Human Gesture Recognition
Phase III Report
Data Mining
CSE 572
Spring 2018

Professor Ayan Banerjee
Ira A. Fulton School of Engineering
Arizona State University

Submitted By: Group 20
Sam Sunder Golla (sgolla2@asu.edu)
Aravind Manoharan (amanoh10@asu.edu)
Anto Oswin Nihal (anihal@asu.edu)
Pravin Kumar Ravi (pravi8@asu.edu)
Madhu Venkatesh (mvenka24@asu.edu)

Table of Contents

S.No	Topic			Page
-	Index			1
1	Introduction			2
2	The Team			2
3	Phase 1 : Data Collection			2
4	Phase 2 : Feature Recognition			2
	4.1	Task 1 : Data Synchronization		2
	4.2	Task 2 : Feature Selection		3
5	Phase 3 : User Dependent Analysis			3
	5.1	Data Preparation		3
	5.2	Accuracy Metrics		4
	5.3	Gestures Legend		5
	5.4	Classification		5
		5.4.1	Decision Tree	6
		5.4.2	Neural Network	17
		5.4.3	Support Vector Machine (SVM)	28
6	Conclusion			39
7	Resources/ Citations			39

1. Introduction

This academic project is a part of the fall 2018 course - Data Mining (CSE 572) at the IRA Fulton School of Engineering at Arizona State University, Tempe, USA. In this project, we attempt to develop an intelligent system that would be able to accurately recognize human gestures, upon its completion. We used two Myo Gesture Control Armbands (one for each hand), from the Impact Lab at ASU, to gather gesture data for each hand, and later analyzed and processed the data to obtain logical patterns within. This report describes our progress while working on the third phase of the project.

2. The Team

The team consists of the following members:

- Sam Sunder Golla
- Aravind Manoharan
- Anto Oswin Nihal
- Pravin Kumar Ravi
- Madhu Venkatesh

3. Phase 1: Data Collection

In phase 1, we recorded the Myo sensor readings for each gesture. One of the team members, performed 10 gestures (and, about, can, cop, deaf, decide, father, find, go out, hearing) 20 times (i.e 20 actions for each gesture) wearing one Myo sensor on each hand. The Myo sensor is equipped with the following sensors to record the hand movements: Accelerometer sensor, Electromyography sensor, Gyroscope sensor, and Orientation sensor. The sensor readings for each action of a gesture is stored in a file (20 actions x 10 gestures x 37 groups = approx. 7400 total files) to serve as input for phase 2.

4. Phase 2 : Feature Recognition

In the second phase, we synchronized the data which we collected in the first phase and we performed feature extraction and feature selection.

4.1 Feature Selection

The objective is to represent the data in frequency domain. We used the following five feature extraction methods:

1) Fast Fourier Transform (FFT)
2) Power Spectral Density (PSD)

3) Discrete Wavelet Transform (DWT)
4) Root Mean Square (RMS)
5) Standard Deviation (STD)

We used the Matlab functions FFT, PSD, DWT, RMS and STD to convert our data into frequency domain. We applied these methods only to attributes for which the feature values are more distinct from each gesture. After applying these methods, we performed feature extraction by selecting top three peak values for FFT, PSD and DWT methods and the scalar values for RMS and STD methods. In the end, we got 55 features for each instance.

4.2 Task 3 : Feature Selection

The objective here is to select best features among the features which we have extracted from the feature extraction method. We performed PCA on our dataset for the feature selection. We used the Matlab inbuilt function PCA to find the Eigen vectors of the corresponding data. We selected those Eigen vectors which contains 90% variance. Then we obtained the new features by multiplying those Eigen vectors with the input data. We finally end up with 8 features. Thus, we performed dimensionality reduction from 55 features to 8 important features.

5 Phase 3 : User Dependent Analysis

Since we are going to perform classification based on user dependent data, we couldn't use the output data of second phase since it has data from all the users. Hence, we performed the second phase again but this time we didn't do it with all the users' data whereas we performed individually on each user data.

5.1 Data preparation

We are doing user dependent classification only on 10 users rather than all 37 users. We have selected 10 users who are User1, User5, User11, User16, User19, User20, User22, User24, User28 and User36. We have selected these users based on who has more number of actions for all the gestures and also each action has exactly 45 values (at the frequency of 15Hz for 3 seconds). After selecting the 10 users, we performed feature extraction and feature selection individually same as the second phase. After the feature selection, we get new features for each gesture. We just appended each gesture's data and assign a class label of 1 to 10 for each gesture.

5.2 Accuracy Metrics

We used the accuracy metrics of Precision, Recall, F Value and Accuracy Percentage for measuring the accuracy of every classification machine. They are described below:

Precision: the number of correctly classified positive examples compared with all the examples classified as positive. Precision defines the amount of relevant data retrieved.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Recall: the number of correctly classified positive examples compared with all the positive examples in the data. Recall defines the how much of retrieved data is relevant.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

F Score: The harmonic mean of precision and recall. To avoid the confusion based on precision and recall while selecting a model we use F1 score as baseline for decision.

$$F1\ Score = \frac{2 * (Recall * Precision)}{Recall + Precision}$$

$$Accuracy\ \% = \frac{(True\ Positives + True\ Negatives) * 100}{True\ Positives + False\ Positives + True\ Negatives + False\ Negatives}$$

The metrics are reported for every model corresponding to the gestures performed by every user.

5.3 Gesture Legend

The 10 gestures were assigned class numbers for the sake of simplicity. The following table describes the assignment:

Class	Gesture
1	ABOUT
2	AND
3	CAN
4	COP
5	DEAF
6	DECIDE
7	FATHER
8	FIND
9	GO OUT
10	HEARING

5.4 Classification

We randomized the data and extracted 60% of it for training the classifier and the remaining 40% for testing the classifier. We used the following machines for classifying the gestures:

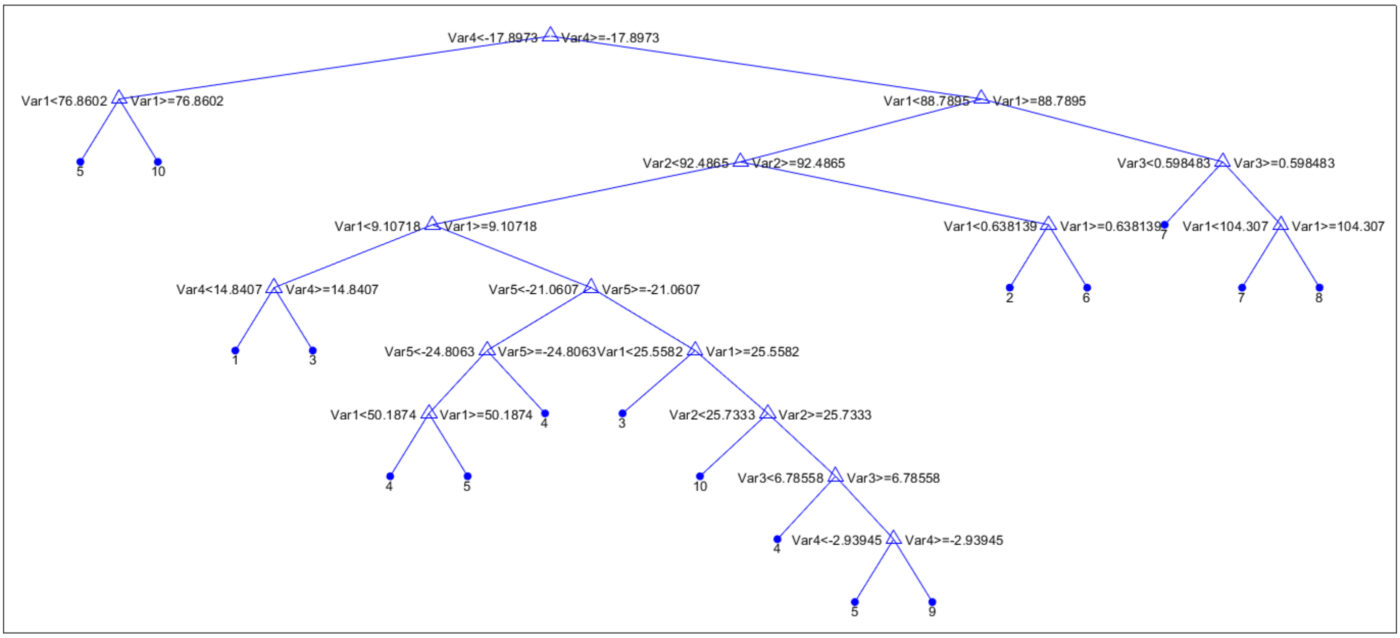
1) Decision Tree
2) Neural Network
3) Support Vector Machine (SVM)

5.4.1 Decision Tree

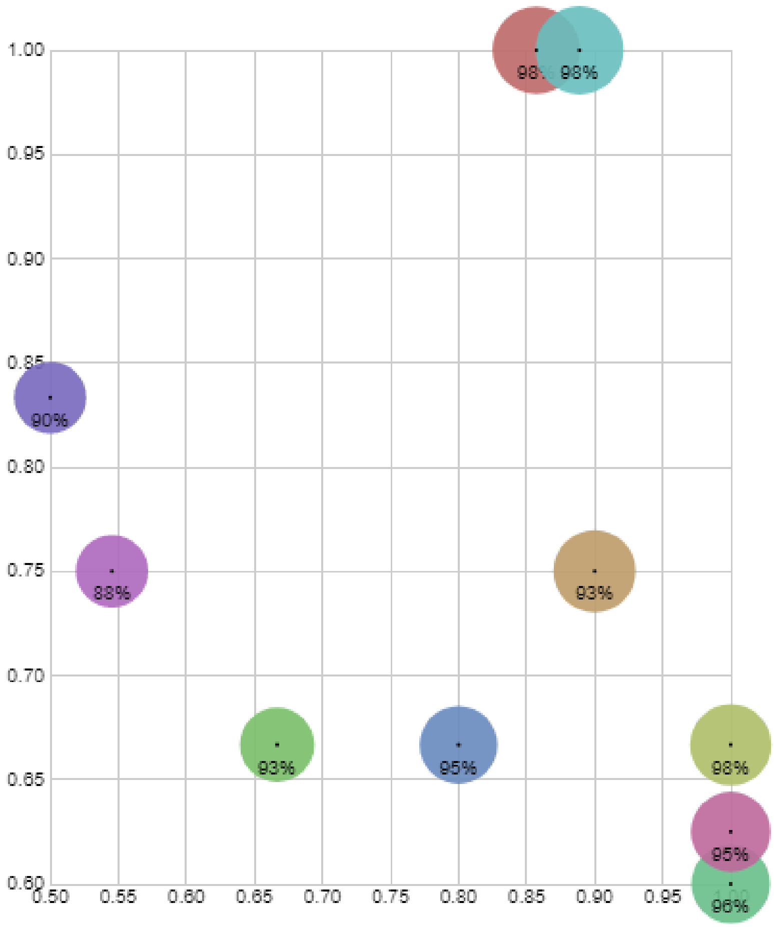
- For Classification and Regression, we use Decision Trees as a non-parametric supervised learning method. Decision Trees predict responses to data by learning decision rules inferred while traversing through the training data.
- A Decision tree is a map of the possible outcomes of a series of related choices. It allows weighing possible actions against one another based on their costs, probabilities, and benefits. We can use them to map out an algorithm that predicts the best choice mathematically.
- A decision tree typically starts with a single node, which branches into possible outcomes. Each of those outcomes leads to additional nodes, which branch off into other possibilities. In our project, nodes represent data rather than decisions. This type of tree is also known as classification tree. Each branch contains a set of attributes, or classification rules, that are associated with a particular class label, which is found at the end of the branch. These rules (decision rules) can be expressed in an if-then clause, with each decision or data value forming a clause, such that, for instance, “if conditions 1,2,3 are fulfilled, then outcome x will be the result with y certainty. Each additional piece of data helps the model more accurately predict which of the finite set of values the subject in question belongs to.
- The cost of using the tree (predicting data) is logarithmic in the number of data points used to train the tree.
- To use decision tree in our project, we used MATLAB’s *fitctree* function to train the decision tree using the train data.
- We then used MATLAB’s *predict* function to predict the classes of test data using the trained decision tree.
- We generated the confusion matrix by comparing the actual results with the predicted results and calculated the accuracy metrics using the formulas defined in the previous section.
- At the end, we used MATLAB’s *view* function to visualize the decision trees and stored the accuracy metrics for each gesture of each user in individual *.csv* files for further use.
- Please refer *DTREE.m* file for the complete code.
- We generated a Precision versus Recall graph for each user. The X-Axis will represent Precision, Y-Axis will represent Recall, size of the node will vary according to F1 value, label on each node will denote accuracy % value and the colors will represent the classes.
- The results that we obtained are as follows:

User 1

USER 1 DECISION TREE



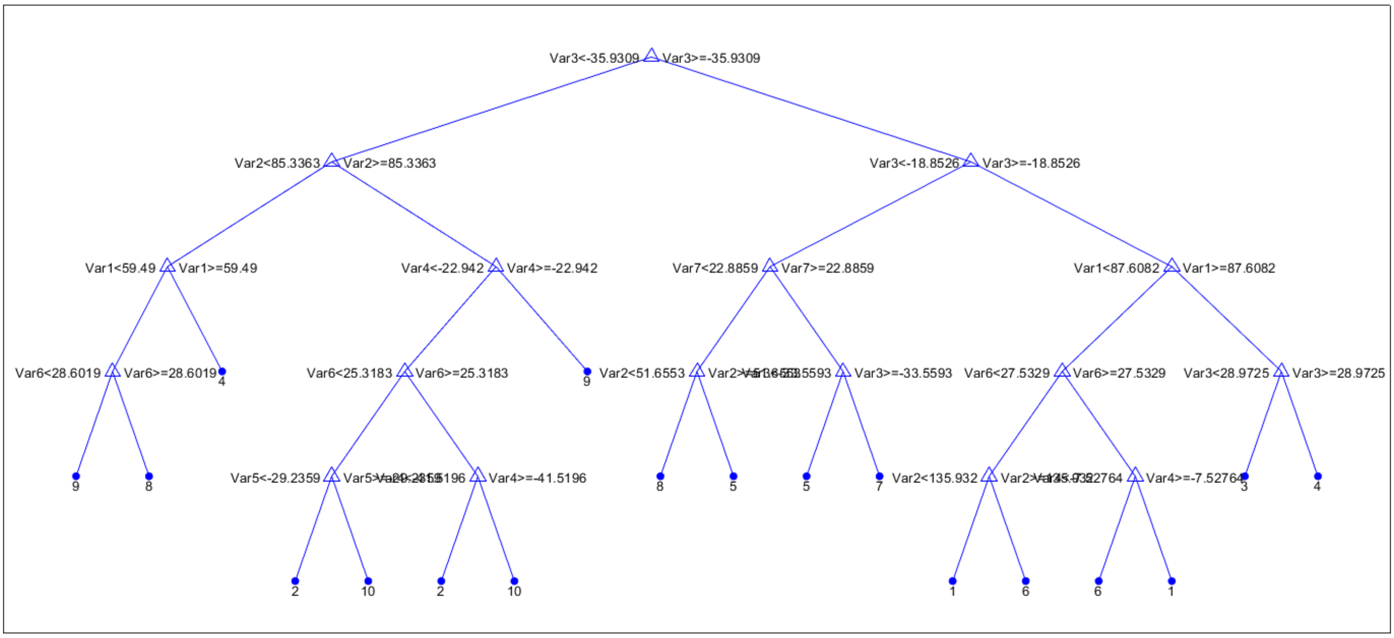
Class	Precision	Recall	F1	Accuracy
1	0.86	1.00	0.92	98%
2	0.90	0.75	0.82	93%
3	1.00	0.67	0.80	98%
4	0.67	0.67	0.67	93%
5	1.00	0.60	0.75	96%
6	0.89	1.00	0.94	98%
7	0.80	0.67	0.73	95%
8	0.50	0.83	0.63	90%
9	0.55	0.75	0.63	88%
10	1.00	0.63	0.77	95%



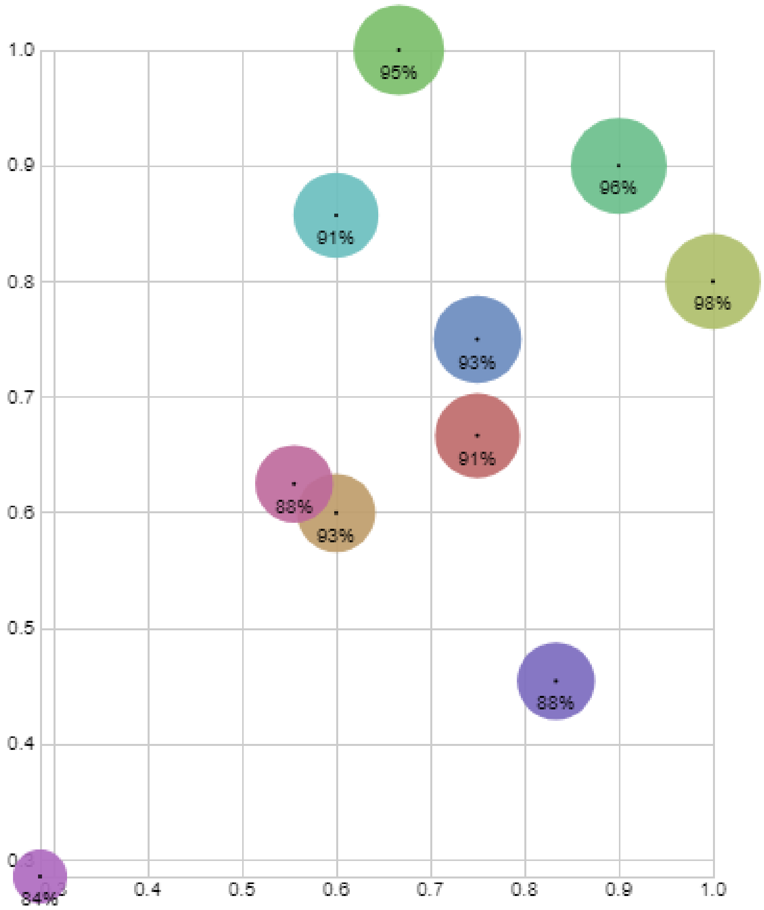
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 5

USER 5 DECISION TREE



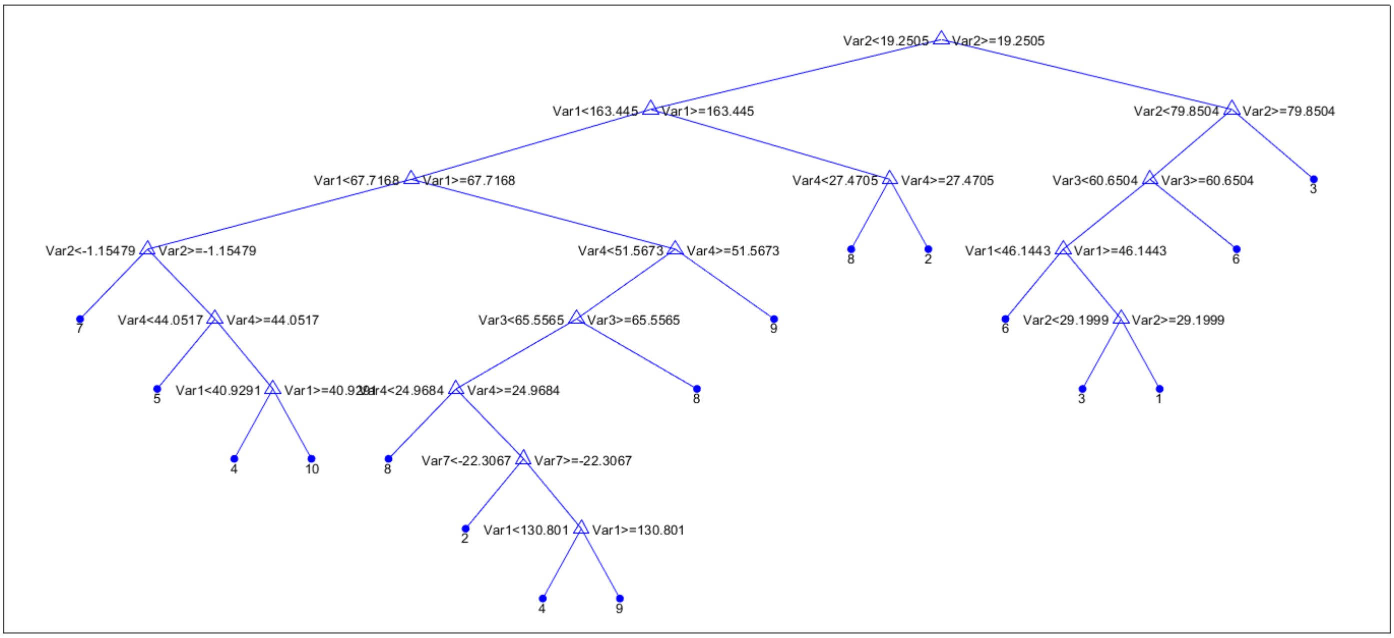
Class	Precision	Recall	F1	Accuracy
1	0.75	0.67	0.71	91%
2	0.60	0.60	0.60	93%
3	1.00	0.80	0.89	98%
4	0.67	1.00	0.80	95%
5	0.90	0.90	0.90	96%
6	0.60	0.86	0.71	91%
7	0.75	0.75	0.75	93%
8	0.83	0.45	0.59	88%
9	0.29	0.29	0.29	84%
10	0.56	0.63	0.59	88%



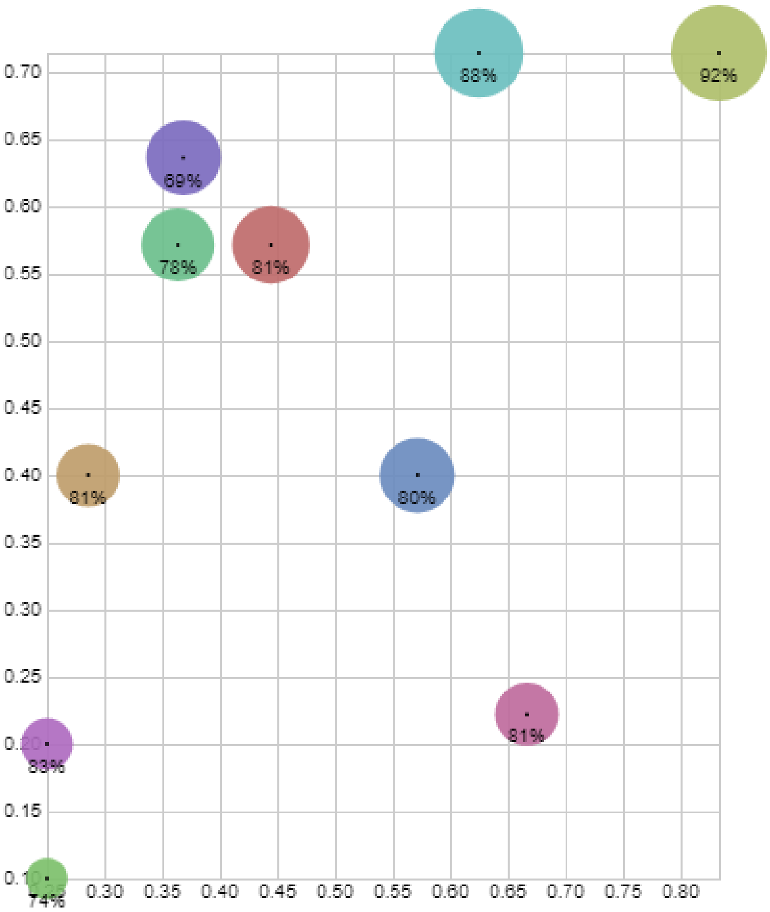
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 11

USER 11 DECISION TREE



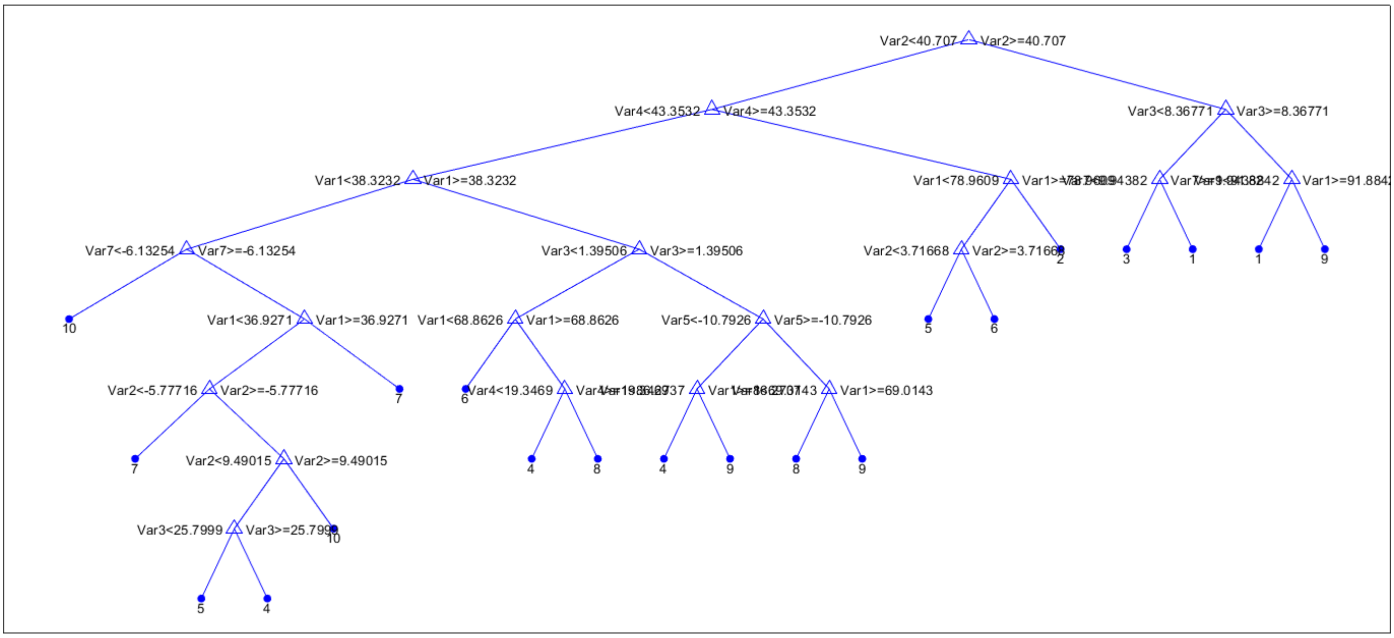
Class	Precision	Recall	F1	Accuracy
1	0.44	0.57	0.50	81%
2	0.29	0.40	0.33	81%
3	0.83	0.71	0.77	92%
4	0.25	0.10	0.14	74%
5	0.36	0.57	0.44	78%
6	0.63	0.71	0.67	88%
7	0.57	0.40	0.47	80%
8	0.37	0.64	0.47	69%
9	0.25	0.20	0.22	83%
10	0.67	0.22	0.33	81%



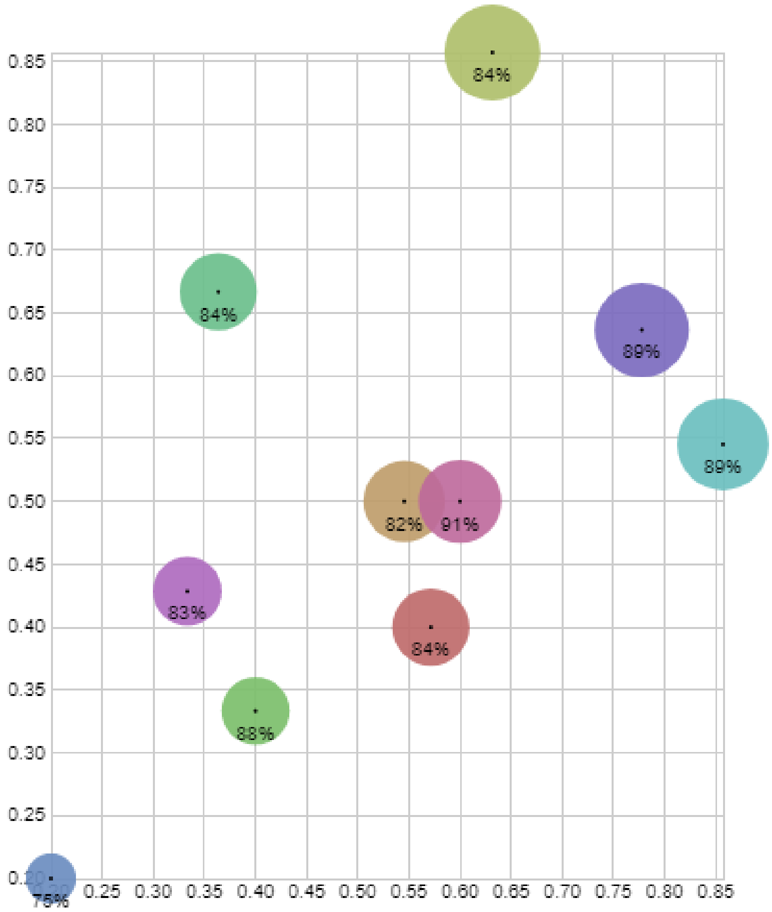
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

<u>User 16</u>

USER 16 DECISION TREE



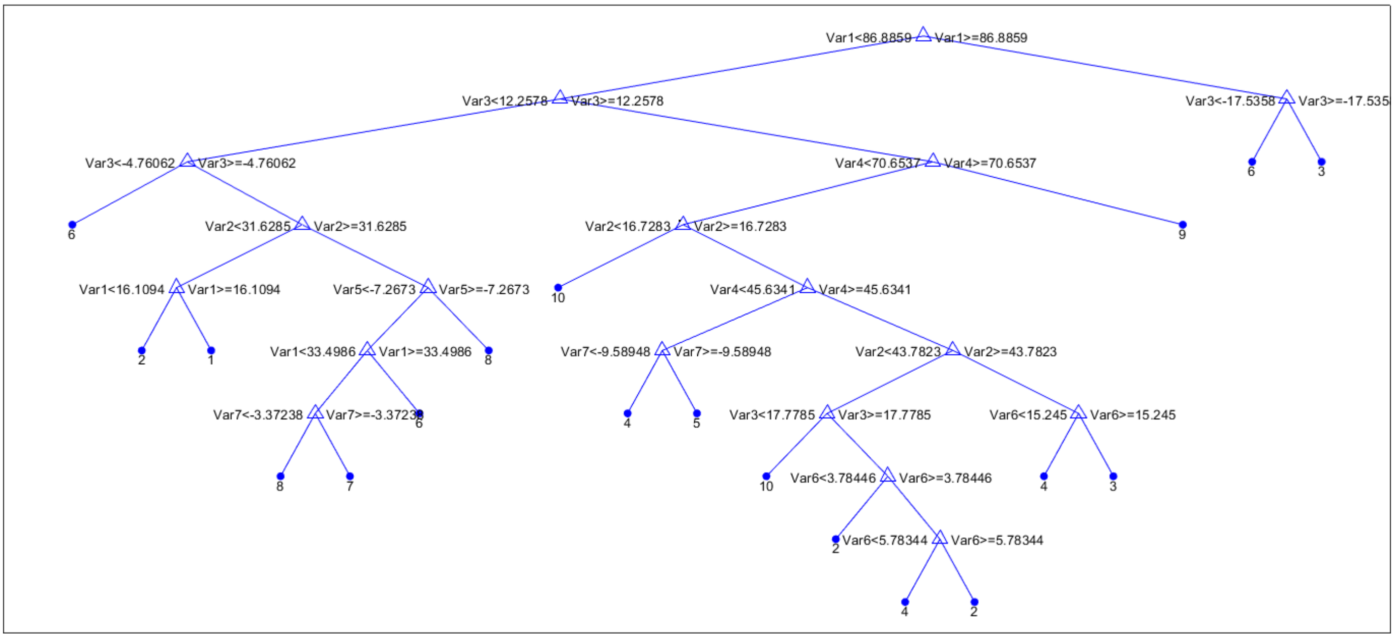
Class	Precision	Recall	F1	Accuracy
1	0.57	0.40	0.47	84%
2	0.55	0.50	0.52	82%
3	0.63	0.86	0.73	84%
4	0.40	0.33	0.36	88%
5	0.36	0.67	0.47	84%
6	0.86	0.55	0.67	89%
7	0.20	0.20	0.20	75%
8	0.78	0.64	0.70	89%
9	0.33	0.43	0.38	83%
10	0.60	0.50	0.55	91%



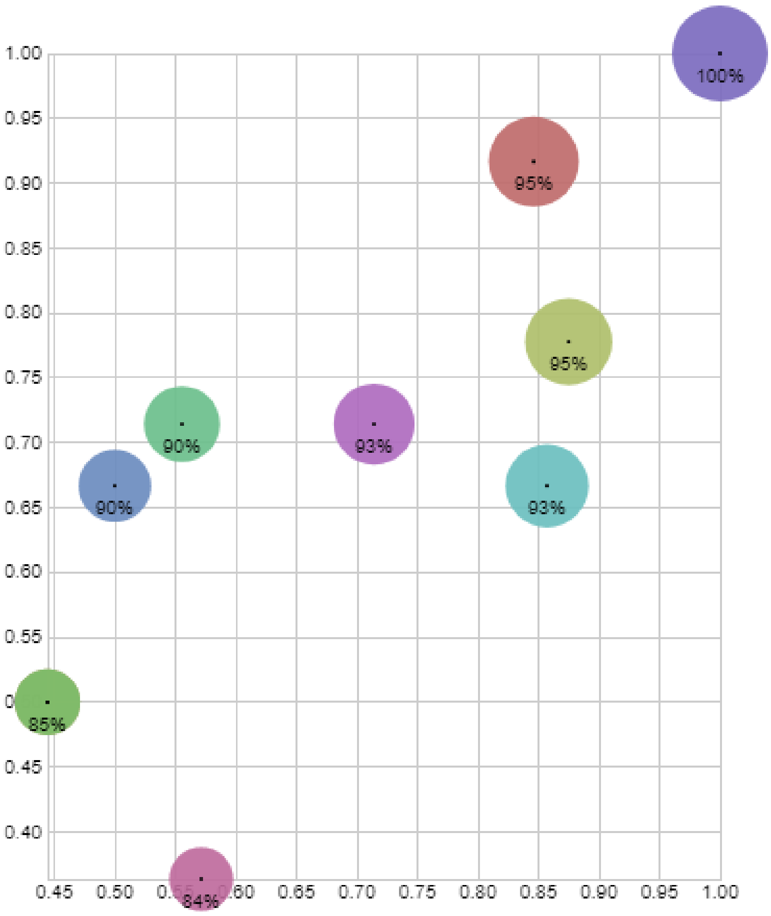
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 19

USER 19 DECISION TREE



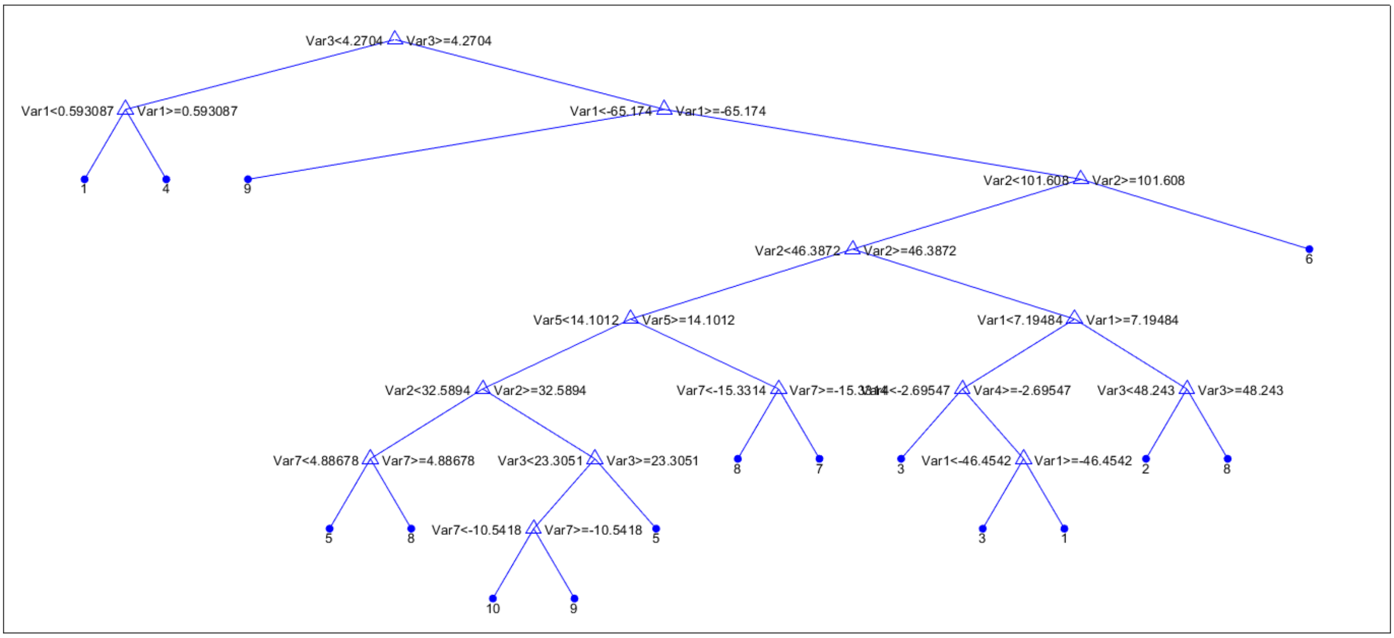
Class	Precision	Recall	F1	Accuracy
1	0.85	0.92	0.88	95%
2	0.44	0.50	0.47	85%
3	0.88	0.78	0.82	95%
4	0.44	0.50	0.47	85%
5	0.56	0.71	0.63	90%
6	0.86	0.67	0.75	93%
7	0.50	0.67	0.57	90%
8	1.00	1.00	1.00	100%
9	0.71	0.71	0.71	93%
10	0.57	0.36	0.44	84%



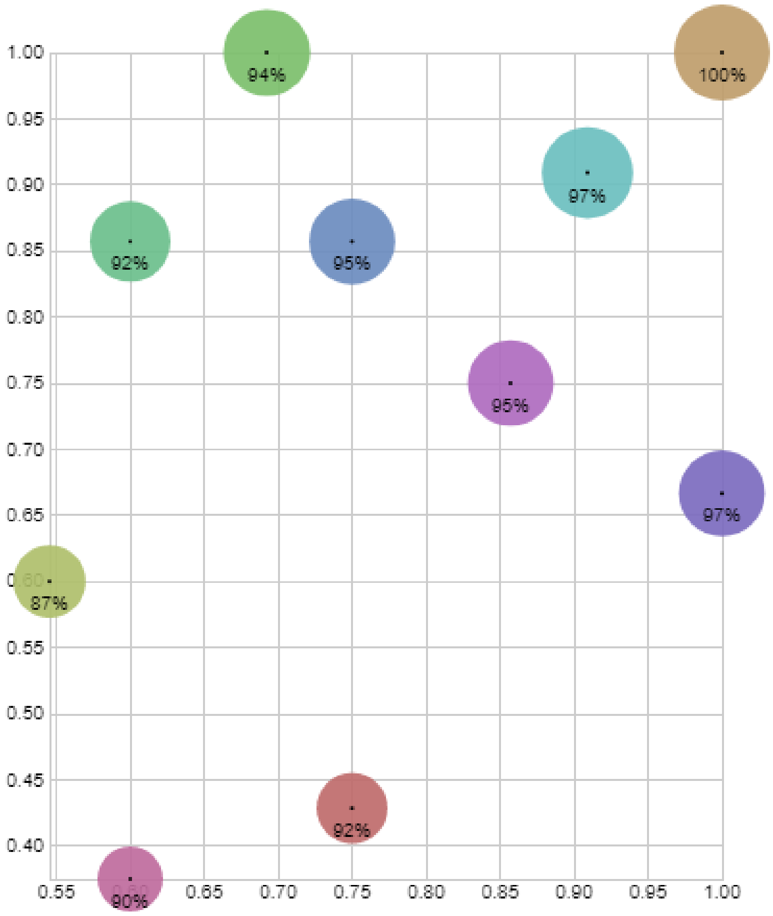
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 20

USER 20 DECISION TREE



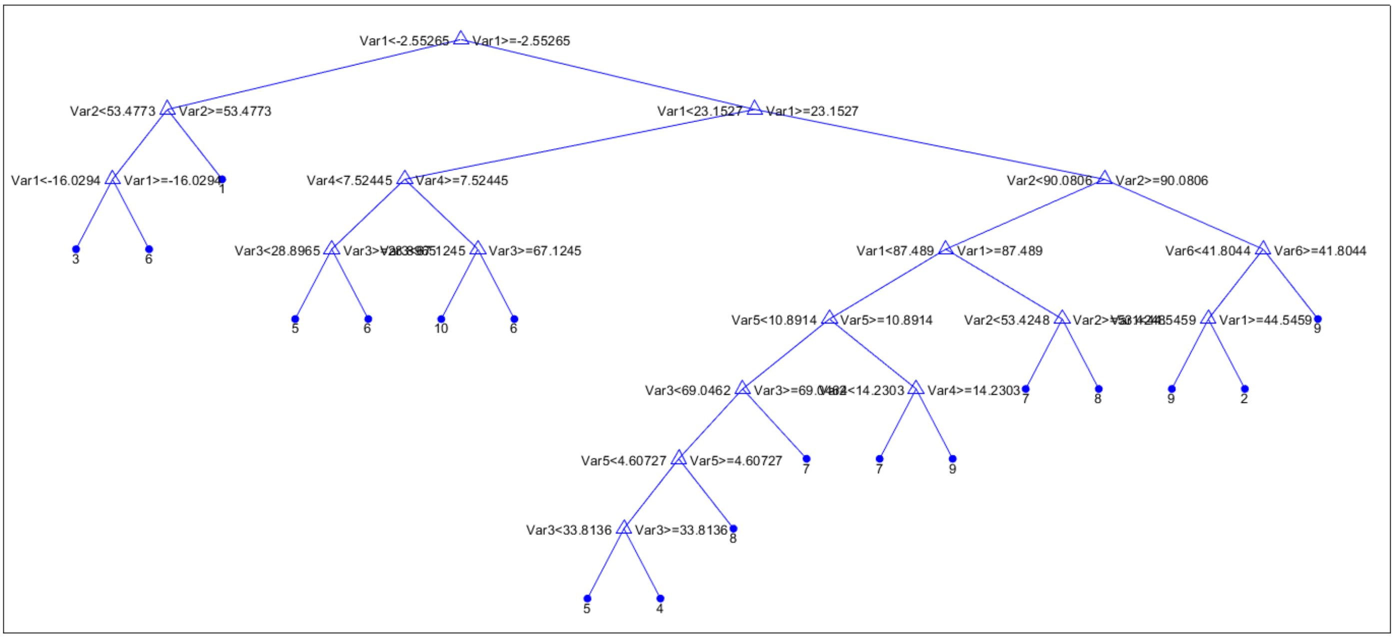
Class	Precision	Recall	F1	Accuracy
1	0.75	0.43	0.55	92%
2	1.00	1.00	1.00	100%
3	0.55	0.60	0.57	87%
4	0.69	1.00	0.82	94%
5	0.60	0.86	0.71	92%
6	0.91	0.91	0.91	97%
7	0.75	0.86	0.80	95%
8	1.00	0.67	0.80	97%
9	0.86	0.75	0.80	95%
10	0.60	0.38	0.46	90%



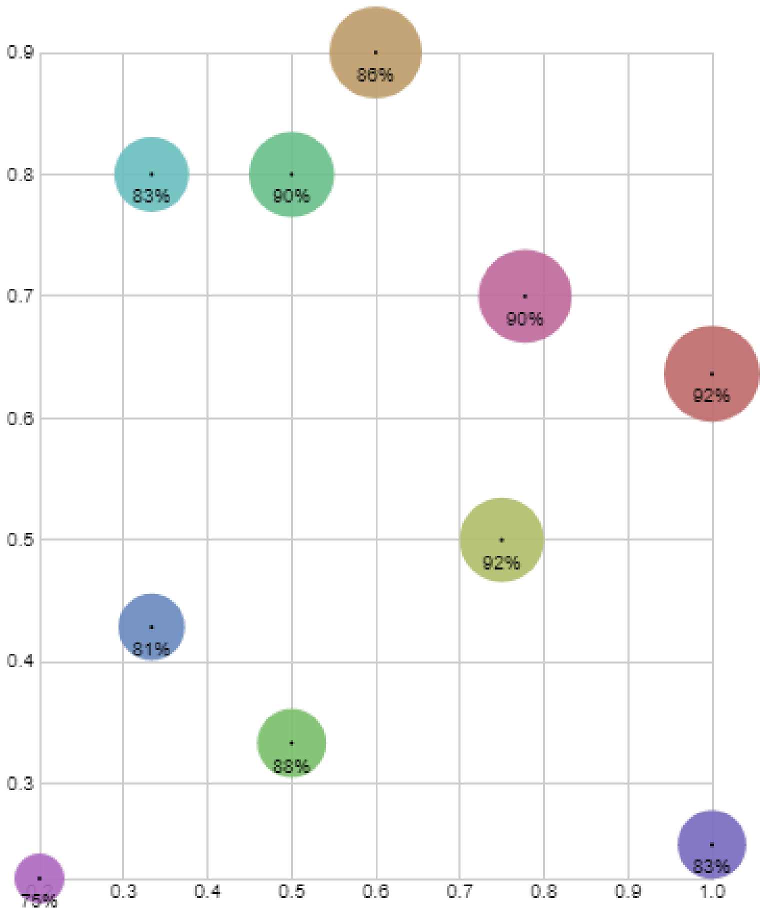
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 22

USER 22 DECISION TREE



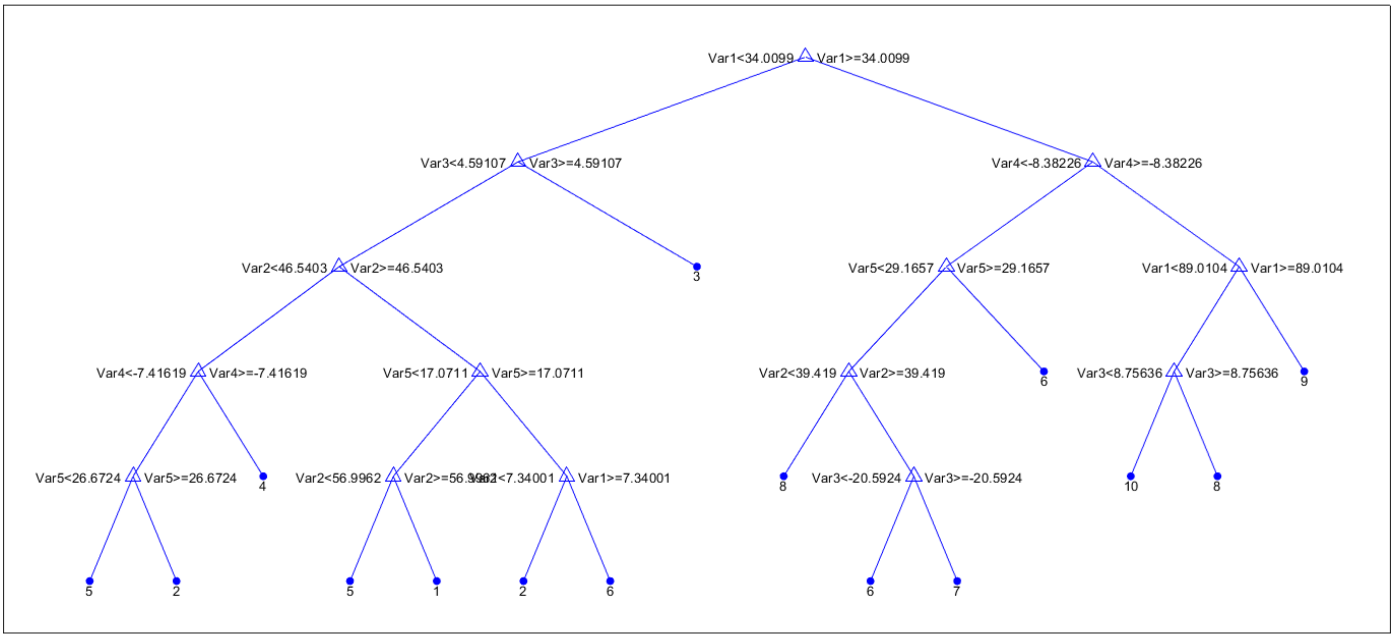
Class	Precision	Recall	F1	Accuracy
1	1.00	0.64	0.78	92%
2	0.60	0.90	0.72	86%
3	0.75	0.50	0.60	92%
4	0.50	0.33	0.40	88%
5	0.50	0.80	0.62	90%
6	0.33	0.80	0.47	83%
7	0.33	0.43	0.38	81%
8	1.00	0.25	0.40	83%
9	0.20	0.22	0.21	75%
10	0.78	0.70	0.74	90%



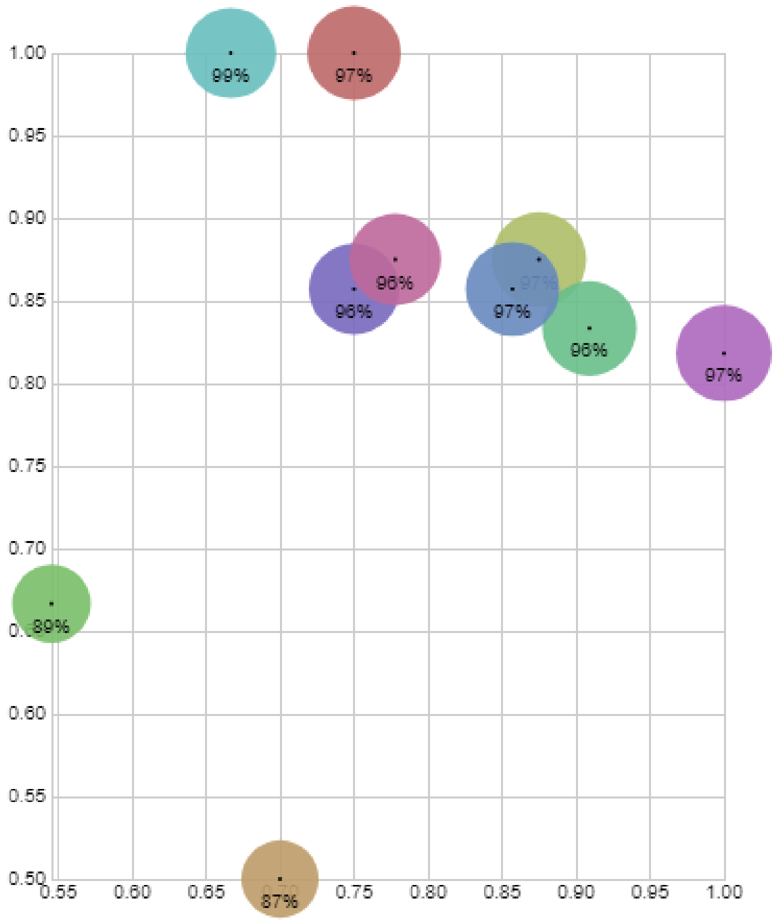
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 24

USER 24 DECISION TREE



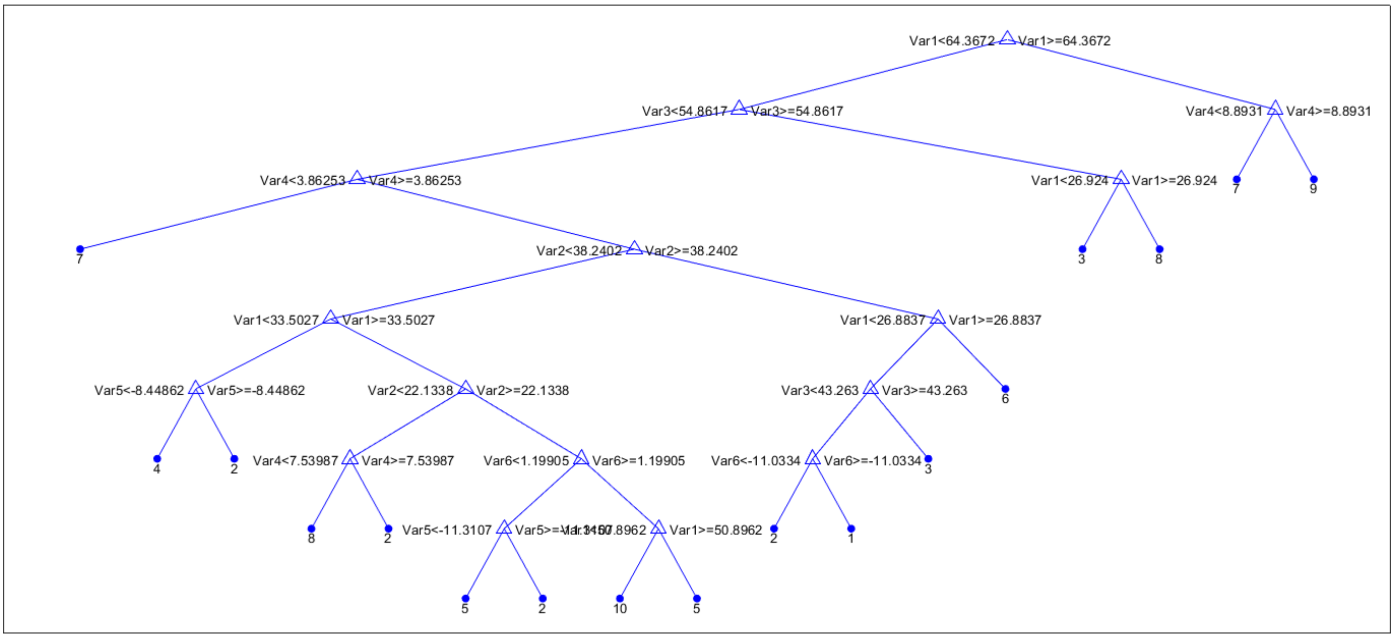
Class	Precision	Recall	F1	Accuracy
1	0.75	1.00	0.86	97%
2	0.70	0.50	0.58	87%
3	0.88	0.88	0.88	97%
4	0.55	0.67	0.60	89%
5	0.91	0.83	0.87	96%
6	0.67	1.00	0.80	99%
7	0.86	0.86	0.86	97%
8	0.75	0.86	0.80	96%
9	1.00	0.82	0.90	97%
10	0.78	0.88	0.82	96%



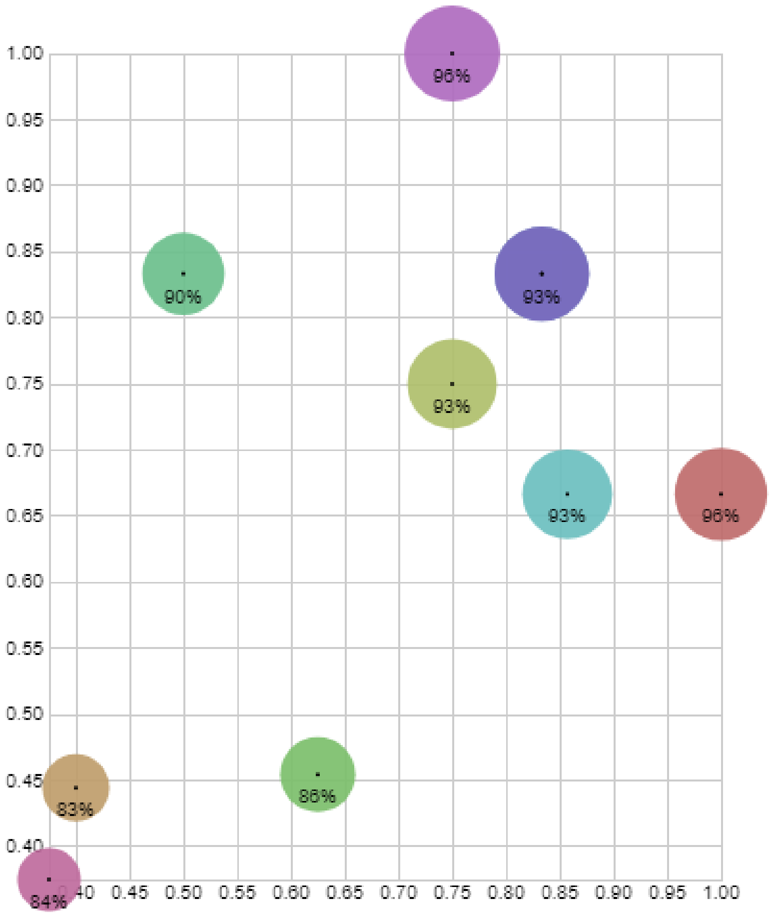
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 28

USER 28 DECISION TREE



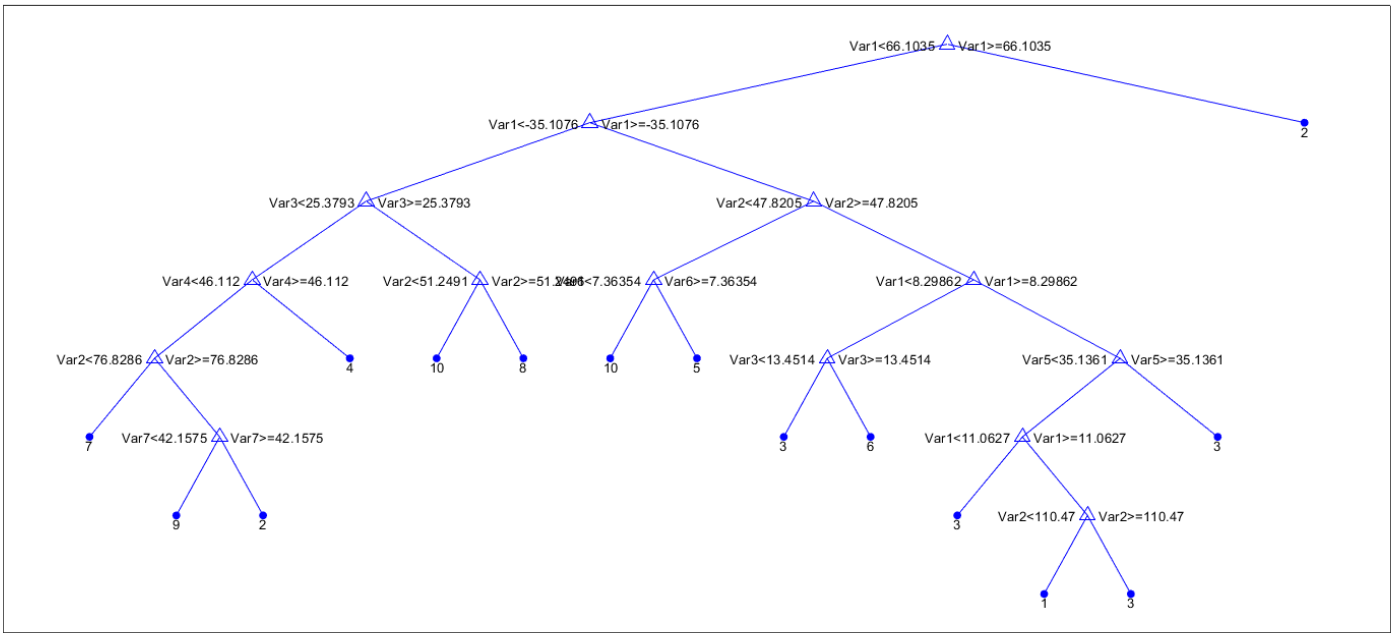
Class	Precision	Recall	F1	Accuracy
1	1.00	0.67	0.80	96%
2	0.40	0.44	0.42	83%
3	0.75	0.75	0.75	93%
4	0.63	0.45	0.53	86%
5	0.50	0.83	0.63	90%
6	0.86	0.67	0.75	93%
7	0.83	0.83	0.83	96%
8	0.83	0.83	0.83	93%
9	0.75	1.00	0.86	96%
10	0.38	0.38	0.38	84%



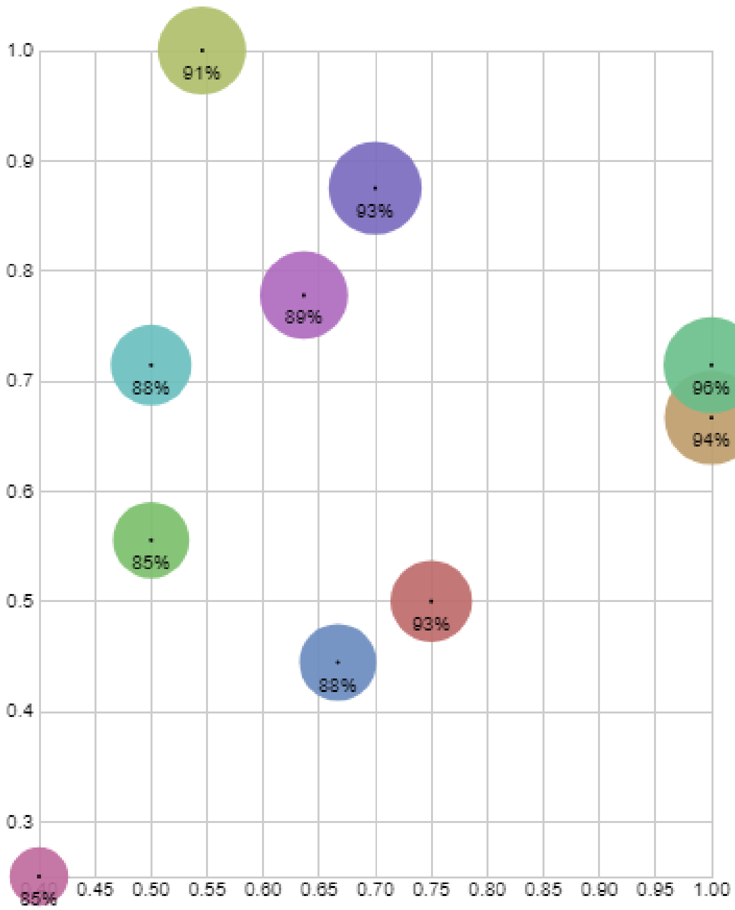
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 36

USER 36 DECISION TREE



Class	Precision	Recall	F1	Accuracy
1	0.75	0.50	0.60	93%
2	1.00	0.67	0.80	94%
3	0.55	1.00	0.71	91%
4	0.50	0.56	0.53	85%
5	1.00	0.71	0.83	96%
6	0.50	0.71	0.59	88%
7	0.67	0.44	0.53	88%
8	0.70	0.88	0.78	93%
9	0.64	0.78	0.70	89%
10	0.40	0.25	0.31	85%

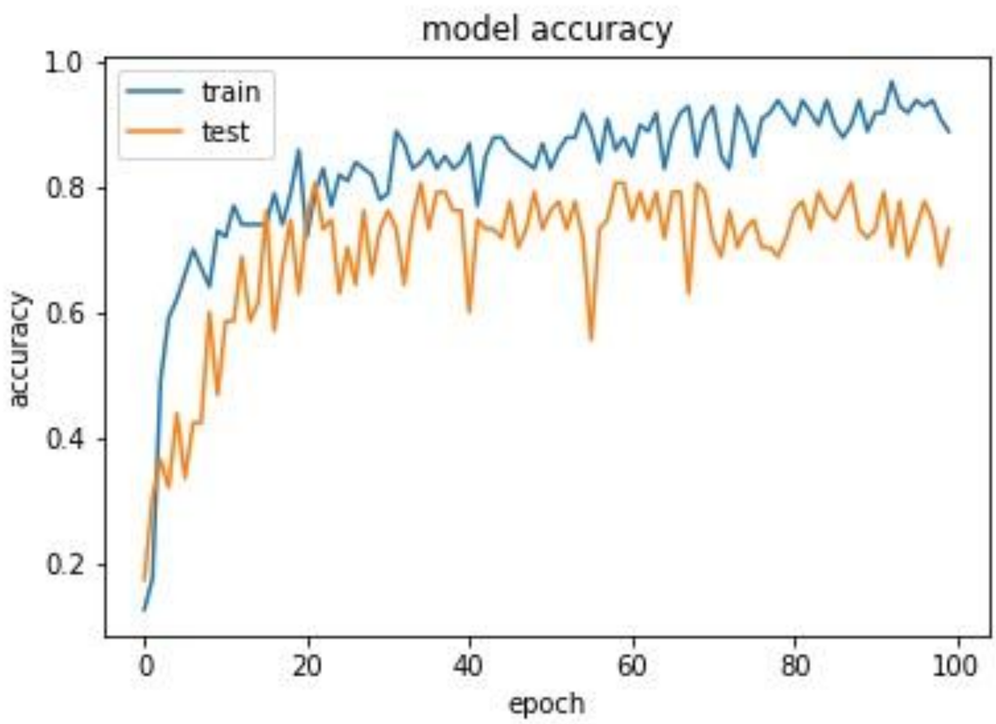


X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

5.4.2 Neural Network

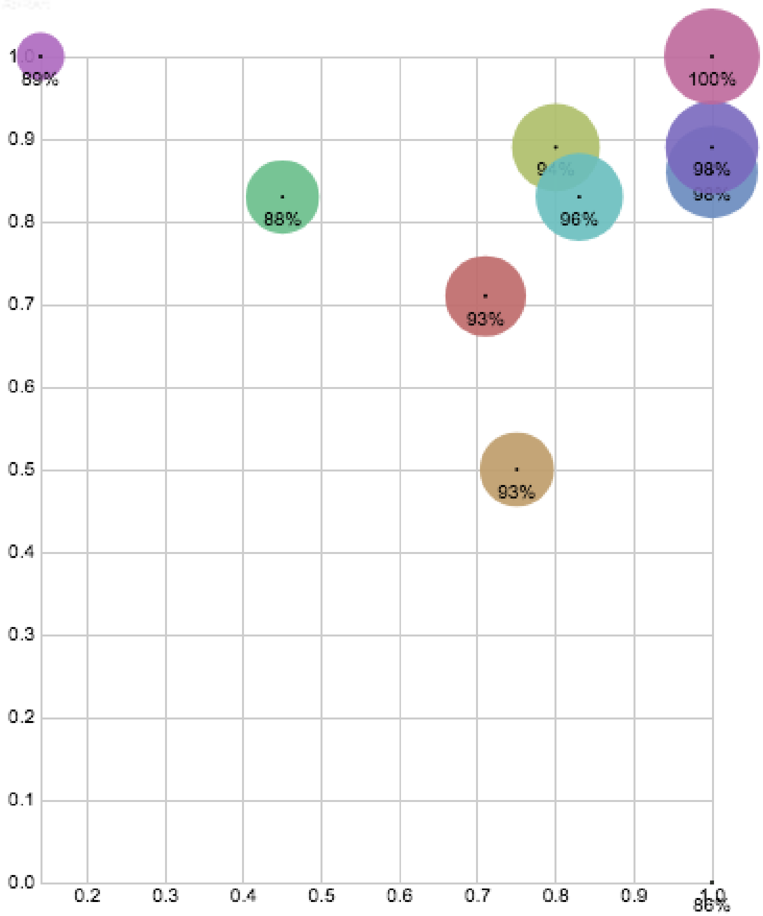
- Neural network is the computing system that is vaguely inspired by animal brains [1]. Recent research in data mining and machine learning are focusing on deep layers neural network because of its ability to perform complicated tasks on different data set domain.
- The fully connected network is one in which every neuron in one layer in connected to all the neuron in next layer. One of the reasons why neural networks perform very well compared to baseline is that every layer is tasked to extract different feature. An early layer of the network is known for recognizing simple features and layer layers of the network are known for recognizing complex feature. the neuron connections are initialized with weights and data is passed through the network which is called forward propagation. The loss function is defined as per objective and backward propagation gradient derivatives are calculated to update the weights. the idea is to reduce the loss function almost to zero. So that the network is pretty well generalized for the specified task.
- The weights are updated using gradient derivates and data is passed through the network using updated weights. different activation functions like 'sigmoid', 'relu', or 'tanh'. relu actioned is shown to be performing well by avoiding gradient values to perform in boundary cases. to use this network for multi-class clasiffier, we can use softmax layer on the output layer.
- In this project, we used the neural network as a multi-class classification for gesture recognition. We built the user dependent classification of 10 gestures.
- The network has two hidden layers with 512 neurons in each. Sigmoid activation is used for hidden layers. Output layer has softmax activations for 10 classes. The network is trained for 100 epochs.
- Keras is tensor framework used to build and test and model in a rapid manner. We used Keras to build this neural network. RMS propagation is used to improve performance. Categorical cross-entropy is used as loss function. Network in trained for 100 epochs for each user and metrics like accuracy, precision, recall, and f1 are observed.
- We generated a Precision versus Recall graph for each user. The X-Axis will represent Precision, Y-Axis will represent Recall, size of the node will vary according to F1 value, label on each node will denote accuracy % value and the colors will represent the classes.
- The results are described in the next few pages.

USER 1

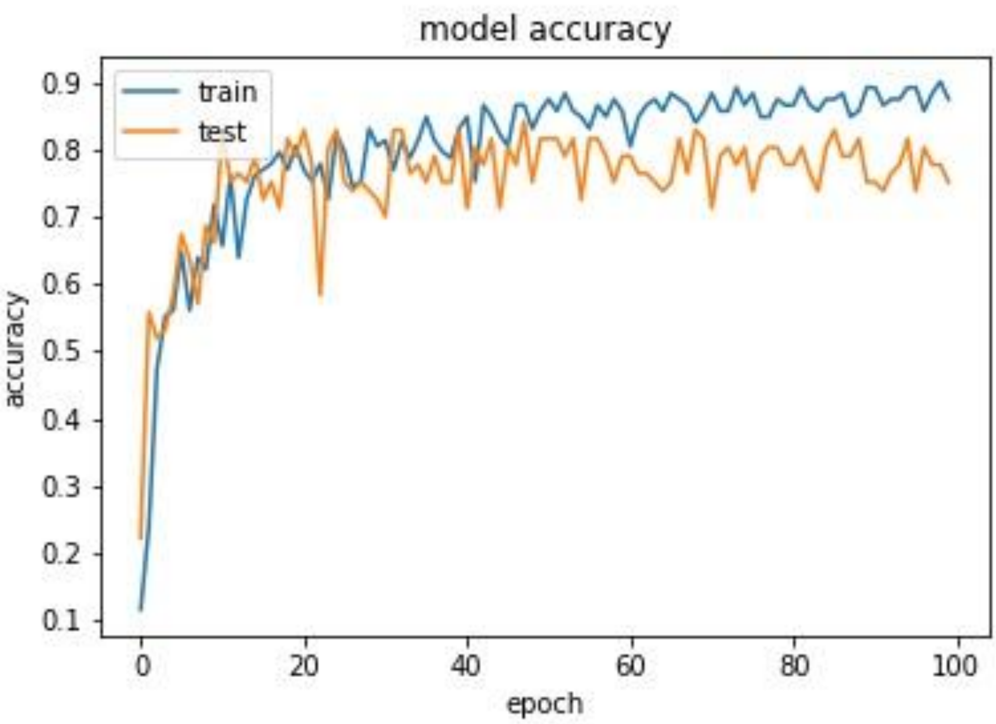


Class	Precision	Recall	F1	Accuracy
1	0.71	0.71	0.71	93%
2	0.75	0.50	0.60	93%
3	0.80	0.89	0.84	94%
4	1.00	0.00	0.00	86%
5	0.45	0.83	0.59	88%
6	0.83	0.83	0.83	96%
7	1.00	0.86	0.92	98%
8	1.00	0.89	0.94	98%
9	0.14	1.00	0.25	89%
10	1.00	1.00	1.00	100%

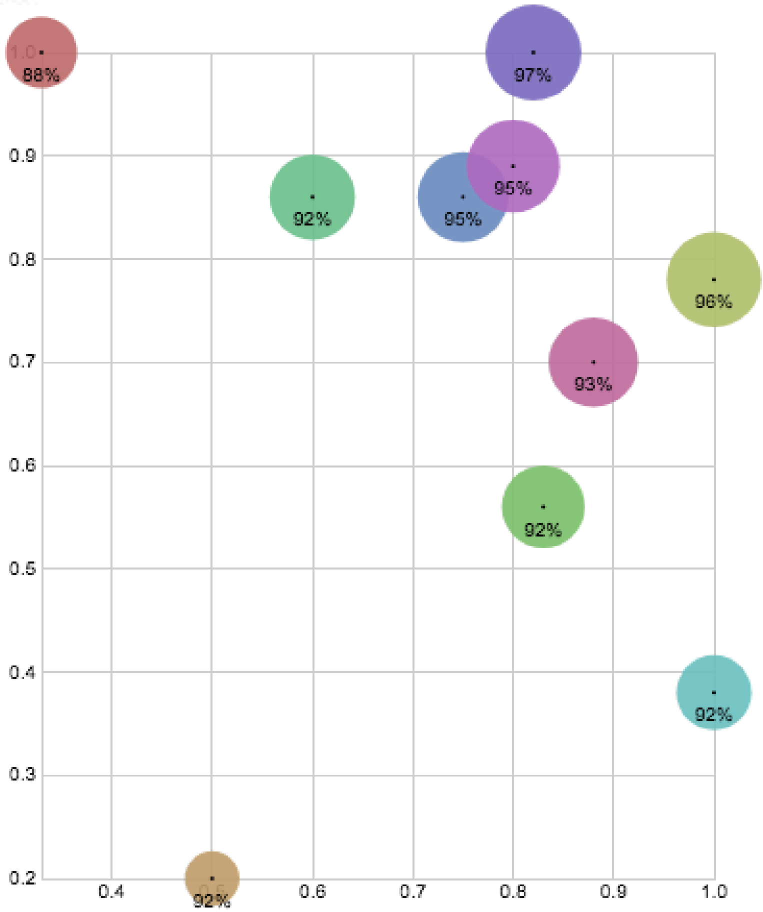
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



USER 5

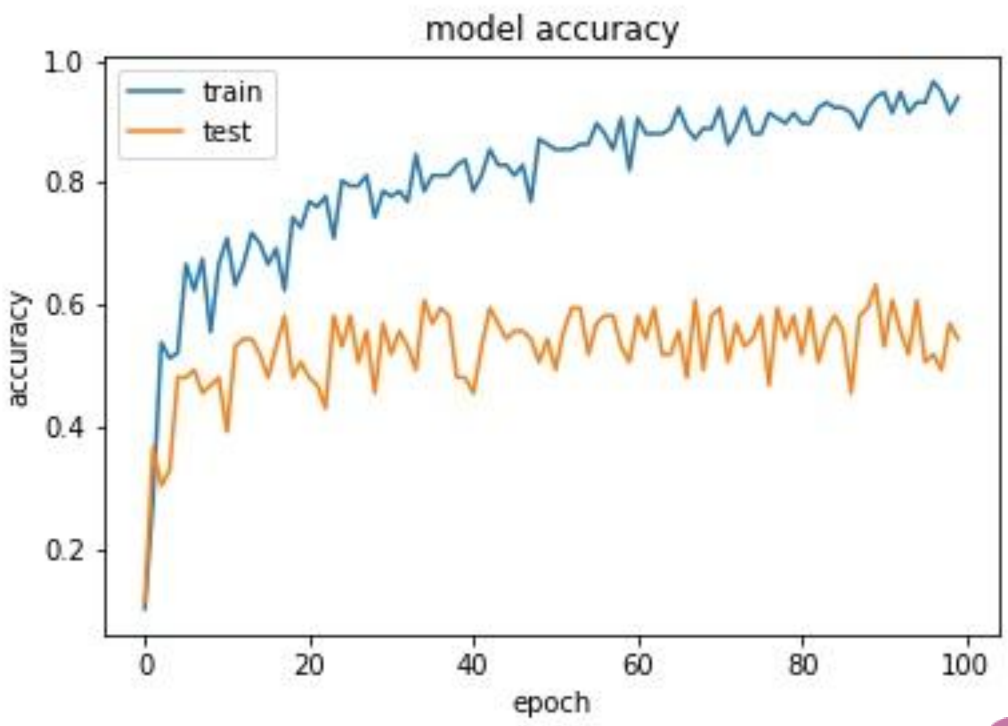


Class	Precision	Recall	F1	Accuracy
1	0.33	1.00	0.50	88%
2	0.50	0.20	0.29	92%
3	1.00	0.78	0.88	96%
4	0.83	0.56	0.67	92%
5	0.60	0.86	0.71	92%
6	1.00	0.38	0.55	92%
7	0.75	0.86	0.80	95%
8	0.82	1.00	0.90	97%
9	0.80	0.89	0.84	95%
10	0.88	0.70	0.78	93%

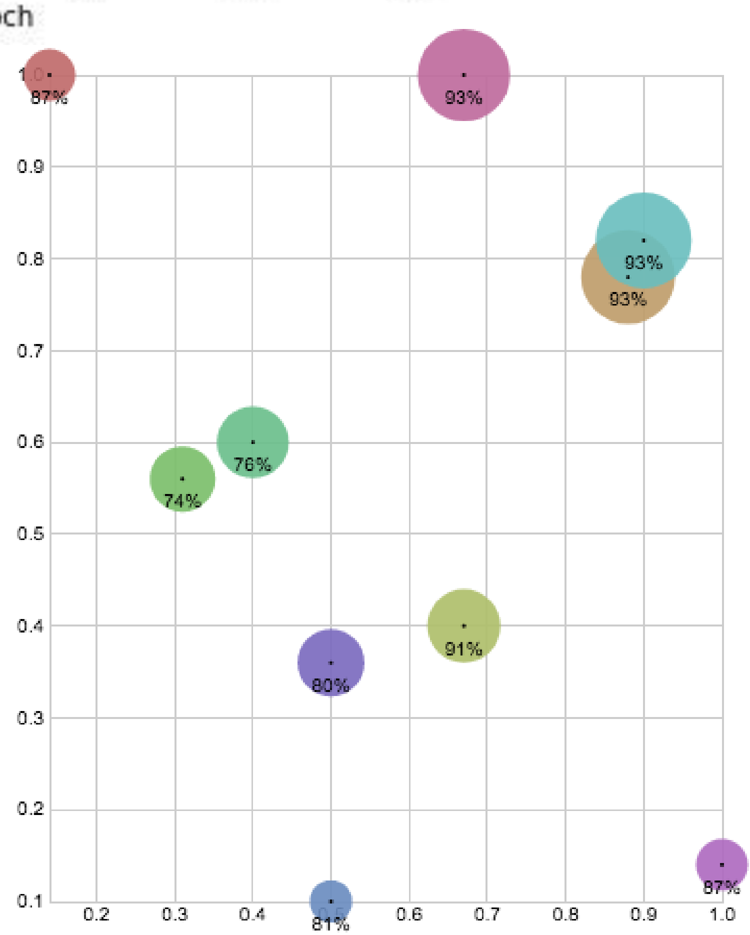


X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

USER 11

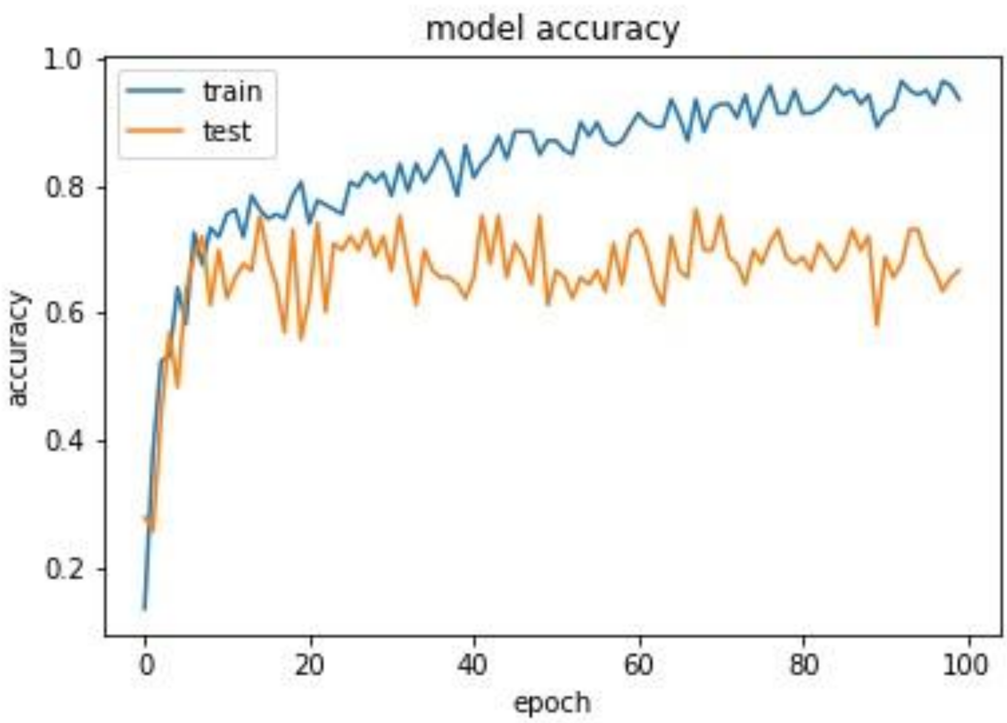


Class	Precision	Recall	F1	Accuracy
1	0.14	1.00	0.25	87%
2	0.88	0.78	0.82	93%
3	0.67	0.40	0.50	91%
4	0.31	0.56	0.40	74%
5	0.40	0.60	0.48	76%
6	0.90	0.82	0.86	93%
7	0.50	0.10	0.17	81%
8	0.50	0.36	0.42	80%
9	1.00	0.14	0.25	87%
10	0.67	1.00	0.80	93%

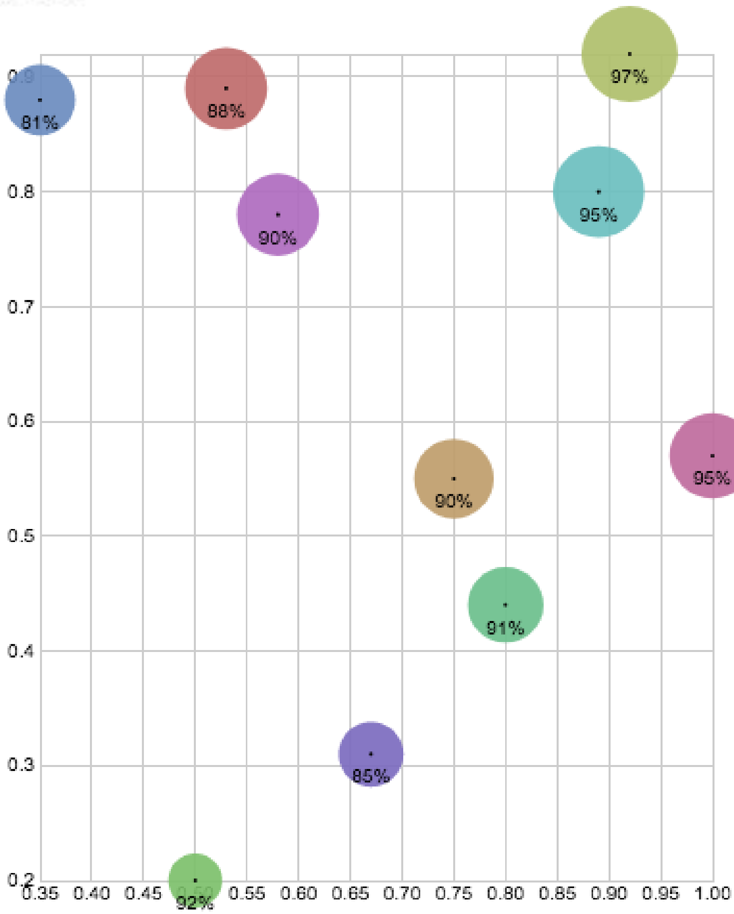


X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

USER 16

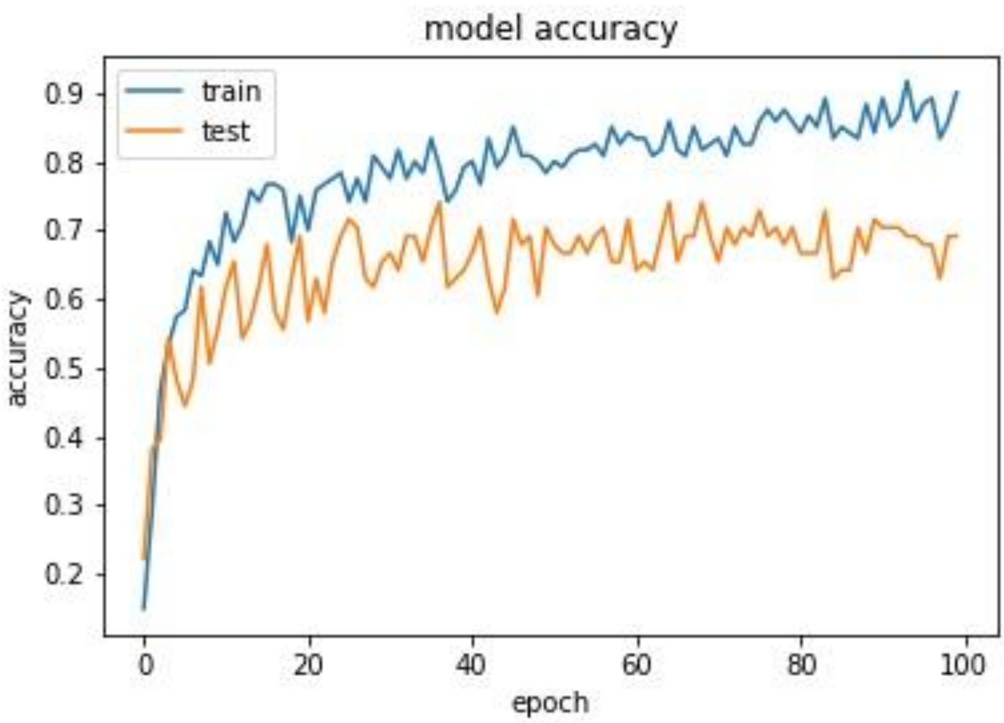


Class	Precision	Recall	F1	Accuracy
1	0.53	0.89	0.67	88%
2	0.75	0.55	0.63	90%
3	0.92	0.92	0.92	97%
4	0.50	0.20	0.29	92%
5	0.80	0.44	0.57	91%
6	0.89	0.80	0.84	95%
7	0.35	0.88	0.50	81%
8	0.67	0.31	0.42	85%
9	0.58	0.78	0.67	90%
10	1.00	0.57	0.73	95%

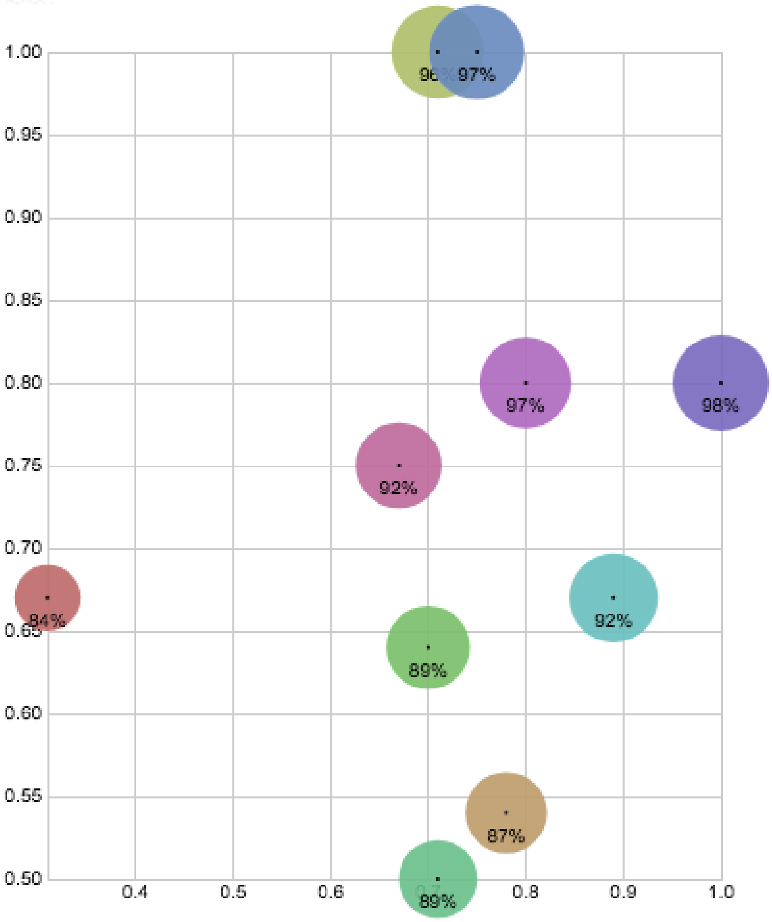


X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

USER 19

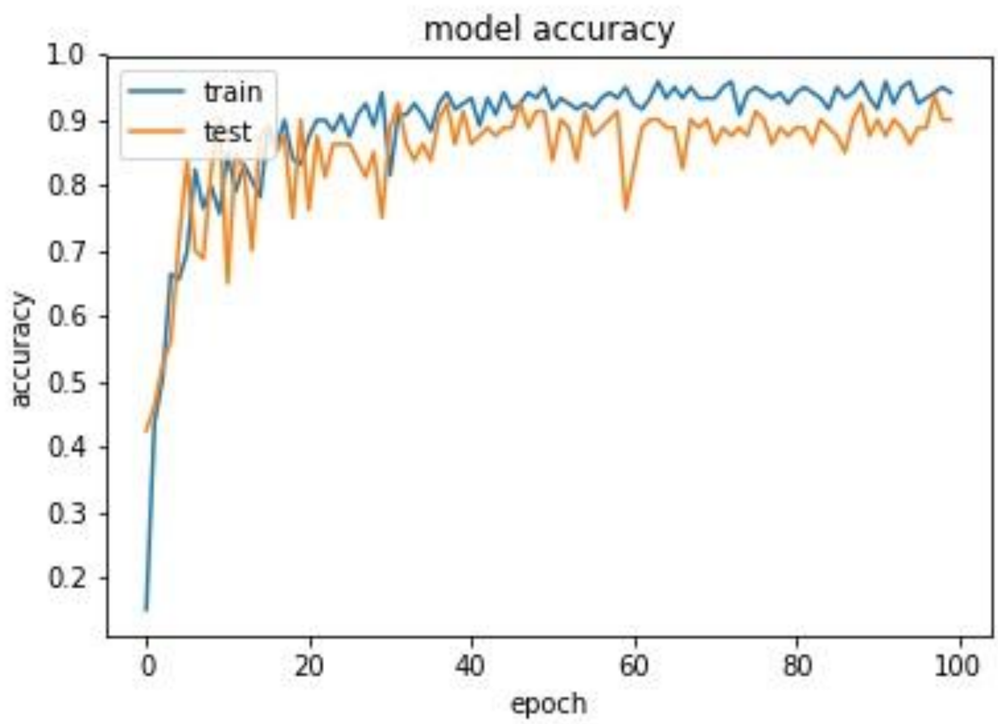


Class	Precision	Recall	F1	Accuracy
1	0.31	0.67	0.42	84%
2	0.78	0.54	0.64	87%
3	0.71	1.00	0.83	96%
4	0.70	0.64	0.67	89%
5	0.71	0.50	0.59	89%
6	0.89	0.67	0.76	92%
7	0.75	1.00	0.86	97%
8	1.00	0.80	0.89	98%
9	0.80	0.80	0.80	97%
10	0.67	0.75	0.71	92%



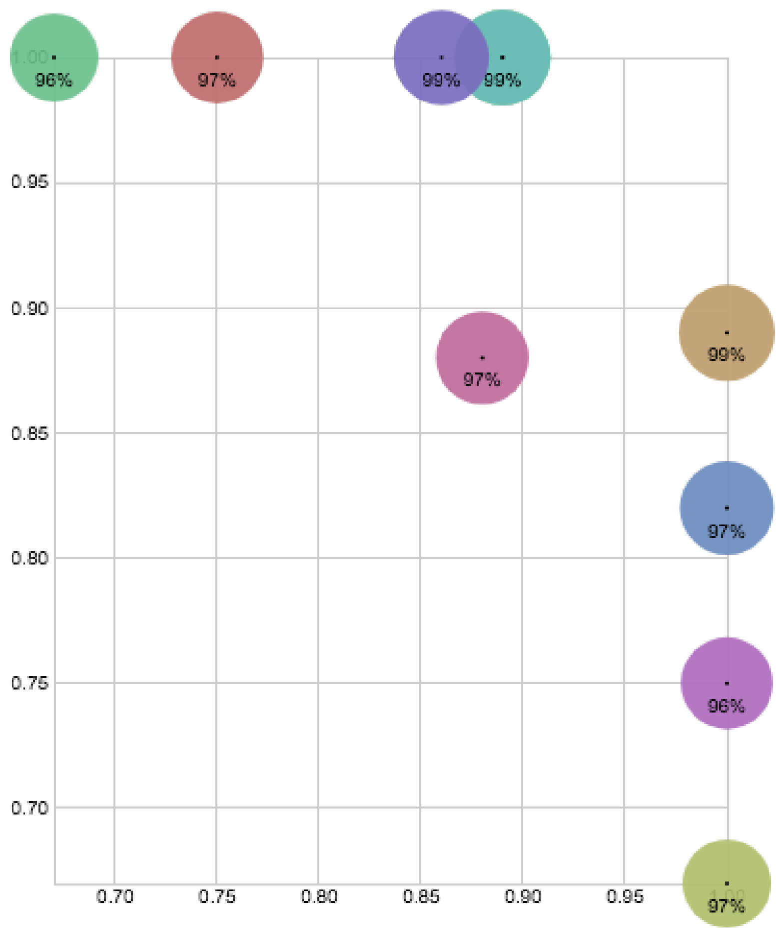
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

USER 20

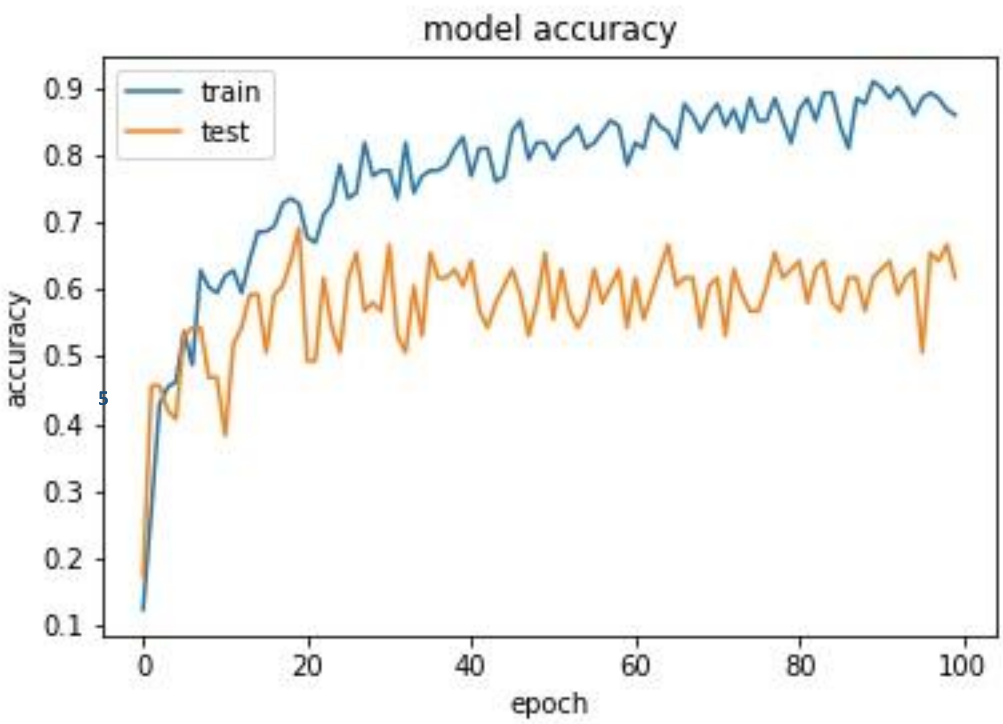


Class	Precision	Recall	F1	Accuracy
1	0.75	1.00	0.86	97%
2	1.00	0.89	0.94	99%
3	1.00	0.67	0.80	97%
4	0.89	1.00	0.94	99%
5	0.67	1.00	0.80	96%
6	0.89	1.00	0.94	99%
7	1.00	0.82	0.90	97%
8	0.86	1.00	0.92	99%
9	1.00	0.75	0.86	96%
10	0.88	0.88	0.88	97%

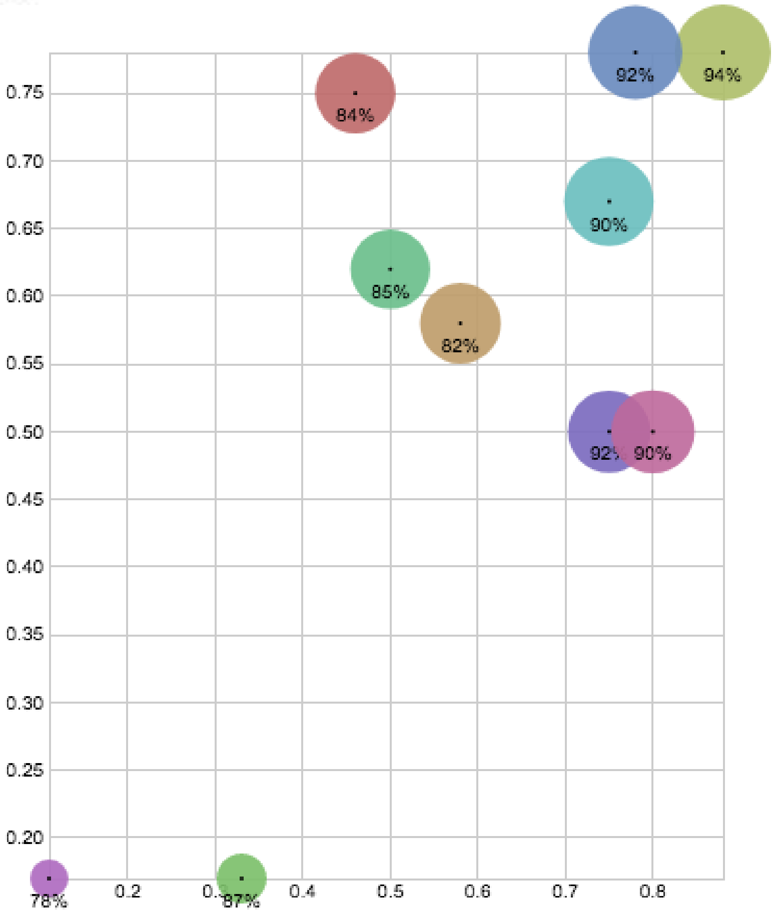
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



USER 22

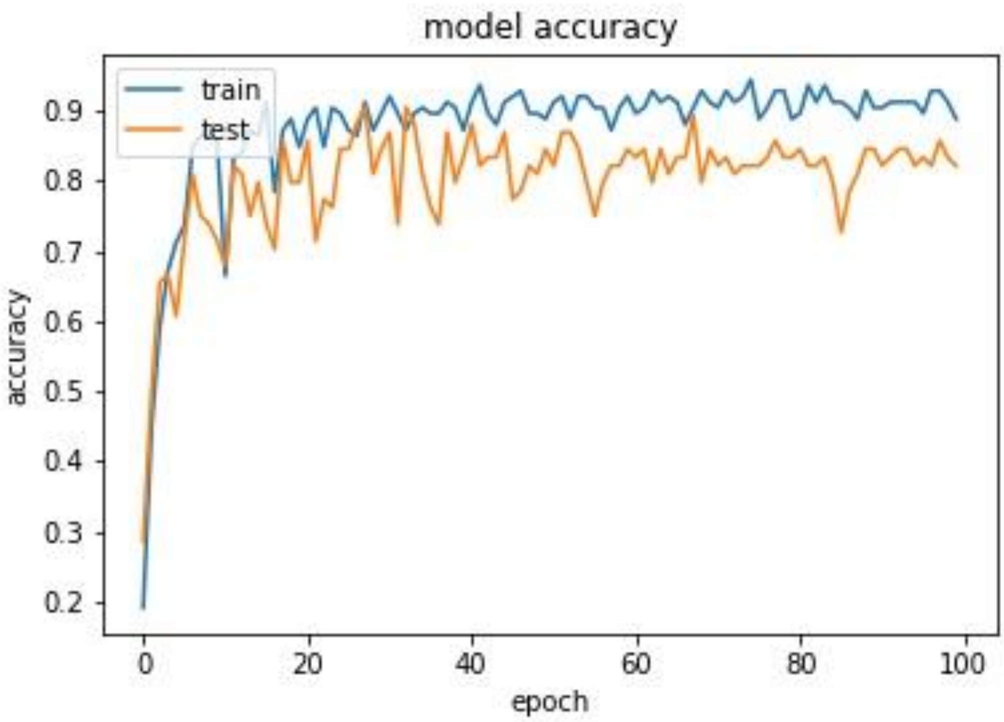


Class	Precision	Recall	F1	Accuracy
1	0.46	0.75	0.57	84%
2	0.58	0.58	0.58	82%
3	0.88	0.78	0.82	94%
4	0.33	0.17	0.22	87%
5	0.50	0.62	0.56	85%
6	0.75	0.67	0.71	90%
7	0.78	0.78	0.78	92%
8	0.75	0.50	0.60	92%
9	0.11	0.17	0.13	78%
10	0.80	0.50	0.62	90%



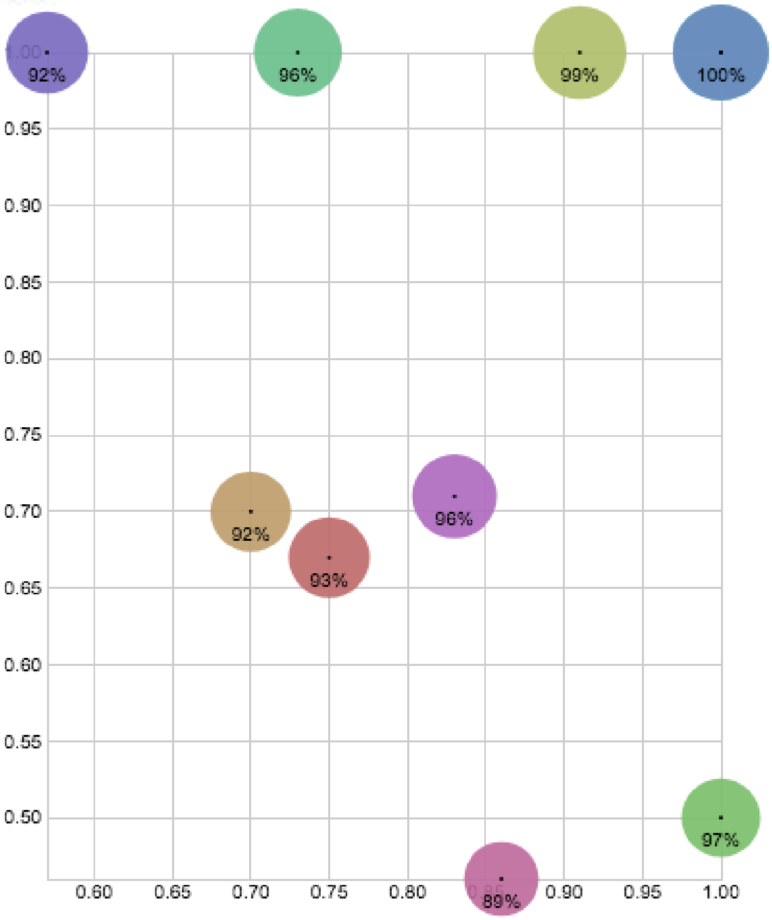
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

USER 24

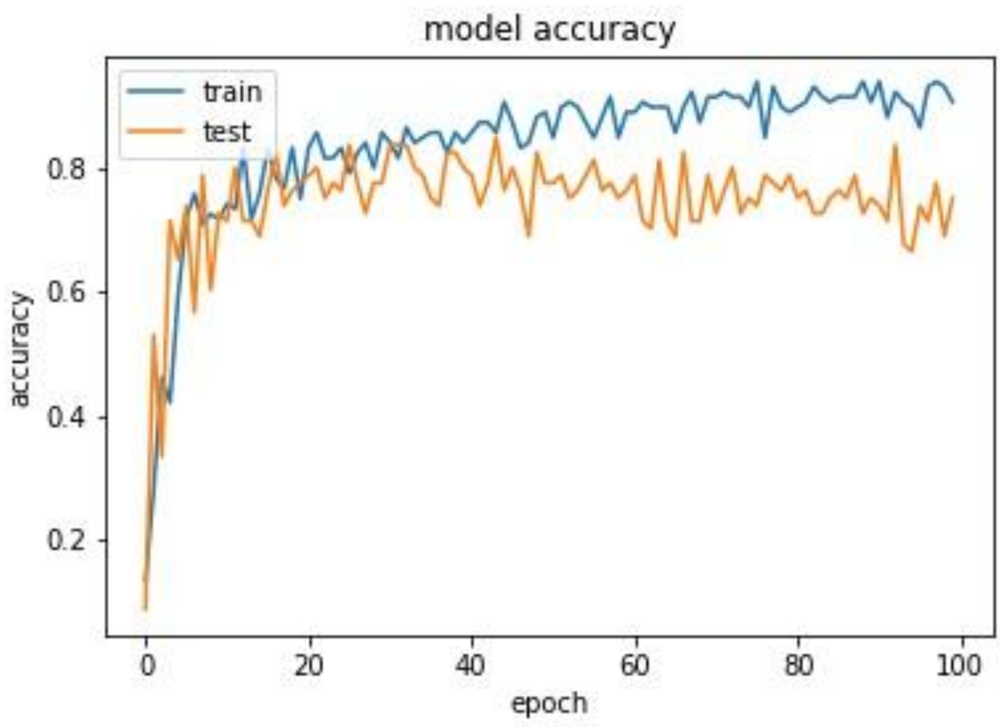


Class	Precision	Recall	F1	Accuracy
1	0.75	0.67	0.71	93%
2	0.70	0.70	0.70	92%
3	0.91	1.00	0.95	99%
4	1.00	0.50	0.67	97%
5	0.73	1.00	0.84	96%
6	1.00	1.00	1.00	100%
7	1.00	1.00	1.00	100%
8	0.57	1.00	0.73	92%
9	0.83	0.71	0.77	96%
10	0.86	0.46	0.60	89%

X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



USER 28



Class	Precision	Recall	F1	Accuracy
1	0.83	0.83	0.83	97%
2	0.62	0.80	0.70	90%
3	0.86	0.86	0.86	97%
4	1.00	0.50	0.67	95%
5	0.71	0.45	0.56	88%
6	0.88	0.88	0.88	97%
7	0.89	0.89	0.89	97%
8	1.00	0.83	0.91	97%
9	0.43	0.86	0.57	87%
10	0.75	0.60	0.67	95%

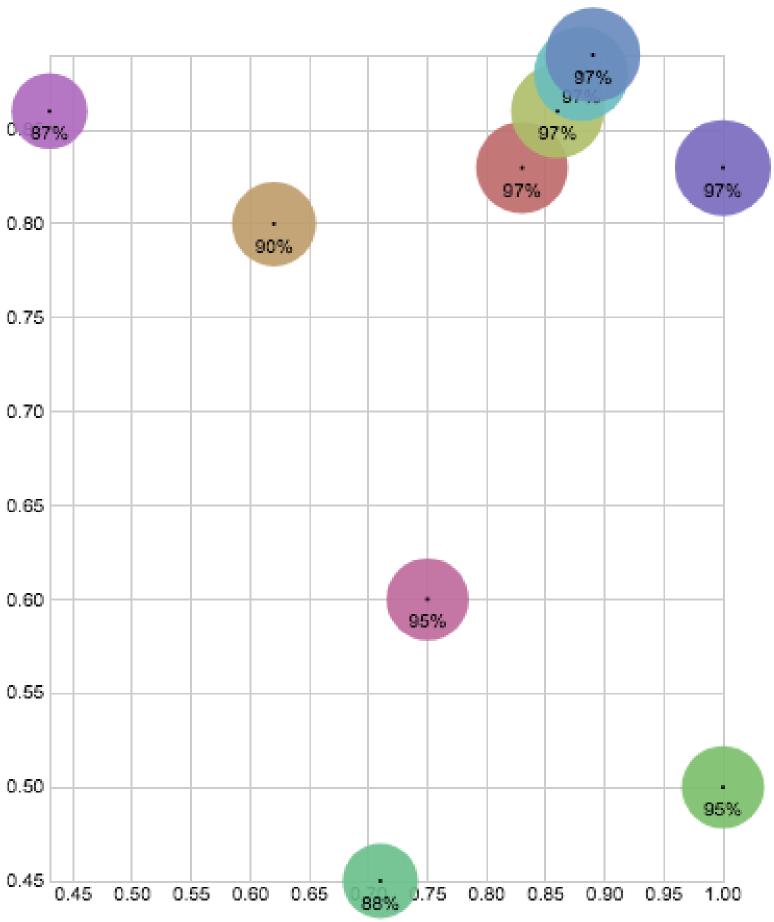
X Axis : Precision Values

Y Axis : Recall Values

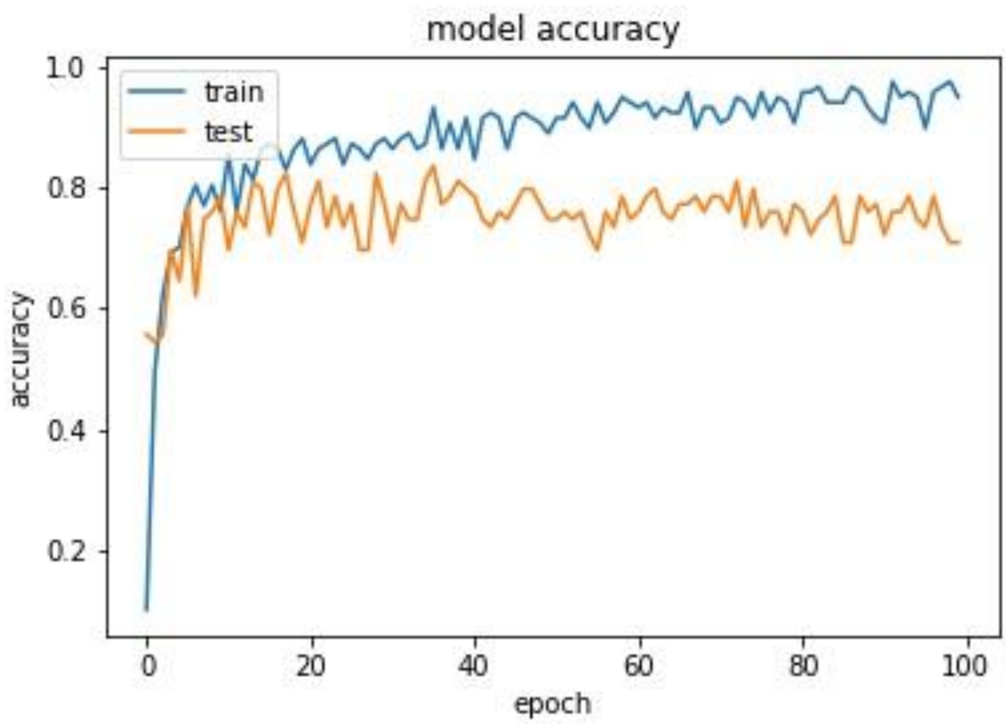
Size of points : F1 Values

Labels : Accuracy Values

Colors : Classes

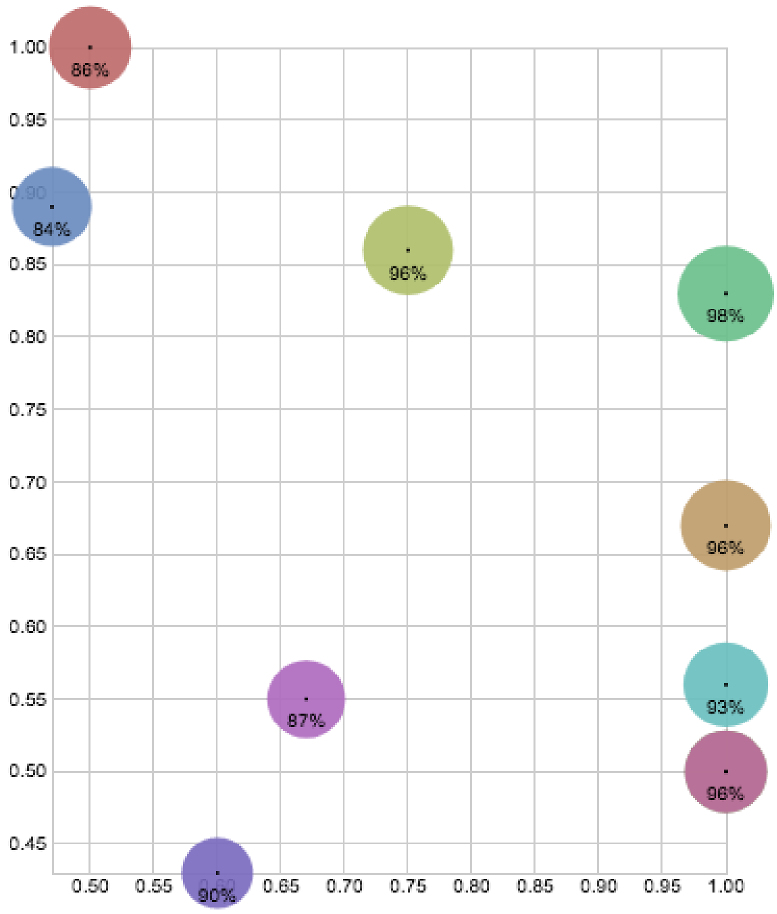


USER 36



Class	Precision	Recall	F1	Accuracy
1	0.50	1.00	0.67	86%
2	1.00	0.67	0.80	96%
3	0.75	0.86	0.80	96%
4	1.00	0.50	0.67	93%
5	1.00	0.83	0.91	98%
6	1.00	0.56	0.71	93%
7	0.47	0.89	0.62	84%
8	0.60	0.43	0.50	90%
9	0.67	0.55	0.60	87%
10	1.00	0.50	0.67	96%

X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

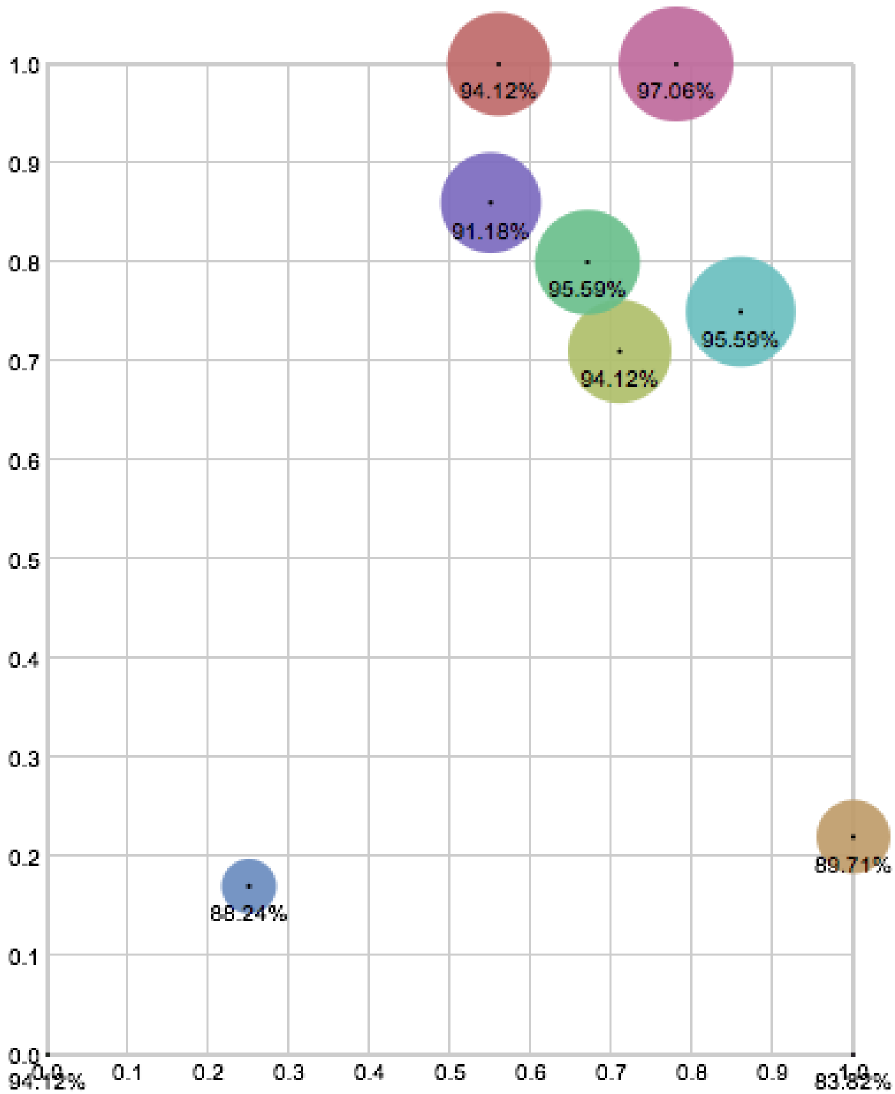


5.4.3 Support Vector Machine

- Support Vector Machines are supervised learning methods used for classification and regression analysis. Each data point in SVM is viewed as a p-dimensional vector and a (p-1) dimensional hyper plane can be used to separate the data set into two classes. A two dimensional dataset is separated by a 1 dimensional hyper plane defined by w. x-b=0. The hyper plane is chosen such that its distance from the data points lying on the margin is maximum. The margins are defined by w. x-b=1 and w. x-b=-1 and the data points lying on them are support vectors. In some cases where the data is not linearly separable, the data need to be mapped to a higher dimensional space so that it is linearly separable in that dimension.
- Since SVMs can only be used for binary classification and we have 10 classes, we need to build multiple binary classification SVMs. This can be achieved by coding designs like One-Versus-All (OVA), One-Versus-One (OVO) etc. Matlab already has multi-class models for support vector machines which are available by the function “fitcecoc”.
- We have used “fitcsvm” which is a binary SVM model and we are training multiple binary SVM classifiers following One-Versus-All model. For every user, an SVM model is built for every gesture, keeping the gesture class as positive and all the other class as negative. The predictions made by all these models need to be accumulated to get the final prediction.
- Please refer DTREE.m file for the complete code.
- We generated a Precision versus Recall graph for each user. The X-Axis will represent Precision, Y-Axis will represent Recall, size of the node will vary according to F1 value, label on each node will denote accuracy % value and the colors will represent the classes.
- The results that we obtained are described in the next few pages.

User 1

Class	Precision	Recall	F1	Accuracy
1	0.56	1.00	0.71	94%
2	1.00	0.22	0.36	90%
3	0.71	0.71	0.71	94%
4	0.00	0.00	0.00	94%
5	0.67	0.80	0.73	96%
6	0.86	0.75	0.80	96%
7	0.25	0.17	0.20	88%
8	0.55	0.86	0.67	91%
9	1.00	0.00	0.00	84%
10	0.78	1.00	0.88	97%

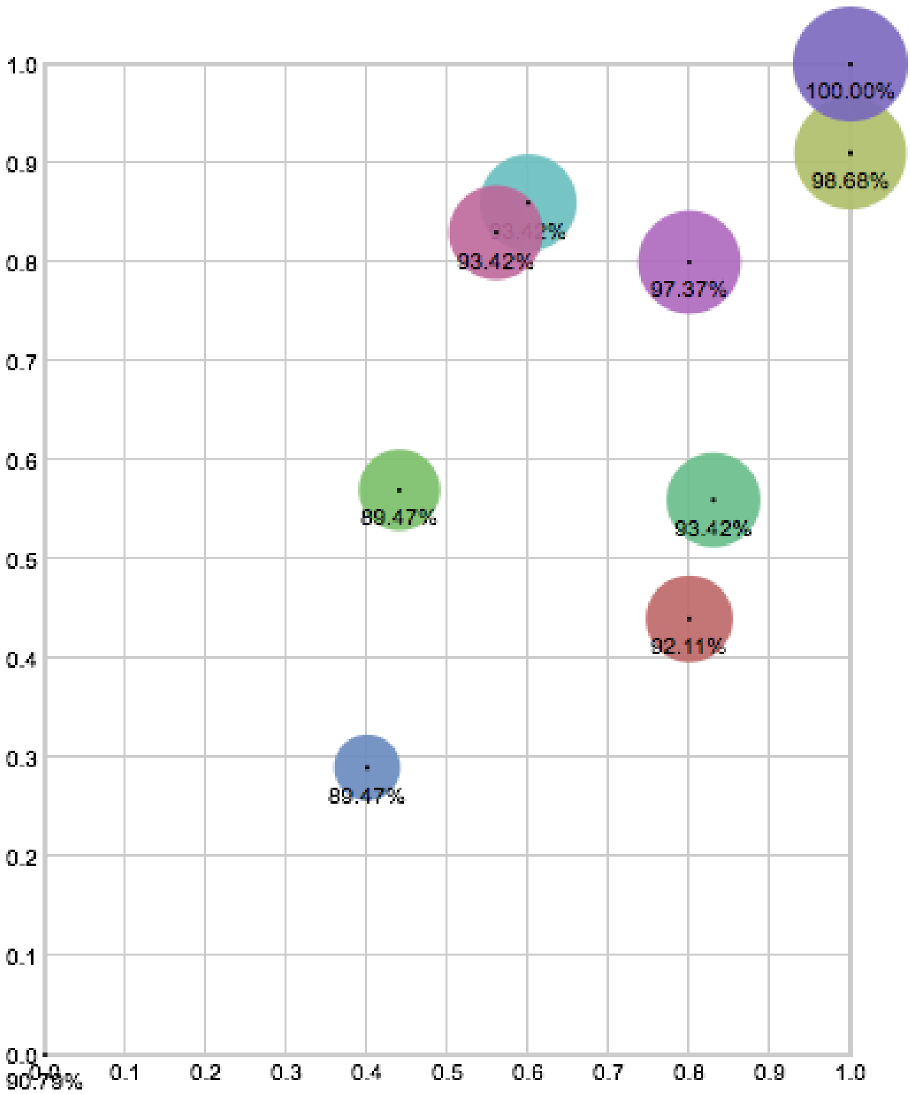


X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 5

Class	Precision	Recall	F1	Accuracy
1	0.80	0.44	0.57	92%
2	0.00	0.00	0.00	91%
3	1.00	0.91	0.95	99%
4	0.44	0.57	0.50	89%
5	0.83	0.56	0.67	93%
6	0.60	0.86	0.71	93%
7	0.40	0.29	0.33	89%
8	1.00	1.00	1.00	100%
9	0.80	0.80	0.80	97%
10	0.56	0.83	0.67	93%

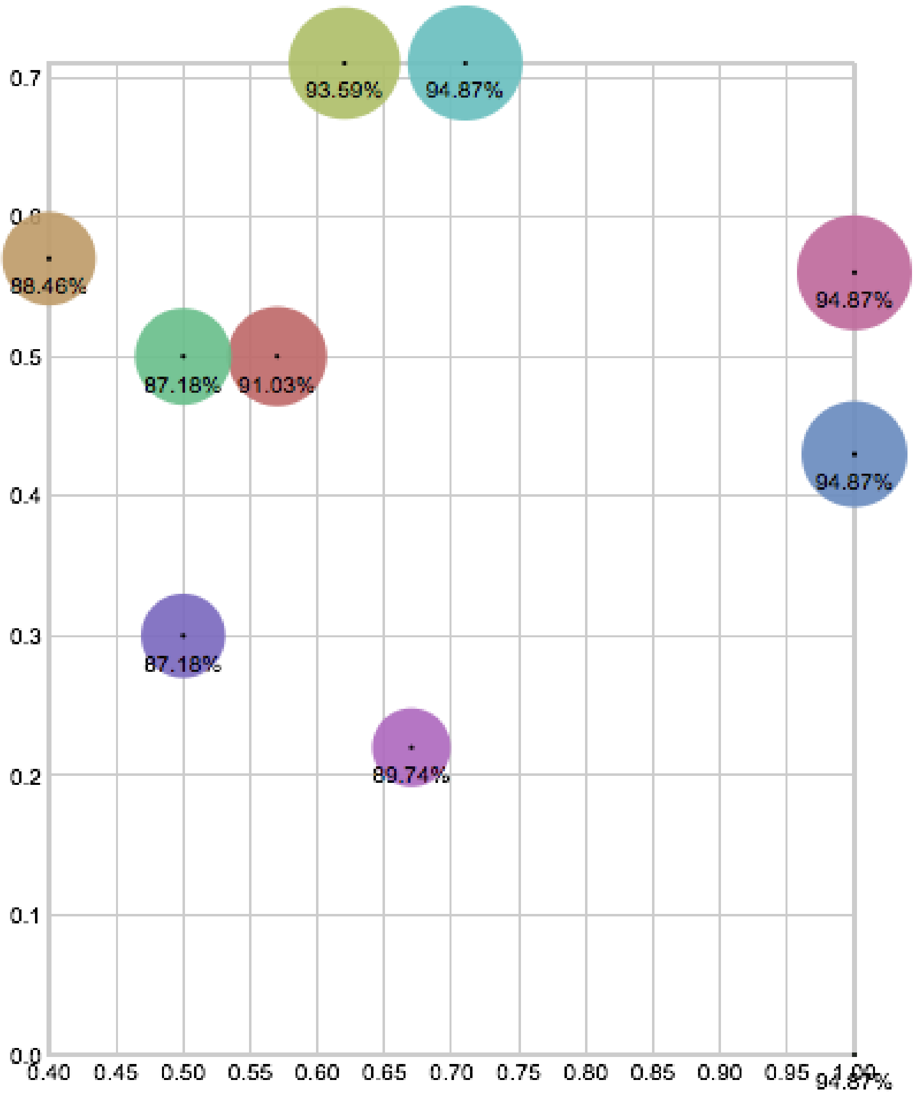
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



User 11

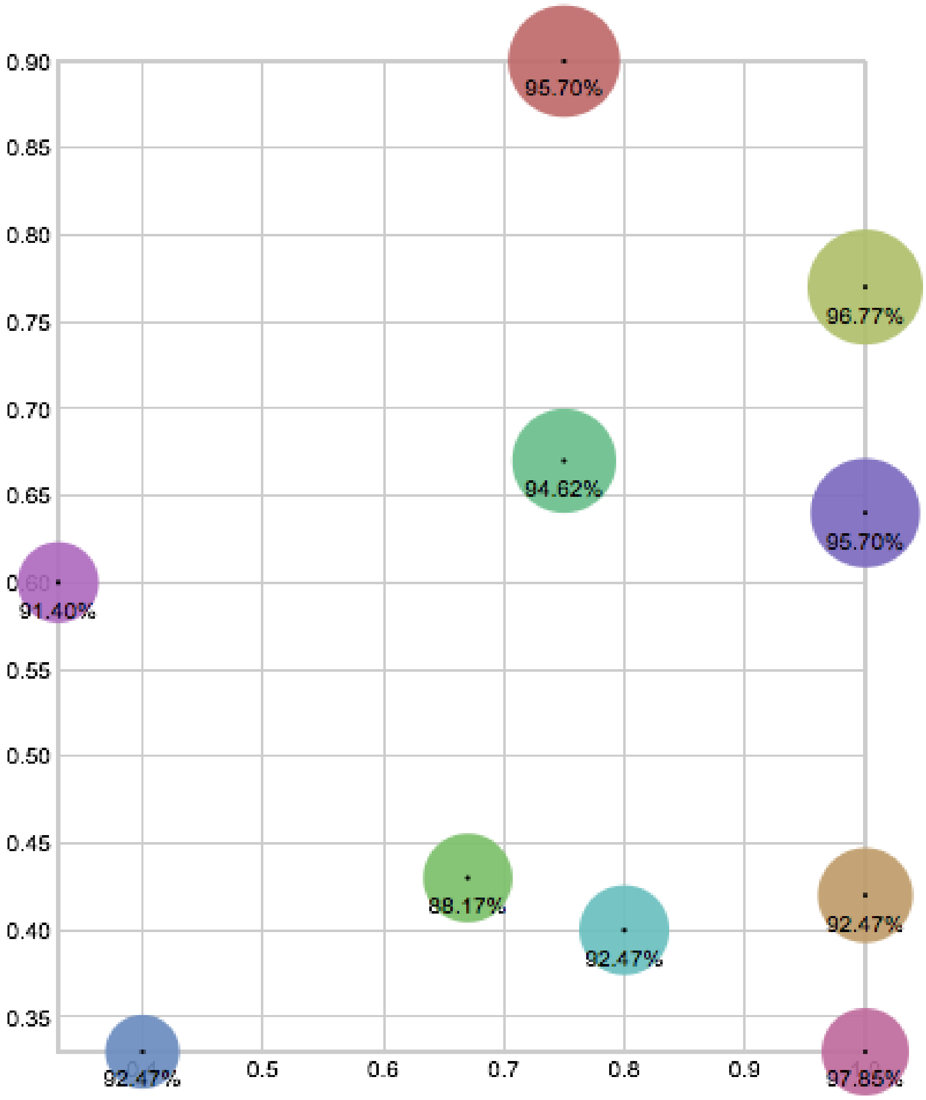
Class	Precision	Recall	F1	Accuracy
1	0.57	0.50	0.53	91%
2	0.40	0.57	0.47	88%
3	0.62	0.71	0.67	94%
4	1.00	0.00	0.00	95%
5	0.50	0.50	0.50	87%
6	0.71	0.71	0.71	95%
7	1.00	0.43	0.60	95%
8	0.50	0.30	0.38	87%
9	0.67	0.22	0.33	90%
10	1.00	0.56	0.71	95%

X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



User 16

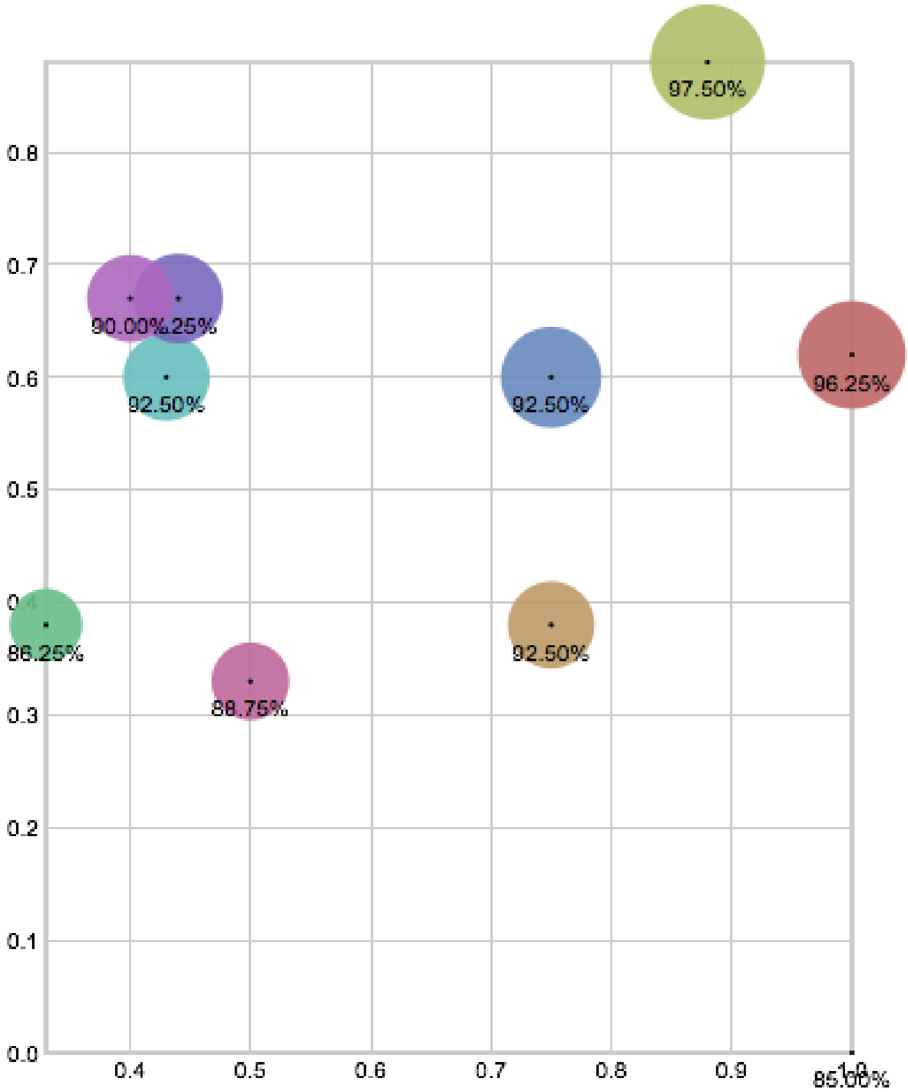
Class	Precision	Recall	F1	Accuracy
1	0.75	0.90	0.82	96%
2	1.00	0.42	0.59	92%
3	1.00	0.77	0.87	97%
4	0.67	0.43	0.52	88%
5	0.75	0.67	0.71	95%
6	0.80	0.40	0.53	92%
7	0.40	0.33	0.36	92%
8	1.00	0.64	0.78	96%
9	0.33	0.60	0.43	91%
10	1.00	0.33	0.50	98%



X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 19

Class	Precision	Recall	F1	Accuracy
1	1.00	0.62	0.77	96%
2	0.75	0.38	0.50	93%
3	0.88	0.88	0.88	98%
4	1.00	0.00	0.00	85%
5	0.33	0.38	0.35	86%
6	0.43	0.60	0.50	93%
7	0.75	0.60	0.67	93%
8	0.44	0.67	0.53	91%
9	0.40	0.67	0.50	90%
10	0.50	0.33	0.40	89%

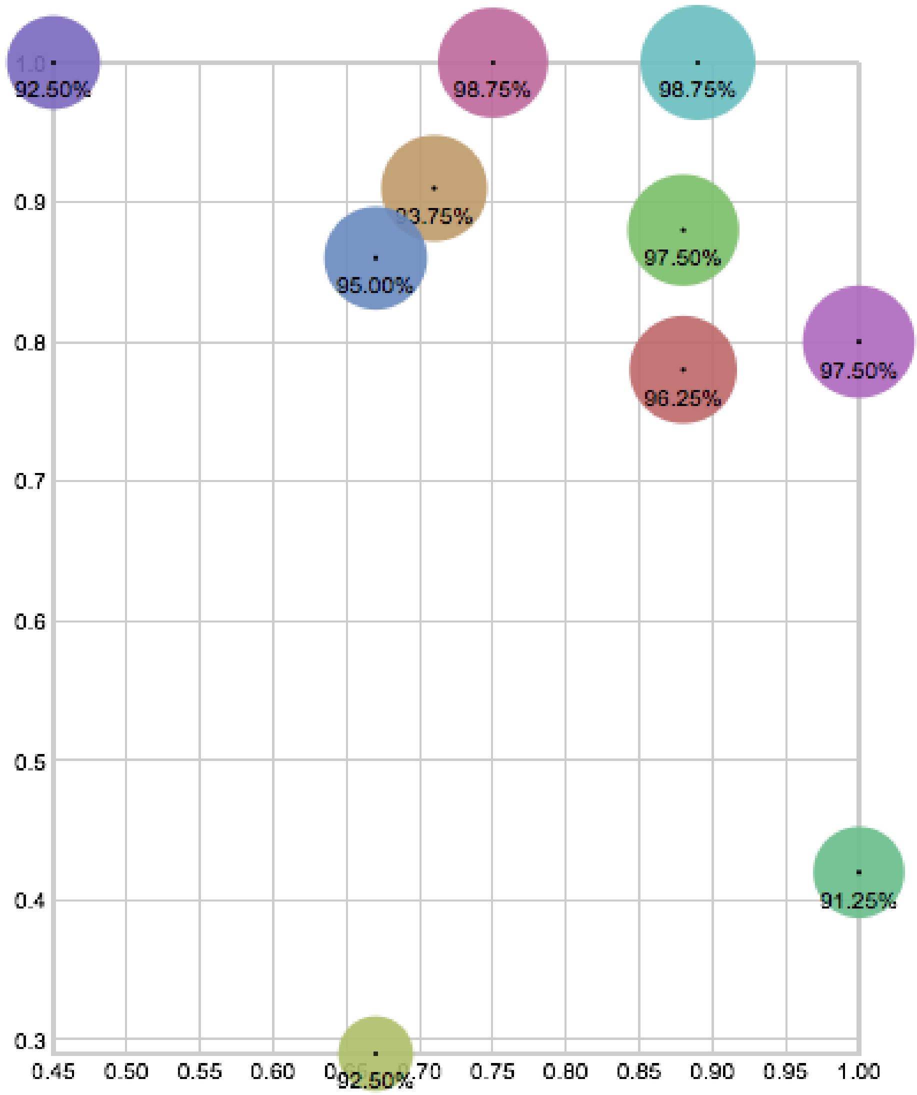


X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes

User 20

Class	Precision	Recall	F1	Accuracy
1	0.88	0.78	0.82	96%
2	0.71	0.91	0.80	94%
3	0.67	0.29	0.40	93%
4	0.88	0.88	0.88	98%
5	1.00	0.42	0.59	91%
6	0.89	1.00	0.94	99%
7	0.67	0.86	0.75	95%
8	0.45	1.00	0.62	93%
9	1.00	0.80	0.89	98%
10	0.75	1.00	0.86	99%

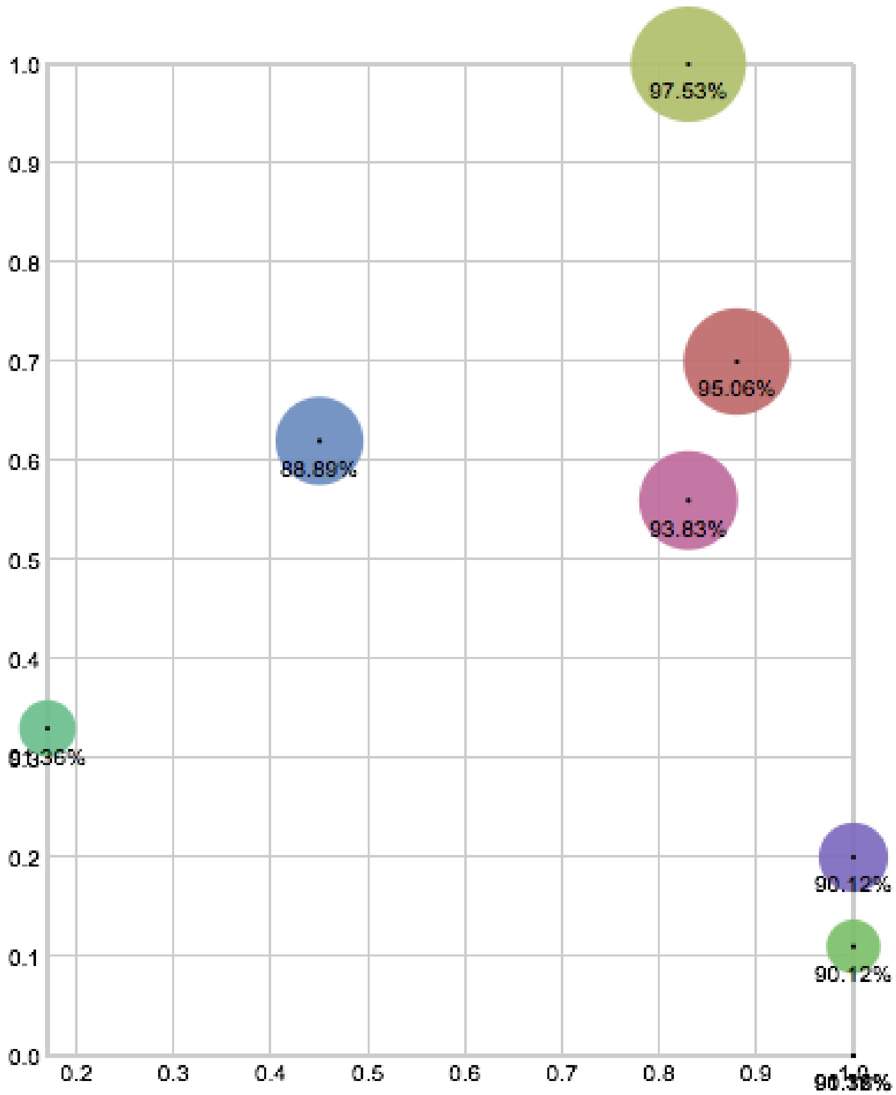
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



User 22

Class	Precision	Recall	F1	Accuracy
1	0.88	0.70	0.78	95%
2	1.00	0.00	0.00	91%
3	0.83	1.00	0.91	98%
4	1.00	0.11	0.20	90%
5	0.17	0.33	0.22	91%
6	1.00	0.00	0.00	91%
7	0.45	0.62	0.53	89%
8	1.00	0.20	0.33	90%
9	1.00	0.00	0.00	90%
10	0.83	0.56	0.67	94%

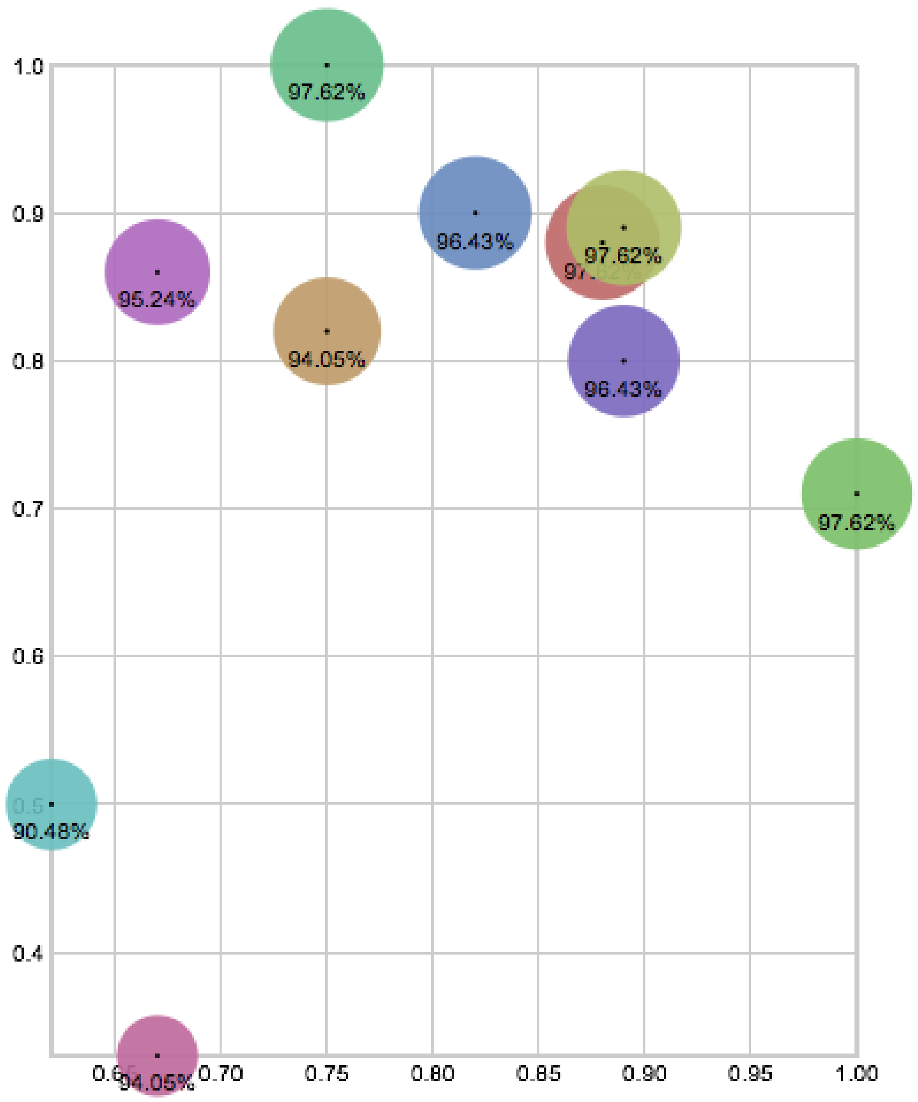
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



User 24

Class	Precision	Recall	F1	Accuracy
1	0.88	0.88	0.88	98%
2	0.75	0.82	0.78	94%
3	0.89	0.89	0.89	98%
4	1.00	0.71	0.83	98%
5	0.75	1.00	0.86	98%
6	0.62	0.50	0.56	90%
7	0.82	0.90	0.86	96%
8	0.89	0.80	0.84	96%
9	0.67	0.86	0.75	95%
10	0.67	0.33	0.44	94%

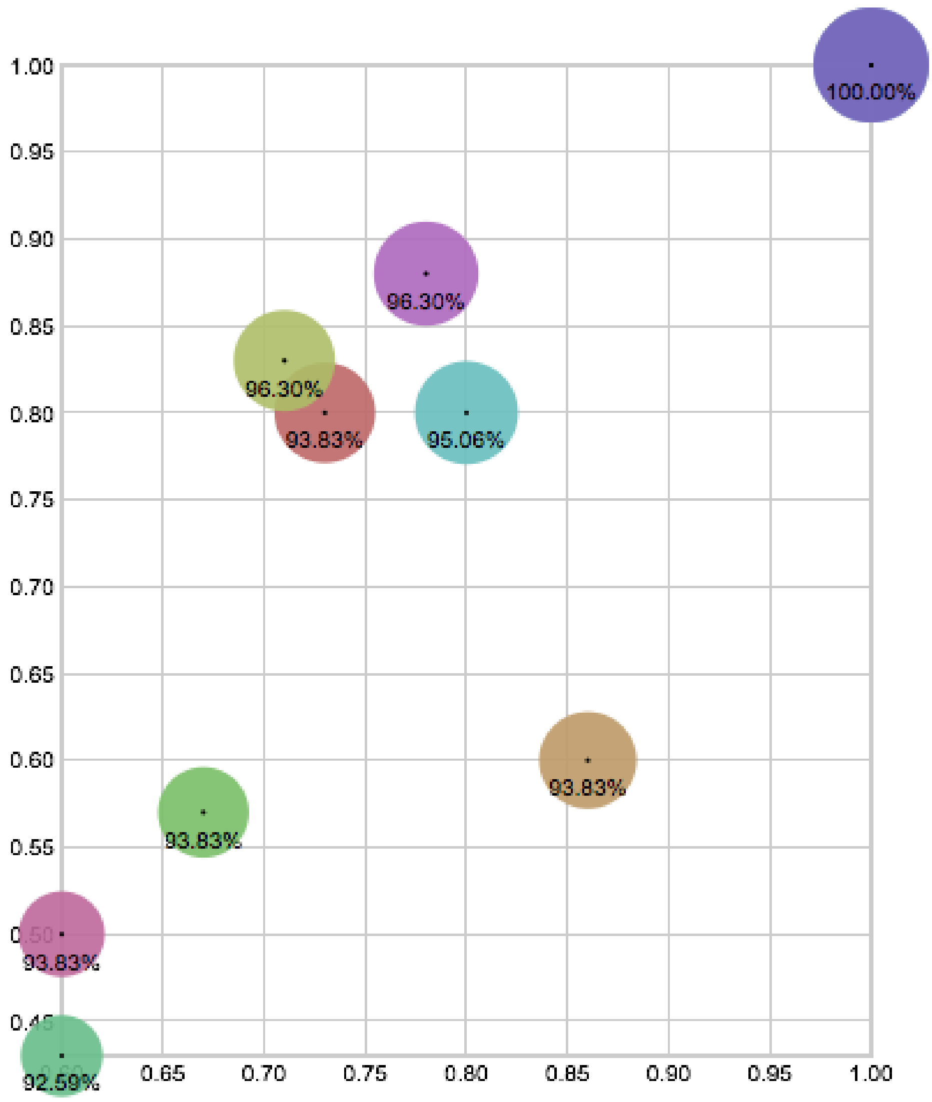
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



User 28

Class	Precision	Recall	F1	Accuracy
1	0.73	0.80	0.76	94%
2	0.86	0.60	0.71	94%
3	0.71	0.83	0.77	96%
4	0.67	0.57	0.62	94%
5	0.60	0.43	0.50	93%
6	0.80	0.80	0.80	95%
7	1.00	1.00	1.00	100%
8	1.00	1.00	1.00	100%
9	0.78	0.88	0.82	96%
10	0.60	0.50	0.55	94%

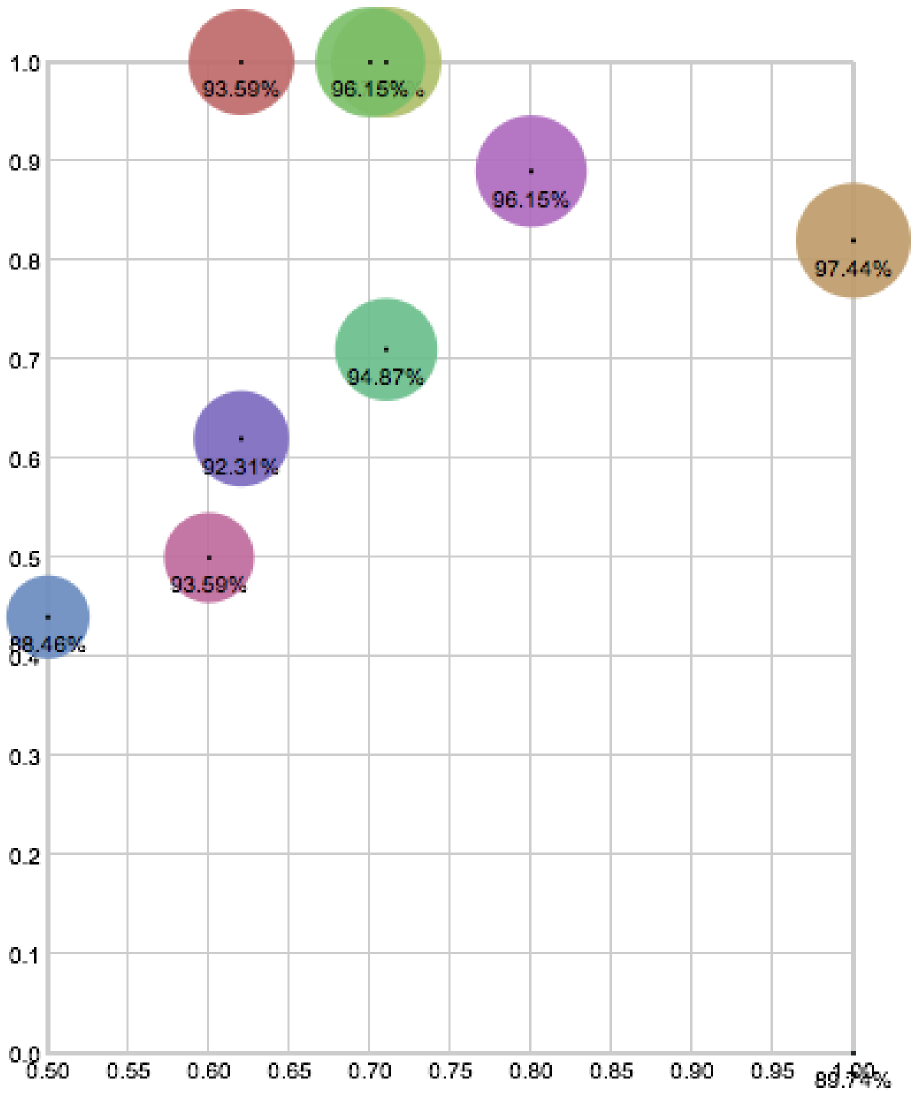
X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



User 36

Class	Precision	Recall	F1	Accuracy
1	0.62	1.00	0.76	94%
2	1.00	0.82	0.90	97%
3	0.71	1.00	0.83	97%
4	0.70	1.00	0.82	96%
5	0.71	0.71	0.71	95%
6	1.00	0.00	0.00	90%
7	0.50	0.44	0.47	88%
8	0.62	0.62	0.62	92%
9	0.80	0.89	0.84	96%
10	0.60	0.50	0.55	94%

X Axis : Precision Values
Y Axis : Recall Values
Size of points : F1 Values
Labels : Accuracy Values
Colors : Classes



6 Conclusion

The objective of this project is to build a user dependent classifier using different techniques in data mining. We used techniques like decision tree, SVM, Neural Networks to build the same. Model properties like accuracy, precision, recall and F1 score are observed for each model in each technique. Even though neural networks are robust enough to perform complex tasks, it yielded low F1 score across different user modules. On the contrary, SVM had a better performance on user dependent model followed by neural networks and decision tree. We believe SVM outperforms neural networks because of deficiency in the data. Given sufficient amount of observations represented in good feature dimension, we believe neural network will outperform SVM.

7 Resources/ Citations

- MYO Sensor : https://www.myo.com/
- MATLAB Decision Tree : https://www.mathworks.com/help/stats/fitctree.html
- MATLAB Predict https://www.mathworks.com/help/stats/compactclassificationdiscriminant.predict.html
- MATLAB View : https://www.mathworks.com/help/matlab/ref/view.html
- MATLAB SVM : https://www.mathworks.com/help/stats/fitcsvm.html
- Decision Tree : https://www.youtube.com/watch?v=eKD5gxPPeY0
- Raw Data Visualization : https://rawgraphs.io/
- [1] Artificial Neural Network : https://en.wikipedia.org/wiki/Artificial_neural_network
- Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures : http://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/
- Beyond Accuracy: Precision and Recall : https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c
- A Comparison of Methods for Multi-class Support Vector Machines : https://www.csie.ntu.edu.tw/~cjlin/papers/multisvm.pdf
- MATLAB fitcecoc : https://www.mathworks.com/help/stats/fitcecoc.html?s_tid=gn_loc_drop#bufm0zb-1
- A systematic analysis of performance measures for classification tasks : http://rali.iro.umontreal.ca/rali/sites/default/files/publis/SokolovaLapalme-JIPM09.pdf
- https://blog.keras.io/building-autoencoders-in-keras.html

- https://github.com/keras-team/keras/commit/a56b1a55182acf061b1eb2e2c86b48193a0e88f7
- https://medium.com/@pushkarmandot/build-your-first-deep-learning-neural-network-model-using-keras-in-python-a90b5864116d
- https://medium.com/@pushkarmandot/build-your-first-deep-learning-neural-network-model-using-keras-in-python-a90b5864116d
- https://github.com/keras-team/keras/issues/5794
- https://stackoverflow.com/questions/43076609/how-to-calculate-precision-and-recall-in-keras
- https://keras.io/metrics/
- https://machinelearningmastery.com/display-deep-learning-model-training-history-in-keras/
- https://github.com/keras-team/keras/blob/master/examples/mnist_mlp.py
- https://keras.io/scikit-learn-api/
- https://keras.io/visualization/
- https://keras.io/getting-started/sequential-model-guide/
- https://machinelearningmastery.com/custom-metrics-deep-learning-keras-python/
- https://github.com/keras-team/keras/issues/2644
- https://github.com/keras-team/keras/issues/2607
- https://keras.io/optimizers/
- https://machinelearningmastery.com/multi-class-classification-tutorial-keras-deep-learning-library/