**DATA MINING – ASSIGNEMENT 3 REPORT**

**CSE 572**

**SPRING 2018**

**GROUP - 12**

**Submitted To: Dr. Ayan Banerjee**

**Arizona State University**


Submitted by:

Aritra Mitra - 1211172453 – amitra8@asu.edu
Arun Karthick Manickam Alagar Muthumanickam - 1213135077 - amanick4@asu.edu
Vinay Bhargav Arni Ragunathan - 1212712330 - varnirag@asu.edu
Giriraj NoLastName - 1213350721 - ggirira2@asu.edu
Deepthi Parvathy Puducode Radhakrishnan - 1213171633 - dpuducod@asu.edu

# Contents

# 1. Introduction:

As part of the Data Mining course requirement project, we are trying to recognize various American Sign language (ASL) gestures by reading the gestures through various sensors and recognizing them by using Data Mining algorithms. We are trying to extract features which are dominant to a action. The extracted features are used to classify through various classification techniques like SVM, Decision trees and Neural Networks. The extracted feature data is divided into train and test. Based on the model results we calculate precision, recall, F1 score and Accuracy.

# 2. Phase 1:

**Data Collection:**

In the Data Collection phase, we worked on collecting the raw data by performing ASL gestures using wrist bands. In this phase, one member from each team wore the wrist bands and performed the given 20 ASL gestures. Each gesture was performed 20 times and the wrist band readings were read. The 20 gestures which we performed were – About, And, Can, Cop, Deaf, Decide, Father, Find, Go out, Hearing, Here, Hospital, If, Cat, Cost, Day, Gold, Good Night, Hurt and Large. The actions were done for about 3s and were recorded at 15Hz frequency.

The raw data of the wrist bands had data of various sensors such as – Accelerometer – X, Y, Z; EMG pod (0-7); Gyroscope - X, Y, Z; Orientation roll, Pitch, Yaw and Kinect data for both the hands.

# 3. Phase 2

In the Phase 2 of the project, we worked on reading the data and segregate it for different gestures. The next part of this phase is to extract interesting features of each gesture. We are studying the gesture and intuitively trying to find out its features, then using MATLAB we are trying to extract these features using different feature extraction methods. Post this we are using Principle Component Analysis to perform feature reduction. This gives us the top latent features for an action and the eigen vector analysis shows which features have high variance and define a gesture uniquely. In this phase we are concentrating only on 10 ASL gestures – About, And, Can, Cop, Deaf, Decide, Father, Find, Go out, Hearing.

# 4. Phase 3

In the Phase 3 of the project, we are working with the feature extracted data in phase 2 to get the user specific predictions. We are building ten different machines for 10 different gestures. Machines are built to classify one gesture from all others for a specific user. We are using classification techniques in MATLAB such as Decision trees, SVM and Neural Networks. We are calculating metrics such as Accuracy, precision, Recall and F1 score to predict how good the classification technique is for the given data.  In this phase we are concentrating only on 10 ASL gestures – About, And, Can, Cop, Deaf, Decide, Father, Find, Go out, Hearing.

## *4.1 Task 1: Data Preparation*

In this task we are reading the raw data from phase 1 and separating the reading for the different users. The data contains the actions done by 37 groups. We are collating all the actions for a particular user in a single file. At this point of time we are considering 10 users. The reason for using 10 users is, collating the data for all the users is very time consuming as each group has done each gesture 20 times. Secondly, the data for all users is too huge and further processing will also be time consuming as we are running the codes in our Personal computers. Finally, as each team has done each action 20 times with 5 persons we are getting the data for each gesture for 20*5 = 100 times, this gives a good person to person variation and will suffice in extracting important features for the action.

Steps done in MATLAB for task 1 (*Task1.m*):

a) Reading the data for gestures: About, And, Can, Cop, Deaf, Decide, Father, Find, Go out, Hearing for folders DM12, DM02, DM03, DM04, DM05 individually using *xlsread().*
b) From this we are extracting the numeric data and the text data. The numeric data contains all the sensor values and the text data contain the sensor names.
c) We are then transposing the numeric data because we need the sensor data row wise. We are taking only 50 readings for each sensor to maintain the consistency. If the readings are less than 50 we do zero padding.
d) The row header contains the action name. The Row header for About looks like 'About'.
e) We are appending the data for 10 actions for 20 times for one user.
f) For a given Action we add a label column at the end for supervised learning. The Label of the current action is set as 1 and Label of all the other actions is set as 0.
g) This total data is stored in a csv file named as the user name 'DM02.csv' using *xlswrite*() and then stored in a folder named 'Output2'.

**Some sample output data:**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| About | -0.96729 | -0.96729 | -0.96729 | -0.96729 | -0.96729 | -0.96729 | -0.96729.. |
| About | 0.211914 | 0.211914 | 0.211914 | 0.211914 | 0.211914 | 0.211914 | 0.211914.. |
| About | -0.15283 | -0.15283 | -0.15283 | -0.15283 | -0.15283 | -0.15283 | -0.15283.. |
| About | -0.97803 | -0.97803 | -0.97803 | -0.97803 | -0.97803 | -0.97803 | -0.97803.. |
| About | -0.25684 | -0.25684 | -0.25684 | -0.25684 | -0.25684 | -0.25684 | -0.25684.. |
| About | -0.08008 | -0.08008 | -0.08008 | -0.08008 | -0.08008 | -0.08008 | -0.08008.. |
| About | -5 | -5 | -5 | -5 | -5 | -5 | -5.. |
| About | -3 | -3 | -3 | -3 | -3 | -3 | -3.. |

…
…
.
…
…
.

## *4.2 Task 2: Decision Tree*

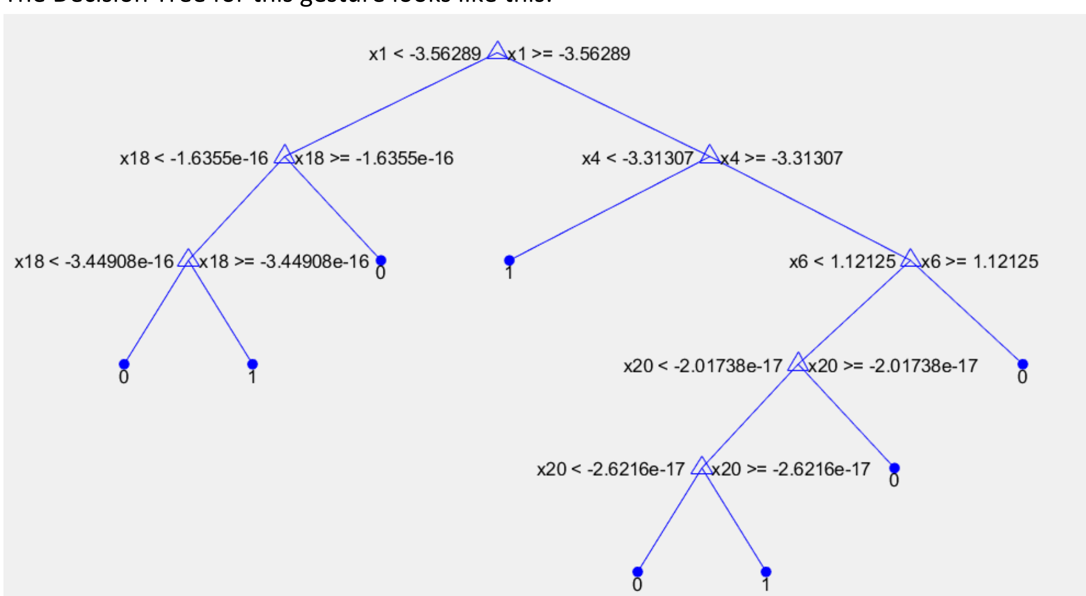
In Task 2 we are building a Machine for classifying the given 10 gestures using Decision Trees. In this phase we are using the Feature set which we obtained in Assignment two. We extract the eigen vectors from the PCA output for the input data and multiply this with the input matrix to get the new feature set. We are using this feature set of all actions performed by one user as our input for Task 2 classification.

Decision Trees: It's a tree-like graph, where the condition at node decides the path to be taken for classification. It's a supervised classification algorithm, which uses different purity metrics like Gini impurity, Information gain to form the conditions for the nodes. As the layers of the tree increases the classification error of training data tends to decrease, but we don't want a deep tree because it increases the chance of overfitting. The classification at each node is based on any one feature which best classifies the data at that point.

### 4.2.1 Decision Tree for User 1

**ABOUT**

a) For Sign 'About' for user 1 (DM02) we obtained the CSV file from Task 1 and divided the data into Test and Training. We have 20 latent features for About.
b) Training set contains 60% of the data with label 1 and 60% of data with label 0. Test set contains the remaining.
c) The Decision tree is created using 'fitctree' functionality of matlab using the Training Data set.
d) The Decision Tree for this gesture looks like this:




e) The accuracy metrics are calculated by getting the predictions of the test dataset by the decision tree.
f) The metrics we are calculating:
Accuracy % : Total correct predictions/Total no of Inputs * 100
True Positives = Total no of correct predictions for label 1
True Negatives = Total no of Correct predictions for label 0
False Positives = Total no of predicted 1 which are incorrect

False Negative = Total no of Predicted 0 which are incorrect

Precision = True Positive / (True Positive + False Positive)
Recall = True Positive / (True Positive + False Negative)
F1 Score = 2* ((precision*recall)/ (precision + recall))

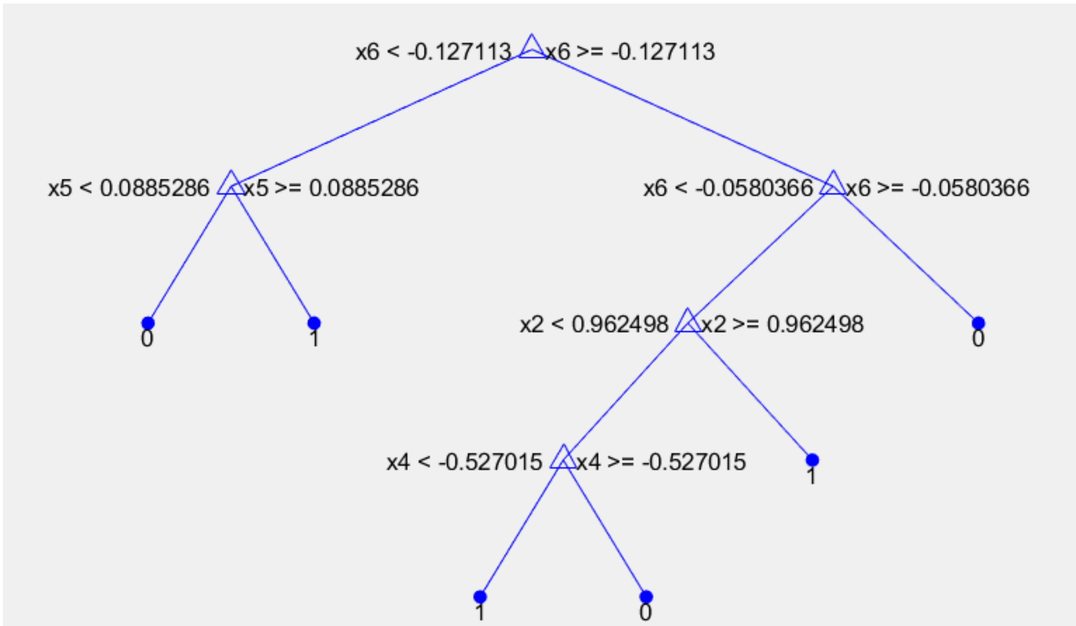
The metrics we obtained are:

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.375 | 0.5 | 0.42857 |

All the other signs we have followed the same process for generating the Decision tree and Accuracy metrics.

**AND**

a) The Decision Tree for AND looks like this:

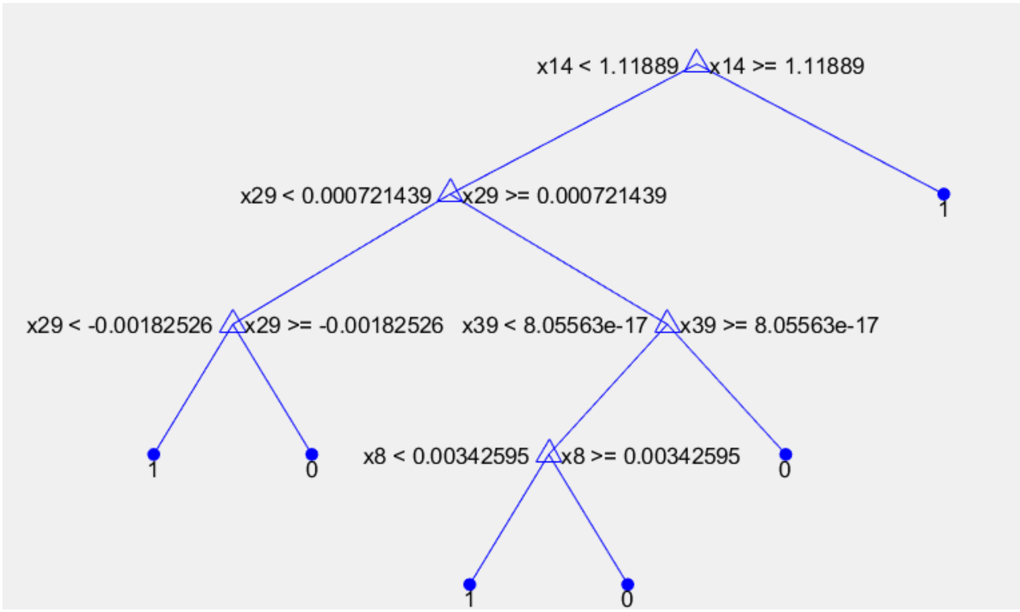



b) The metrics we obtained are:

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 0.9125 | 0.625 | 0.55556 | 0.58824 |

**CAN**

a) The Decision Tree for CAN looks like this:

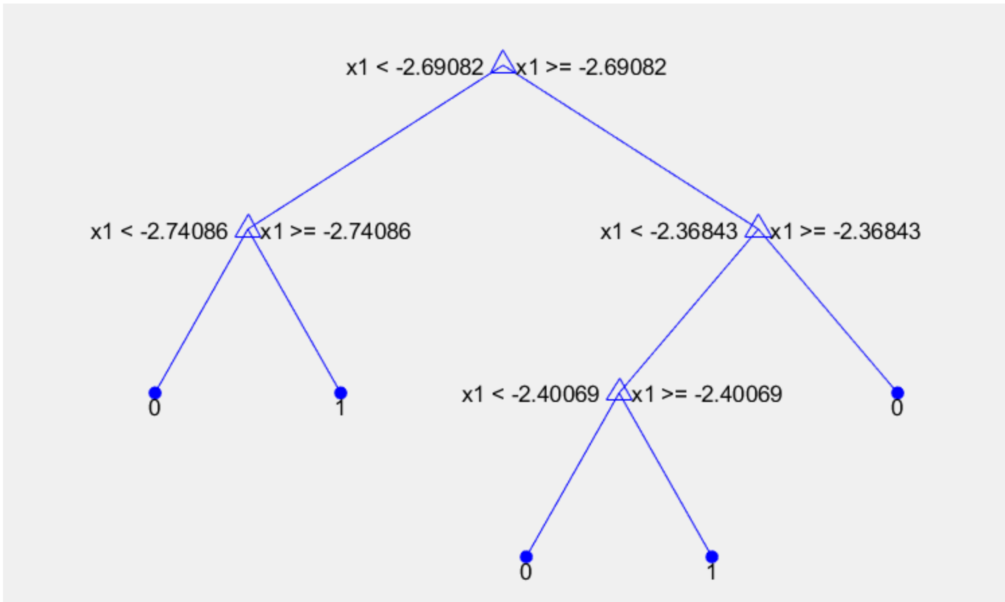



b) The metrics we obtained are:

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 1 | 1 | 1 | 1 |

**COP**

a) The Decision Tree for COP looks like this:

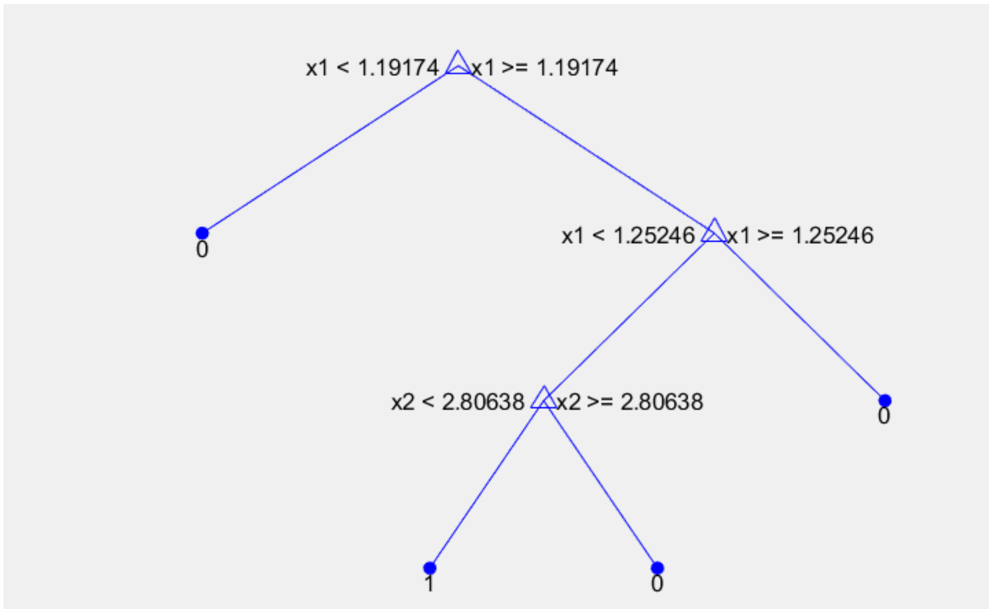



b) The metrics we obtained are :

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0 | | |

**DEAF**

a) The Decision Tree for DEAF looks like this :

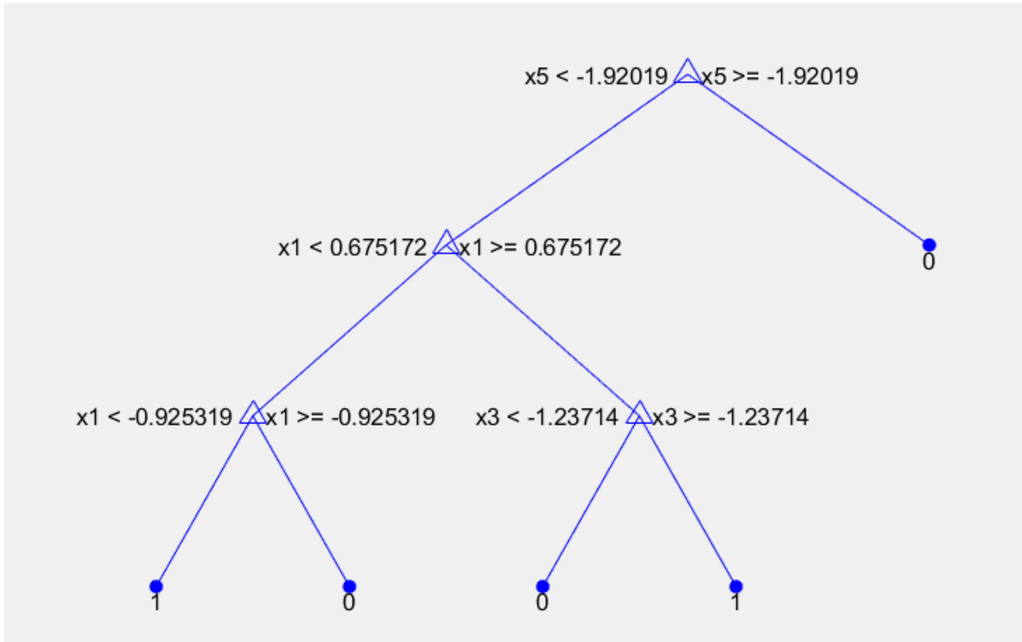



b) The metrics we obtained are :

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.25 | 0.5 | 0.33333 |

**DECIDE**

a) The Decision Tree for DECIDE looks like this :

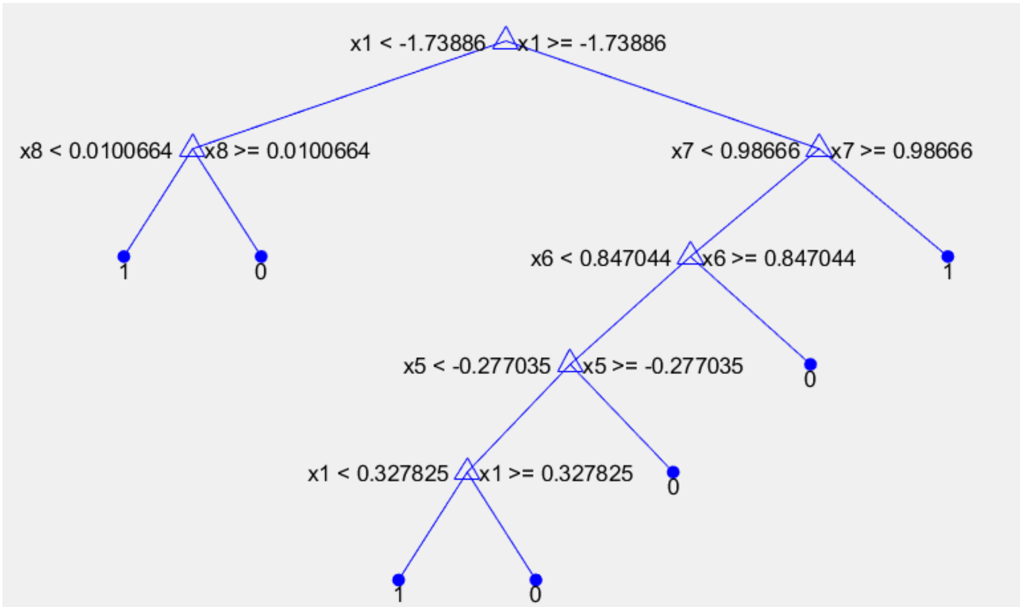



b) The metrics we obtained are :

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 0.8625 | 0.625 | 0.38462 | 0.47619 |

**FATHER**

a) The Decision Tree for FATHER looks like this :

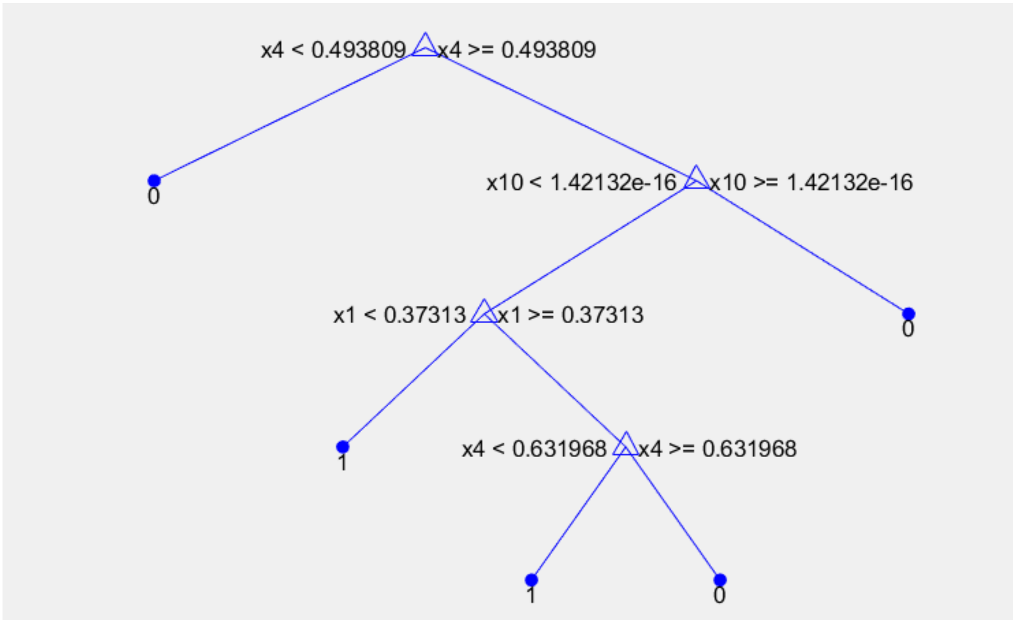



b) The metrics we obtained are :

| **User** | **DT_Accuracy** | **DT_Precision** | **DT_Recall** | **DT_F1** |
|---|---|---|---|---|
| DM02 | 0.925 | 0.25 | 1 | 0.4 |

**FIND**

a) The Decision Tree for FIND looks like this :

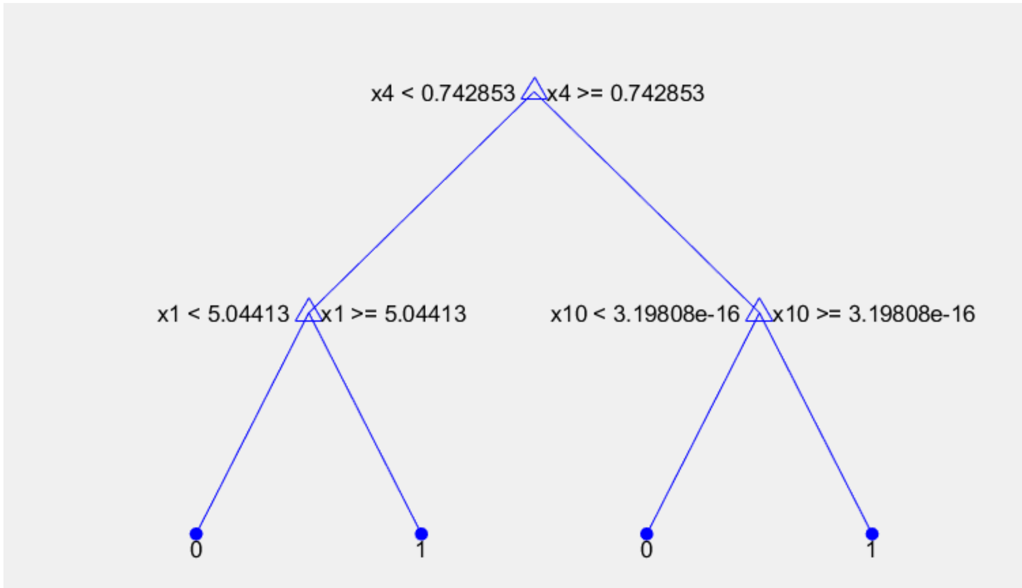



b) The metrics we obtained are :

| **User** | **DT_Accuracy** | **DT_Precision** | **DT_Recall** | **DT_F1** |
|---|---|---|---|---|
| DM02 | 0.9 | 0.125 | 0.5 | 0.2 |

**GO OUT**

a) The Decision Tree for GO OUT looks like this :

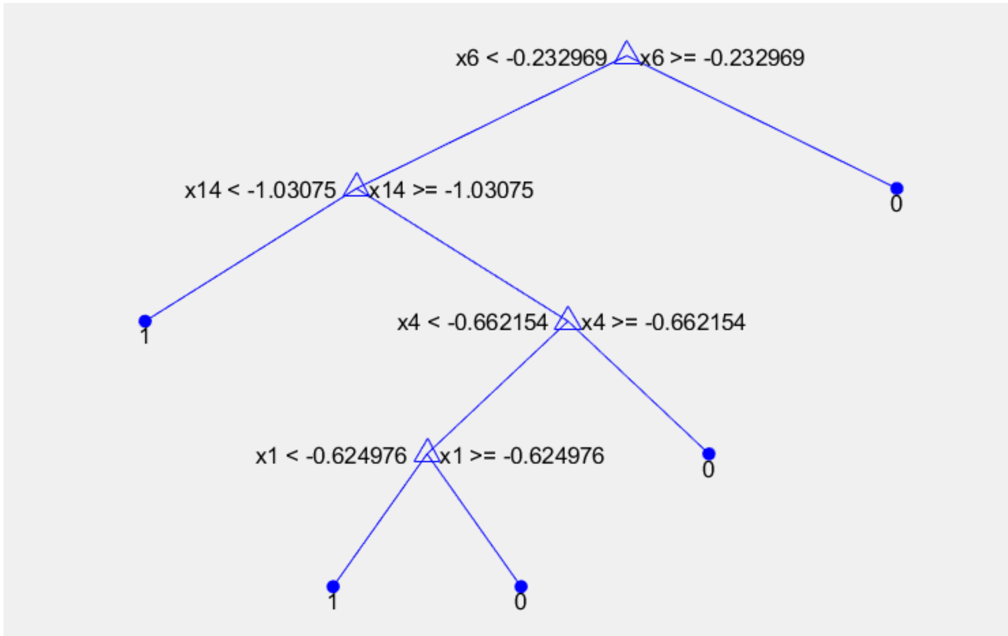



b) The metrics we obtained are :

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 0.7625 | 0.375 | 0.17647 | 0.24 |

**HEARING**

a) The Decision Tree for HEARING looks like this :




b) The metrics we obtained are:

| User | DT_Accuracy | DT_Precision | DT_Recall | DT_F1 |
|---|---|---|---|---|
| DM02 | 0.85 | 0.25 | 0.25 | 0.25 |

The metrics for all the other 9 users are given at the end.

## *4.3 Task 3: Support Vector Machines*

In Task 3 we are building a Machine for classifying the given 10 gestures using Support Vector Machines (SVM). In this phase we are using the Feature set which we obtained in Assignment two. The PCA output for the input data gives us the eigen vectors, when we multiply this with the input matrix we get the feature set. We are using this feature set of all actions performed by one user as our input for Task 3 classification.

Support Vector Machines: SVM is a classification technique which divides the data with a hyperplane into two broad categories. It's a supervised learning technique if we know what the possible class labels are. SVM identifies the hyper plane which divides the data into categories in such a way that it is closer to neither of the boundary.

### 4.3.1 Support Vector Machine model for User 1:

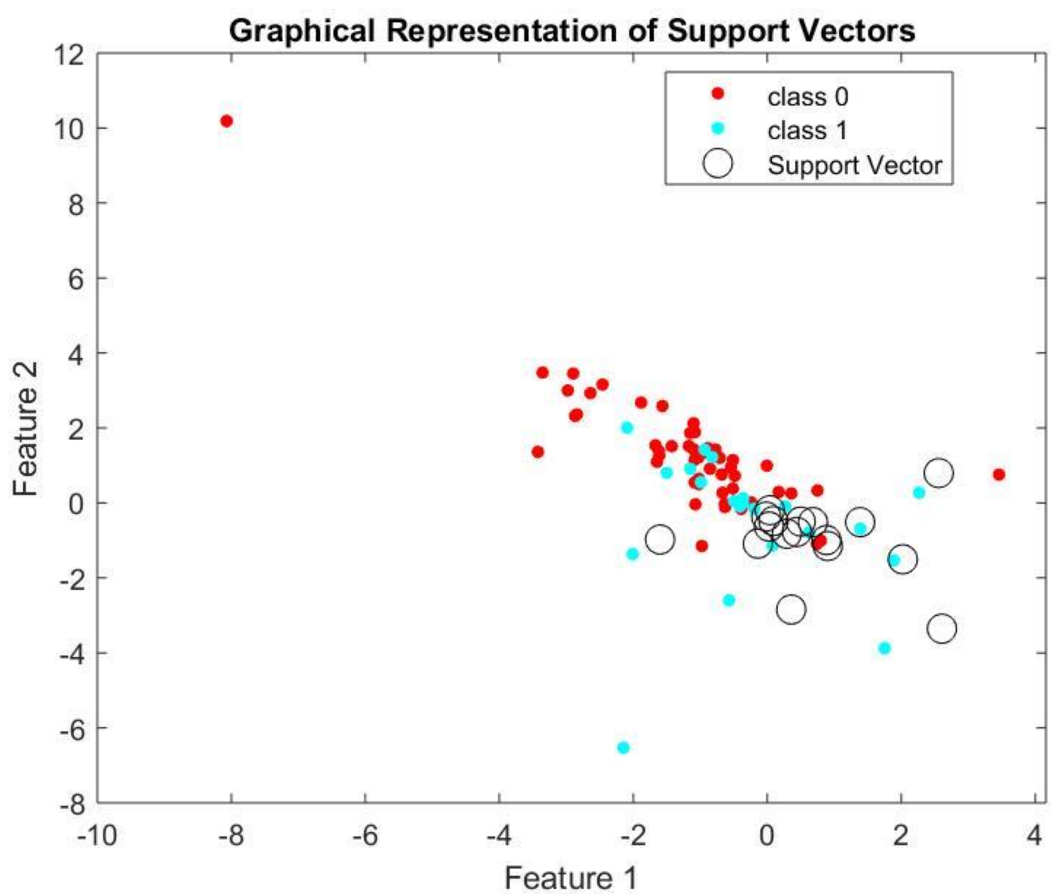
**ABOUT**

g) For Sign About for user 1 (DM02) we obtained the CSV file from Task 1 and divided the data into Test and Training. We have 20 latent features for About.

h) Training set contains 60% of the data with label 1 and 60% of data with label 0. Test set contains the remaining.

i) The SVM model is created using 'fitcsvm' functionality of matlab using the Training Data set.

j) The SVM model looks like this :




k) The accuracy metrics are calculated by getting the predictions of the test dataset by the decision tree.

l) The metrics we are calculating :

Accuracy % : Total correct predictions/Total no of Inputs * 100
True Positives = Total no of correct predictions for label 1
True Negatives = Total no of Correct predictions for label 0
False Positives = Total no of predicted 1 which are incorrect
False Negative = Total no of Predicted 0 which are incorrect

Precision = True Positive /(True Positive + False Positive)
Recall = True Positive /(True Positive + False Negative)

F1 Score = 2* ((precision*recall)/(precision + recall))

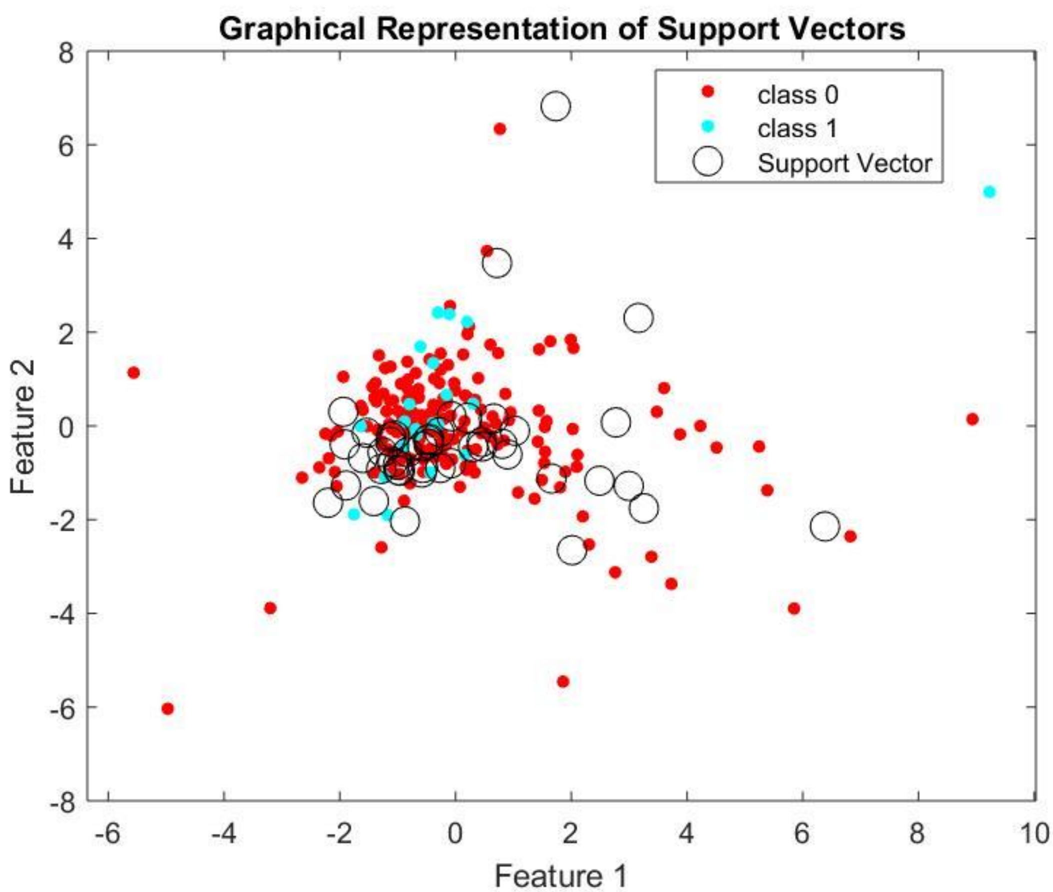
The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.375 | 0.5 | 0.42857 |

All the other signs we have followed the same process for generating the SVM model and Accuracy metrics.

**AND**

c) The SVM model for AND looks like this:

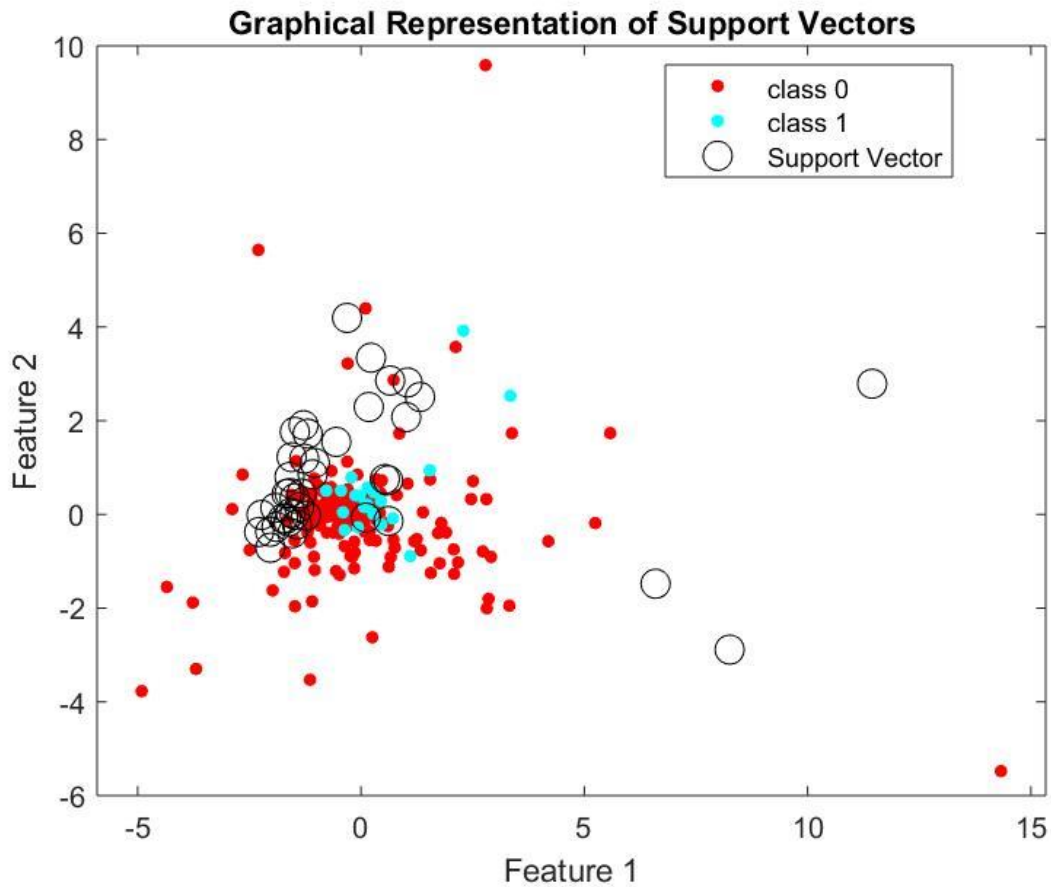



d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.9125 | 0.625 | 0.55556 | 0.58824 |

**CAN**

c) The SVM model for CAN looks like this:

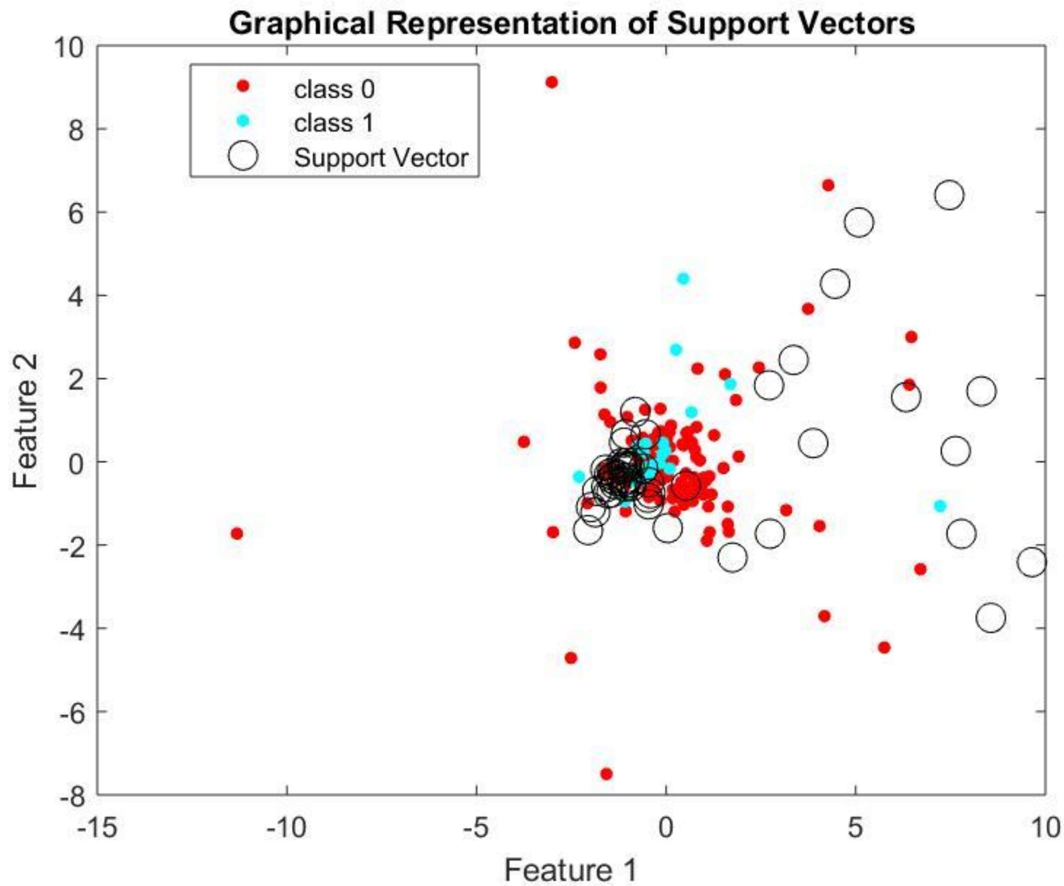



d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 1 | 1 | 1 | 1 |

**COP**

c) The SVM model for COP looks like this :

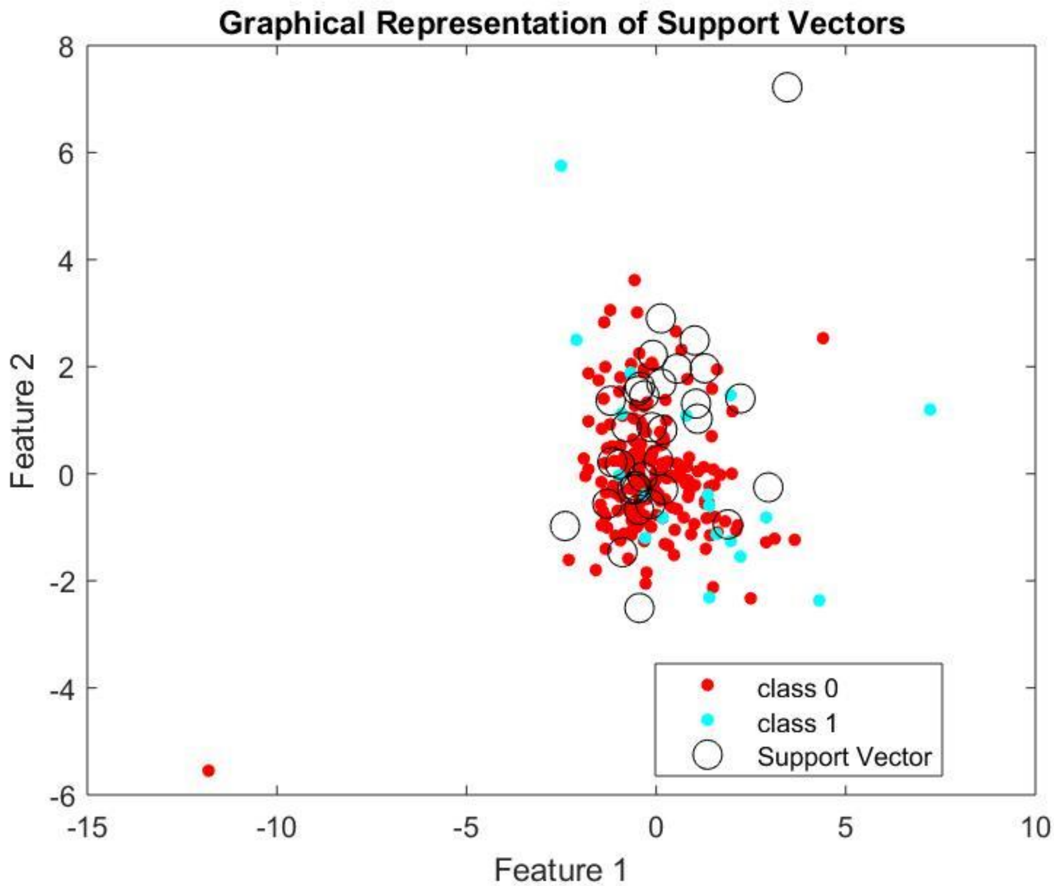



d) The metrics we obtained are :

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0 | | |

**DEAF**

c) The SVM model for DEAF looks like this :

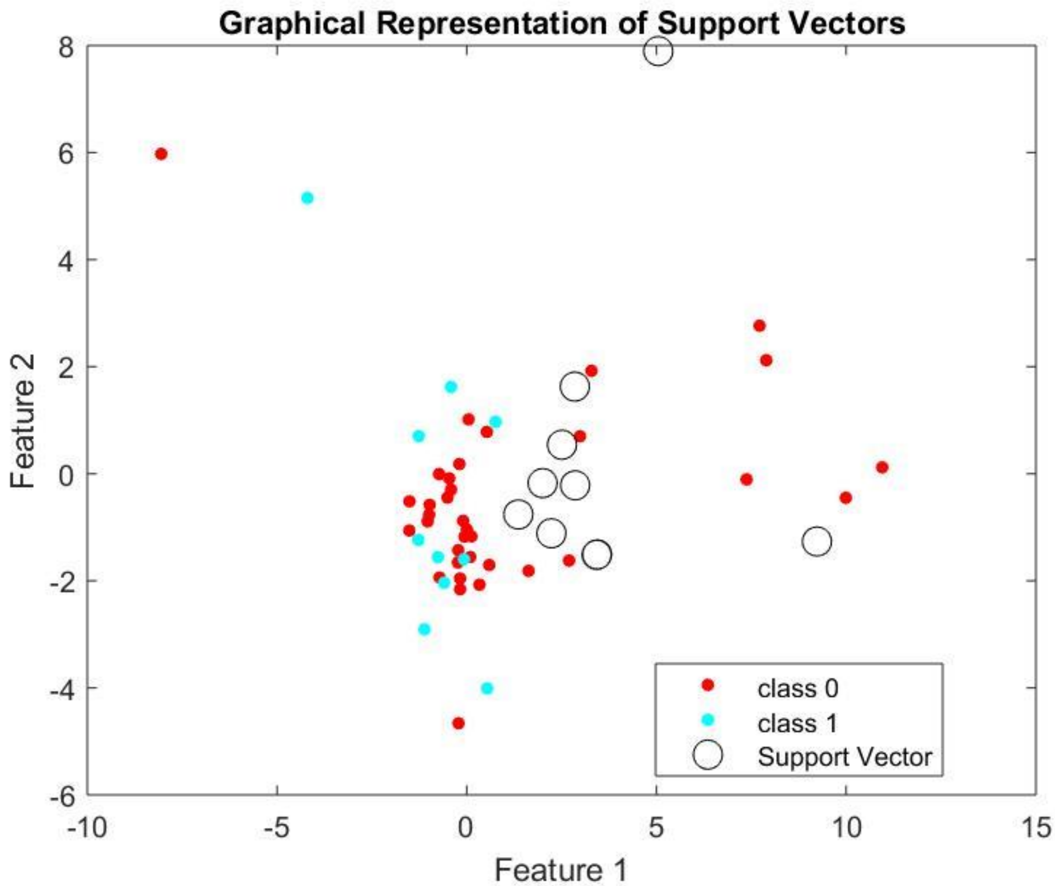



d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.25 | 0.5 | 0.33333 |

**DECIDE**

c) The SVM model for DECIDE looks like this :

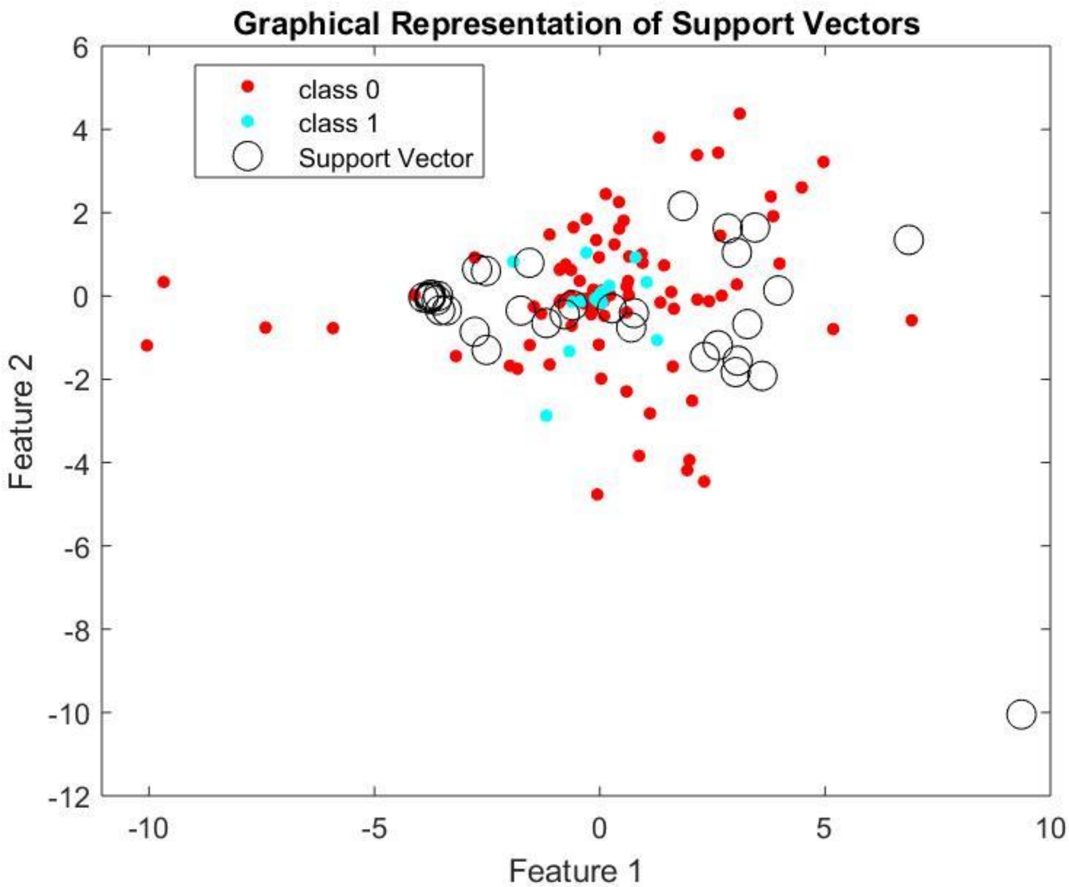



d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.8625 | 0.625 | 0.38462 | 0.47619 |

**FATHER**

c) The SVM model for FATHER looks like this :

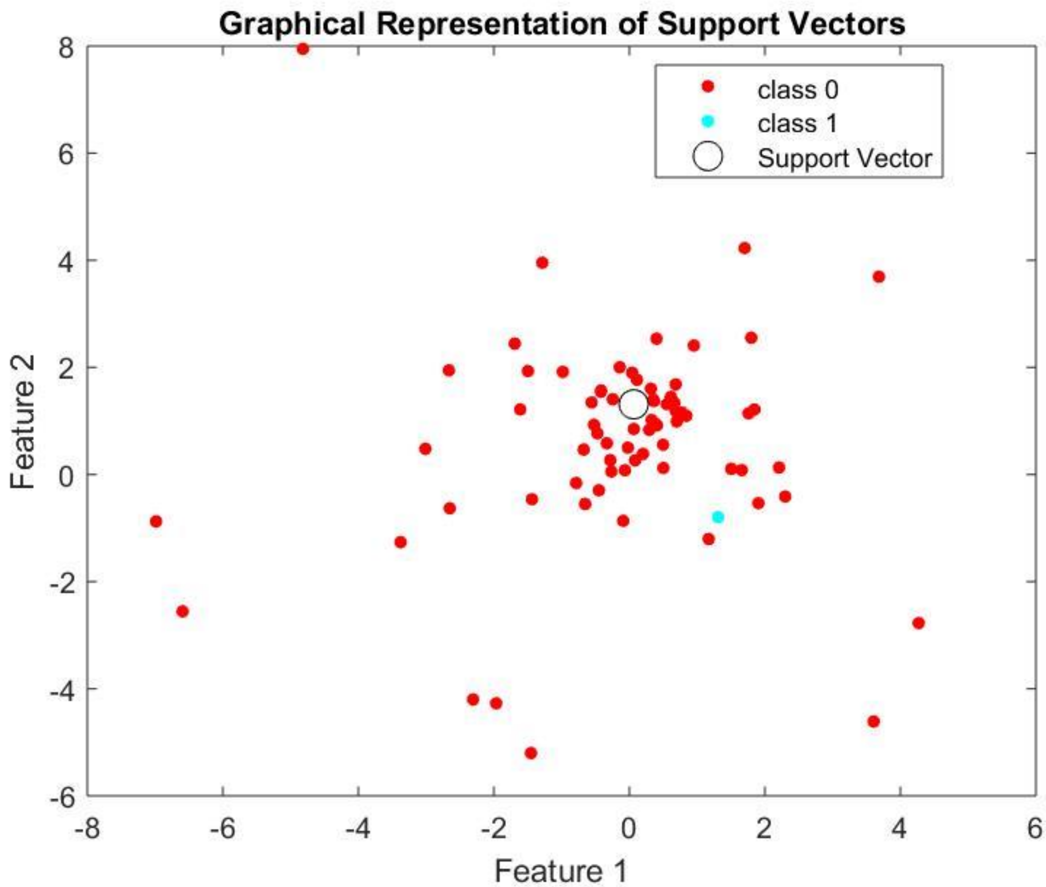



d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.925 | 0.25 | 1 | 0.4 |

**FIND**

c) The SVM model for FIND looks like this :

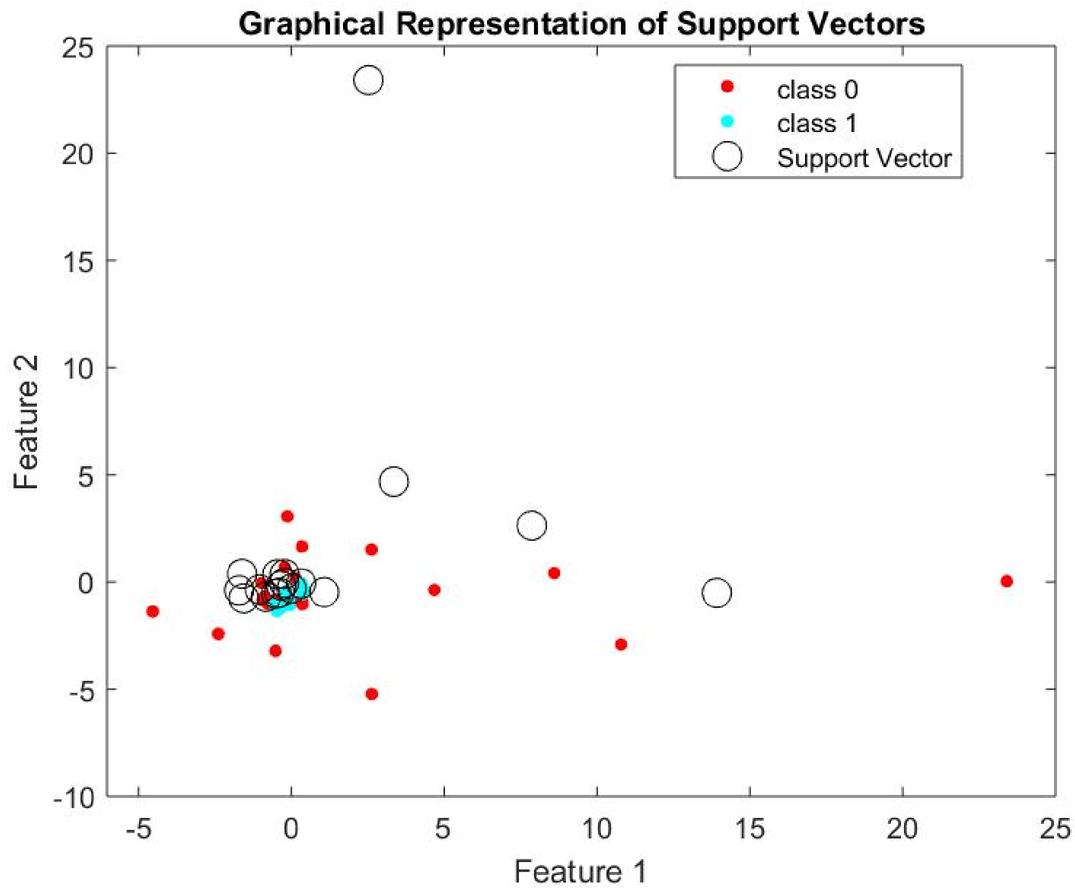



d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.125 | 0.5 | 0.2 |

**GO OUT**

c) The SVM model for GO OUT looks like this :

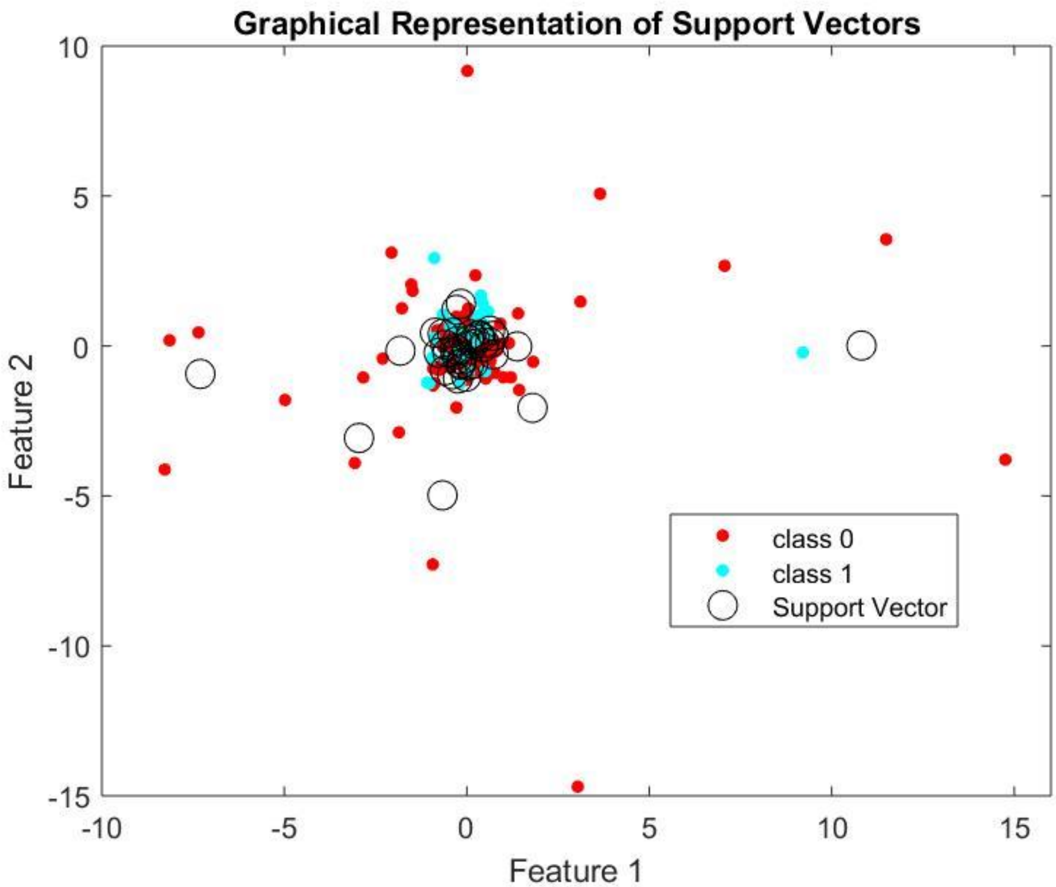



d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.7625 | 0.375 | 0.17647 | 0.24 |

**HEARING**

c) The SVM model for HEARING looks like this :




d) The metrics we obtained are:

| User | SVM_Accuracy | SVM_Precision | SVM_Recall | SVM_F1 |
|---|---|---|---|---|
| DM02 | 0.85 | 0.25 | 0.25 | 0.25 |

The metrics for all the other 9 users are given at the end.

## *4.4 Task 4: Neural Networks*

In Task 4 we are building a Machine for classifying the given 10 gestures using Neural Networks. In this phase we are using the Feature set which we obtained in Assignment two. The PCA output for the input data gives us the eigen vectors, when we multiply this with the input matrix we get the feature set. We are using this feature set of all actions performed by one user as our input for Task 3 classification.

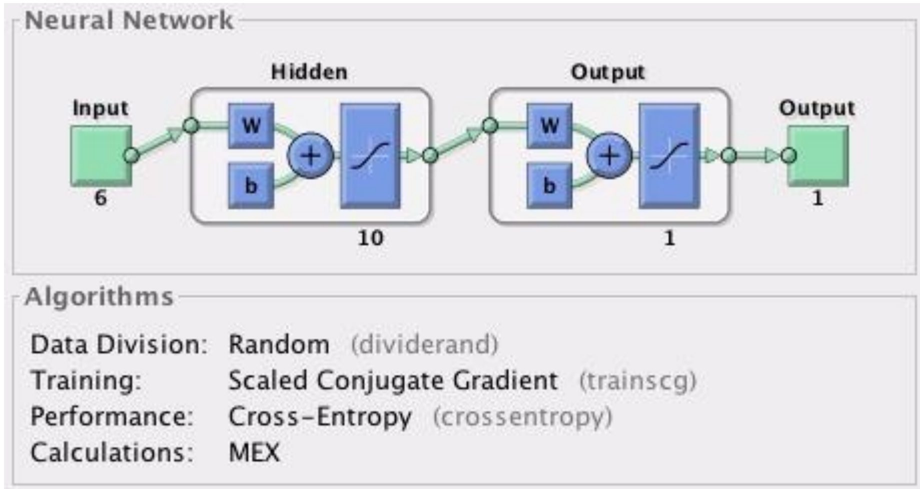
Neural Networks: Neural Networks are inspired by brain cells. Neural Networks have nodes or neurons. The nodes can be a part of Visible layers or Hidden layers. Based on the type and complexity we can add hidden layers in the Neural Network. The Nodes are interconnected by edges and we have an activation function which activates the nodes. The Neural Networks provides non-linear classification. Here we are using 10 Layers and 1000 epochs. We are using the patternnet function, commonly known as Pattern recognition network. These are feed forward networks which can be trained to classify inputs based on the target class.

### 4.4.1 Neural Network model for User 1:

**ABOUT**

a) For Sign ‘About’, user 1 (DM02) we obtained the CSV file from Task 1 and divided the data into Test and Training. We have 20 latent features for About.
b) We split the dataset into three. One is the training set contains which contains 40% of the data, next is the validation set containing 10% of the data, and the remaining 50% forms the test data.
c) The Neural Network model is created using ‘patternnet’ functionality of matlab.
d) We use the features and class variables of the data separately and the number of layers to train the model.
e) We modify some of the parameters of the Neural Network to achieve better accuracy and build a more efficient model. Some of the parameters which were modified are
   1. The TransferFcn block, which is used to model a linear system by a transfer function of the Laplace-domain variable. We chose the “tansig” model.
   2. We set the minimum gradient value to 1.0000e-15. We keep a very low value so that the learning is not stopped, and sufficient learning is done before this value is reached.
   3. We set the learning rate to 0.0001.
   We started with a very basic neural network and improved the efficiency of the model by iteratively changing the values of the aforementioned parameters.

f) The Pattern Recognition Neural Network of this gesture looks like this:




g) The accuracy metrics are calculated by getting the predictions of the test dataset by the Neural Network.

h) We use the confusion matrix to calculate the following metrics:

Accuracy% = Total correct predictions/Total no of Inputs * 100

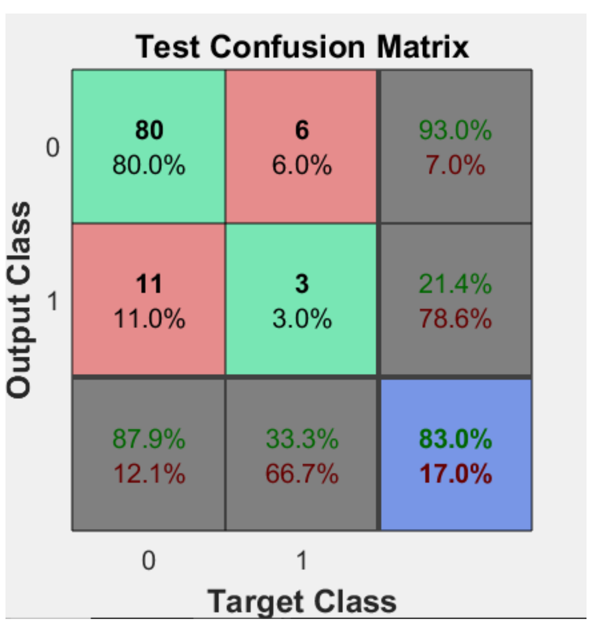
True Positives = Total no of correct predictions for label 1
True Negatives = Total no of Correct predictions for label 0
False Positives = Total no of predicted 1 which are incorrect
False Negative = Total no of Predicted 0 which are incorrect

Precision = True Positive / (True Positive + False Positive)
Recall = True Positive / (True Positive + False Negative)

F1 Score = 2* ((precision*recall)/ (precision + recall))

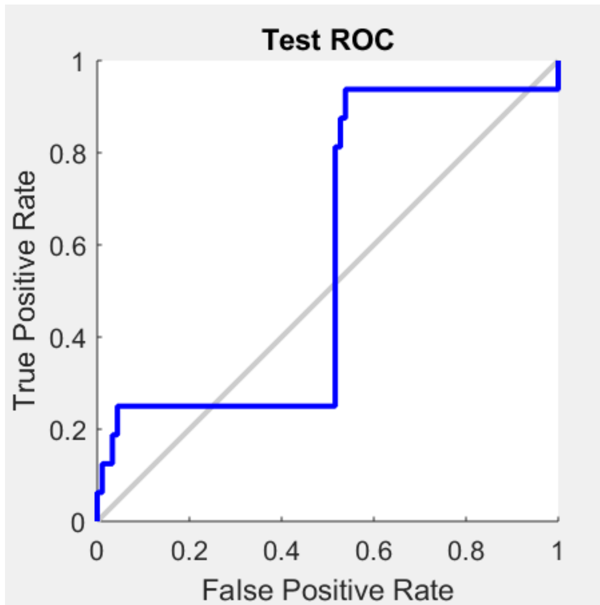
i) We also compute the confusion matrix for training, validation and test data separately.

j) The confusion matrix of ABOUT gesture for this user is:



k) We generate the ROC curve to illustrate the diagnostic ability of a binary classifier system as its discrimination threshold is varied. We create the ROC curve by plotting the True Positive Rate against the False Positive Rate. The ROC curve of the About gesture is given below.

l) The ROC curve of the ABOUT gesture is:



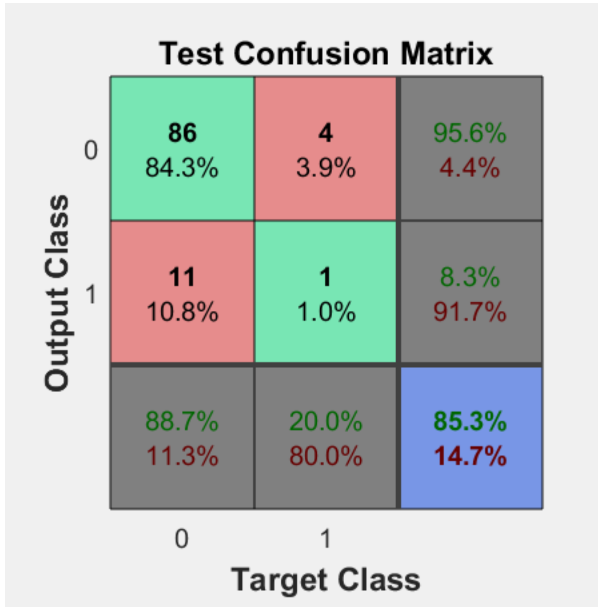

The metrics we obtained are:

| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.375 | 0.5 | 0.42857 |

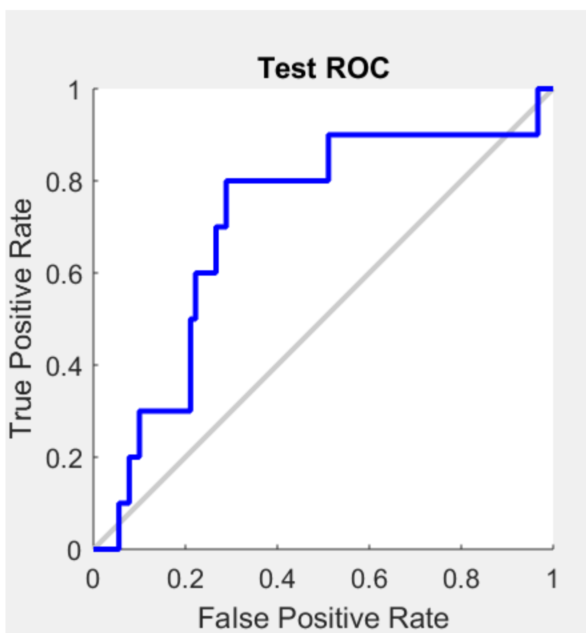
All the other signs we have followed the same process for generating the SVM model and Accuracy metrics.

**AND**

a) The confusion matrix of AND gesture for this user is:




b) The ROC curve of the AND gesture is:

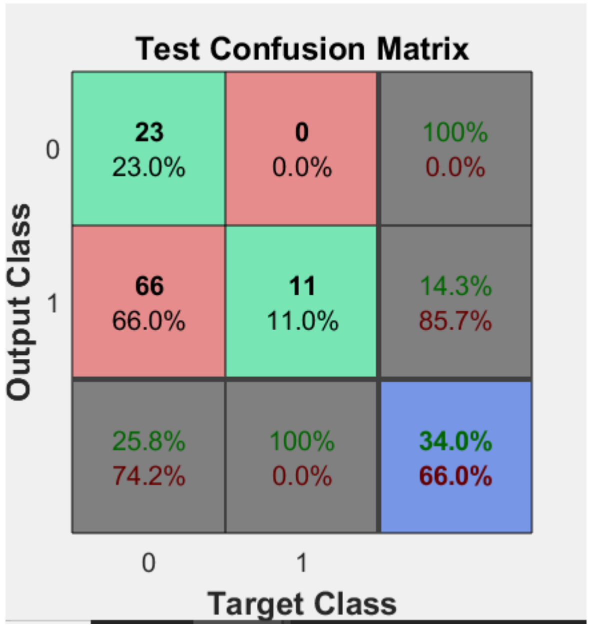



The metrics we obtained are:

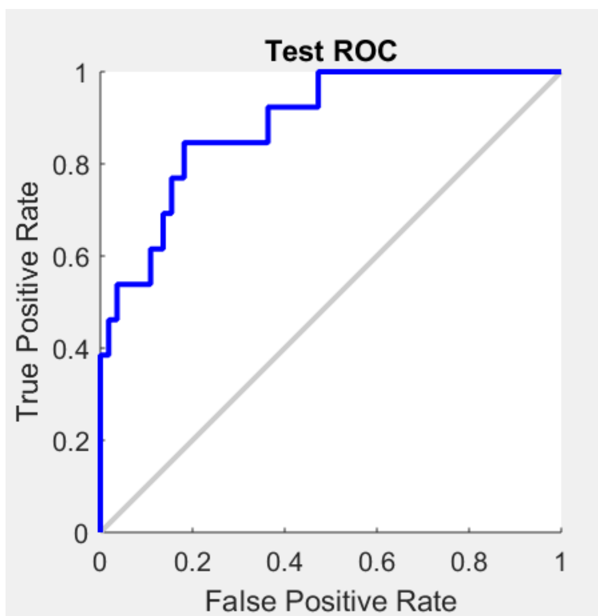
| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.9125 | 0.625 | 0.55556 | 0.58824 |

**CAN**

a) The confusion matrix of CAN gesture for this user is:




b) The ROC curve of the CAN gesture is:

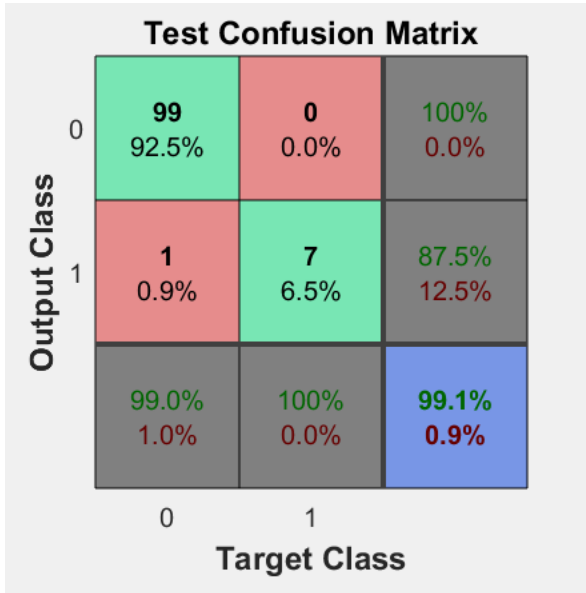



The metrics we obtained are:

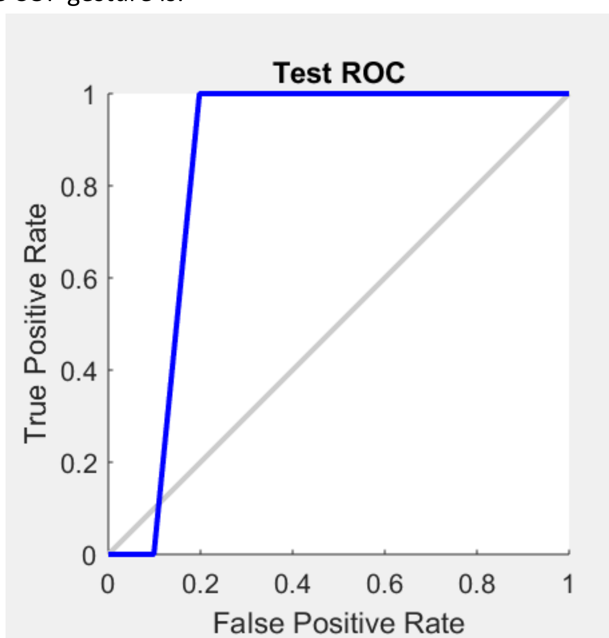
| **User** | **NN_Accuracy** | **NN_Precision** | **NN_Recall** | **NN_F1** |
|---|---|---|---|---|
| DM02 | 1 | 1 | 1 | 1 |

**COP**

a) The confusion matrix of COP gesture for this user is:




b) The ROC curve of the COP gesture is:

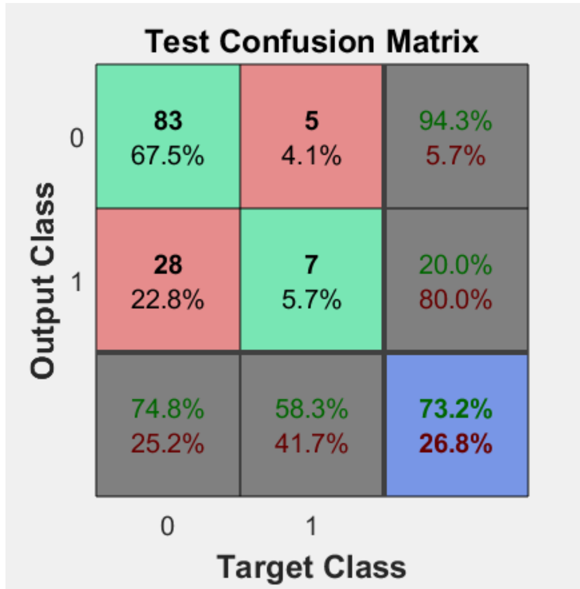



The metrics we obtained are:

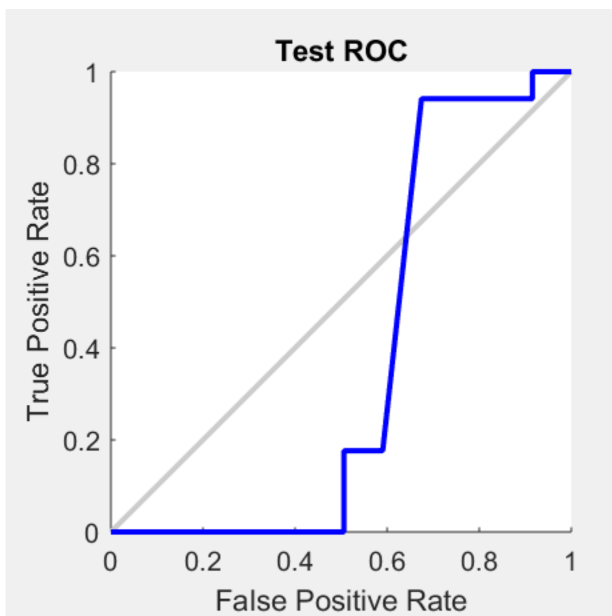
| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0 | | |

**DEAF**

a) The confusion matrix of DEAF gesture for this user is:




b) The ROC curve of the DEAF gesture is:

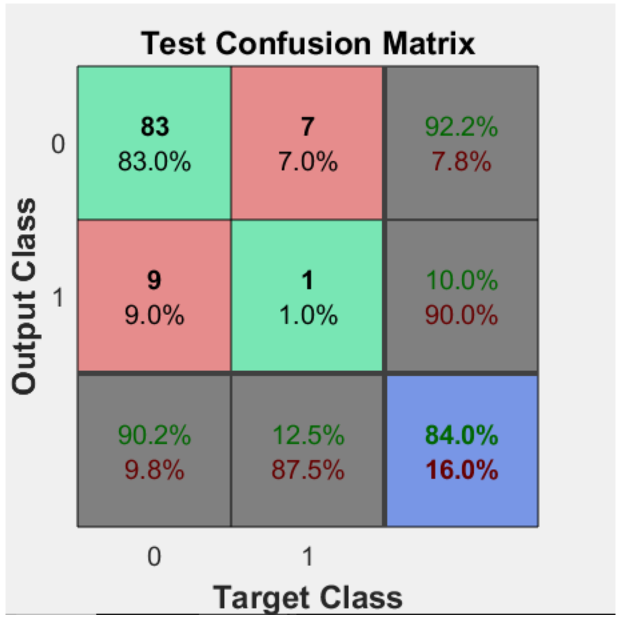



The metrics we obtained are:

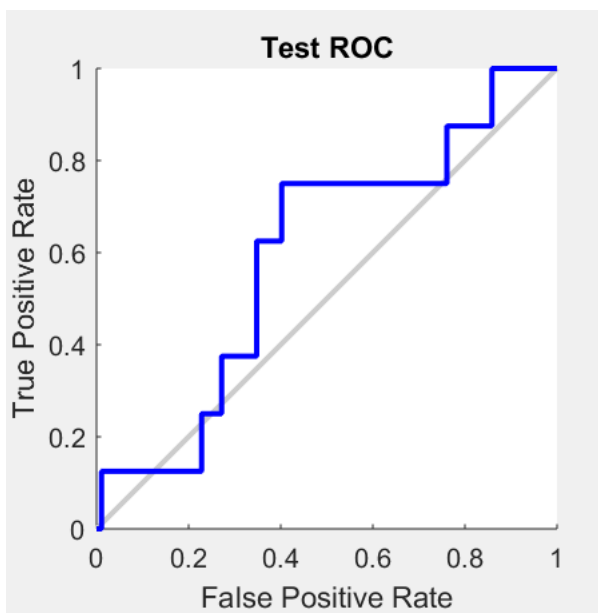
| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.25 | 0.5 | 0.33333 |

**DECIDE**

a) The confusion matrix of DECIDE gesture for this user is:




b) The ROC curve of the DECIDE gesture is:

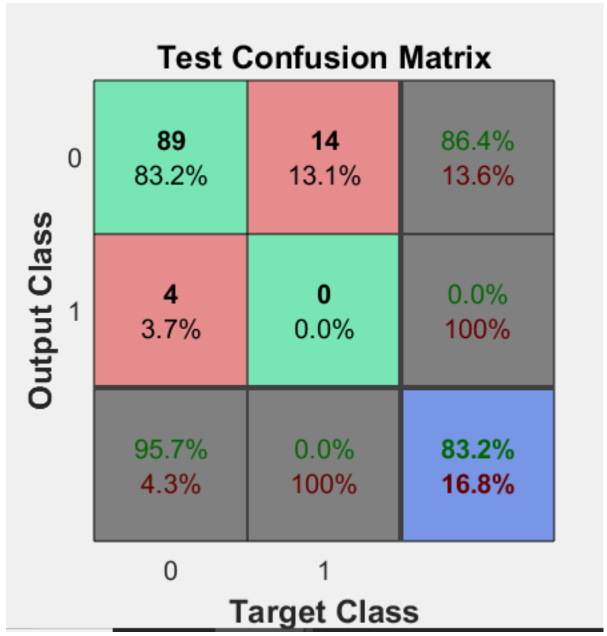



The metrics we obtained are:

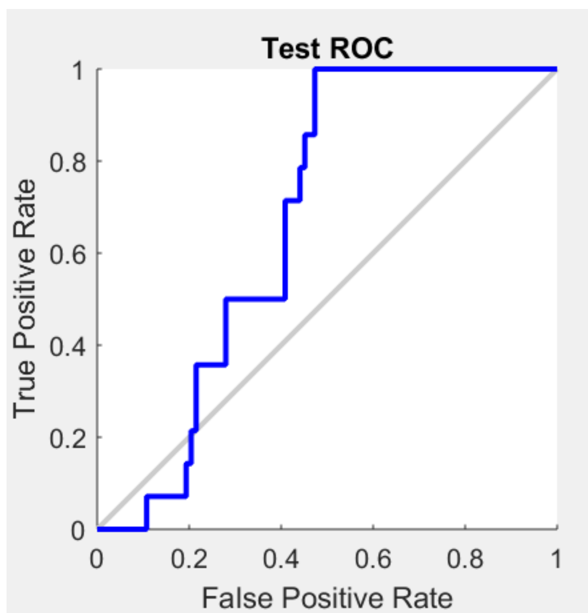
| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.8625 | 0.625 | 0.38462 | 0.47619 |

**FATHER**

a) The confusion matrix of FATHER gesture for this user is:




b) The ROC curve of the FATHER gesture is:

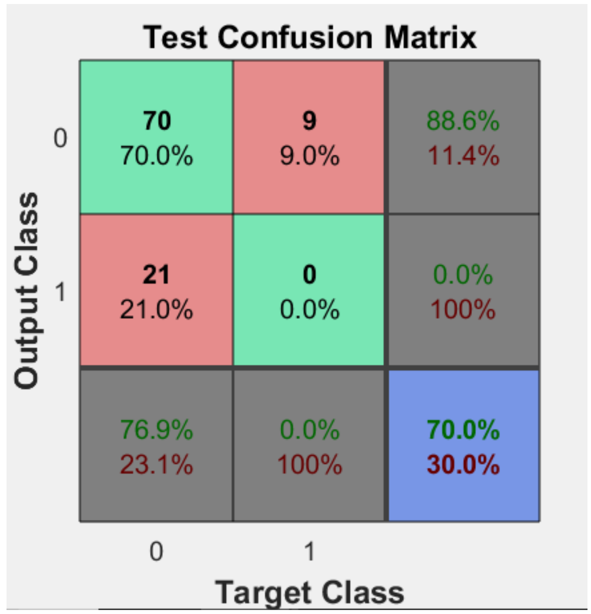



The metrics we obtained are:

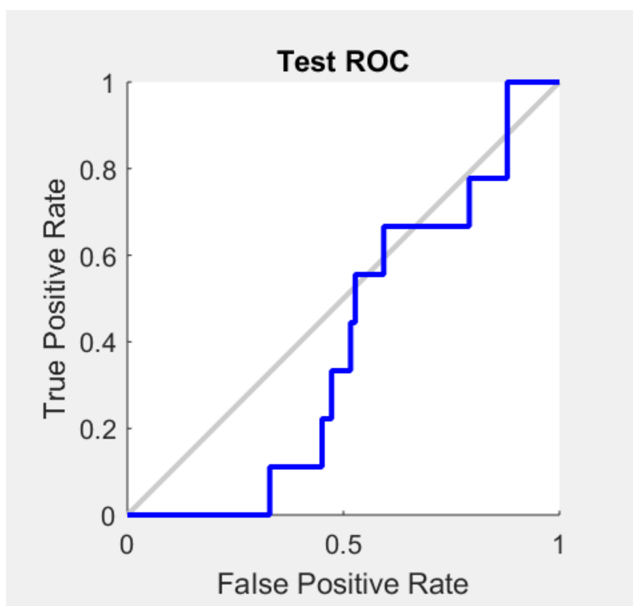
| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.925 | 0.25 | 1 | 0.4 |

**FIND**

a) The confusion matrix of FIND gesture for this user is:




b) The ROC curve of the FIND gesture is:

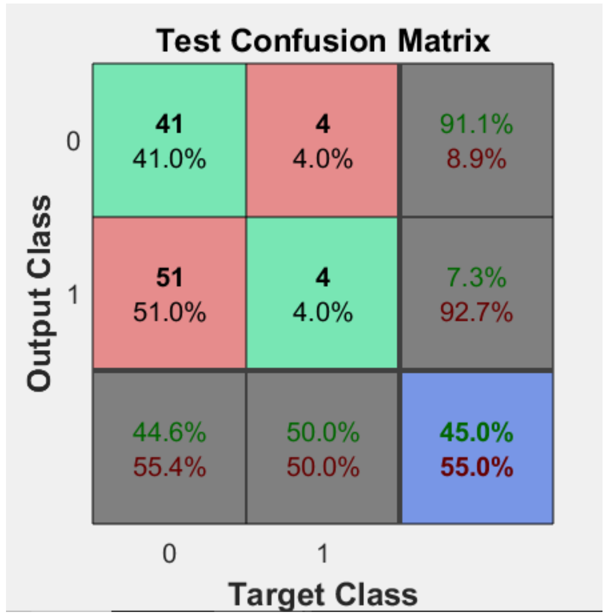



The metrics we obtained are:

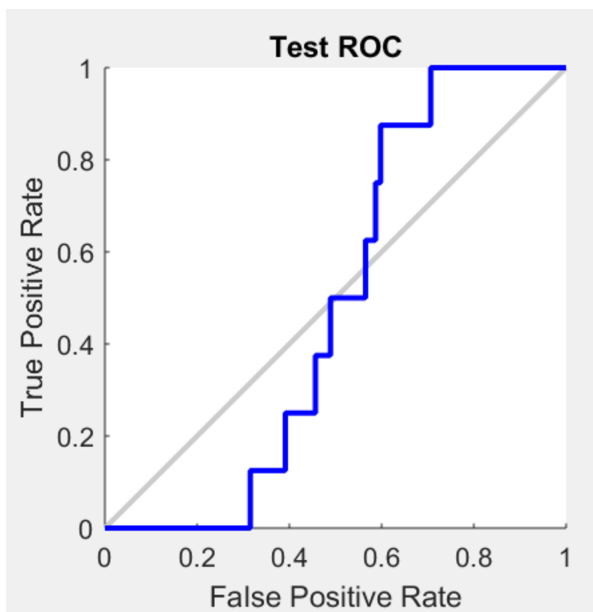
| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.9 | 0.125 | 0.5 | 0.2 |

**GO OUT**

a) The confusion matrix of AND gesture for this user is:




b) The ROC curve of the AND gesture is:

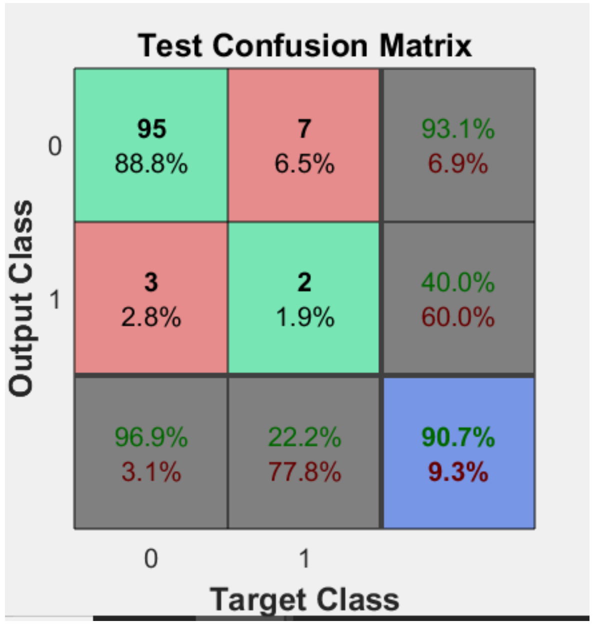



The metrics we obtained are:

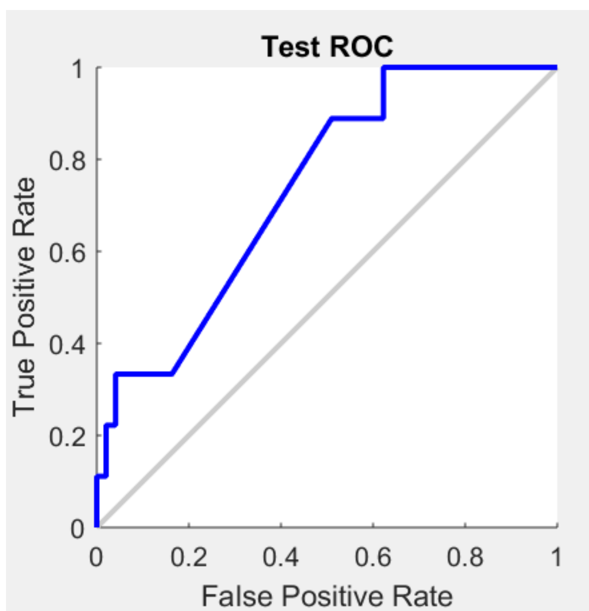
| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.7625 | 0.375 | 0.17647 | 0.24 |

**HEARING**

a) The confusion matrix of AND gesture for this user is:




b) The ROC curve of the AND gesture is:




The metrics we obtained are:

| User | NN_Accuracy | NN_Precision | NN_Recall | NN_F1 |
|---|---|---|---|---|
| DM02 | 0.85 | 0.25 | 0.25 | 0.25 |

The metrics for all the other 9 users are given at the end.

# 5. Statistical Measures:

a) Below excel contains the statistical measure of all the gestures for 10 different users.

performance.xlsx

# 6. Conclusion

a) Classification of gestures using all the 3 algorithms on an average gave an Accuracy of about 80 – 90% for majority of the users.

b) Due to sparse data (user specific) for positive class, it was difficult to classify the gesture with user dependency and the precision and recall values were not up to the expectations. Though for some of the gestures we still got precision value of 1, at the same time for some users we also got '0/0' value for recall and F1-score. Even though the accuracy was high, as the number of class 0 data was very high when compared to class 1 data, we can't be sure that the machine which we built is very efficient as it can be seen from the precision, recall and F1 scores.

c) The above results show that the feature extraction and PCA performed during phase 2 of the project was useful in classifying the data.

# 7. Bibliography


a) https://www.mathworks.com/

b) https://www.wikipedia.com/

c) https://www.stackoverflow.com