PATTERN RECOGNITION AND MACHINE LEARNING

Indian Institute of Technology Madras
M.Tech (Computer Science & Engineering)
SESSION 2018-2019

SUBMITTED BY:
OJAS MEHTA - CS18M038
PRANAV MURALI - CS18M041

SUBMITTED TO
Prof. C. CHANDRA SEKHAR
IIT Madras

GROUP NUMBER : 5

TASK 1:- Discrete HMM

1.METHOD

Hidden markov model is a statistical markov model in which the system being modeled is assumed to be a markov process with hidden states.

HMM model consists of:-

- N = number of states in model
- M = number of observation symbols.
- Q= $\{q_0, q_1, q_2, \ldots\ldots, q_{N-1}\}$ (Distinct states of markov model)
- B= {0,1,.....,M-1} set of observations
- A= state transition probabilities.
- B=observation probability matrix.
- π = initial state observation
- O=$\{O_0, O_1, \ldots, O_{T-1}\}$ (Set of observation sequences).

We can calculate the probability of the observation sequence given model λ i.e $P(O|\lambda)$ using forward method (inductively) as shown below:-

$$\alpha_{t+1}(j) = (\sum_{i=1}^{N} \alpha_t(i) a_{ij}) * b_j(O_{t+1})$$

Where $\alpha_t(i)$ is the probability of the partial sequence $\{O_0, \ldots, O_T\}$ and state S_i at time t given model λ.

2.RESULTS

1) **On-line Handwritten Data**:-

Training data			Validation Data		
N	**M**	**Accuracy**	**N**	**M**	**Accuracy**
11	11	81.8%	11	11	83.23%
13	12	83.3%	12	12	87.66%
14	13	85.0%	14	13	87.15%
14	14	88.2%	14	14	86.91%

Fig 1.2.1

2) **Spoken digit data(Isolated)**

Training Data			Validation Data		
N	**M**	**Accuracy**	**N**	**M**	**Accuracy**
2	2	96.20%	2	2	96.31%
4	4	100%	4	4	99.95%
10	10	100%	10	10	100%
12	12	100%	12	12	100%

Fig 1.2.2

3) **Spoken digit data(Connected)**

Test-1 Data		
N	**M**	**Accuracy**
10	10	61.50%
12	11	61.50%
12	12	69.23%
14	14	69.23%

Fig 1.2.3

Classification Accuracies on Test data:-

1)Online handwritten data:-86% (N=14,M=14)
2)Spoken-digit(Isolated):-100% (N=12,M=12)
3)Spoken-digit(Connected, Test2-Data):-
(First sequence is the original digit sequence and second sequence is the predicted digit sequence)

398->134	403->333	408->143
399->144	404->344	409->443
400->14	405->341	410->444
401->13	406->431	
402->334	407->44	

Fig 1.2.4

1) **Confusion Matrix for On-line Handwritten Data**

Training Data				Test Data			
	Class1	Class2	Class3		Class1	Class2	Class3
Class1	80	1	0	Class1	20	0	0
Class2	3	65	10	Class2	0	14	15
Class3	4	10	66	Class3	0	3	17

Fig 1.2.5

2) Confusion Matrix for Spoken digit data(Isolated)

Training Data				Test Data			
	Class1	Class2	Class3		Class1	Class2	Class3
Class1	42	0	0	Class1	10	0	0
Class2	0	42	0	Class2	0	10	0
Class3	0	0	42	Class3	0	0	10

Fig 1.2.6

3) Prediction for Test-1 Data for Spoken digit data(Connected)

First sequence is the original sequence of digits and second sequence is the predicted sequence of digits

13->13	34->34	141->111	434->434
14->14	41->413	331->331	
31->31	43->43	344->344	
33->333	44->44	413->411	

Fig 1.2.7

TASK 2:- MULTI-CLASS LOGISTIC REGRESSION

1) METHOD

Multinomial logistic regression is a classification method that generalizes logistic regression to multi class problems. It is a model that is used to predict the probabilities of the different possible outcomes of a categorically distributed dependent variable, given a set of independent variables.

Probability of an input vector $\overline{x}$ belong to class i is given by:

$$y_i = \frac{e^{a_i}}{\sum_{j=1}^{M} e^{a_j}}$$

where $a_i = \overline{w_i}^T \phi(\overline{x})$ and $\phi(\overline{x})$ can be gaussian or polynomial basis function.

To measure the error between target and predicted output we use cross entropy function given by:-

$$\varepsilon(\overline{w_1}, \overline{w_2}, ..., \overline{w_n}) = -\sum_{n=1}^{N}\sum_{i=1}^{M} t_{ni} log(y_{ni})$$

Where N=number of examples, M=number of classes.

To find the appropriate appropriate $\overline{w_i}$, we use gradient descent method. Update equation is given by:-

$$\overline{w_i}^{new} = \overline{w_i}^{old} - \eta \frac{\partial \varepsilon(\overline{w_1}, \overline{w_2}, ..., \overline{w_m})}{\partial \overline{w_i}}$$

Class of input vector is given by $argmax(y_i)$.

2) PLOTS

Polynomial basis functions

1) Linearly Separable Data:-

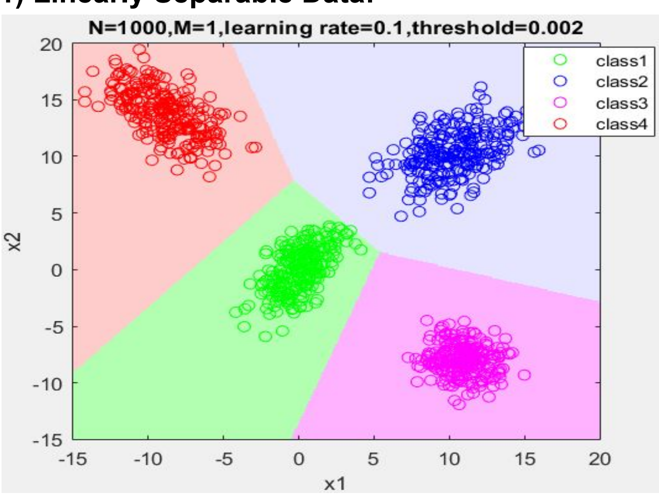


Fig 2.2.1

2) Non-Linearly Separable Data:-

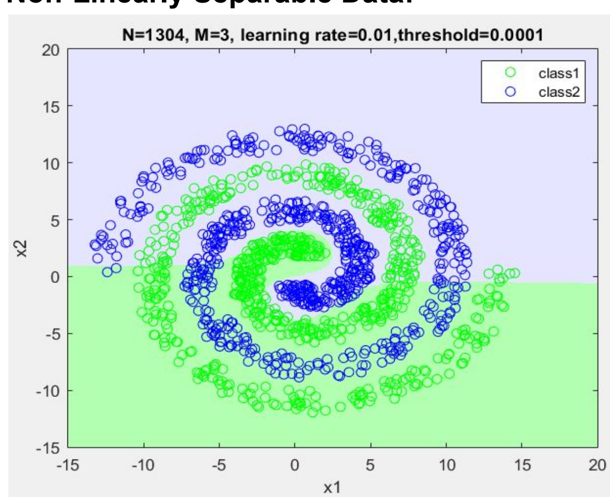


Fig 2.2.2

Gaussian Basis function

1)Linearly Separable Data:-

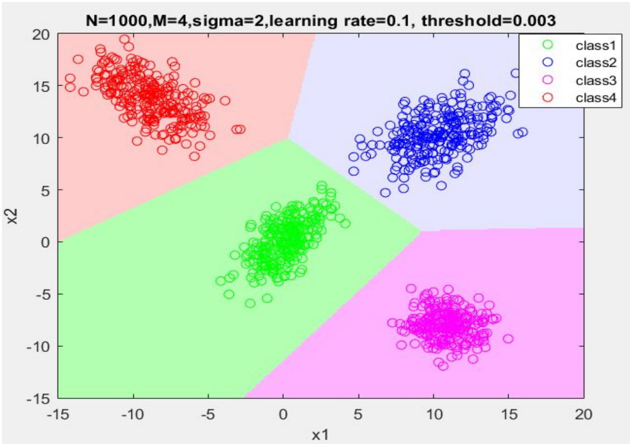


Fig 2.2.3

2)Non Linearly Separable Data:-

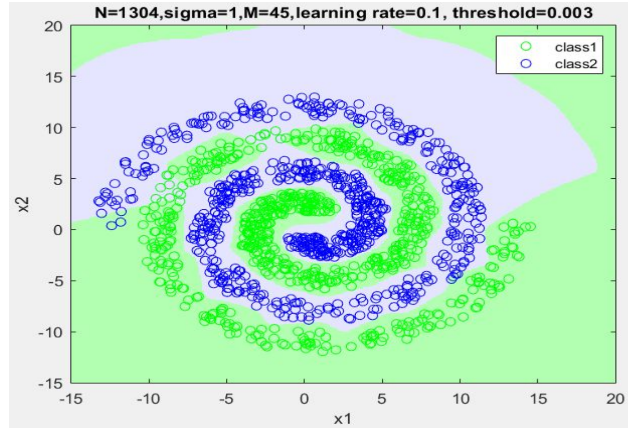


Fig 2.2.4

3) RESULTS

In Fig 2.3.1 and Fig 2.3.2 shown below, A stands for Accuracy in percentage (%),M for number of gaussian basis functions and σ for standard deviation.

For Figures 2.3.3, 2.3.4, 2.3.5, 2.3.6, C1,C2,C3 and C4 stands for class1, class2, class3 and class4 respectively.

i) Polynomial Basis function

Training Data						Validation Data					
Linearly Separable Data		Non-Linearly Separable Data		Image Data		Linearly Separable Data		Non-Linearly Separable Data		Image Data	
M	A	M	A	M	A	M	A	M	A	M	A
1	100	2	46.70	1	40.23	1	100	2	44.23	1	38.96
2	100	3	70.71			2	99.75	3	69.55		
3	100	4	53.60	2	62.87	3	99.5	4	53.58	2	61.33
4	100	6	53.52			4	99.5	6	55.56		

Fig 2.3.1

ii) Gaussian Basis function

Training Data									Validation Data								
Linearly Separable Data			Non-Linearly Separable Data			Image Data			Linearly Separable Data			Non-Linearly Separable Data			Image Data		
σ	M	A	σ	M	A	σ	M	A	σ	M	A	σ	M	A	σ	M	A
3	2	76.75	0.8	40	91.46	1	9	60.99	3	2	70.75	0.8	40	90.49	1	9	60.23
4	3	82.75	0.8	45	98.24	1.25	10	62.34	4	3	81.25	0.8	45	96.72	1.25	10	61.33
4	4	100	1	45	99.16	1.0	11	66.95	4	4	100	1	45	98.63	1.0	11	65.55
6	4	100	1.2	45	99.52	0.8	12	65.92	6	4	99.75	1.2	45	98.12	0.8	12	64.01

Fig 2.3.2

iii) Confusion Matrix for Linearly Separable Data (Polynomial Basis function)

Training Data					Test Data				
	C1	C2	C3	C4		C1	C2	C3	C4
C1	250	0	0	0	C1	100	0	0	0
C2	0	250	0	0	C2	0	100	0	0
C3	0	0	250	0	C3	0	0	100	0
C4	0	0	0	250	C4	0	0	0	100

Fig 2.3.3

iv) Confusion Matrix for Linearly Separable Data (Gaussian Basis function)

Training Data					Test Data				
	C1	C2	C3	C4		C1	C2	C3	C4
C1	250	0	0	0	C1	100	0	0	0
C2	0	250	0	0	C2	0	100	0	0
C3	0	0	250	0	C3	0	0	100	0
C4	0	0	0	250	C4	0	0	0	100

Fig 2.3.4

v) Confusion Matrix for Non Linearly Separable Data

Polynomial Basis functions						Gaussian Basis functions					
Training Data			Test Data			Training Data			Test Data		
	C1	C2		C1	C2		C1	C2		C1	C2
C1	442	210	C1	184	76	C1	652	0	C1	257	3
C2	172	480	C2	84	176	C2	7	645	C2	5	255

Fig 2.3.5

vi) Confusion Matrix for Image Data

Polynomial Basis functions								Gaussian Basis functions							
Training Data				Test Data				Training Data				Test Data			
	C1	C2	C3		C1	C2	C3		C1	C2	C3		C1	C2	C3
C1	199	34	57	C1	52	20	9	C1	193	3	87	C1	55	12	17
C2	57	166	67	C2	4	41	15	C2	16	192	82	C2	10	46	14
C3	34	20	180	C3	12	10	36	C3	28	46	160	C3	14	16	38

Fig 2.3.6

vii) Test Data Accuracy

Polynomial Basis functions			Gaussian Basis functions		
Linearly separable data	Non-Linearly separable data	Image Data	Linearly separable data	Non-Linearly separable data	Image Data
100%(M=1)	69.231%(M=3)	60.85(M=2)	100%(M=4,σ=1)	98.5%(M=45,σ=1	65.56(M=11,σ=1)

TASK 3:- Perceptron Model

1) METHOD

Perceptron is an algorithm for supervised learning of binary classifiers. Generally, Perceptron models are neural networks with single hidden layer. Multi-layer perceptron model is also known as feed-forward neural network.

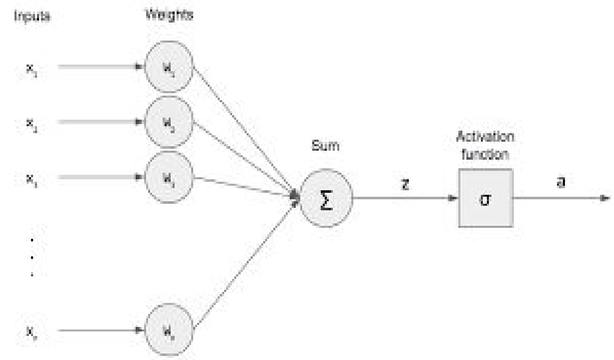
Weights of the network are updated using perceptron learning rule.

$$\overline{w}^{new} = \overline{w}^{old} - \eta(a_n - t_n)\overline{x_n}$$

where η is the learning rate.

t_n is the target output.

$a_n = \phi(z_n)$ and ϕ is activation function.



2) PLOTS AND RESULTS

1) Linearly Separable Data

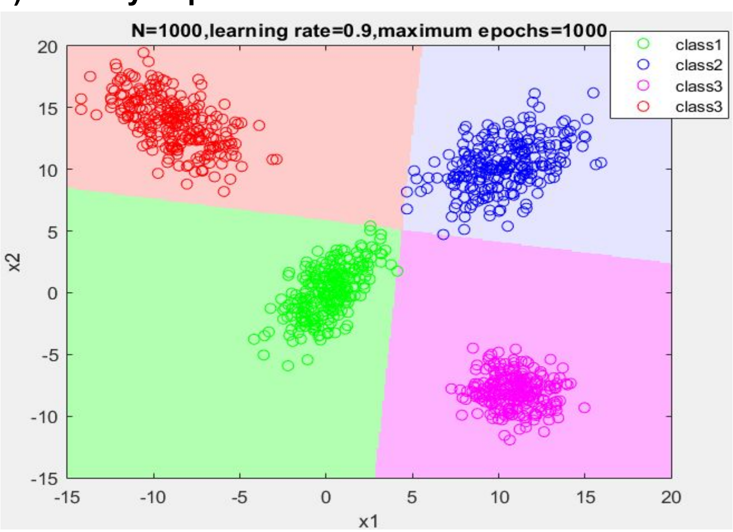


Fig 3.2.1

2) Accuracy For Training Data:-

Learning Rate	Accuracy
0.2	100%
0.5	100%
0.9	100%

Fig 3.2.2

3) Accuracy for Validation Data:-

Learning Rate	Accuracy
0.2	100%
0.5	100%
0.9	100%

Fig 3.2.3

iv) Confusion Matrix for Linearly Separable Data

Training Data					Test Data				
	C1	C2	C3	C4		C1	C2	C3	C4
C1	250	0	0	0	C1	100	0	0	0
C2	0	250	0	0	C2	0	100	0	0
C3	0	0	250	0	C3	0	0	100	0
C4	0	0	0	250	C4	0	0	0	100

Fig 3.2.4

Classification Accuracy on Test Data = 100% (Learning rate=0.9).

TASK 4:- Multi-layer feed forward network

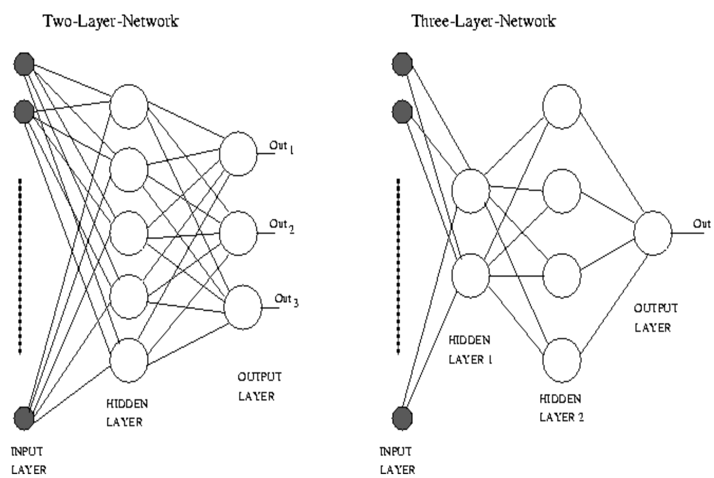
1) METHOD

Multilayer networks solve the classification problem for non-linear sets by employing “hidden layers”, whose neurons are not directly connected to the output.The additional hidden layers can be seen as additional hyperplanes which enhance the separation capacity of the network.

To compute the error at the nodes in the output layer, we can either use sum of squared error or cross-entropy error function.

Generally, if we use sum of squared error function, activation function of the output layer is logistic function and for cross entropy function, activation function at the output layer is soft-max function.

Three Layer Network



Instantaneous error(sum of squared error) for a particular input $\overline{x_n}$ is given by:-

$$\tilde{\varepsilon_n} = \frac{1}{2}\sum_{k=1}^{K}(t_{nk} - y_{nk})^2$$

Where k is total number of nodes in the output layer.

t_{nk} is the actual output of $\overline{x_n}$ at k^{th} node of the output layer

y_{nk} is the predicted output of $\overline{x_n}$ at k^{th} node of the output layer.

To update the weights of the neural network,, we can use backpropagation algorithm using pattern mode of learning. Weight update equation for weight w_{jk} from “node j” of hidden layer to “node k” of output layer equation is given by:-

$$\Delta w_{jk} = -\eta\frac{\partial\tilde{\varepsilon_n}}{\partial w_{jk}}$$

2) PLOTS AND RESULTS

In Fig 4,2.3, "A” stands for Accuracy in %, H1 and H2 stand for number of nodes in hidden layers 1 and 2 respectively. Learning rate=0.9.

1)Linearly Separable Data

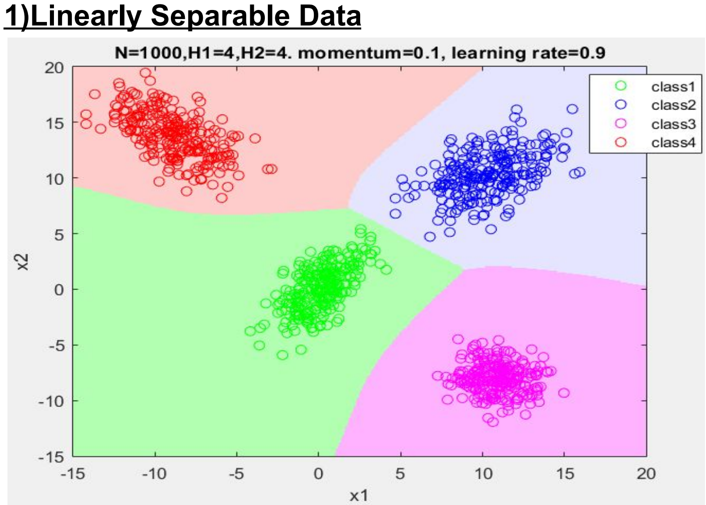


Fig 4.2.1

2)Non-Linearly Separable Data

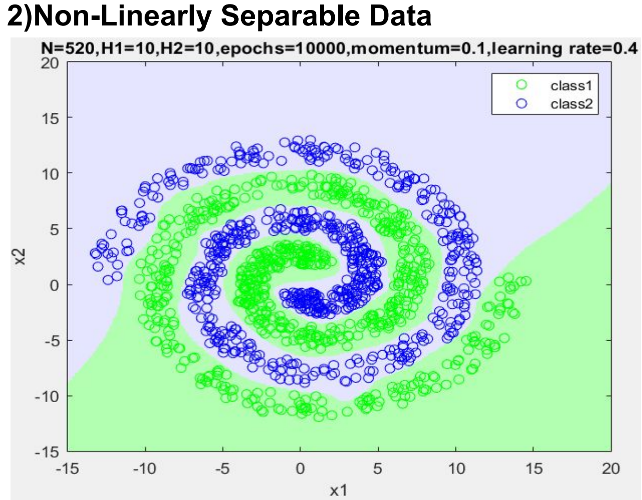


Fig 4.2.2

i) Classification Accuracy for training and Validation Data:-

Training Data									Validation Data								
Linearly Separable Data			Non-Linearly Separable Data			Image Data			Linearly Separable Data			Non-Linearly Separable Data			Image Data		
H1	H2	A	H1	H2	A	H1	H2	A	H1	H2	A	H1	H2	A	H1	H2	A
3	3	100	8	8	93.62	100	100	86.55	3	3	100	8	8	92.18	100	100	86.33
4	4	100	10	10	100	150	150	93.12	4	4	100	10	10	100	150	150	92.72
5	5	100	10	10	100	175	175	93.67	5	5	100	11	11	100	175	175	92.73

Fig 4.2.3

ii) Confusion Matrix for Linearly Separable Data

Training Data					Test Data				
	C1	C2	C3	C4		C1	C2	C3	C4
C1	250	0	0	0	C1	100	0	0	0
C2	0	250	0	0	C2	0	100	0	0
C3	0	0	250	0	C3	0	0	100	0
C4	0	0	0	250	C4	0	0	0	100

Fig 4.2.4

iii) Confusion Matrix for Non-Linearly Separable Data

Training Data			Test Data		
	C1	C2		C1	C2
C1	652	0	C1	260	0
C2	0	652	C2	0	260

Fig 4.2.5

iv) Confusion Matrix for Image Data:-

Training Data				Test Data			
	C1	C2	C3		C1	C2	C3
C1	264	10	16	C1	81	1	2
C2	6	271	13	C2	4	65	1
C3	8	3	223	C3	8	0	50

Fig 4.2.6

iii)Value of Hidden and Output layer Nodes at different epochs

Epochs	H1-1	H1-2	H2-1	H2-2	O1-1	O1-2
1	-0.9875	0.9442	-0.1481	-0.9172	0.00035	0.9639
2	-0.9874	0.9442	-0.1409	0.9171	0.00038	0.9616
10	-0.9875	0.9443	-0.0707	0.916	0.0013	0.9346
50	-0.9864	0.9446	0.2072	0.9147	0.1985	0.5181
10000	-0.9444	0.8922	-0.2295	0.8325	0.0023	0.8997

Fig 4.3.4

iv) Test Data Accuracy on Best model of data:-

- Linearly Separable Data:- 100% (H1=10,H2=10, learning rate=0.9)
- NonLinearly Separable Data:-100% (H1=13,H2=13,learning rate=0.4)
- Image Data:-92.4528%(H1=150,H2=150, learning rate=0.9)

TASK 5:- SVM

1) METHOD

objective

$$\min_{\mathbf{w},b,\xi} \quad \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i$$

$$\text{subject to} \quad y_i(\mathbf{w}^T\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i,$$

$$\xi_i \geq 0.$$

Kernel functions:

- linear: $K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T\mathbf{x}_j$.
- polynomial: $K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma\mathbf{x}_i{}^T\mathbf{x}_j + r)^d$, $\gamma > 0$.
- radial basis function (RBF): $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$, $\gamma > 0$.

Libsvm (conventions):

-s svm_type 0 -- C-SVC
-t kernel_type : set type of kernel function (default 2)
    0 -- linear: u'*v
    1 -- polynomial: (gamma*u'*v + coef0)^degree
    2 -- radial basis function: exp(-gamma*|u-v|^2)
-d degree : set degree in kernel function (default 3)
-g gamma : set gamma in kernel function (default1/num_features)
-r coef0 : set coef0 in kernel function (default 0)
-c cost : set the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
-usv: unbounded support vector
-bsv :bounded support vector.

2) PLOTS AND RESULTS

1) Linearly Separable Data

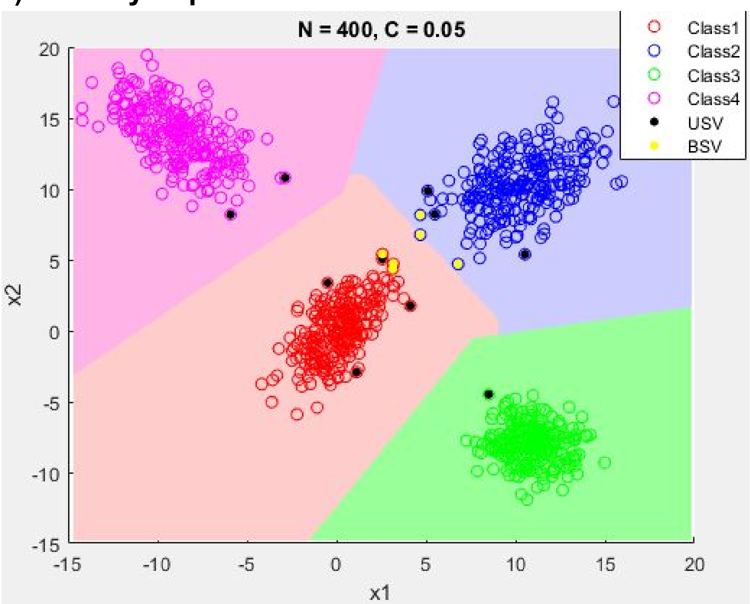


Fig 5.2.1

2) Accuracy For Training Data:-

Cost(C)	Accuracy
0.05	100%
1	100%
100	100%

Fig 5.2.2

3) Accuracy for Validation Data:-

Cost(C)	Accuracy
0.05	100%
1	100%
100	100%

Fig 5.2.3

i) <u>Confusion Matrix for Linearly Separable Data</u>

Training Data					Test Data				
	C1	C2	C3	C4		C1	C2	C3	C4
C1	250	0	0	0	C1	100	0	0	0
C2	0	250	0	0	C2	0	100	0	0
C3	0	0	250	0	C3	0	0	100	0
C4	0	0	0	250	C4	0	0	0	100

Fig 5.2.4

Classification Accuracy on Test Data = 100% for C=0.05.

2) Nonlinearly Separable Data

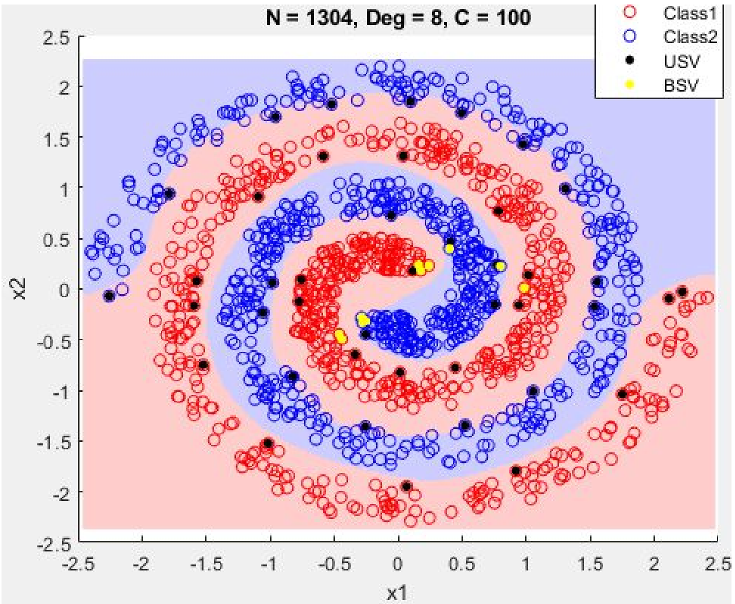


Fig 5.2.5: polynomial kernel

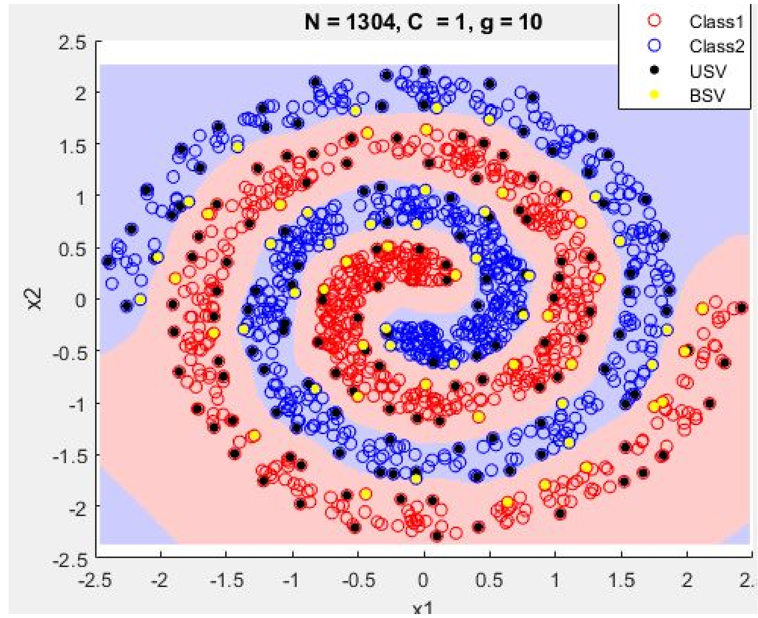


Fig 5.2.6:RBF kernel

I) <u>Accuracy For Training and Validation Data:-</u>

C	Degree	Training Accuracy	Validation Accuracy
1	7	95.9356%	92.711%
1	10	99.7699%	99.3606%
1	11	100%	99.4885%
100	7	99.8466%	99.3606%
100	8	100%	100%
0.1	11	99.2331%	98.3376%
0. 1	7	79.8313%	76.2148%

Fig 5.2.7: polynomial kernel

C	Gamma	Training Accuracy	Validation Accuracy
1	0.5	69.862%	63.1714%
1000	0.5	99.0798%	99.1049%
10000	0.5	100%	100%
1	1	95.092%	82.7366%
700	1	100%	100%
0.1	10	100%	100%
1	10	100%	100%

Fig 5.2.8: RBF kernel

ii)<u>Confusion Matrix:</u>

Training Data			Test Data		
	C1	C2		C1	C2
C1	652	0	C1	260	0
C2	0	652	C2	0	260

Fig 5.2.9: polynomial kernel

Training Data			Test Data		
	C1	C2		C1	C2
C1	652	0	C1	260	0
C2	0	652	C2	0	260

Fig 5.2.10: RBF kernel

Classification Accuracy on Test Data for best model is 100% for both the models having Polynomial kernel with C = 100, degree = 8 & RBF kernel with g = 10 C = 1.

3) Image Dataset

i)Accuracy For Training and Validation Data:-

C	Degree	Training Accuracy	Validation Accuracy
100	2	80.1676	82.3529
100	3	82.5419	84.8039
100	4	84.0782	85.2941
1000	3	84.9162	85.7843

Fig 5.3.1: polynomial kernel

C	Gamma	Training Accuracy	Validation Accuracy
1	0.5	69.862%	63.1714%
1000	0.5	99.0798%	99.1049%
10000	0.5	100%	100%
1	10	100%	87.7451%

Fig 5.3.2: RBF kernel

ii)Confusion Matrix:

Training Data				Test Data			
	C1	C2			C1	C2	C3
C1	188	12	51	C1	32	1	4
C2	1	178	25	C2	2	20	8
C3	12	7	242	C3	0	3	36

Fig 5.3.3: polynomial kernel

Training Data				Test Data			
	C1	C2	C3		C1	C2	C3
C1	251	0	0	C1	36	0	1
C2	0	204	0	C2	0	17	13
C3	0	0	261	C3	2	1	36

Fig 5.3.4: RBF kernel

Classification Accuracy on Test Data for best model is
1)Polynomial kernel with C = 100, degree 8 = 83.01%
2)RBF kernel with g = 10 C = 1 = 83.9623%

4) <u>On-line Handwritten Data:</u>- C-SVM with Linear Kernel is used for this dataset

i)<u>Accuracy For Training and Validation Data</u>:-

C	Training Accuracy	Validation Accuracy
0.2	74.8792%	72.8814%
0.8	88.8889%	86.4407%
1	88.8889%	88.1356%
10	98.5507%	93.2203%

Fig 5.4.1

ii)<u>Confusion Matrix:</u>

Training Data				Test Data			
	C1	C2			C1	C2	C3
C1	67	2	1	C1	11	0	0
C2	7	55	5	C2	0	7	4
C3	4	4	62	C3	0	0	10

Fig 5.4.2

Classification Accuracy on Test Data for best model is 87.5% for Linear kernel with C = 0.8

INFERENCES

1) **Linearly Separable Data:-**
   The accuracies using Logistic Regression, Perceptron, MLFFNN and C-SVM with linear, polynomial and RBF kernel is 100%. Therefore the model with less complexity should be prefered. Hence, C-SVM with linear kernel is best for this type of data. Logistic regression being more sensitive to outliers does not have as good generalizing ability as does C-SVM.

2) **NonLinearly Separable Data:-**
   In Logistic regression accuracy was only 69.231%. In gaussian basis functions accuracy is 98.5.Although accuracy using gaussian basis functions is good, it takes considerable amount of time in computation Both MLFFNN and C-SVM gave 100 % accuracy. Since MLFFNN (with h1=13 and h2=13) took much more time to converge (around 6000 epochs) and C-SVM with RBF kernel(with gamma = 10 and C=1) function was much more faster and hence should be more preferable.

3) **Image Dataset:**
   In logistic regression with polynomial basis = 60.85% and with gaussian basis is 65.56%. With feedforward neural network the accuracy is 92.85%. With C-SVM the best accuracy obtained was 83.96%. So for image classification, multilayer feedforward neural network is more preferable because it yields higher accuracy.

4) **On-line Handwritten Data:-**
   Accuracy using SVM is 87.5% and accuracy using HMM is 86%. As the difference between the accuracies is very less, it would be preferable to use the model with less complexity. So using SVM would be preferable.

5) **Spoken digit data(Isolated and Connected):-**
   In case of spoken digit data, we are given a set of features vectors representing the voice signals. Here the sequence of the vectors matter as it is speech signal. So HMM would be a good candidate in solving this problem.