Audio Recognition using FeedForward Neural Network optimized by Principle Component Analysis

Supervisor: Dr. Jia Uddin

Abdullah	13301061
Nusrat Suzana Momo	13301059

Department of Computer Science and Engineering,

BRAC University.

Submitted on: 18th April 2017

DECLARATION

We, hereby declare that this thesis is based on the results found by ourselves. Materials of work found by other researcher are mentioned by reference. This Thesis, neither in whole or in part, has been previously submitted for any degree.

Signature of Supervisor

Signature of Author

Dr. Jia Uddin

Abdullah

Nusrat Suzana Momo

ACKNOWLEDGEMENTS

All thanks to Almighty ALLAH, the creator and the owner of this universe, the most merciful, beneficent and the most gracious, who provided us guidance, strength and abilities to complete this research.

We are especially thankful to Dr. Jia Uddin, our thesis supervisor, for his help, guidance and support in completion of my project. We also thankful to the BRAC University Faculty Staffs of the Computer and Communication Engineering, who have been a light of guidance for us in the whole study period at BRAC University, particularly in building our base in education and enhancing our knowledge.

Finally, we would like to express our sincere gratefulness to our beloved parents, brothers and sisters for their love and care. We are grateful to all of our friends who helped us directly or indirectly to complete our thesis.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In our proposed model we have used PCA as dimension reduction technique and neural network for pattern recognition. Our goal was to recognize audios of two vowels spoken by Parkinson’s disease Patient. The vocal of these patients becomes unclear in later stage of the disease, therefore understanding them becomes difficult and hence our model is targeted to help them communicate. PCA was run to get the finest number of features to train the classifier. The classifier takes 30 percent of the feature to train and the rest 70% for testing and validation. Our model has yield a very high accuracy compared to other models.

CHAPTER 01

INTRODUCTION

1.1 Motivations

Parkinson's sickness (PD) can impact a man's voice, making them talk gradually or encounter issues surrounding sounds clearly. Individuals may not think about these and distinctive changes to the voice, yet a large number individuals with PD will experience them at some point or another over the traverse of the disease. So also as PD impacts improvement in various parts of the body, it in like manner impacts the muscles in the face, mouth and throat that are used as a piece of talking. Besides, many individuals with PD fight to find words, subsequently they may talk bit by bit. Moreover, in various cases, PD makes people Sutter. In light of this their addresses turn out to be less justifiable for other individuals who are not experiencing PD. Thus it turns out to be truly hard to recognize the letters particularly the vowels in their talks.

So separating the elements just from their speeches won't give an ideal outcome with high accuracy. Along these lines we have to choose the features which are more distinguishable, to help us recognize the vowels with most elevated accuracy. This features will help the classifier to perceive the vowels effectively. Thus our proposed model will help communicate with the patients and understand their words properly.

1.2 Contribution Summary

The summary of the main contributions is as follows:

- Extract the best features for 2D audio signal
- Apply dimension reduction technique to create optimal feature vector.
- Use Classifier to recognize the correct vowel.

1.3 Thesis Orientation

The rest of the thesis is organized as follows:

- Chapter 02 includes the necessary background information regarding the used algorithm.
- Chapter 03 presents our proposed model and its implementation.
- Chapter 04 demonstrates the experimental results and comparison.
- Chapter 05 concludes the thesis and states the future research directions.

CHAPTER 02

BACKGROUND INFORMATION

2.1 Principle Component Analysis (PCA)

PCA is intensely utilized as a part of highlight determination technique. The determined discernable elements are called as highlight vectors. Another property of PCA is this is a worldwide calculation and has the best recreation property which preferably imply that loss of imperative data is very far-fetched. So to reduce dimensionality PCA can be utilized with no loss of data. It diminishes the measurements by expelling immaterial parts and cutting down the information into its unique parts. Rather than processing on a typical x-y pivot it is for the most part accommodating to figure information concerning its primary segments. Vital parts are an arrangement of estimations of straightly uncorrelated factors changed from an arrangement of connected factors. Central parts are headings demonstrating where the information is most spread out and has the greatest fluctuation. PCA dependably finds an arrangement of commonly orthogonal pivot so that the primary main segment has the greatest fluctuation and each after part in this way has the greatest difference possible et cetera. The later chief parts are evacuated for diminishing the quantity of measurements. The idea of eigenvector and eigenvalue is essential in PCA investigation. PCA can be connected by the deterioration of eigenvalues of an information covariance lattice as a rule after mean focusing and normalizing the information network for each characteristic [1]. PCA technique simplifies a large and complex dataset to a much lower dimension. It takes the data then finds a different set of axis just like regular axis but they need to be mutually orthogonal. It lines up the variance of the data with those axis so that we can drop the least significant ones and that gives a way to do feature selection hence the name feature transformation algorithm. So PCA follows several steps. From the whole dataset consisting of d-dimensional samples PCA calculates the d-dimensional mean vector and the covariance matrix of the large dataset. After that it calculates the eigenvectors (e1,e2,..en) of A (if A is an nx nmatrix) known as orthonormal vectors and its corresponding eigenvalues (λ1, λ2,…, λn).. Then these eigenvalues are ordered in terms of significance from highest to lowest. And finally lesser significance principal components are discarded. [2]

Covariance is used to determine a relationship between dimensions among datasets mainly between two dimensions. The covariance matrix will be a diagonal matrix as covariance is non-negative value. A direction in d-dimensional space is chosen by PCA along which X has the maximum variance. This process is repeated and PCA finds another direction with maximum variance Thus n directions are chosen and principal components are identified by the result set. [2]

2.2 FeedForward Neural Network (FNN)

Feedforward neural systems are all inclusive approximators. Without a doubt, a three-or four-layered feedforward neural system, which is just permitted associations between neighboring layers, can accomplish any constant mapping utilizing sigmoid-like soaking unit yield capacities when there are endlessly many shrouded units [3,4,5]. Considering this, it is clear that with unendingly many concealed units a four-layered feedforward neural system is proportional to a three-layered feedforward neural system. Genuine applications, in any case, confine use to feedforward neural systems having a limited number of concealed units The execution of a system can be separated into two principle classes, one in light of the speculation abilities of the organize and the other in light of its mapping abilities. Villiers also, Barnard [6] concentrated the speculation capacities of three-and four-layered feedforward systems given the same number of weights for grouping errands. Their decision was against the utilization of four-layered systems in everything except the most recondite applications. The architecture of the ANN was formed by an input and an output layer and a series of hidden layers, each of which was formed by a determined number of nodes. A node is defined mathematically as follows [7]:

$$y = f\ (\Sigma iwjixi + \theta i) \tag{1}$$

here x and y are the inputs and outputs respectively of the j^{th} node, the weights for each input is determined by wji, θi is the bias/threshold value and f (x) is referred to as the activation function [8].

$$f\ (x) = 1/(1 + e - x) \tag{2}$$

This function was used to introduce a non-linearity in the predictable output. The output layer also contained a node, whose transfer function was a linear function instead of a sigmoid function, meaning that all the non-linear calculations occurred within the hidden layers.

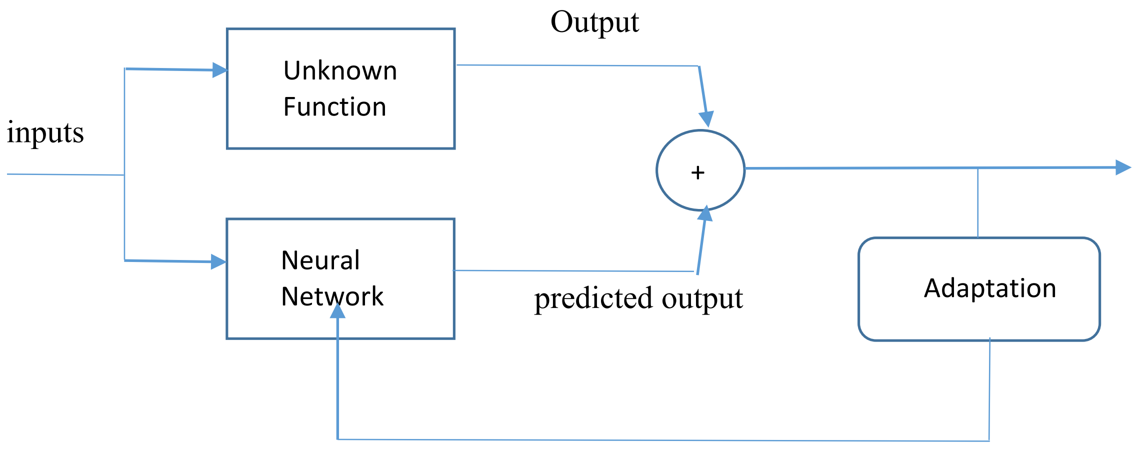


Figure 2.1 Neural network as function approximator.

As shown in Figure 1, we have some unknown function that we wish to approximate. We want to adjust the parameters of the network so that it will produce the same response as the unknown Function, if the same input is applied to both systems.

CHAPTER 03

PROPOSED MODEL

3.1 Introduction

Figure 2 demonstrates a detailed implementation of our proposed model. It demonstrates how the algorithm is set up. Initially we extracted 20 features from two sets of audio signals, one set contains audio recording of letter ‘a’ and the other set contained letter ‘o’. Then we did dimension reduction using PCA which lead to optimal feature selection. Finally, using neural network to train and test our model.

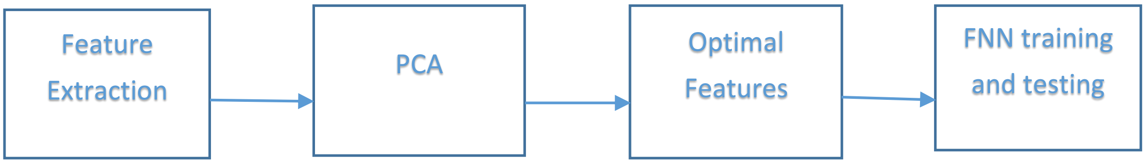


Fig 3.1: Our Proposed model

3.2 Optimizing Feature Extractions

Sound signs have numerous properties like amplitudes, most extreme amplitudes, frequencies and length of sound and some more. Features are numeric properties of a flag that could possibly be one of a kind. It is a quantifiable property that has been seen in wonder [9]. Diverse signs have distinctive components, which could possibly be one of a kind. Like for instance, there can be two totally isolate signals with same normal adequacy. Subsequently it is imperative to extricate however much elements as could reasonably be expected so that the most unmistakable components among them can be worked out. As these measurements assume a critical part in example acknowledgment and machine learning, having however many as would be prudent expands that precision of foreseeing the right letters, which prompt the extraction of the accompanying 20 more: root mean square (rms), standard deviation, kurtosis, Energy Entropy, Signal Energy, Zero Crossing Rate, Spectral Rolloff, Spectral Centroid, Spectral Flux, maximum amplitude, minimum amplitude, mean amplitude, median amplitude, mean frequency, median frequency, skewness, peak to peak-the difference between maximum and minimum peak, peak to rms, root sum of square (rssq) and variance. These elements alongside the past 6 highlights made the letters more discernable by the classifiers. Clearly utilizing every one of the components together does not give the most elevated precision in light of the fact that there are elements that have values, which are regular to both sounds. Therefore we had to use PCA to find the optimal features for training for classifiers. The following way is used for extracting best features. If you use

[V] = pca(M)

Where M is 140x20 matrix containing all 20 features of 140 audios, the output is one argument, it will return the principal coefficients, sometimes called the loadings. The 20x140 matrix you received contains the first loading in the first row, the second in the second row and so on. If you ask for two outputs, you obtain

[V, U] = pca(X)

where V contains the loadings and U the score values. You reconstruct the input data by U*V'.

In order to perform dimensionality reduction, you must select the first n components of both matrices as U(:, 1:n) and V(:, 1:n) and perform the approximated reconstruction as:

U(:, 1:n)*V(:, 1:n)'

3.3 Training Classifier

We used Matlab's built-in feedforward neural network. The app takes a feature vector and another matrix that has the labels of the feature vector. The feature vector is a matrix with the features as rows and the sample values as the column values. The label is a 2x140 matrix, with the row denoting label and the column denoting true or false for the samples. We have used 30 percent of the features to train the classifiers and used the rest 70 percent for testing and validation. We started with 3 features from our feature vector and incremented one feature at a time until we achieved a high accuracy.

CHAPTER 04

EXPERIMENTAL ANALYSIS

The datasets used in this research are voice recordings of 28 patients saying the letters ‘a’ and ‘o’ [9]. We have retrieved the following graphs: performance graph, error histogram, receiver operating characteristic (ROC) and confusion matrix for evaluating our model. We will use the results of 3 features to explain the graphs.

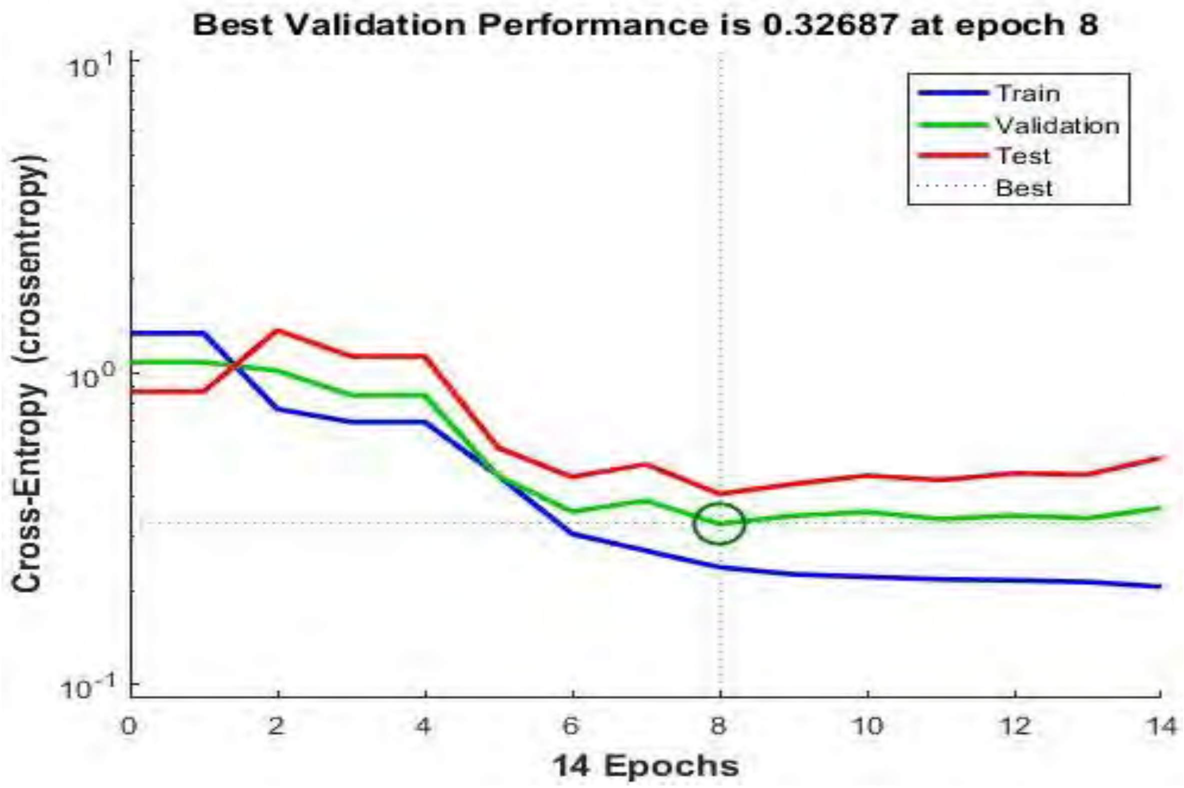


Fig 4.1 Performance Graph of 3 features.

Figure 4 shows the performance graph when using 3 features. It tells us how much overfitting was done to match the data. It also tells us the best validation performance which in this case is 0.32687.

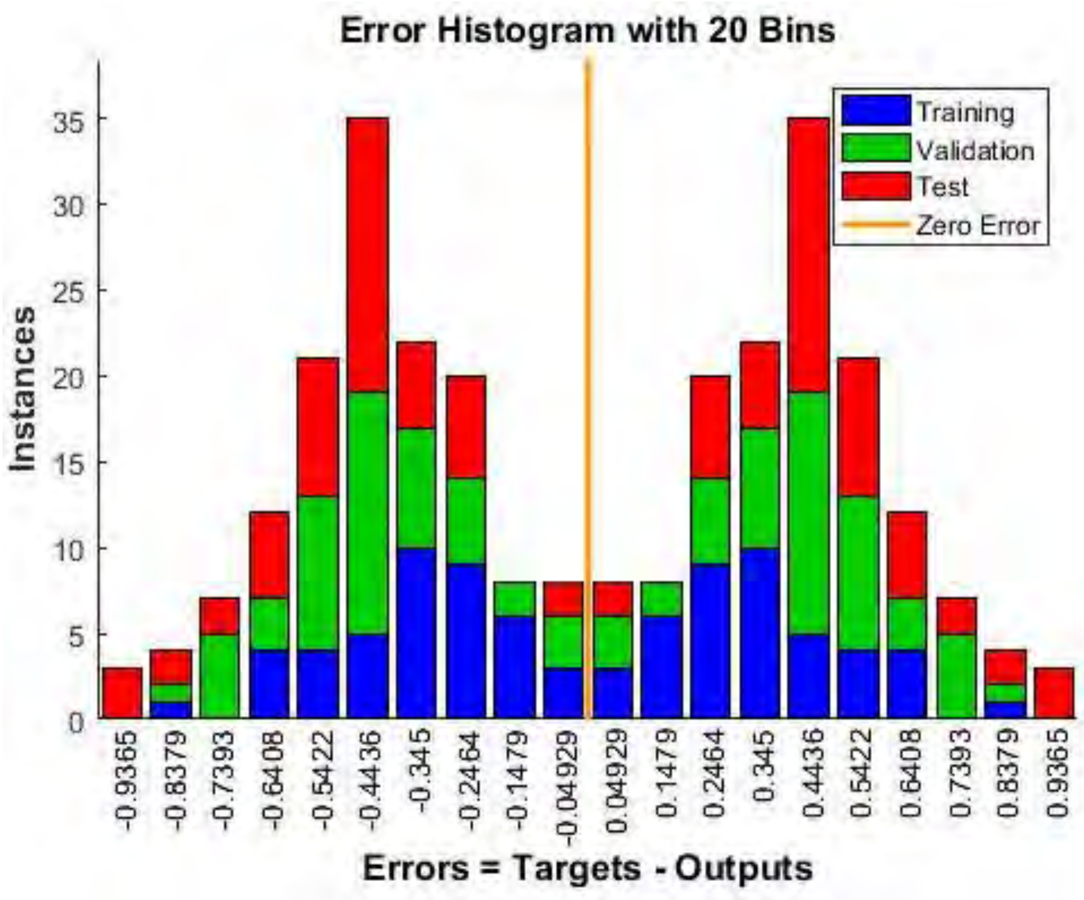


Fig 4.2 Error Histogram

The Figure 5 shows an error histogram which indicates the range of errors and the data that are being overfitted, like the ones at 0.9365 and -0.9365.

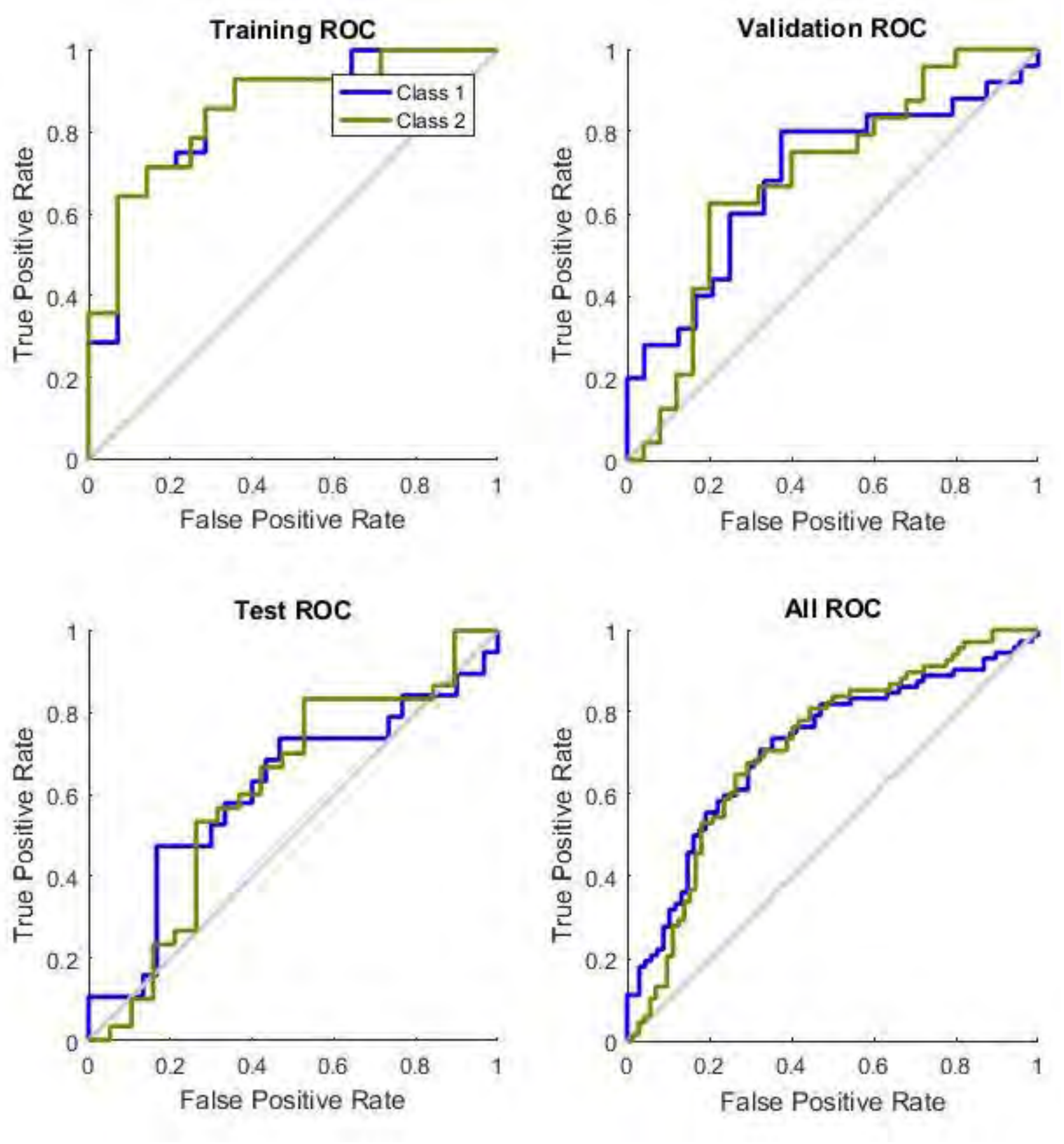


Fig 4.3 Reciever Operating Characteristics

Figure 6 shows the ratio of true positive to false positive for both cases. For a better classifier the graphs need to be closer to y-axis, true positive. In case of using only 3 features does not give good results.

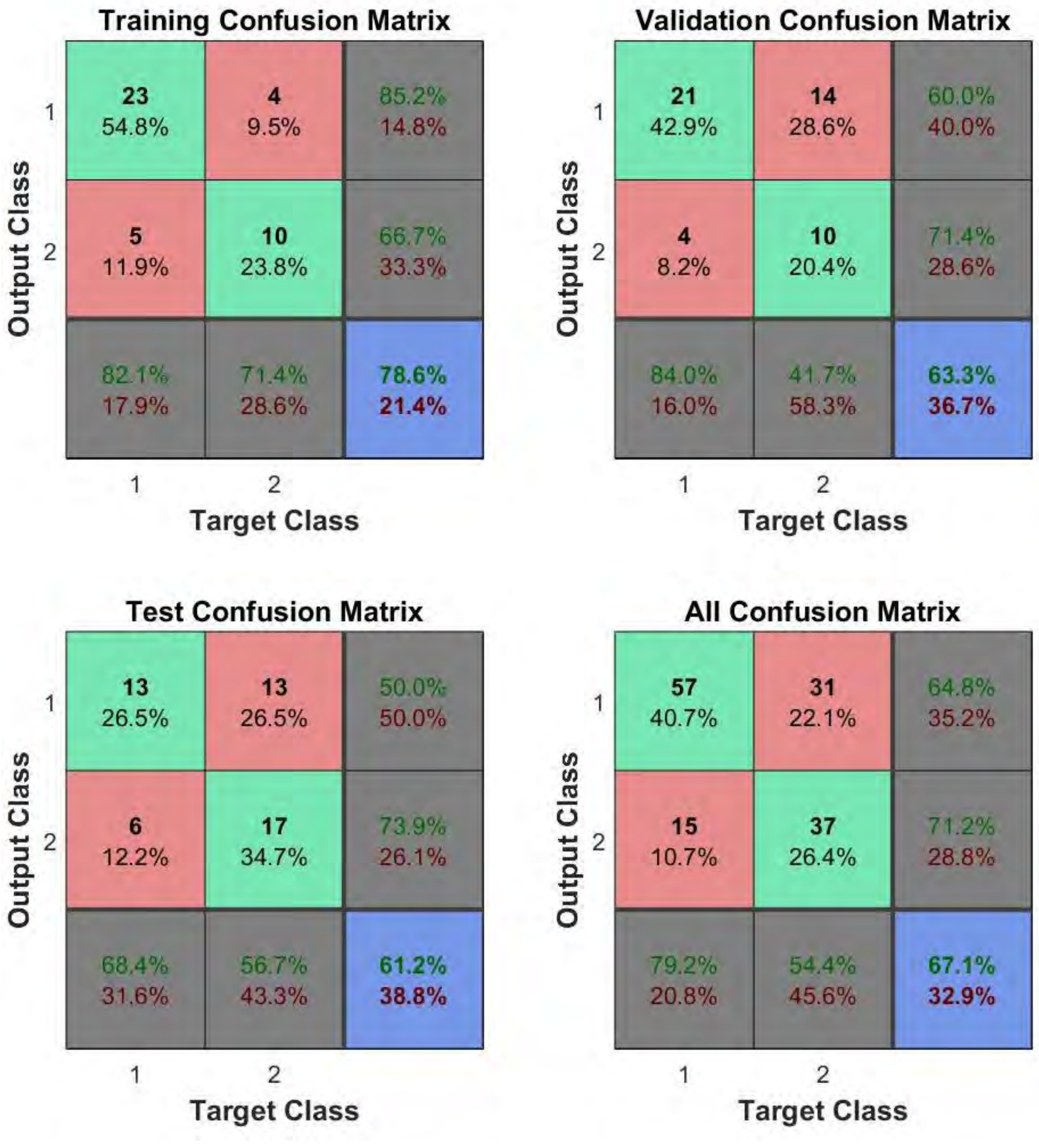
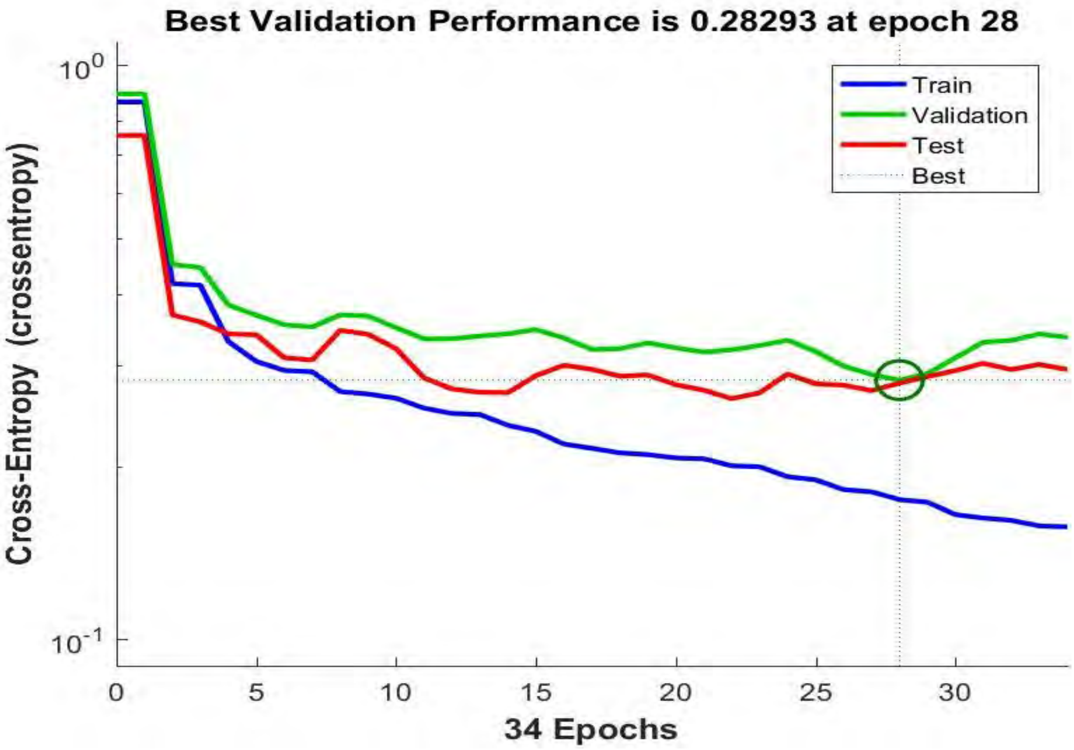


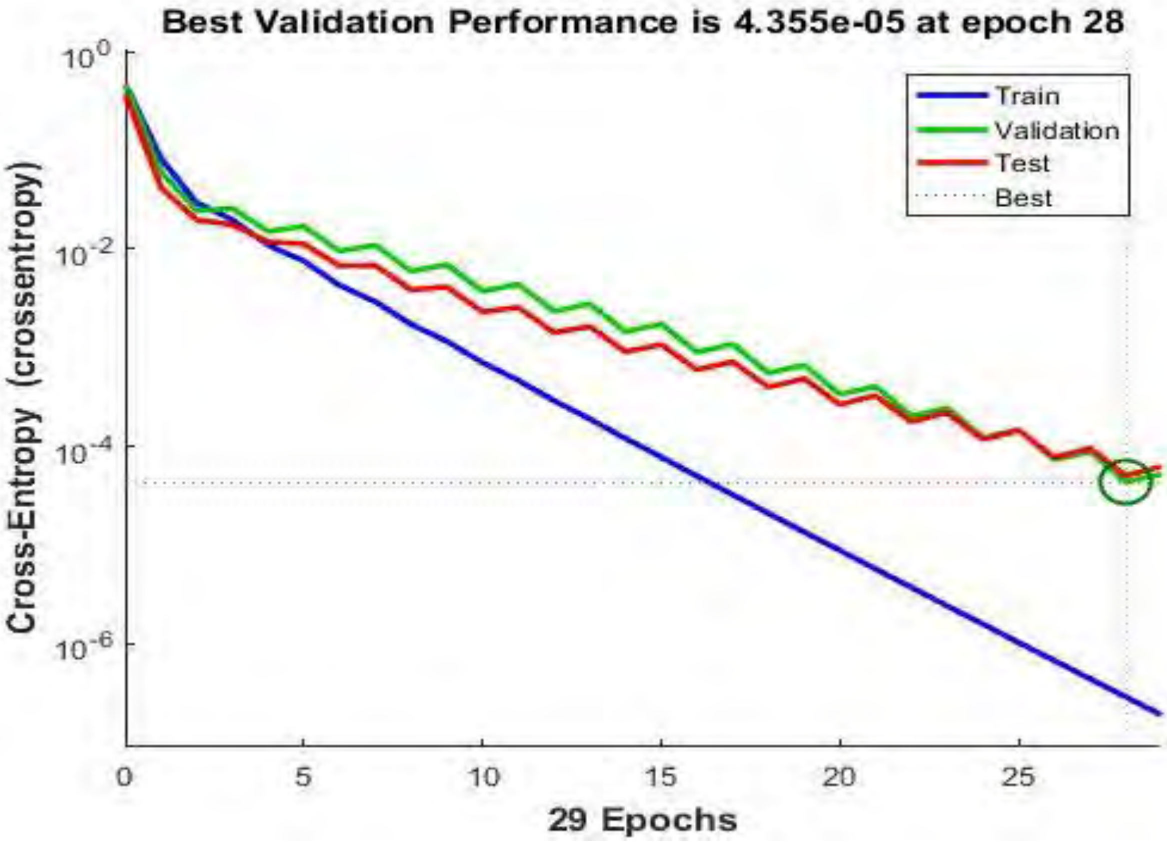
Fig 4.4 Confusion matrix

Confusion matrix gives us an idea of the accuracy with diagonal boxes, green and blue boxes, showing the matches. Now we shall see the other diagram for features 4, 5, 6, 7. Firstly performance graphs:



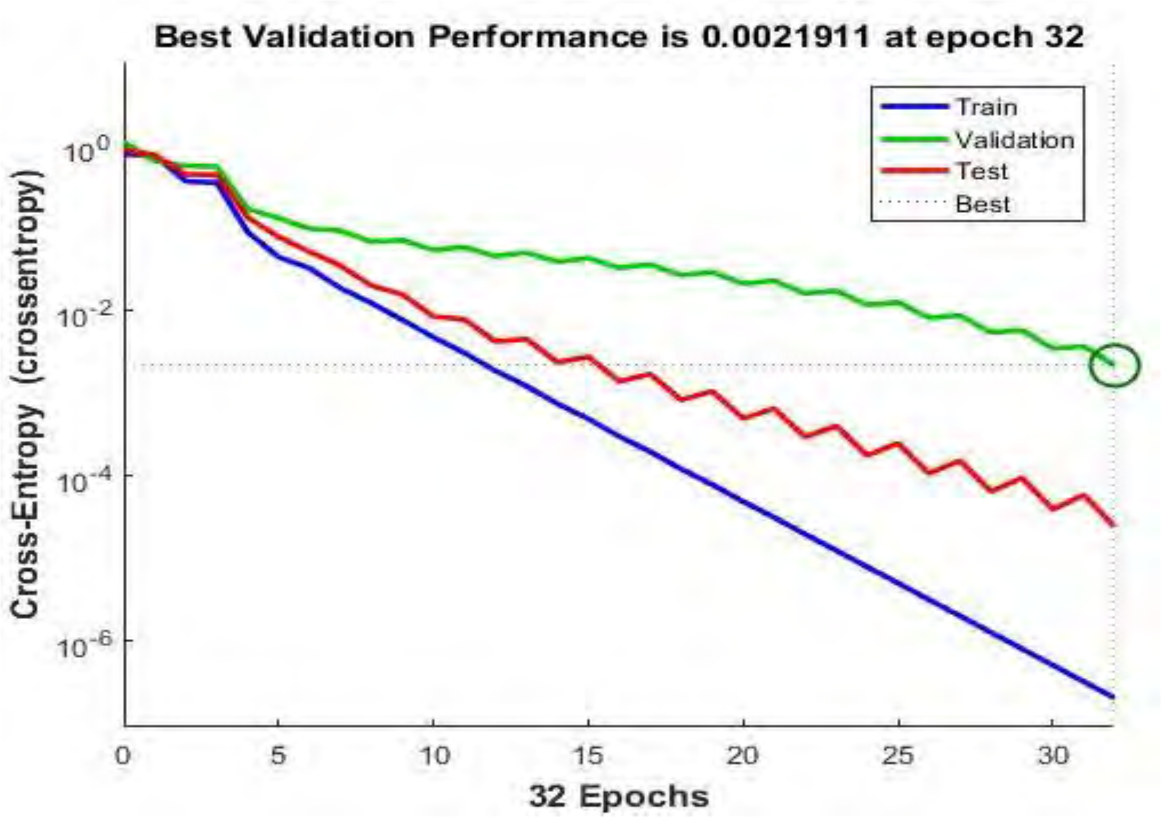
Best Validation Performance is 0.28293 at epoch 28
Cross-Entropy (crossentropy)
10^0
10^{-1}
0
5
10
15
20
25
30
34 Epochs
Train
Validation
Test
Best

(a)

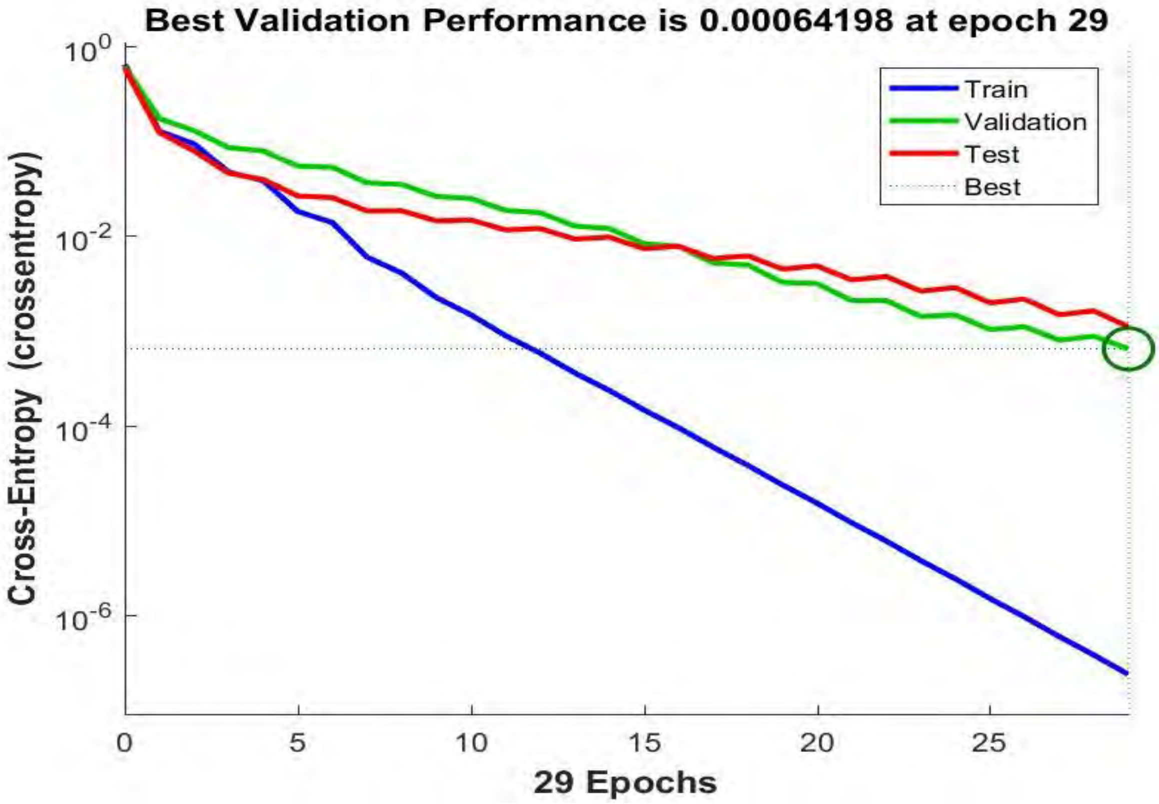


Best Validation Performance is 4.355e-05 at epoch 28
Cross-Entropy (crossentropy)
10^0
10^{-2}
10^{-4}
10^{-6}
0
5
10
15
20
25
29 Epochs
Train
Validation
Test
Best

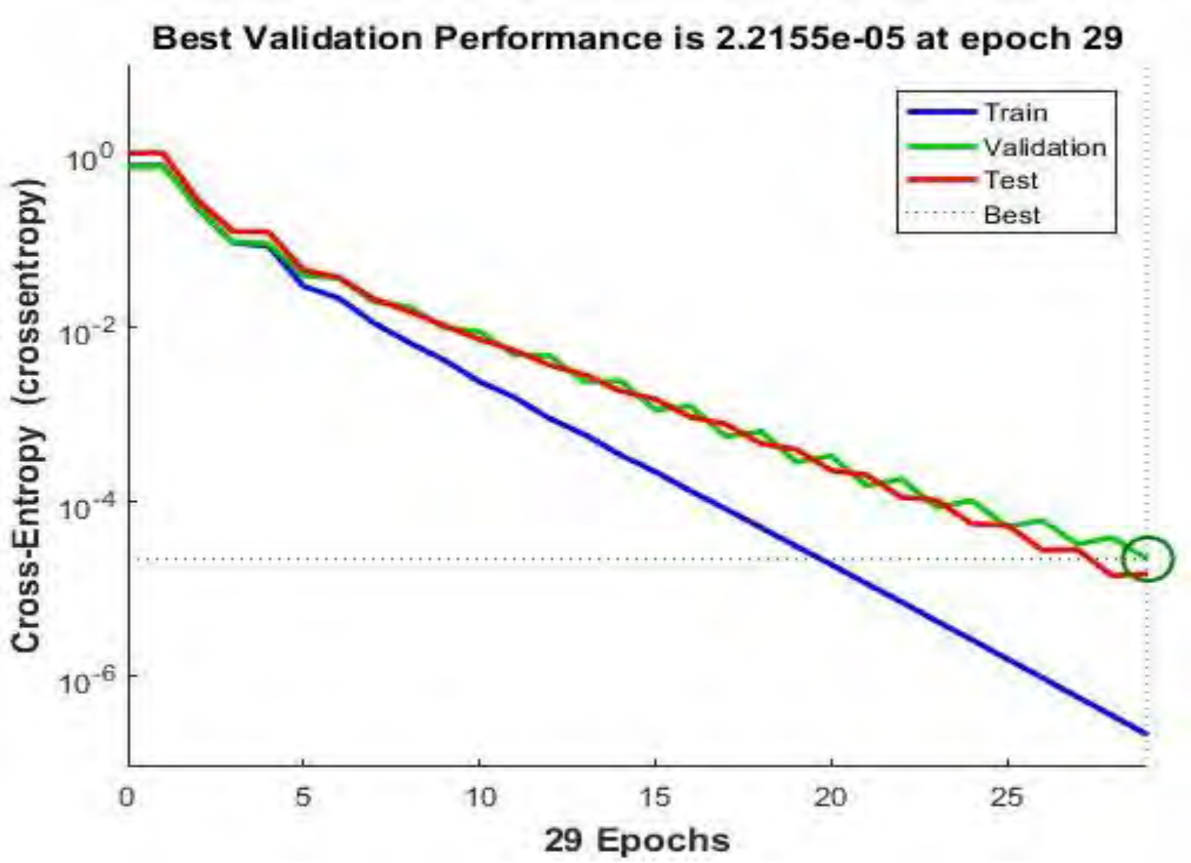
(b)


Best Validation Performance is 0.0021911 at epoch 32
Cross-Entropy (crossentropy)
10^{0}
10^{-2}
10^{-4}
10^{-6}
0
5
10
15
20
25
30
32 Epochs
Train
Validation
Test
Best

(c)


Best Validation Performance is 0.00064198 at epoch 29
Cross-Entropy (crossentropy)
10^{0}
10^{-2}
10^{-4}
10^{-6}
0
5
10
15
20
25
29 Epochs
Train
Validation
Test
Best

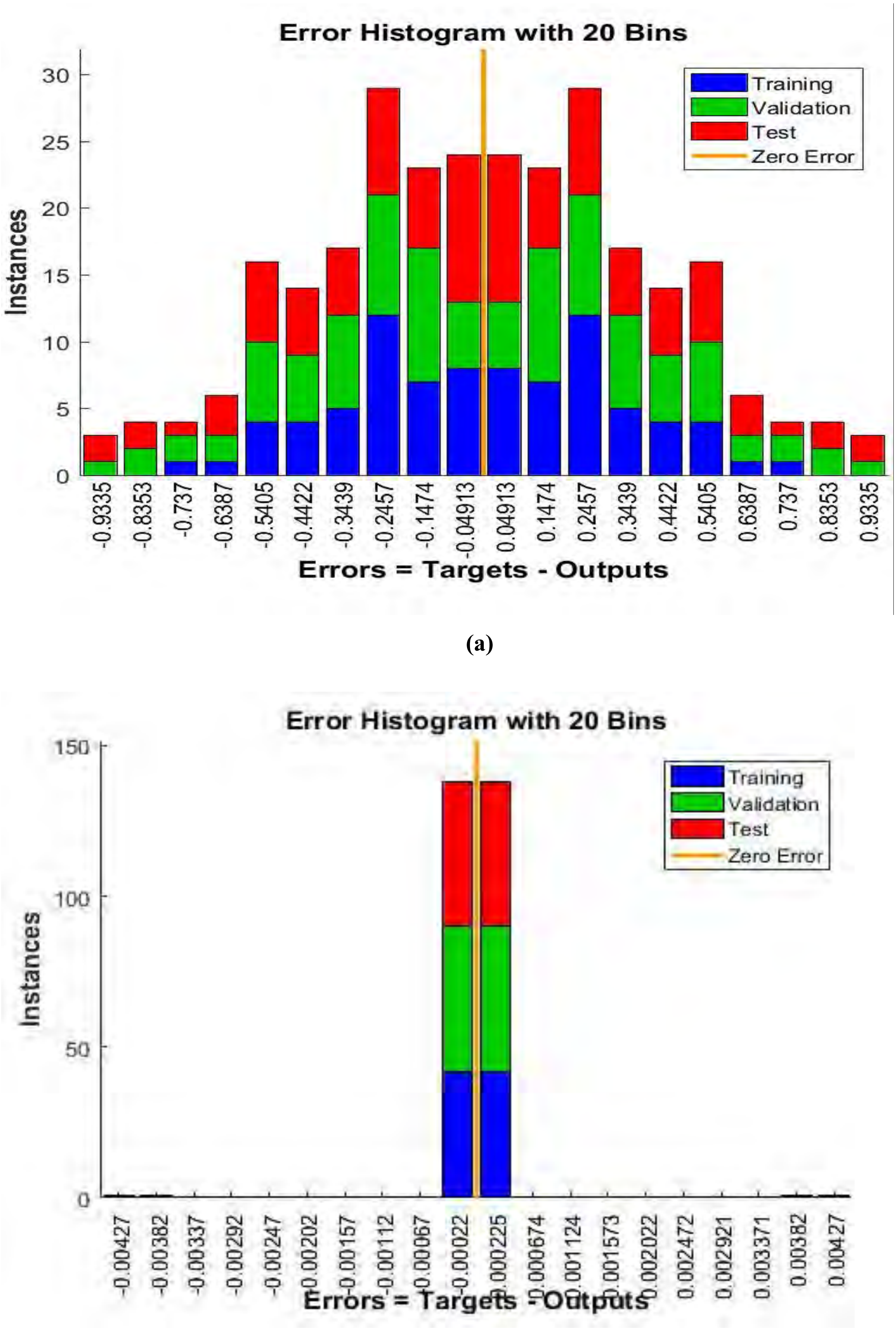
(d)



(e)

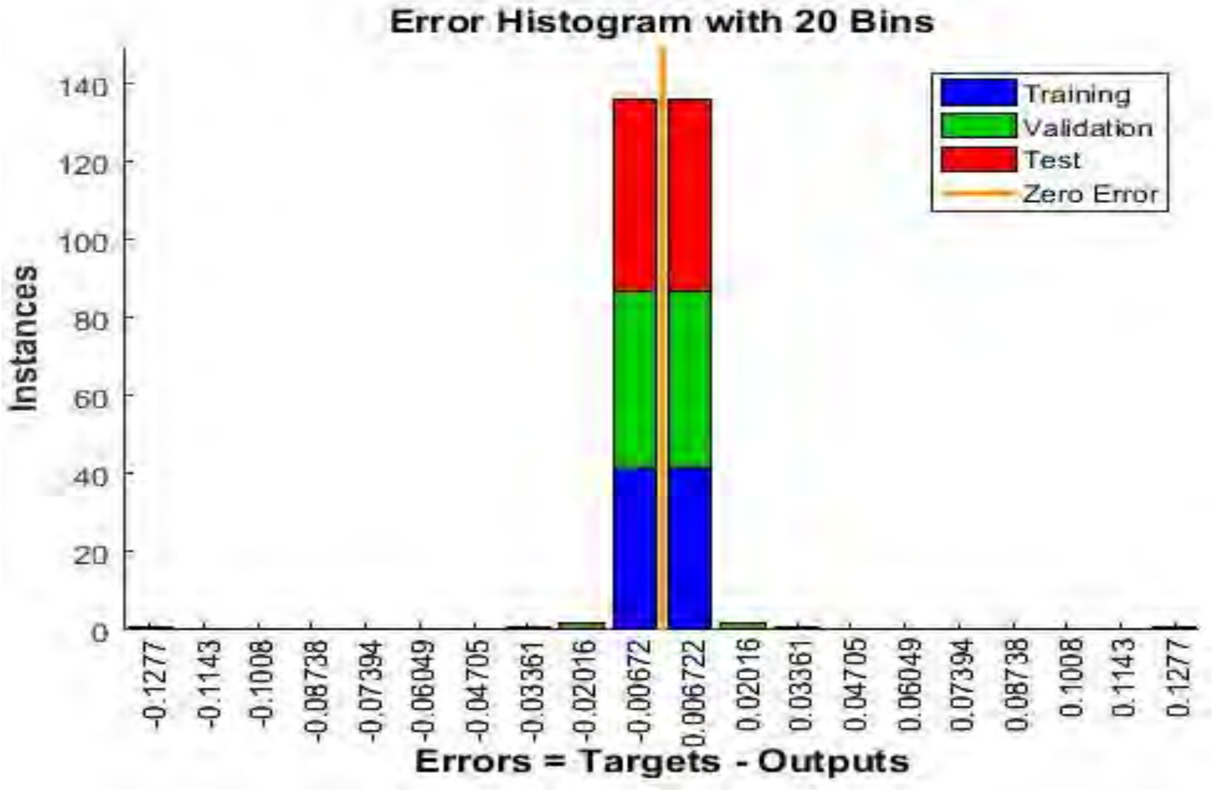
Fig.4.5 (a) Performance Matrix using 4 Features **(b)** Performance Matrix using 5 Features **(c)** Performance Matrix using 6 Features **(d)** Performance Matrix using 7 Features **(e)** Performance Matrix using 8 Features

We can clearly see that as we take more features the overfitting decreases significantly and the cross entropy also decreases down to 2.2155e-05 with 8 features. Now we will discuss error histograms.

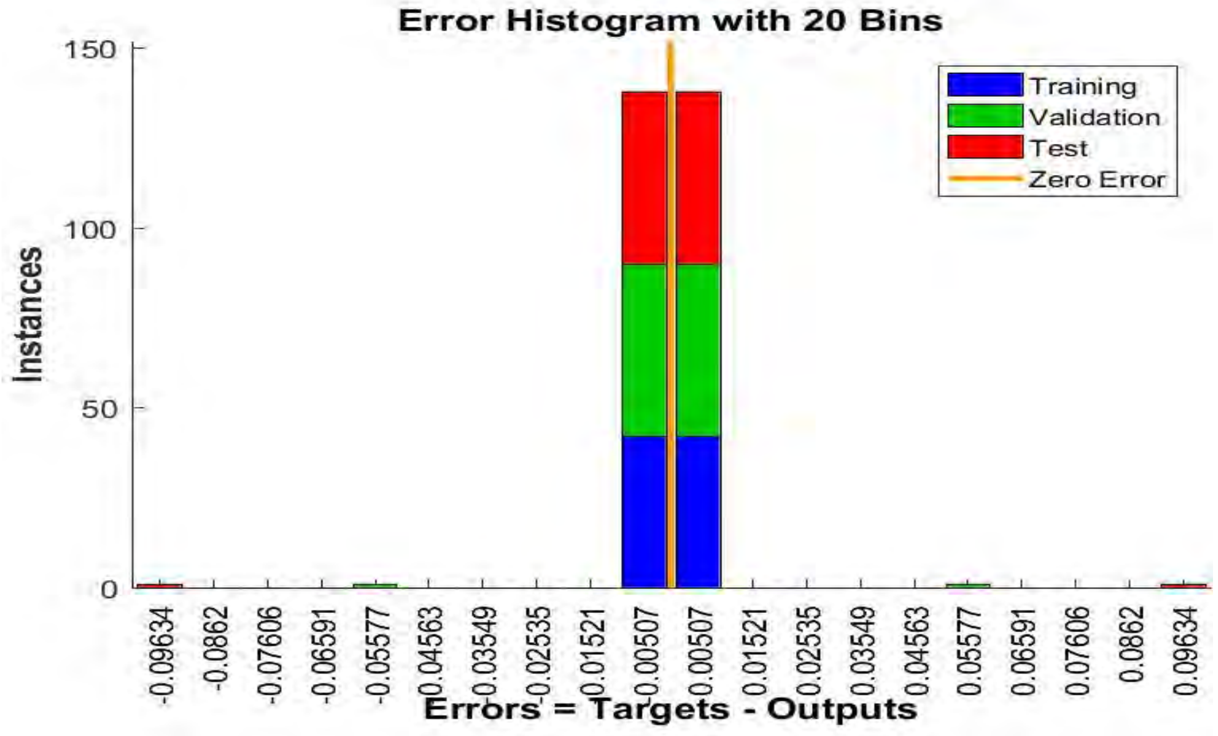

Error Histogram with 20 Bins
Instances
30
25
20
15
10
5
0
Training
Validation
Test
Zero Error
-0.9335
-0.8353
-0.737
-0.6387
-0.5405
-0.4422
-0.3439
-0.2457
-0.1474
-0.04913
0.04913
0.1474
0.2457
0.3439
0.4422
0.5405
0.6387
0.737
0.8353
0.9335
Errors = Targets - Outputs
Error Histogram with 20 Bins
Instances
150
100
50
0
Training
Validation
Test
Zero Error
-0.00427
-0.00382
-0.00337
-0.00292
-0.00247
-0.00202
-0.00157
-0.00112
-0.00067
-0.00022
0.000225
0.000674
0.001124
0.001573
0.002022
0.002472
0.002921
0.003371
0.00382
0.00427
Errors = Targets - Outputs

(a)

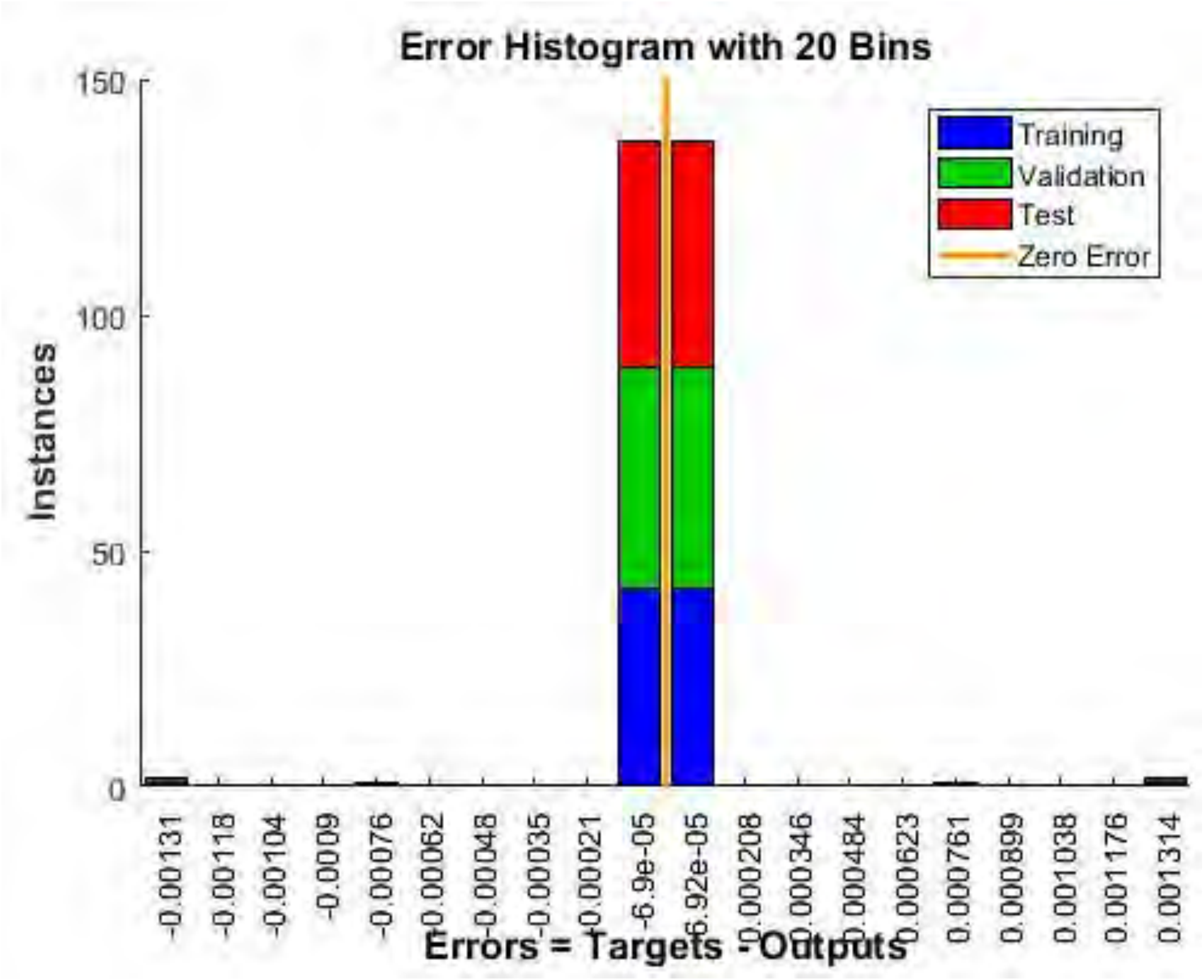
(b)


Error Histogram with 20 Bins
Instances
140
120
100
80
60
40
20
0
Training
Validation
Test
Zero Error
-0.1277
-0.1143
-0.1008
-0.08738
-0.07394
-0.06049
-0.04705
-0.03361
-0.02016
-0.00672
0.006722
0.02016
0.03361
0.04705
0.06049
0.07394
0.08738
0.1008
0.1143
0.1277
Errors = Targets - Outputs

(c)


Error Histogram with 20 Bins
Instances
150
100
50
0
Training
Validation
Test
Zero Error
-0.09634
-0.0862
-0.07606
-0.06591
-0.05577
-0.04563
-0.03549
-0.02535
-0.01521
-0.00507
0.00507
0.01521
0.02535
0.03549
0.04563
0.05577
0.06591
0.07606
0.0862
0.09634
Errors = Targets - Outputs

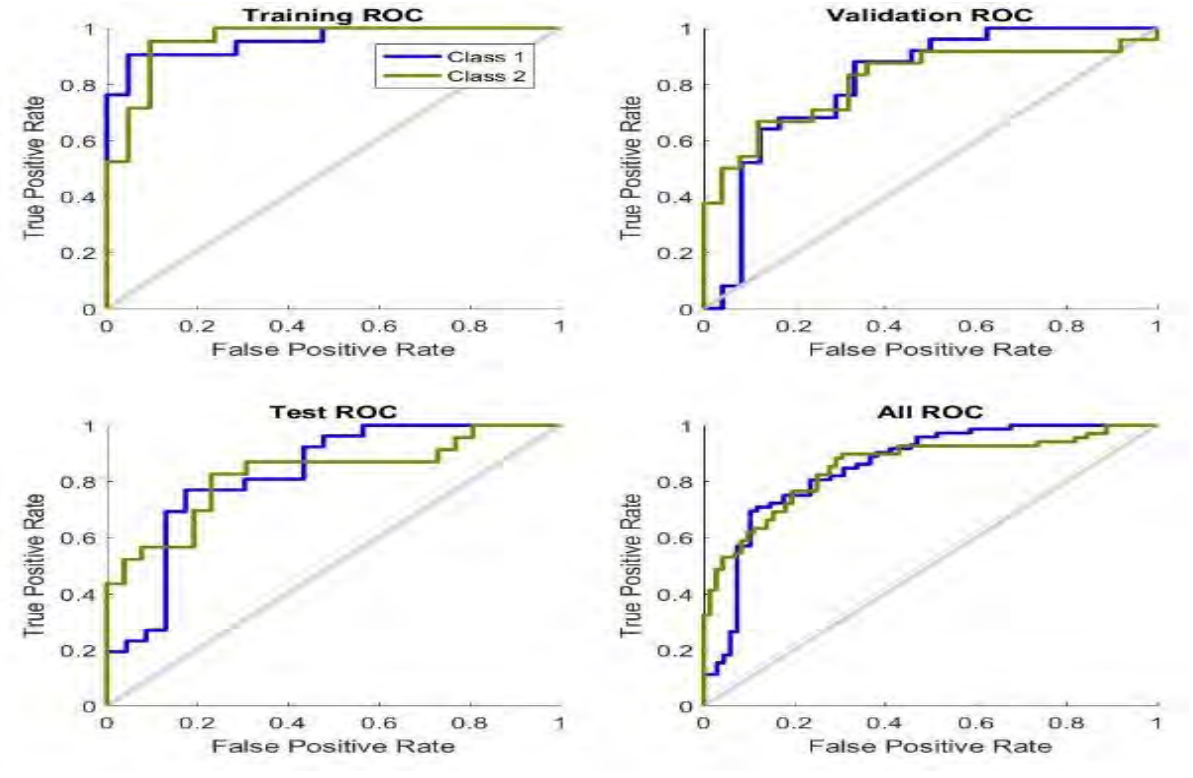
(d)



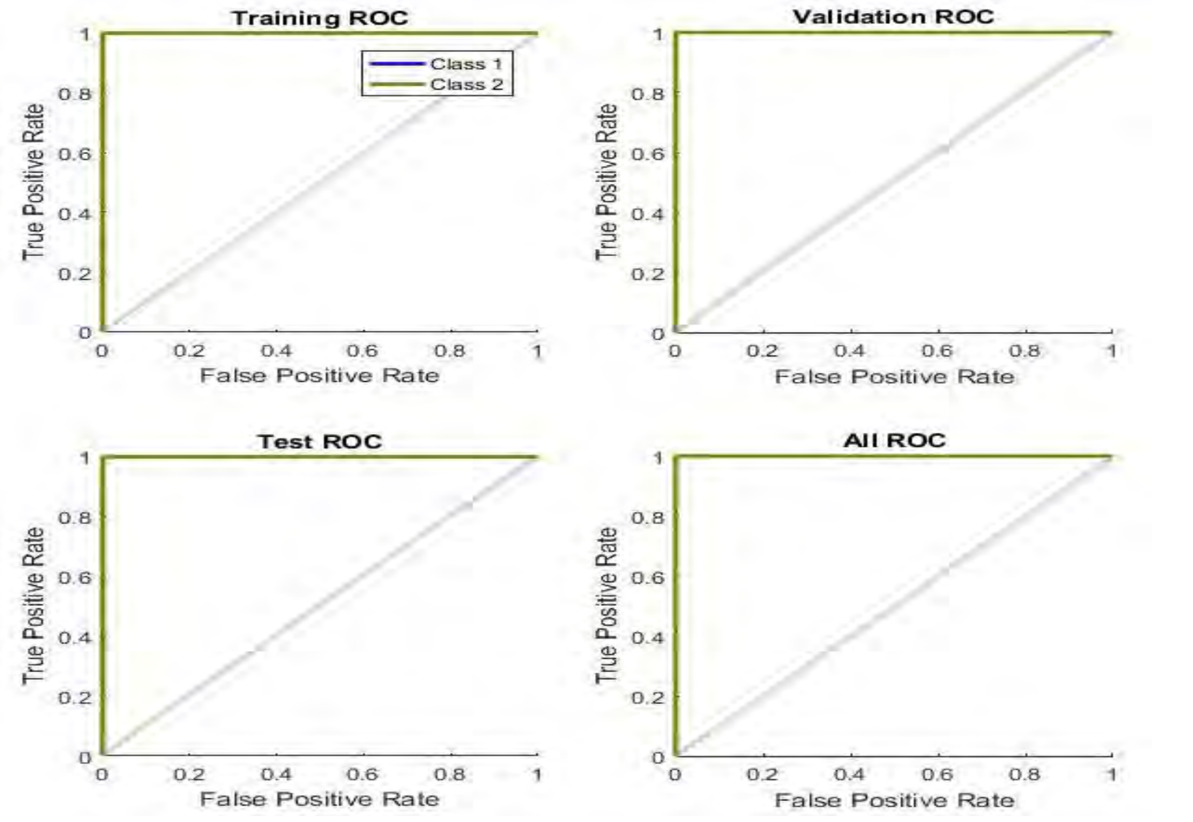
(e)

Fig.4.6 (a) Error Histogram using 4 Features **(b)** Error Histogram using 5 Features **(c)** Error Histogram using 6 Features **(d)** Error Histogram using 7 Features **(e)** Error Histogram using 8 Features

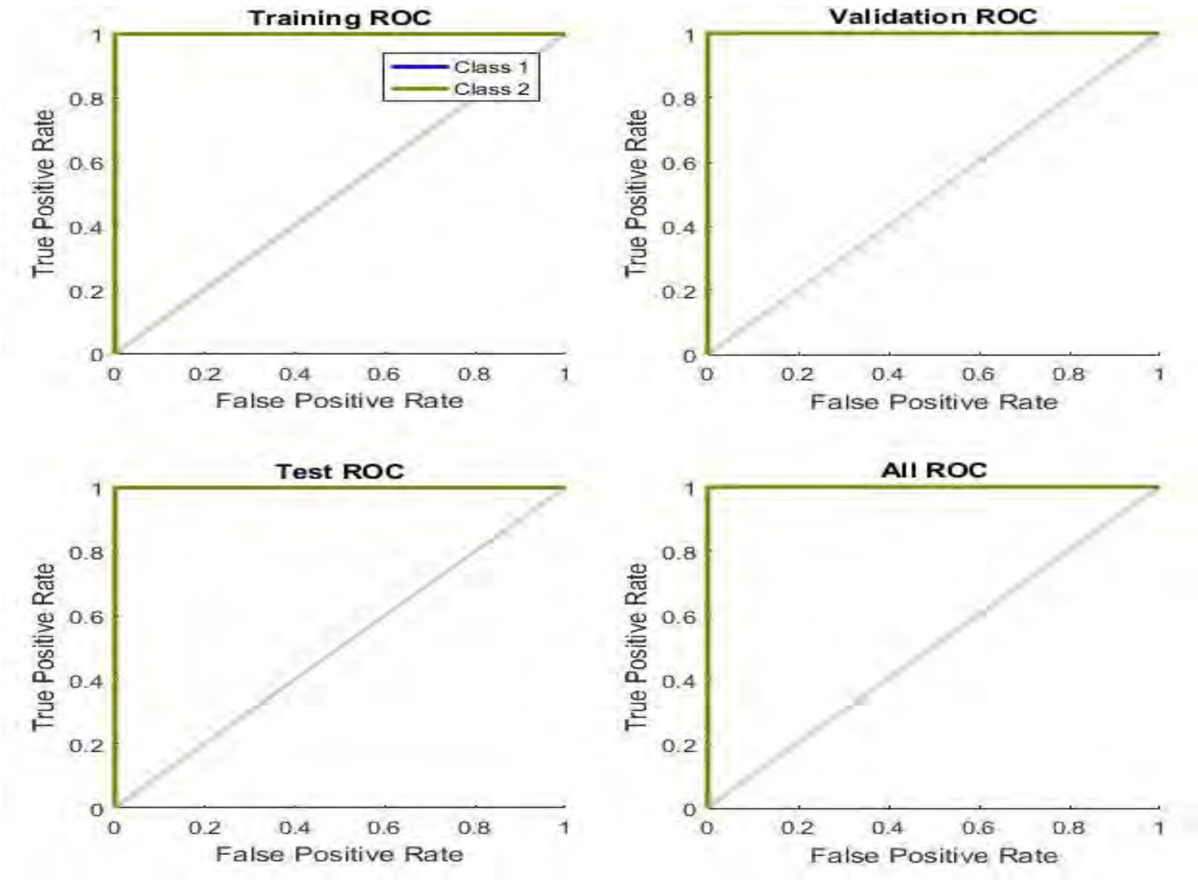
As we can see with increase in usage of features, decreases the error rate and also data which are were overfitted before, are not being overfitted anymore. Now receiver operating characterstic will be dicussed.


Training ROC
Class 1
Class 2
Validation ROC
Test ROC
All ROC
True Positive Rate
False Positive Rate
0
0.2
0.4
0.6
0.8
1

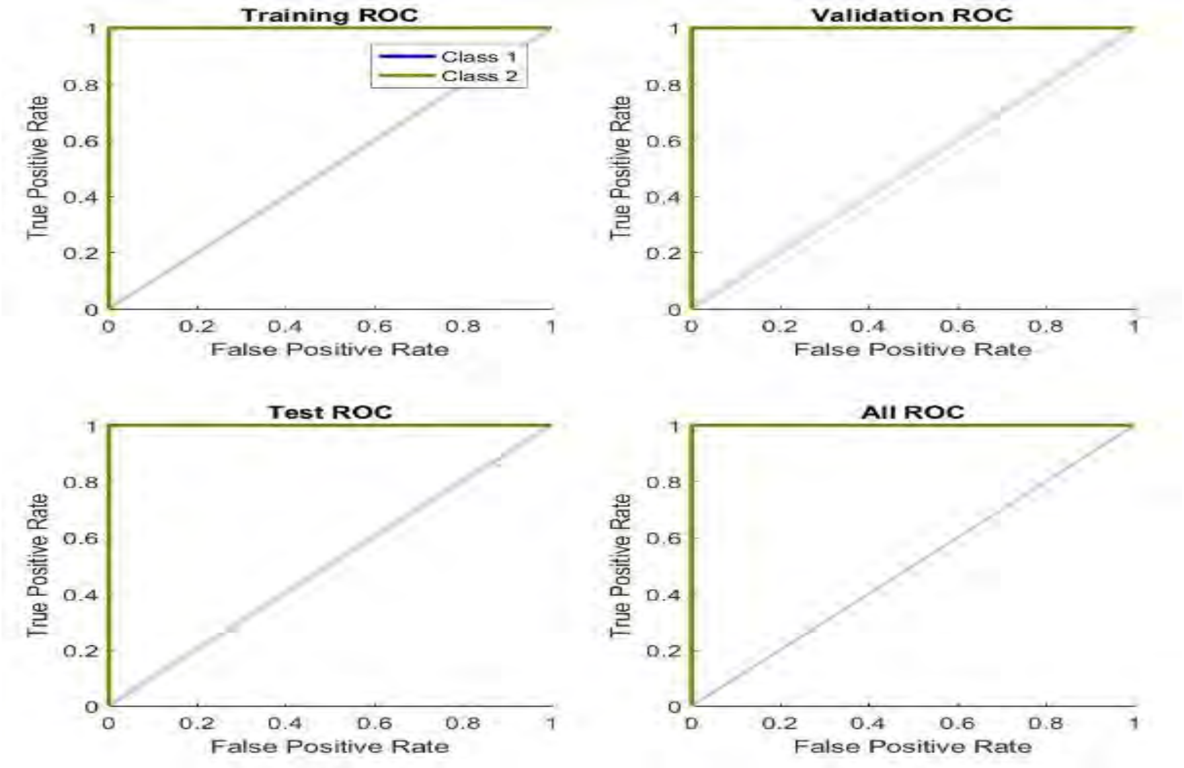
(a)


Training ROC
Class 1
Class 2
Validation ROC
Test ROC
All ROC
True Positive Rate
False Positive Rate
0
0.2
0.4
0.6
0.8
1

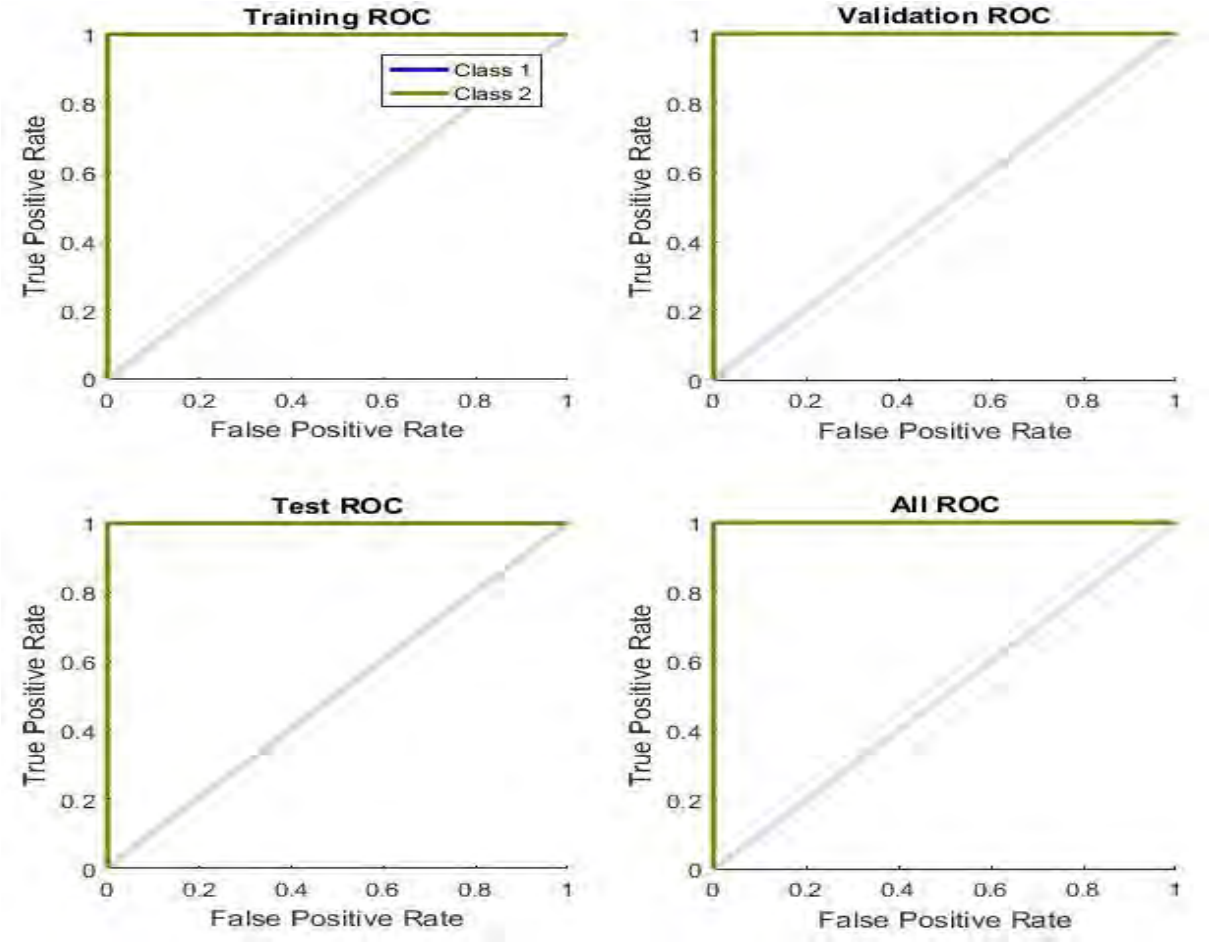
(b)


Training ROC
Validation ROC
Test ROC
All ROC
Class 1
Class 2
True Positive Rate
False Positive Rate
0
0.2
0.4
0.6
0.8
1

(c)


Training ROC
Validation ROC
Test ROC
All ROC
Class 1
Class 2
True Positive Rate
False Positive Rate
0
0.2
0.4
0.6
0.8
1

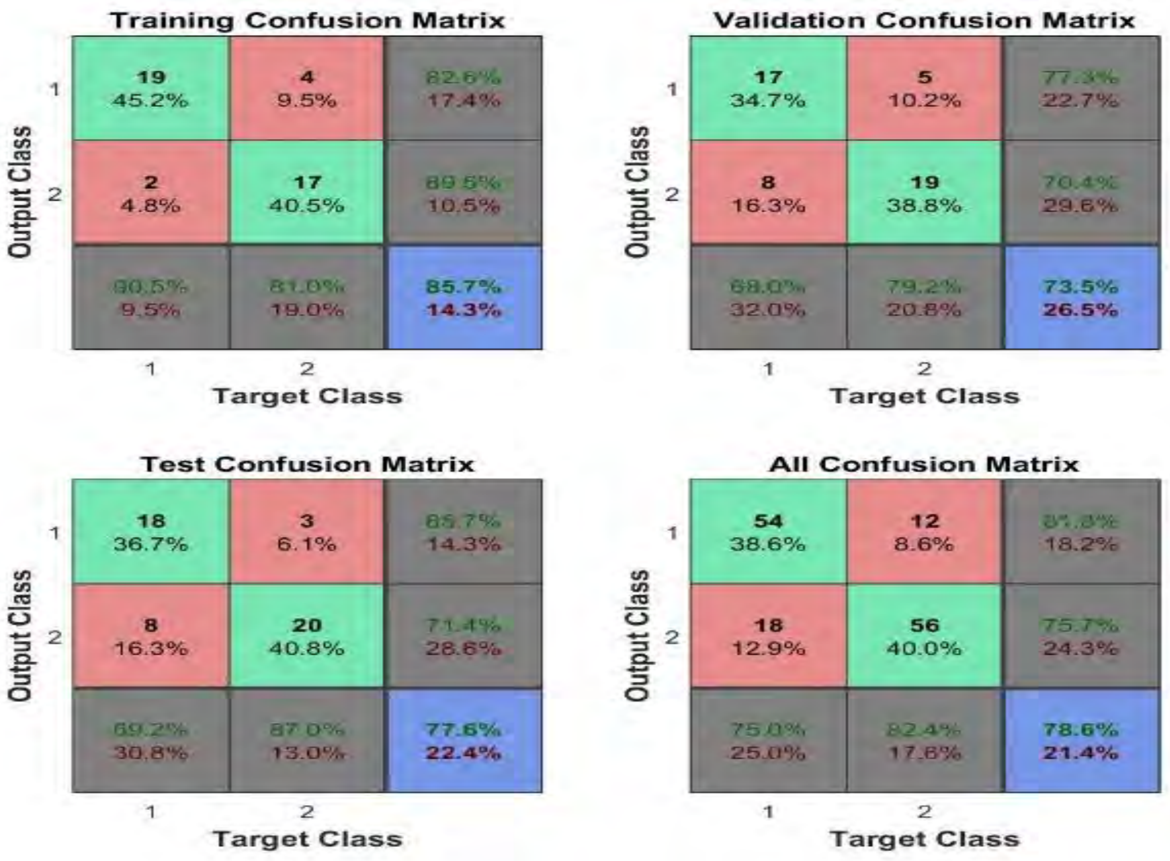
(d)



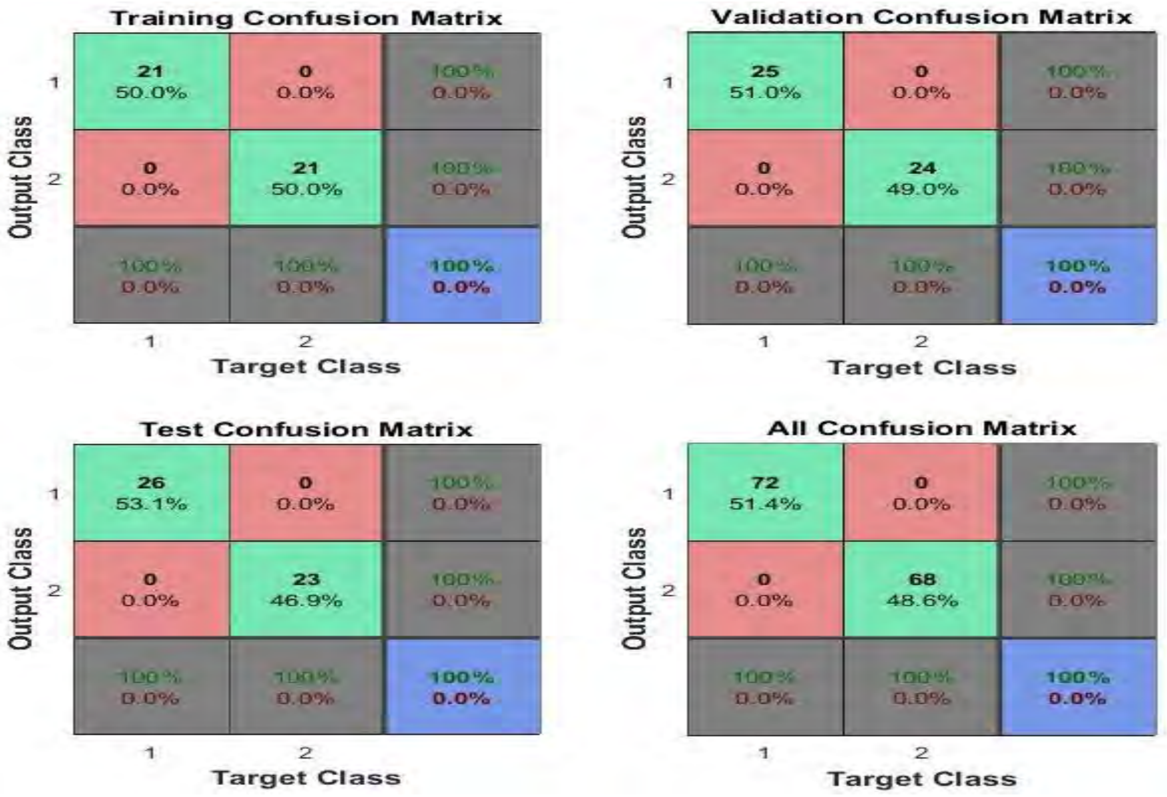
(e)

Fig.4.7 (a) ROC using 4 Features **(b)** ROC using 5 Features **(c)** ROC using 6 Features **(d)** ROC using 7 Features **(e)** ROC using 8 Features

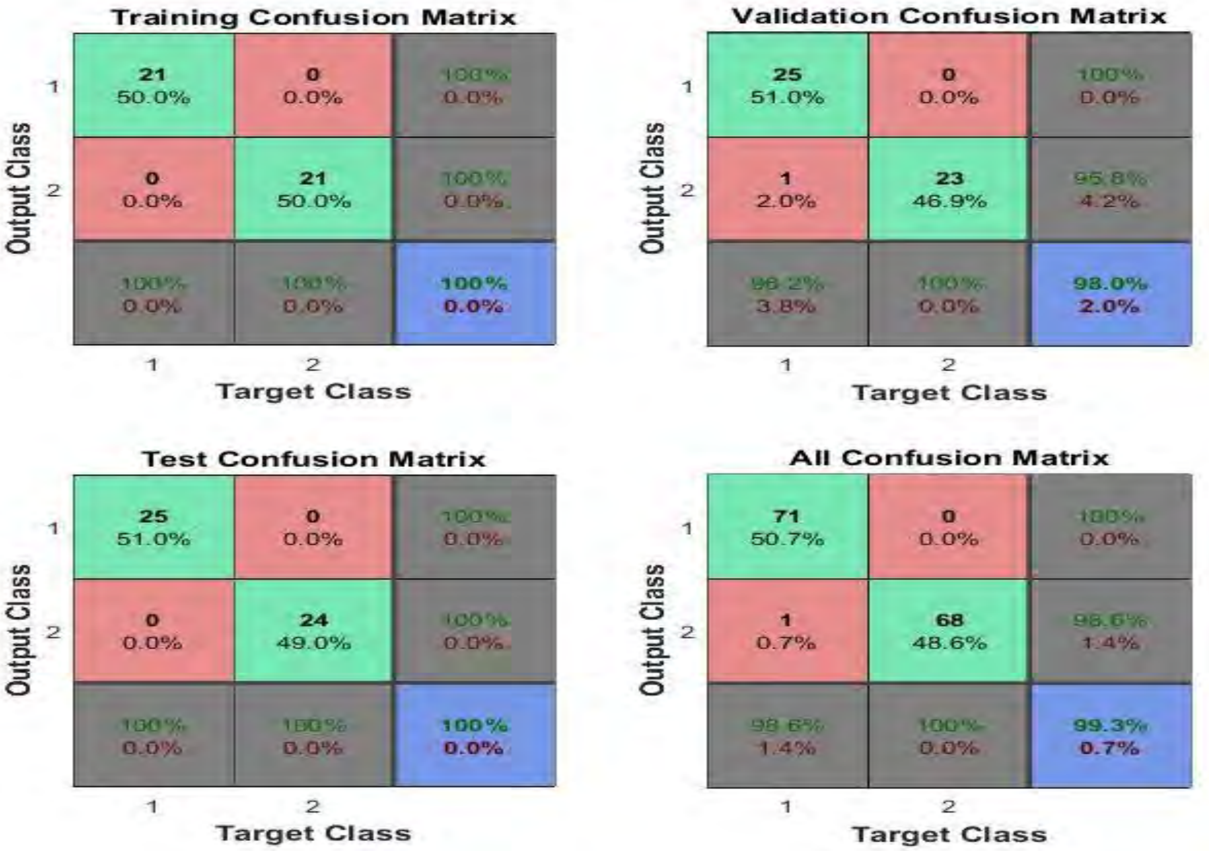
Again as we use more features, the graph gets pushed towards the y-axis, giving us a better result.


Training Confusion Matrix
19
45.2%
4
9.5%
82.6%
17.4%
2
4.8%
17
40.5%
89.5%
10.5%
90.5%
9.5%
81.0%
19.0%
85.7%
14.3%
Validation Confusion Matrix
17
34.7%
5
10.2%
77.3%
22.7%
8
16.3%
19
38.8%
70.4%
29.6%
68.0%
32.0%
79.2%
20.8%
73.5%
26.5%
Test Confusion Matrix
18
36.7%
3
6.1%
85.7%
14.3%
8
16.3%
20
40.8%
71.4%
28.6%
69.2%
30.8%
87.0%
13.0%
77.6%
22.4%
All Confusion Matrix
54
38.6%
12
8.6%
81.8%
18.2%
18
12.9%
56
40.0%
75.7%
24.3%
75.0%
25.0%
82.4%
17.6%
78.6%
21.4%
Output Class
1
2
Target Class
1
2

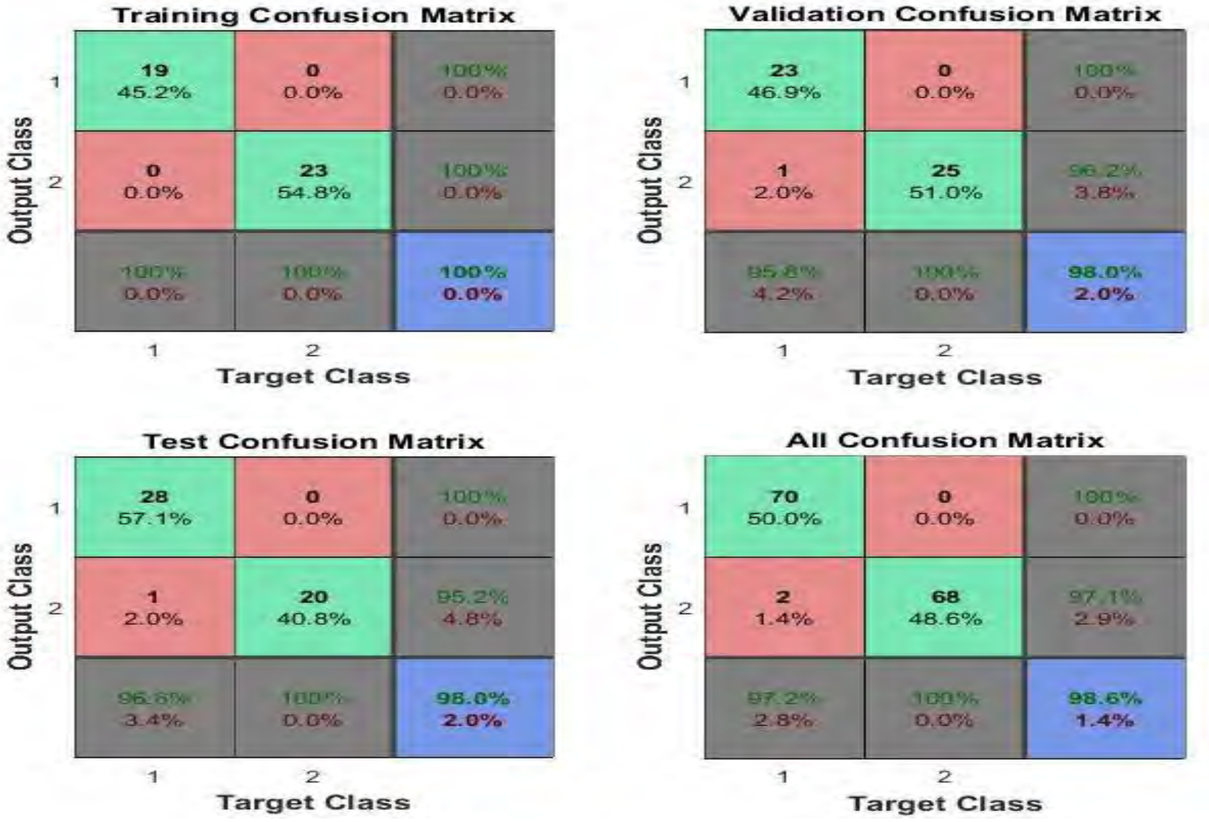
(a)


Training Confusion Matrix
21
50.0%
0
0.0%
100%
0.0%
0
0.0%
21
50.0%
100%
0.0%
100%
0.0%
100%
0.0%
100%
0.0%
Validation Confusion Matrix
25
51.0%
0
0.0%
100%
0.0%
0
0.0%
24
49.0%
100%
0.0%
100%
0.0%
100%
0.0%
100%
0.0%
Test Confusion Matrix
26
53.1%
0
0.0%
100%
0.0%
0
0.0%
23
46.9%
100%
0.0%
100%
0.0%
100%
0.0%
100%
0.0%
All Confusion Matrix
72
51.4%
0
0.0%
100%
0.0%
0
0.0%
68
48.6%
100%
0.0%
100%
0.0%
100%
0.0%
100%
0.0%
Output Class
1
2
Target Class
1
2

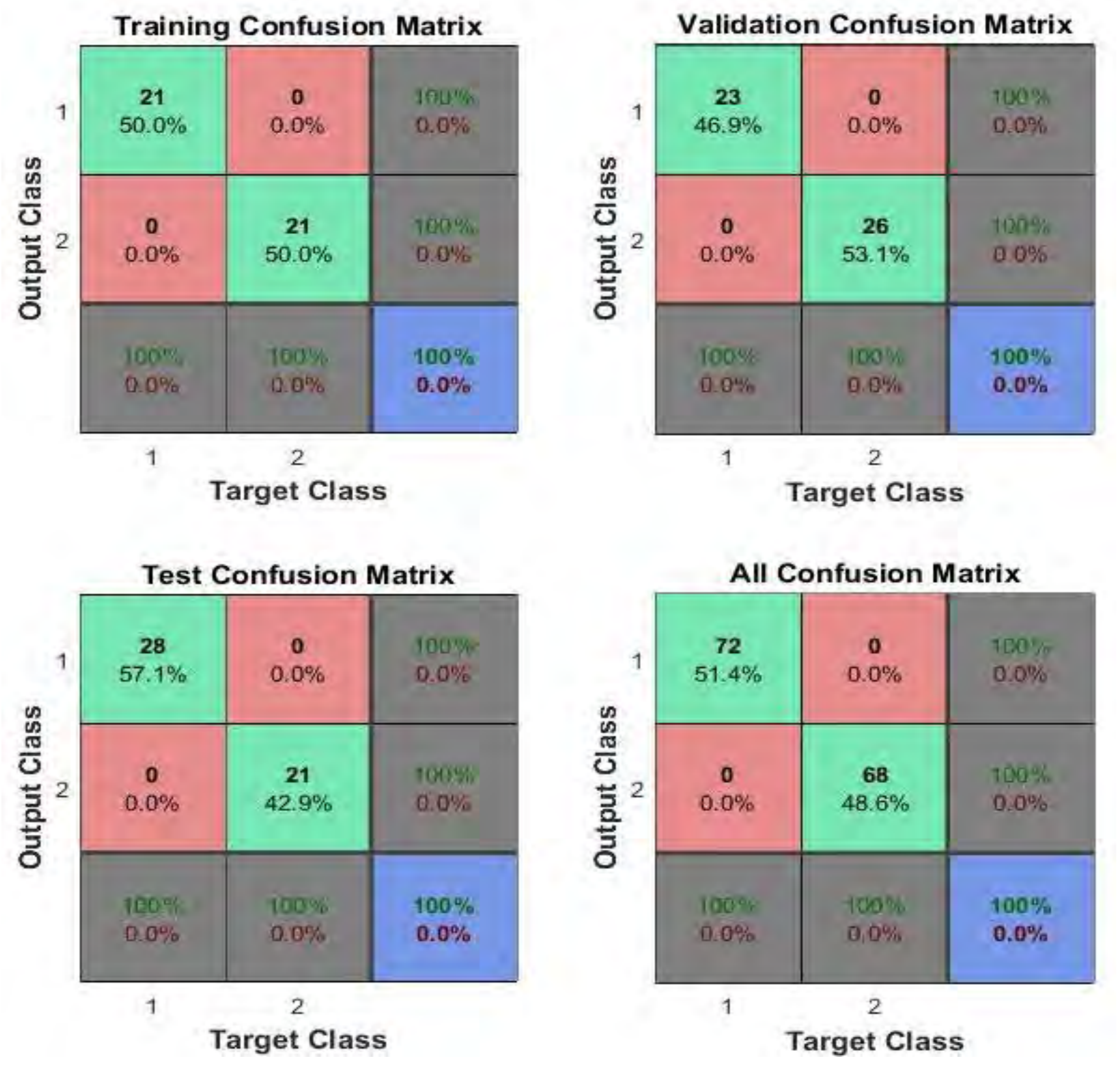
(b)


Training Confusion Matrix
21 50.0%
0 0.0%
100% 0.0%
0 0.0%
21 50.0%
100% 0.0%
100% 0.0%
100% 0.0%
100% 0.0%
Validation Confusion Matrix
25 51.0%
0 0.0%
100% 0.0%
1 2.0%
23 46.9%
95.8% 4.2%
96.2% 3.8%
100% 0.0%
98.0% 2.0%
Test Confusion Matrix
25 51.0%
0 0.0%
100% 0.0%
0 0.0%
24 49.0%
100% 0.0%
100% 0.0%
100% 0.0%
100% 0.0%
All Confusion Matrix
71 50.7%
0 0.0%
100% 0.0%
1 0.7%
68 48.6%
98.6% 1.4%
98.6% 1.4%
100% 0.0%
99.3% 0.7%
Output Class
Target Class
1
2

(c)


Training Confusion Matrix
19 45.2%
0 0.0%
100% 0.0%
0 0.0%
23 54.8%
100% 0.0%
100% 0.0%
100% 0.0%
100% 0.0%
Validation Confusion Matrix
23 46.9%
0 0.0%
100% 0.0%
1 2.0%
25 51.0%
96.2% 3.8%
95.8% 4.2%
100% 0.0%
98.0% 2.0%
Test Confusion Matrix
28 57.1%
0 0.0%
100% 0.0%
1 2.0%
20 40.8%
95.2% 4.8%
96.6% 3.4%
100% 0.0%
98.0% 2.0%
All Confusion Matrix
70 50.0%
0 0.0%
100% 0.0%
2 1.4%
68 48.6%
97.1% 2.9%
97.2% 2.8%
100% 0.0%
98.6% 1.4%
Output Class
Target Class
1
2

(d)



Training Confusion Matrix

Output Class \ Target Class	1	2	
1	21 50.0%	0 0.0%	100% 0.0%
2	0 0.0%	21 50.0%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%

Validation Confusion Matrix

Output Class \ Target Class	1	2	
1	23 46.9%	0 0.0%	100% 0.0%
2	0 0.0%	26 53.1%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%

Test Confusion Matrix

Output Class \ Target Class	1	2	
1	28 57.1%	0 0.0%	100% 0.0%
2	0 0.0%	21 42.9%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%

All Confusion Matrix

Output Class \ Target Class	1	2	
1	72 51.4%	0 0.0%	100% 0.0%
2	0 0.0%	68 48.6%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%

(e)

Fig.4.8 (a) Confusion Matrix using 4 Features **(b)** Confusion Matrix using 5 Features **(c)** Confusion Matrix using 6 Features **(d)** Confusion Matrix using 7 Features **(e)** Confusion Matrix using 8 Features

The same trend can be seen in the confusion matrix. The accuracy increases drastically. Comparing our results with other models such as Particle Swarm Optimization with Naive Bayes Classifier (Model 1), Bat Algorithm with Naive Bayes Classifier (Model 2). Table 1 shows the comparison.

Algorithms	Error Rate
Model 1	0.8889
Model 2	0.8889
Our Model	2.2155e-5

Table 4.1 Model Comparison

CHAPTER 05

CONCLUSIONS AND FUTURE WORKS

5.1 Concluding

In the proposed model a high percent accuracy is achieved. Among all the 20 features PCA has been applied. A matrix or feature vector was returned which contained arrangement of features according to their significance. Then first 8 optimal fearutes were chosen without any outliers. This model has been compared with various other models too. Thus it can be said that this model have a great efficiency with higher accuracy.

REFERENCES

[1] Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. Neural Networks, 2(5), 359-366. doi:10.1016/0893-6080(89)90020-8.

[2] Murali, M. (2015). Principal Component Analysis based Feature Vector Extraction. Indian Journal of Science and Technology.

[3] K. Funahashi, “On the approximate realization of continuous mappingby neural networks,” Neural Networks, vol. 2, pp. 183–192, 1989.

[4] G. Cybenko, “Continuous valued neural networks with two hidden layers are sufficient,” Math. Contr., Signals, Syst., vol. 2, pp. 303–314, 1989.

[5] K. Hornik, M. Stinchcombe, and H. White, “Multilayer feedforward networks are universal approximators,” Neural Networks, vol. 2, pp.359–366, 1989..

[6] J. de Villiers and E. Barnard, “Backpropagation neural nets with one and two hidden layers,” IEEE Trans. Neural Networks, vol. 4, pp. 136–141, Jan. 1992.

[7] Meruelo, A. C., Simpson, D. M., Veres, S. M., & Newland, P. L. (2016). Improved system identification using artificial neural networks and analysis of individual differences in responses of an identified neuron. Neural Networks, 75, 56-65. doi:10.1016/j.neunet.2015.12.002.

[8] Xing, L., & Pham, D. T. (1995). Neural networks for identification, prediction, andcontrol. Springer-Verlag New York, Inc

[9] 18. Sakar, B. E., Isenkul, M., Sakar, C. O., Sertbas, A., Gurgen, F., Delil, S., . . . Kursun, O. (2013). Collection and Analysis of a Parkinson Speech Dataset With Multiple Types of Sound Recordings. IEEE Journal of Biomedical and Health Informatics, 17(4), 828-834. doi:10.1109/jbhi.2013.2245674.

[10] Fong, S., Yang, X., & Deb, S. (2013). Swarm Search for Feature Selection in Classification. 2013 IEEE 16th International Conference on Computational Science and Engineering. doi:10.1109/cse.2013.135.