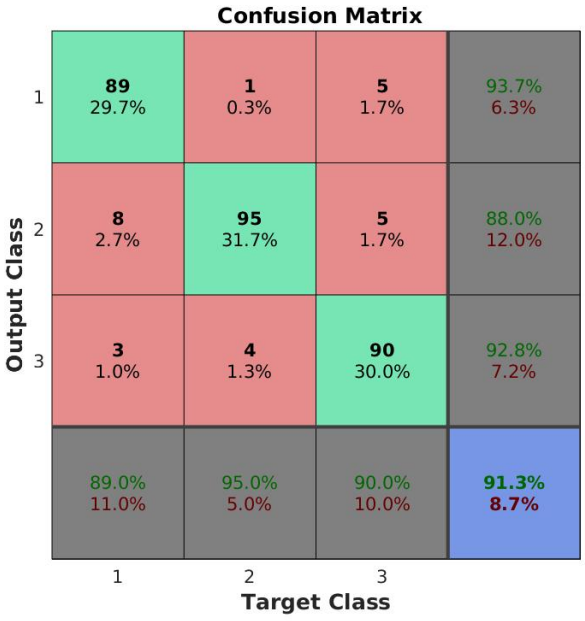

Figure 49: Figure showing confusion matrix for test set for dataset 1c gaussian kernel for $\sigma = 33$,c=1

The accuracy is found to be 91.3% which is slightly lesser than polynomial kernel which yields accuracy 91.7%

3.2 Logistic Regression

In logistic regression the posterior probabilities of the class are ensured to be between 0 and 1 and sum over all classes is 1 by using a softmax acting on a linear function. Cross entropy was used as the error function. And the python library sci-kit learn was used for this implementation.

Parameter estimation: Logistic Regression was done on the dataset with polynomial basis function. The parameter that was estimated was degree of the basis expansion. We tried varying the degree between 0 and 6, we got good results on the test set with degree 3. The best accuracy observed was 92%. Since the dataset is overlapping, a perfect accuracy seems difficult. Hence an accuracy of 92 percent implies that this model is a good model. Here we present our findings.

The error on the validation set vs degree was plot and was used to select what is the best model. We notice that all models perform almost equally between degree 1 and 4 but there is a slight maximum at 3. We notice a maximum accuracy of 92% on validation set when degree is 3. Hence model with degree 3 is chosen as the best model.

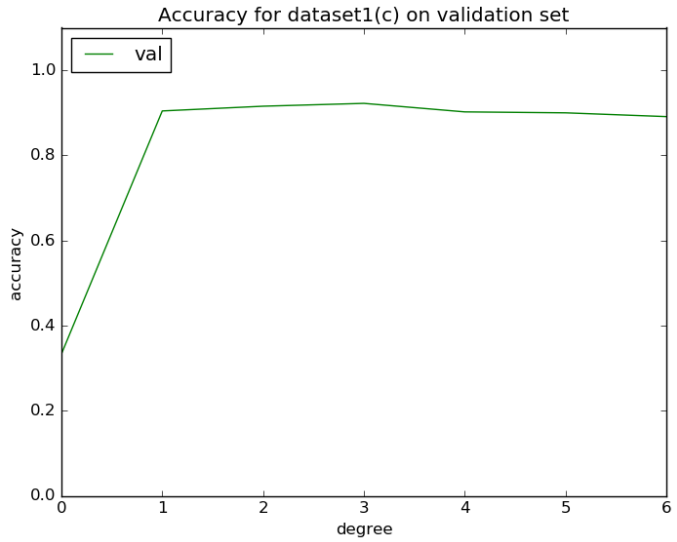

Figure 50: Figure showing accuracy of validation set for different values of degree

We see that accuracy=0.92 for degree equal to 3.

Decision region plot:
The decision region plot for degree=3 looks like:

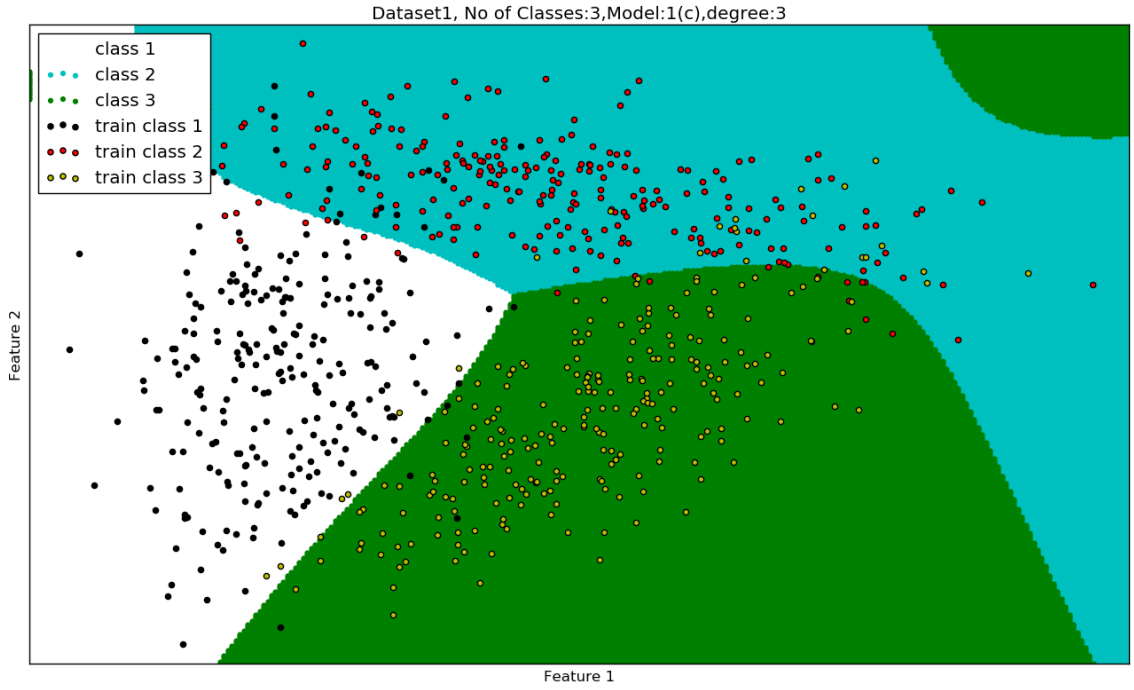

Figure 51: Figure showing decision region plot for dataset 1c using logistic regression with polynomial expansion of degree equal to 3

From the figure we see that, the decision boundaries separating each pair of classes are not linear indicating that a separation function of degree greater than 1 is involved,and the points in the train set overlap closely among them.

Confusion matrix:

The confusion matrix for test set look like:

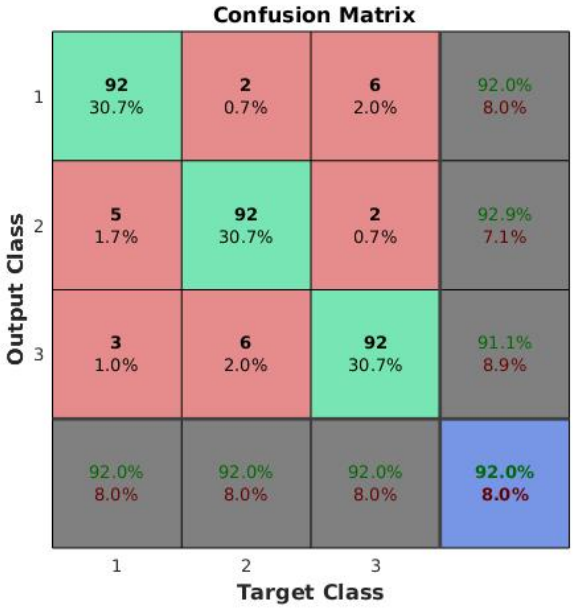

Figure 52: Figure showing confusion matrix for test set

We see that $accuracy = 92\%$ on test set.

3.3 MLFFNN with 2 hidden layers

MLFFNN was trained with 2 hidden layers. The hidden layers uses tanh activation function, and the output layer uses softmax activation function. Cross entropy error function was used.

Model Selection: For the 2 hidden layers the no of nodes were varied from 2 to 15, and the best model complexity obtained using minimum cross validation error was for 7 and 5 nodes in the hidden layers 1 and 2 respectively.

Decision region plot:

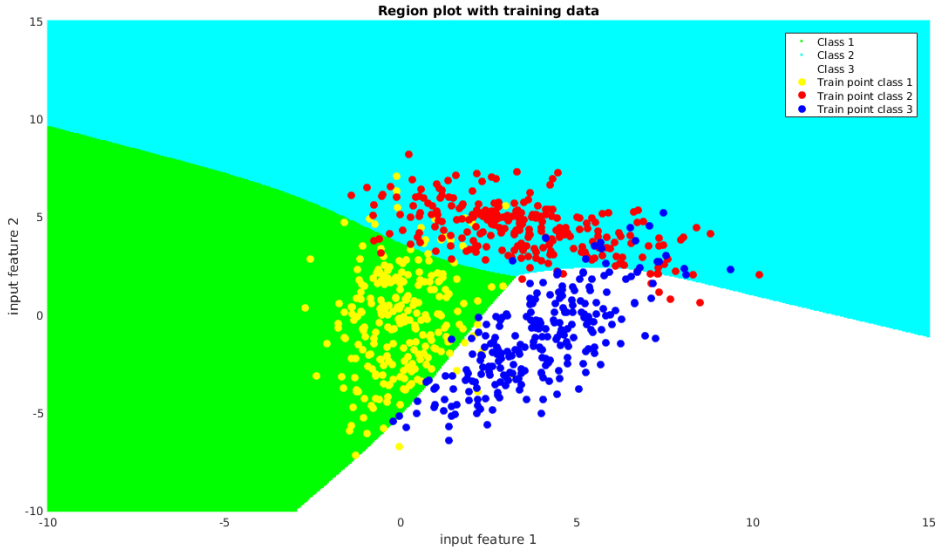

Figure 53: Decision region plot for overlapping dataset

Confusion matrix plot:

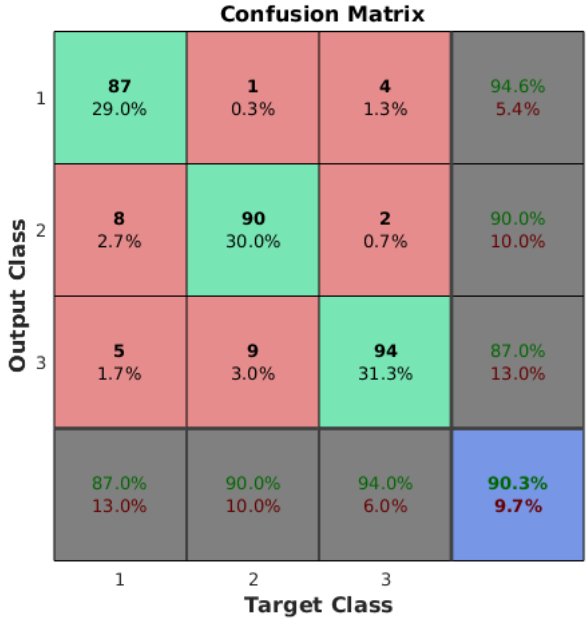

Figure 54: Confusion matrix plot for Dataset 1(c)

The classes are overlapping, neural network model learned doesn't separate the 2 classes com-

pletely. Minimizing validation error, the best model learned was for 7 nodes on first hidden layer, and 5 nodes on second hidden layer. The model gives 90.3 % accuracy on test data.

4 Black and white image data set

Gaussian kernel

Parameter estimation:

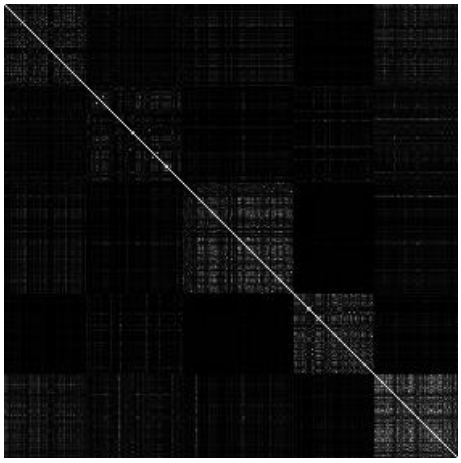

Figure 55: Figure showing grey level image of kernel gram matrix for kernel parameters ,$\sigma = 60$

The best σ obtained is 60 from kernel gram matrix.
Here we see that some of the classes are well separable whereas other pairs of classes are not.Using CNN willhave higher accuracy because it exploits local features and robust to shift errors unlike SVM.

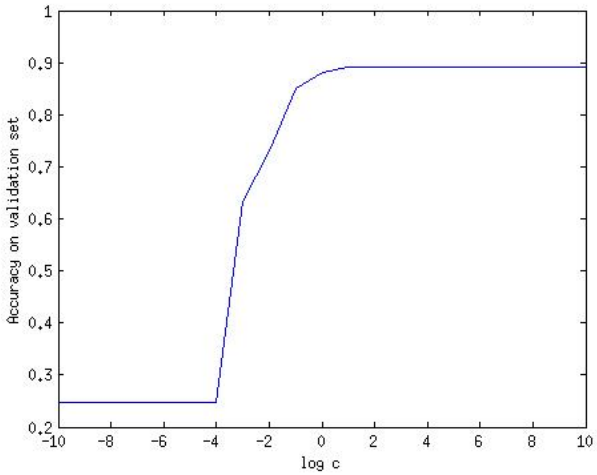

Figure 56: Figure showing accuracy for validation set for different values of c

c is chosen to be "1" as accuracy=100%
Confusion matrix:

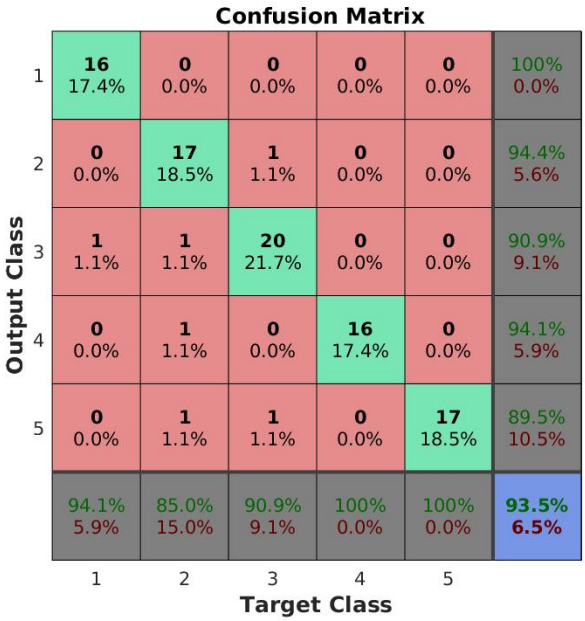

Figure 57: Figure showing confusion matrix for test set for black and white image dataset gaussian kernel for $\sigma = 60$,c=1

The accuracy is observed to be 93.5% on test set.
We can also observe that examples of class a misclassified as class b have greyer patch in kernel gram matrix relative to those pairs of classes which are separable/not misclassified.(ex:class3,class4;class1,class2;)

4.1 CNN

A deep leaning convolutional neural network was used to train this dataset. Convolutional neural networks tend to perform well on image datasets, we use a two layer network here. Each layer included a combination of a filter layer that introduces non linearity through a relu function and a max pooling layer which combined nodes with some drop out probability. Finally after the two layers there was a fully connected layer that generated outputs based on the number of classes and a softmax cross entropy function in the end for multi class classification.

Parameter estimation: The parameters to be varied were the number of feature maps in the first and the second layer. Sir told us to keep the stride and the local receptor field constant. Hence we kept them the same and didn't consider them in out parameter estimation. The value of the LRF(Local Receptor field) was $5 * 5$, and the stride parameter was 1 in the convolutional layer, in the pooling layer the value of the LRF was $2 * 2$, and the stride parameter was 2. After 1 layer the output reduced by 50 percent in both dimensions. Here we present our findings when we vary the number of feature maps to be applied.

The error on the validation set vs number of feature maps in layer 1 was plot and was used to select what is the best model. We decided to keep the number of feature maps in layer 2 as twice the number of feature maps in layer 1 as the CNN was taking a long time to train. Hence instead of estimating on a huge field we varied parameters in first layer $\in \{1, 5, 10, 50, 100\}$. We noticed that the model with 10 performs the best and the models with 50 and 100 tend to overfit on the data. The number of epochs were set at 200 and the exit criteria was if the error changed by a value of less than 0.1 percent after 10 epochs we exited. We notice a maximum accuracy of 96.42% on validation set when number of feature maps in layer 1 are 100 and number of feature maps in layer 2 are 200. Hence this model was chosen as the best model.

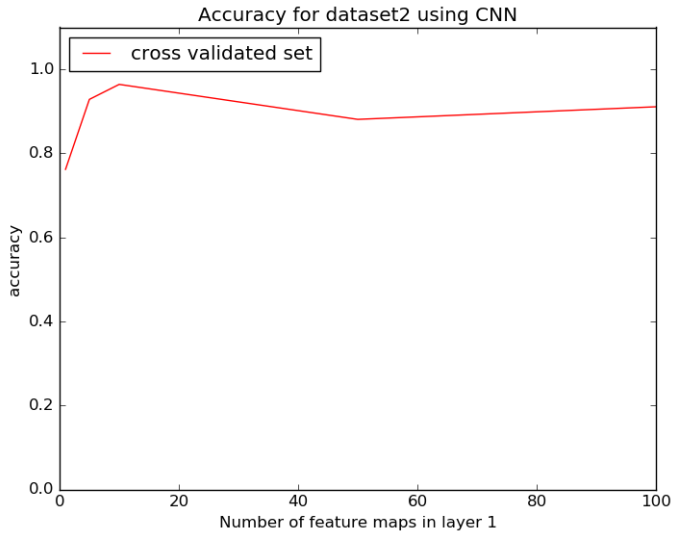

Figure 58: Figure showing accuracy on validation set for different values of feature maps in layer 1, Number of feature maps in layer 2 are twice the number of feature maps in layer 1

Feature Maps:
The feature maps of the two layers are shown below:

Figure 59: Input Image:Class Label 40

Figure 60: Feature Map of layer 1

Figure 61: Feature Map of layer 2

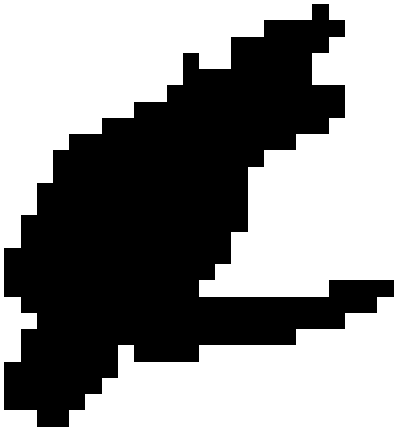

Figure 62: Input Image:Class Label 48

Figure 63: Feature Map of layer 1

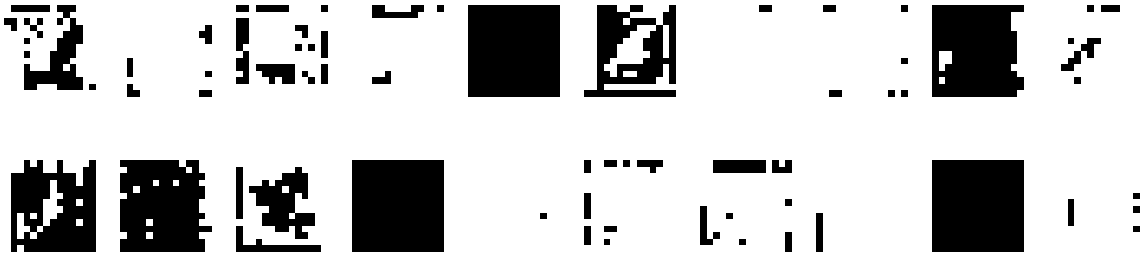

Figure 64: Feature Map of layer 2

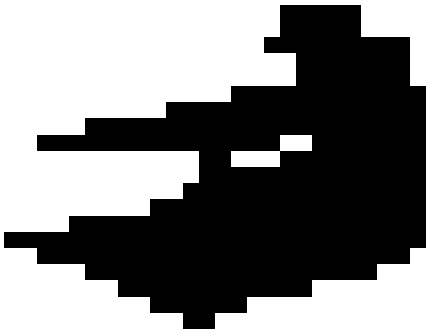

Figure 65: Input Image:Class Label 56

Figure 66: Feature Map of layer 1

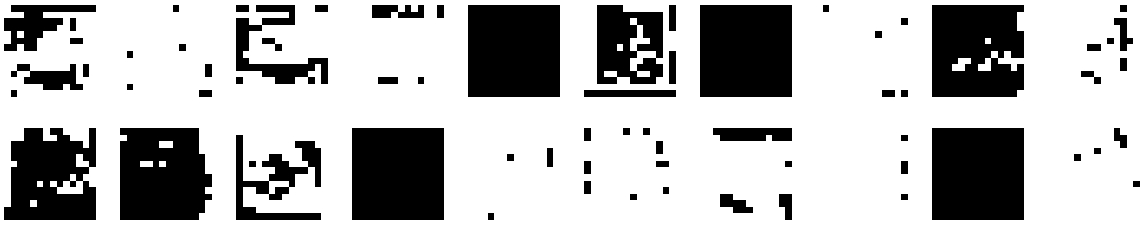

Figure 67: Feature Map of layer 2

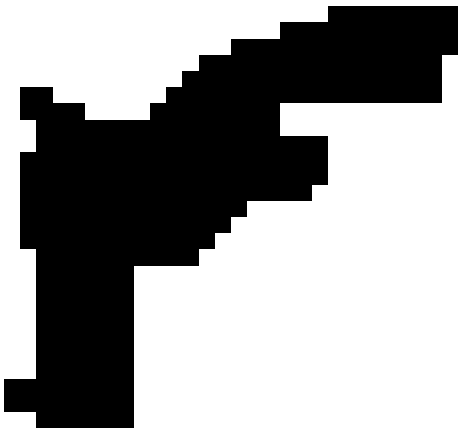

Figure 68: Input Image:Class Label 76

Figure 69: Feature Map of layer 1

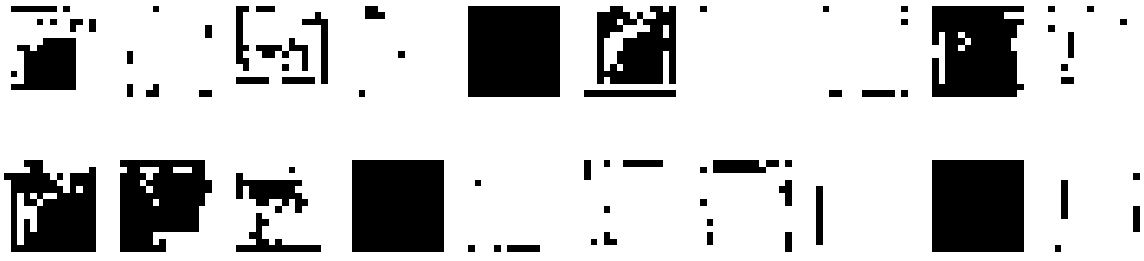

Figure 70: Feature Map of layer 2

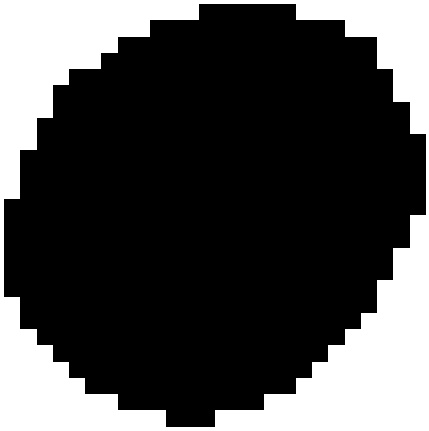

Figure 71: Input Image:Class 93

Figure 72: Feature Map of layer 1

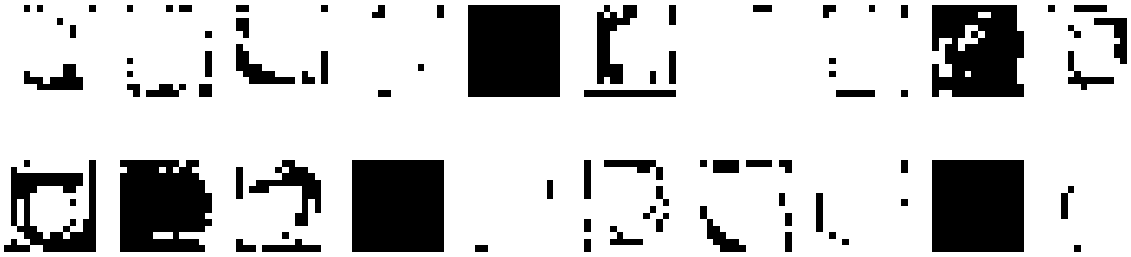

Figure 73: Feature Map of layer 2

The number of feature maps are 10 in first layer and 20 in the second layer, we display each of them for five of the output classes. It is difficult to interpret these images and output some observations from these.

Confusion matrix:

The confusion matrix for test set look like:

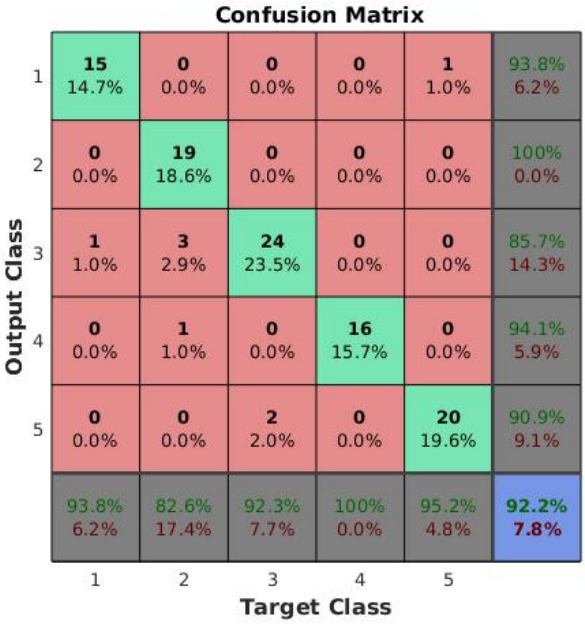

Figure 74: Figure showing confusion matrix for test set

We see that $accuracy = 92.2\%$ on test set.

4.2 MLFFNN with 2 hidden layers

MLFFNN with 2 hidden layers was trained. Cross entropy error function was used. The dataset was split into train, test and validation sets with ratio of 0.6,0.2,0.2 without any class imbalance.

Model selection: The hidden layer nodes were varied in high range(500- 600) ,mid range(300 - 500), and in low range(<300). The best model complexity using cross validation error was obtained for 350, and 200 nodes in the hidden layers 1 and 2 respectively. The test accuracy attained was 87%. The performance in neural network is less compared to CNN because CNN learns local feature, and the shift errors are overcome. And so the classification is better in case of CNN.

Confusion Matrix:

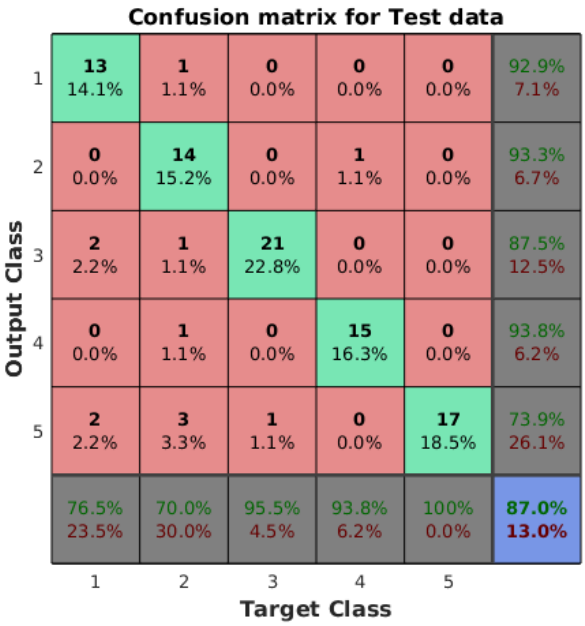

Figure 75: Confusion matrix plot for black and white dataset for test data

4.3 MLFFNN pre-trained with auto-encoder

Dataset: Dataset consists of 784 feature of 467 data points. The given data was split in the ratio of 60%, 20%, 20% of Train, Validation, Test data. The data was split without class imbalance. Model selection: MLFFNN with 3 hidden layers were to be leaned. To arrive at the best model configuration, some configurations in the high range(500 - 600 nodes), some in the middle(300 - 500 nodes), and some in the low range($<$ 200 nodes) were tried.

The model in the middle range gave minimum validation error of 86.7%, for which the test error was 91.3%.

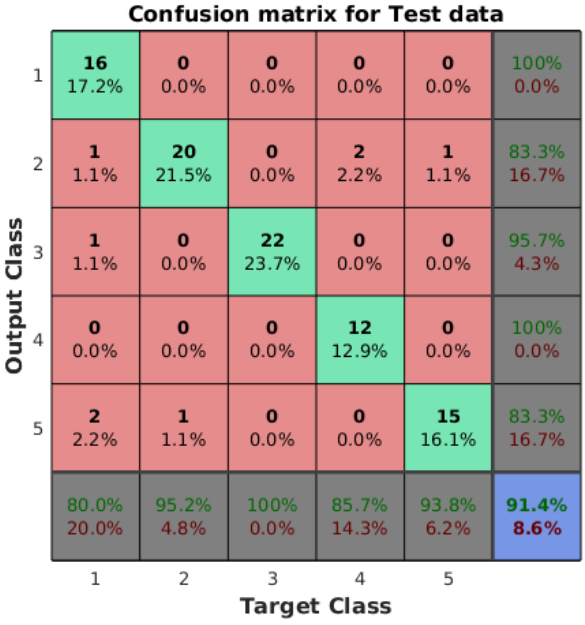

Figure 76: Confusion matrix of Test set of black and white data on MLFFNN pre-trained using auto-encoder

The pre-trained model is slightly better than the model of same configuration but without pre-training.

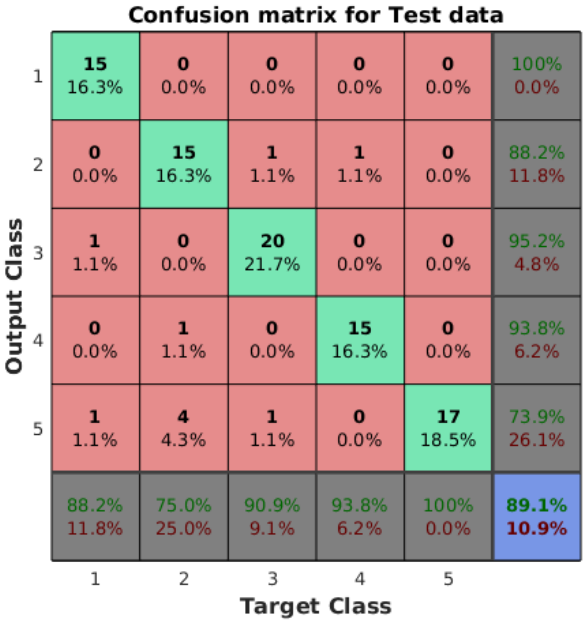

Figure 77: Confusion matrix of Test set of black and white data on MLFFNN without any pre-training

A decrease in no of iterations for convergence is observed when pretraining is done. And the train error is slightly better in case of pretraining.

4.4 MLFFNN pretrained with RBM

Model selection: 3 level stacked RBM was used. The 3 RBMs were leaned separately. The weights were fed to same configuration neural network and trained. Some configuration in high range, medium, and low range of nodes were tried. Out of which the best was selected using cross validation which was the one with 450, 400, 300 nodes in Hidden layers 1 , 2 and 3 respectively.

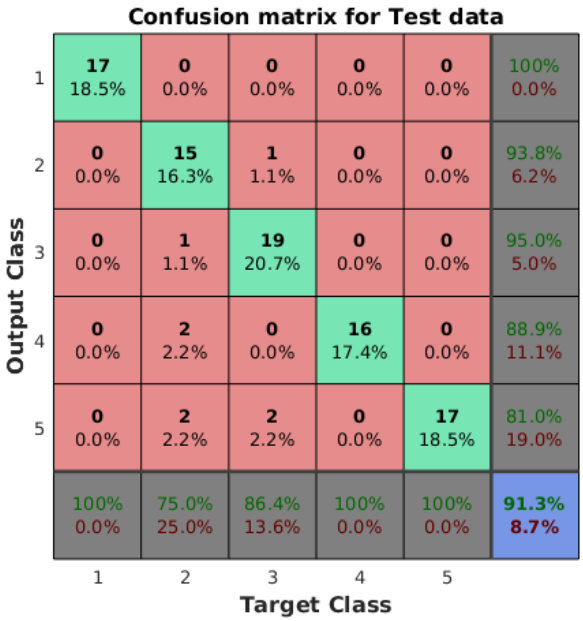

Figure 78: Confusion matrix of Test set of black and white data on MLFFNN pre-trained using stacked RBM

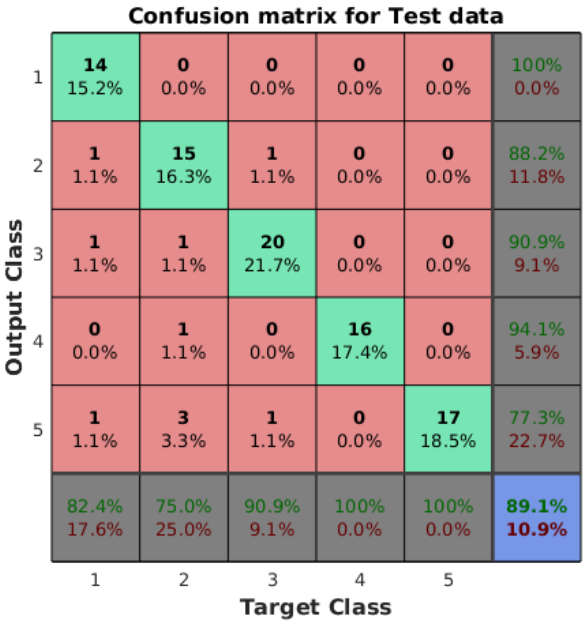

Figure 79: Confusion matrix of Test set of black and white data on MLFFNN without any pre-trained

Comparison between autoencoder and RBM

In both cases pretraining gives slightly better accuracy than without pretraining. Both methods of pretraining works similar interms of better accuracy. But when pretrained using RBM the neural network converges much faster. The reduction in no of iteration is more than 50%, which in auto encoder the decrease iterations because of pretraining is very mild. This way we can say that RBM is learning weights that are more appropriate compared autoencoder.

5 Multivariate data set

Gaussian kernel

Parameter estimation:
the best value of σ is observed to be 0.5.the kernel gram matrix is observed to be:(for $\sigma = 0.5$)

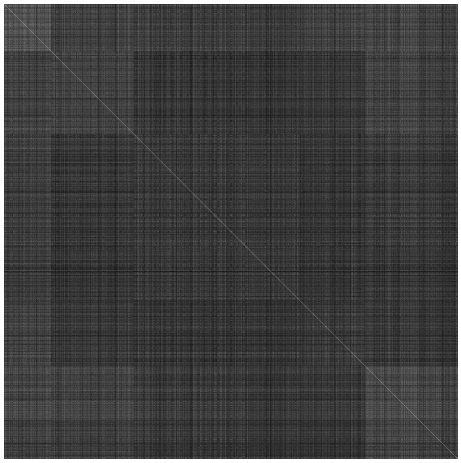

Figure 80: Figure showing grey level image of kernel gram matrix for kernel parameters $\sigma = 0.5$

We observe from the kernel gram matrix that it is far away from block diagonal matrix which means these classes are not separable it means that features used are not sufficient to capture the class of image.

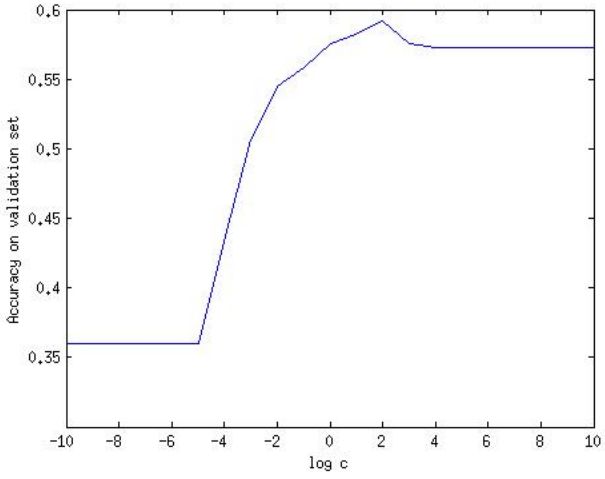

Figure 81: Figure showing accuracy for validation set for different values of c

The best c is observed to be 4.
Confusion matrix:

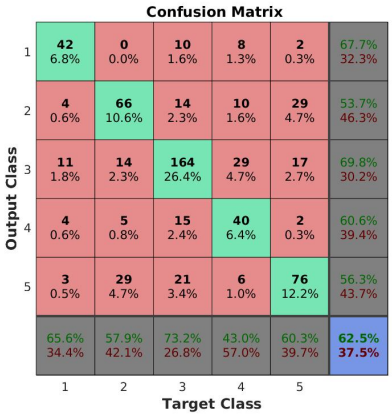

Figure 82: Figure showing confusion matrix for test set for multivariate dataset gaussian kernel for $\sigma = 0.5$,c=4

The accuracy for test set is observed to be 62.5%
We see that class1(sheep) is the class which is highest accuracy among the classes.most of the examples of class2 (dining table) are misclassified as class5(sofa) and viceversa.Similarly most of the examples in class3(car) are misclassified as class4(bicycle) and viceversa.This can also be attributed to granularity in classification.(more granular less separable).

5.1 Logistic Regression

In logistic regression the posterior probabilities of the class are ensured to be between 0 and 1 and sum over all classes is 1 by using a softmax acting on a linear function. Cross entropy was used as the error function. And the python library sci-kit learn was used for this implementation.

Parameter estimation: Logistic Regression was done on the dataset with polynomial basis function. The parameter that was estimated was degree of the basis expansion. We tried varying the degree between 1 and 3. But the number of data points were large and the number of features were 512. As the dimensions after basis expansion are given by $\binom{d+M}{M} = {}^{d+M}C_M$ where C is the combination symbol. Hence even a basis expansion of degree 2 means huge increase in the number of features and the program being too large to get results. Hence we tried reducing the number of features by running PCA and then tried to do basis expansion in order to observe non linear boundaries.

The error on the various sets vs the degree was plot. The validation set vs degree was used to select what is the best model. We notice a maximum accuracy on validation set when degree is 2. Hence model with degree 2 is chosen as the best model. But it should be noted that the PCA was done on the sets differently. The model with degree 1 was done with no PCA applied, the model with degree 2 was done with PCA of 100 features and then a basis expansion. And the model with degree 3 was done with a PCA of 10 features followed by a basis expansion. We did this because an increase in the degree even by one was leading to a huge increase in the features.

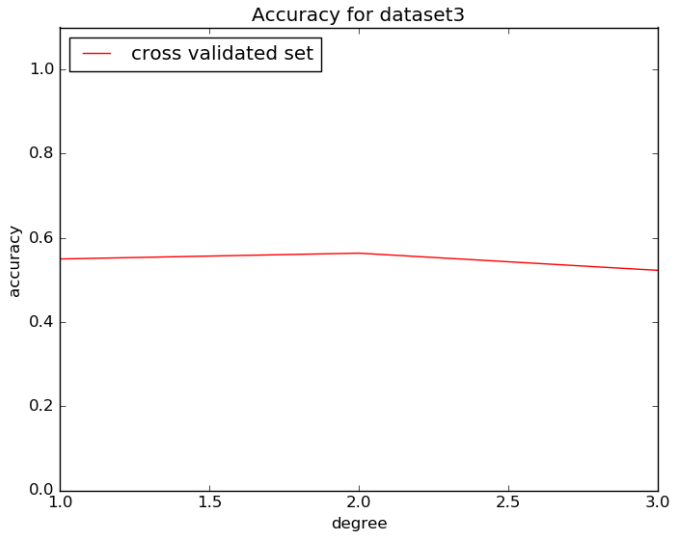

Figure 83: Figure showing accuracy for validation set for different values of degree

Confusion matrices:
Here we present the three confusion matrices for the three degrees the results obtained on these matrices were found on the cross validated sets where for each element in the input, the prediction was obtained for that element when it was in the test set.

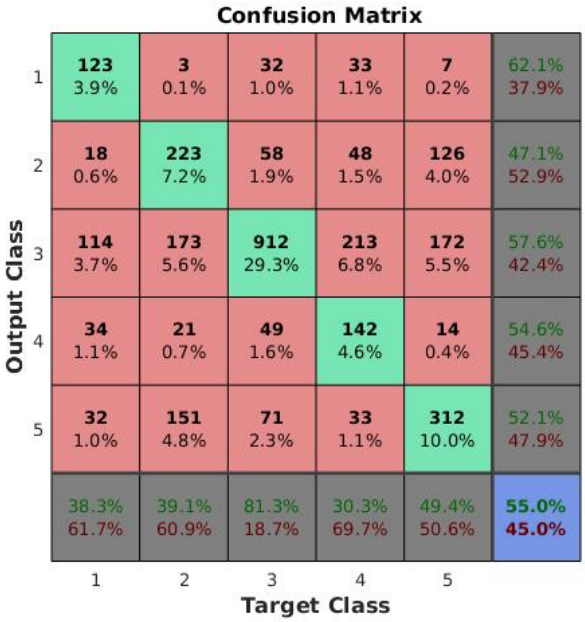

Figure 84: Figure showing confusion matrix when each input was in the test set and degree of basis expansion is 1, No Feature Reduction

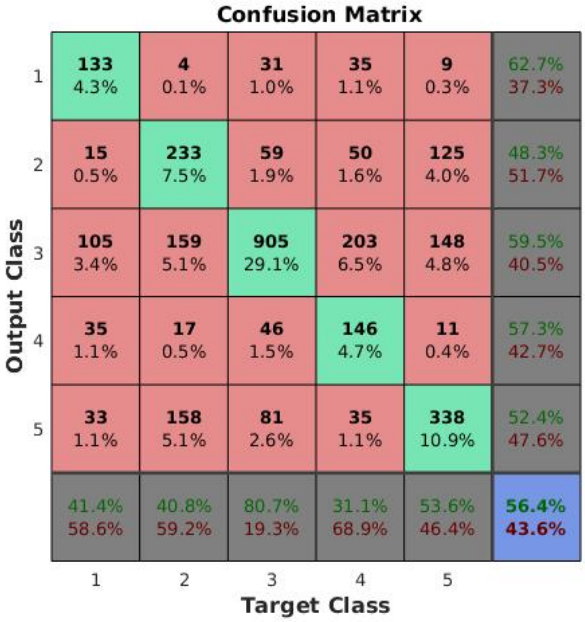

Figure 85: Figure showing confusion matrix when each input was in the test set and degree of basis expansion is 2, Feature Reduction with 100 components

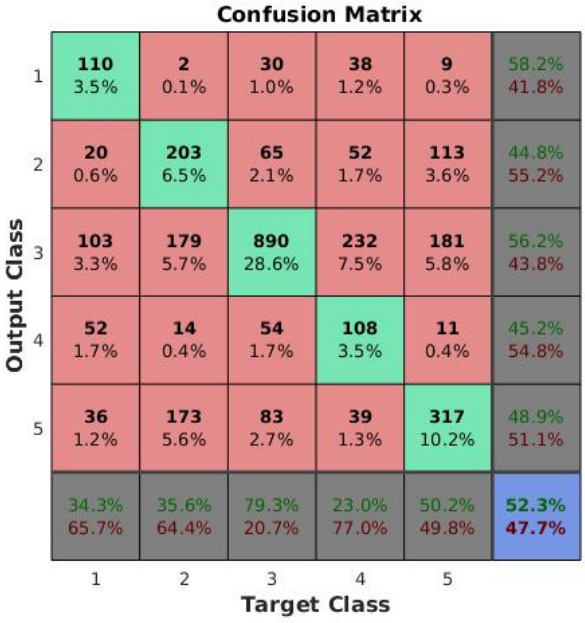

Figure 86: Figure showing confusion matrix when each input was in the test set and degree of basis expansion is 3, Feature reduction with 10 components

We find a best accuracy of about 56 percent through logistic regression with basis expansion on this dataset.

5.2 MLFFNN using 2 hidden layers

The dataset consists of 512 features, so the hidden layer nodes were varied in the range of high (400- 500), mid range(100 - 400) and low(<100). The best accuracy was obtained for 450 and 400 nodes in hidden layers 1 and 2 respectively.

Confusion Matrix:

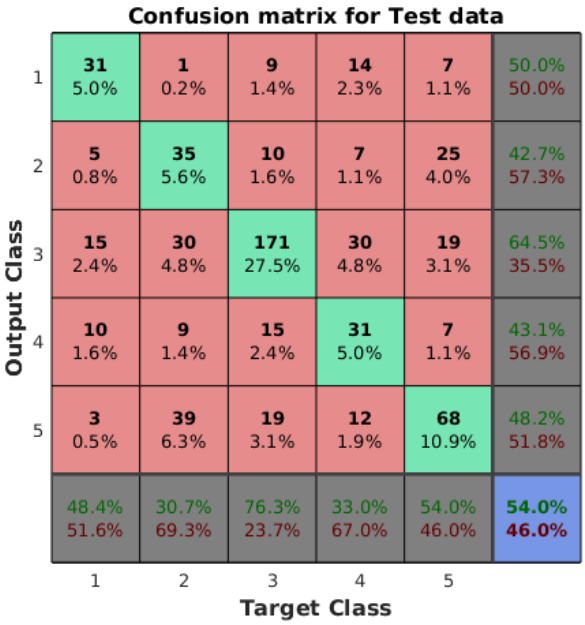

Figure 87: Confusion matrix for Multivariate dataset for test data

Accuracy obtained in case of neural network is around 54%. The accuracy is low because the features are pre-decided and fed to the neural network. The features might not be enough to distinguish between the different classes.