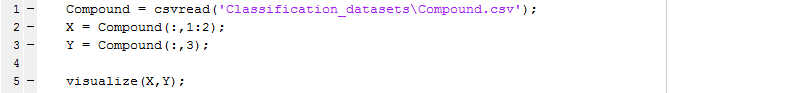
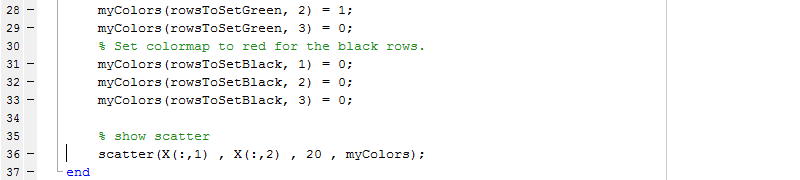
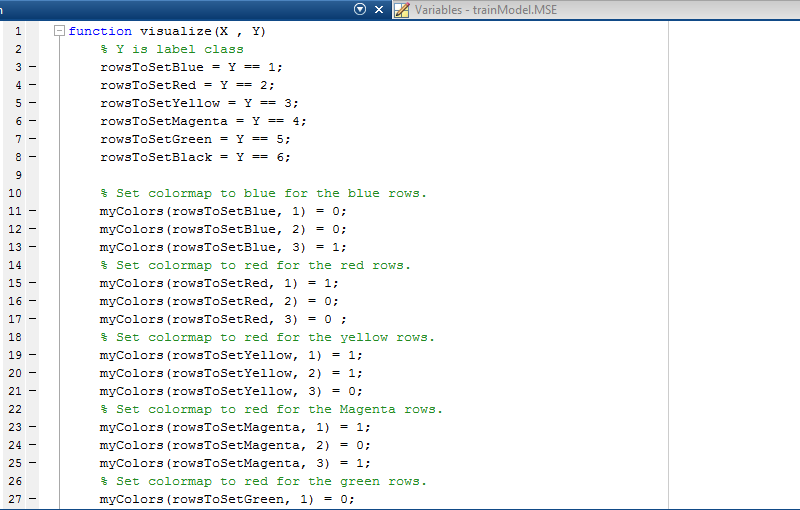
**MULTI LAYER PERCEPTRON**

1. Load and Visualize the Data

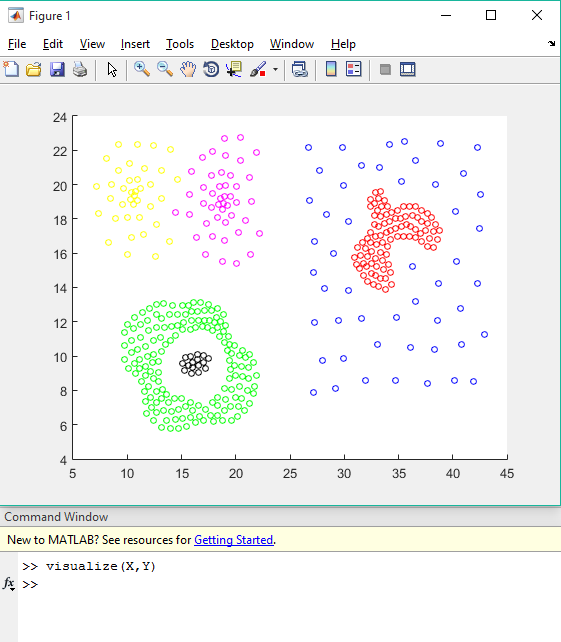
* MLPMain.m



* visualize.m



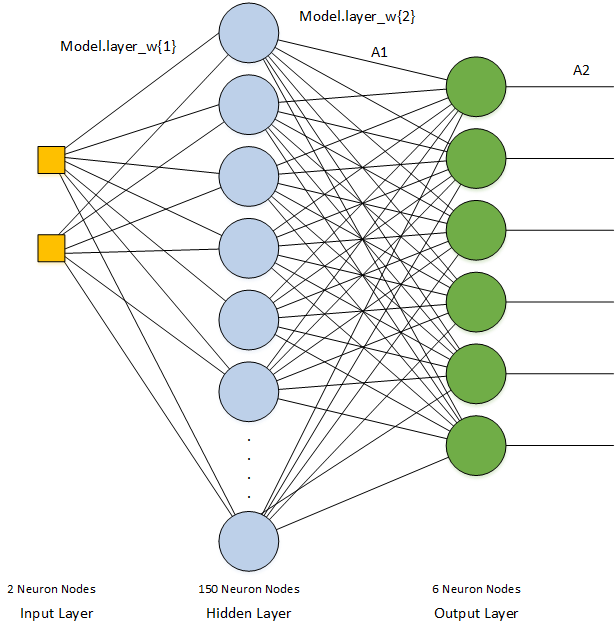
Output :



Analisis Program :

Pada tugas ini saya menggunakan data set dengan nama file Compound.csv, kemudian meload data tersebut kedalam variable X untuk features dan Y untuk labelnya. Dan menjalankan fungsi visualize(X,Y), untuk memvisualisasikan setiap data point. Pada fungsi visualize(x,y) masing-masing data label class disesuaikan dengan data warna RGB. Kemudian memanggil scatter(param1,param2,param3,param4) dengan param1 dan param2 adalah kolom pertama dari data train dan kolom kedua dari data train, kemudian param3 adalah size dari ukuran plot, dan param4 adalah data color.

1. MLP Architecture



Jenis JST yang saya terapkan adalah multi layer perceptron dengan 1 Hidden Layer dan 1 Output layer. Dengan hidden node sebanyak 150 buah dan Output node sebanyak 6 buah. Pada Matlab dibuat sebuah model dari JST tersebut dengan variable objek model.layer\_w{1} antara input layer dan hidden layer dan model.layer\_w{2} antara hidden layer dengan output layer. Dan signal antara hidden layer adalah A1 dan Output layer adalah A2. Kemudian untuk target menerapkan format sebagai berikut :

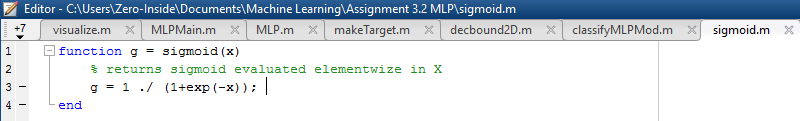
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Label | Format Target | | | | | |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 1 | 0 | 0 | 0 | 0 |
| 3 | 0 | 0 | 1 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 1 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 1 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 1 |

Dengan output sebanyak 6 buah sesuai dengan banyaknya class diharapkan performansi untuk JST meningkat. Untuk learning rate saya set dengan 0.00001 dan inputan tanpa bias. Epoch sebanyak 10000 dengan waktu selesai sekitar 4 jam menggunakan model ini. Activation function yang dipakai adalah sigmoid. Memetakan A1 dan A2 saat feedforward dengan nilai antara hampir 1 dan hampir 0. Dan untuk back propagation adalah turunan pertama dari sigmoid.

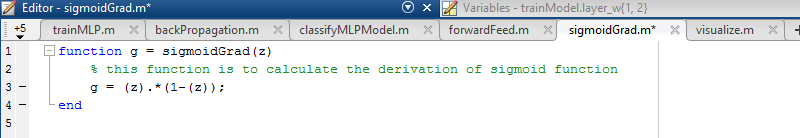


Berikut code MATLAB untuk activation code dan activation code derivative :

* sigmoid.m

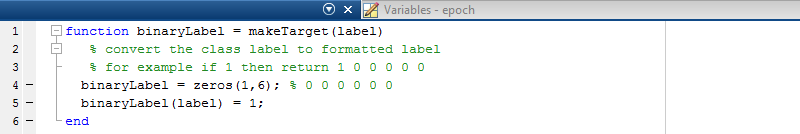


* sigmoidGrad.m

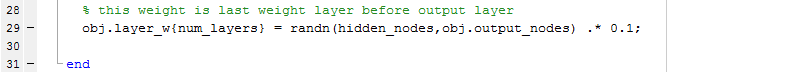
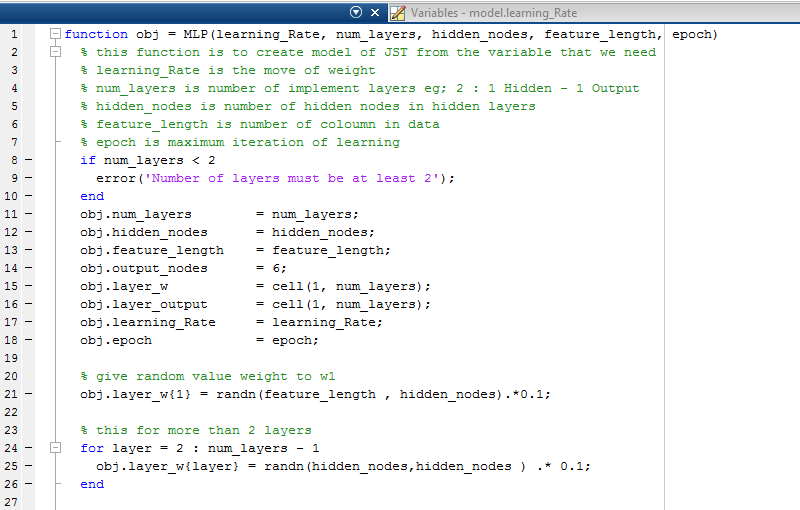


Berikut code MATLAB untuk membuat model JSP diatas.

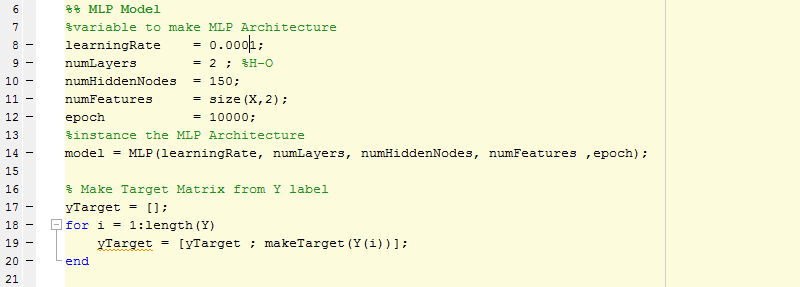
* + - makeTarget.m



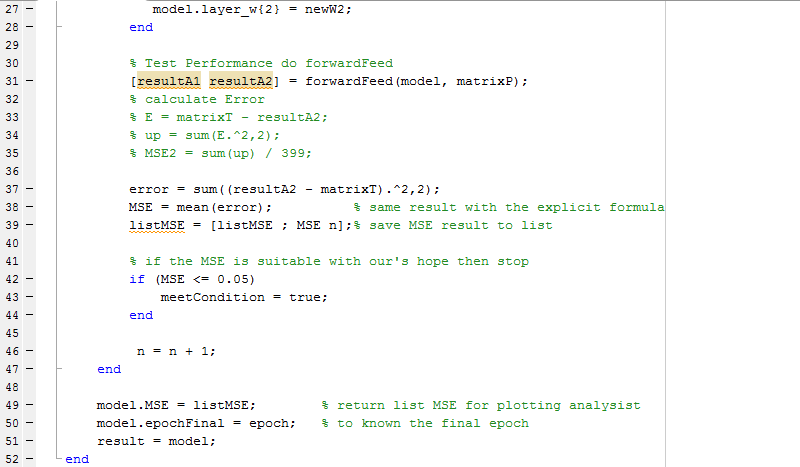
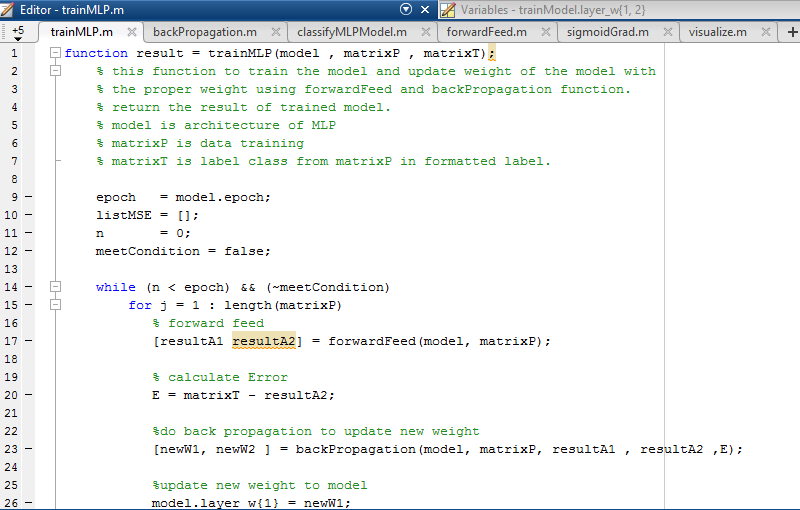
* + - MLP.m



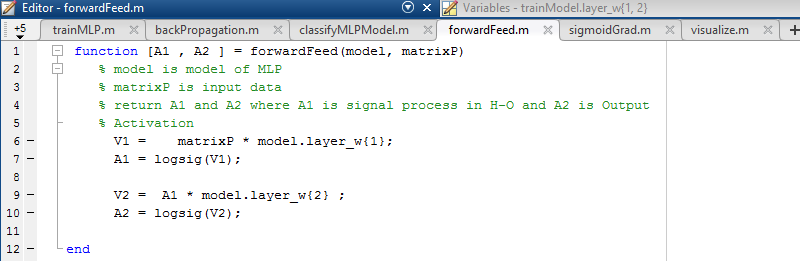
* + - MLPMain.m



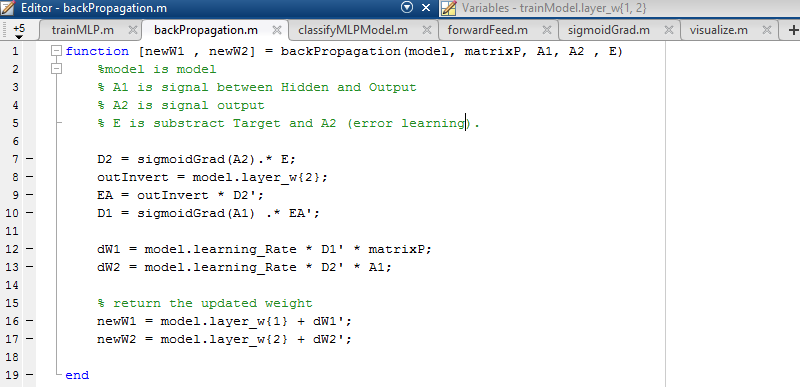
1. Apply Multi-Layer Perceptron
   1. Function for Learning
      * trainMLP.m



* forwardFeed.m



* backPropagation.m



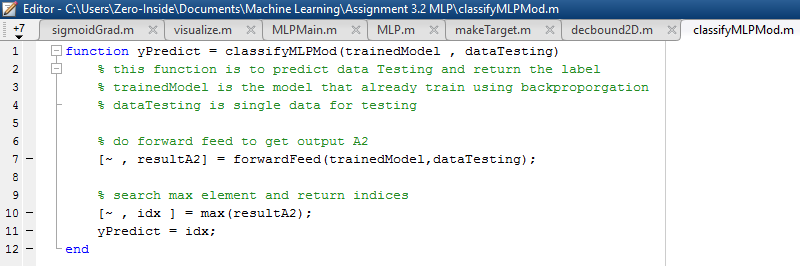
* MLPMain.m



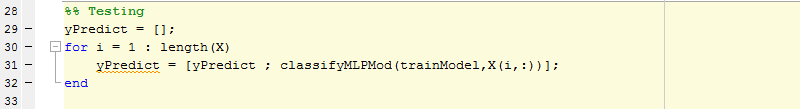
Analisis Program :

Untuk melatih model dengan menerapkan algoritma back propagation, menjalankan trainMLP(param1,param2,param3) dengan param1 adalah model JST, MatrixP adalah data training, dan param3 adalah data label masing-masing data training yang sudah disesuaikan dengan format. Proses pelatihan dengan mengulangi literasi 1 siklus pelatihan sebanyak epoch. Terlebih dahulu model JST yang sudah dibentuk dilakukan forward feed atau pelatihan maju dengan menjalankan fungsi forwardFeed(param1,param2) dengan param1 adalah model, dan param2 adalah data training dihasilkan signal A1 dan signal A2, kemudian untuk mendapatkan error dengan mengurangkan matrix Target dan A2 lalu dikuadratkan dan dijumlah kemudian dirata-rata atau disebut MSE. Hasil A1,A2 dan error akan digunakan untuk menjalankan fungsi backPropagation untuk mengupdate bobot yang ada pada model. Untuk analisis error, dilakukan record MSE pada setiap epoch. Setelah sampai batas epoch atau nilai MSE yang diinginkan dalam hal ini saya mengeset error 0.05 maka didapat model baru pelatihan dengan bobot yang siap untuk ditesting.

* 1. Function for Predicting/Classifying Data
     + classifyMLPMod.m



* MLPMain.m



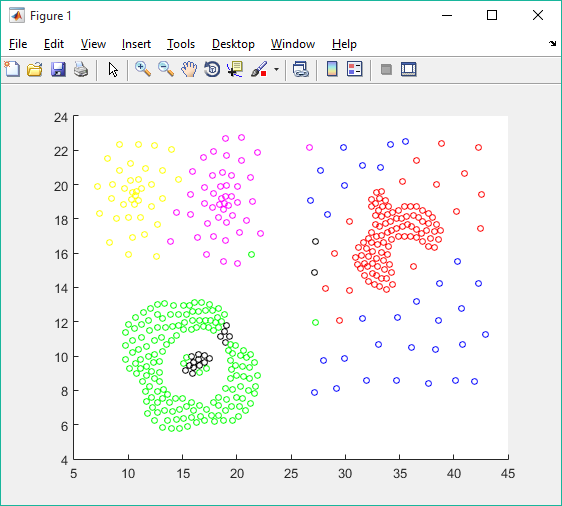
Analisis Program :

Fungsi classifyMLPMod(param1,param2) akan memprediksi data testing dan mengembalikan nilai classnya. Dengan param1 adalah model yang sudah dilatih dan param2 adalah satu data testing yang akan diuji. Untuk memprediksi atau mengklasifikasikan data testing dilakukan pelatihan maju terlebih dahulu dengan menjalankan fungsi forwardFeed dan didapat output neuron atau resultA2. Kemudian mencari nilai index yang ada pada element maximal pada resultA2. Index ini menjadi hasil label yang diprediksi pada data tersebut. Untuk mencari label pada seluruh data testing dilakukan iterasi, bisa dilihat pada file MLPMain.m untuk seluruh data testing.

* 1. Visualize the predicting data
* MLPMain.m



* Output



Analisis Program :

Dengan memanggil visualize(x,yPredict2), dimana x adalah train data dan yPredict adalah class yang dihasilkan melalui testing maka ditampilkan scatter yang memplot posisi dari data train.

* 1. Difference between testing and training

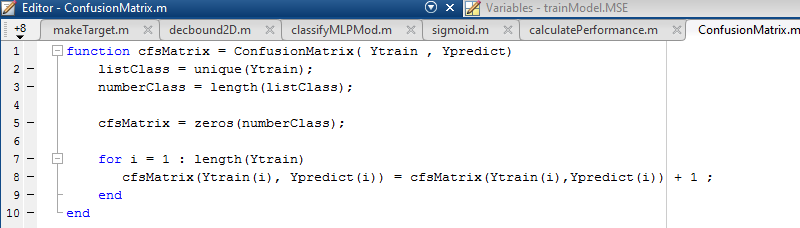
|  |  |
| --- | --- |
| Scatter Data Asli | Scatter Data Prediksi |
|  |  |

Analisis :

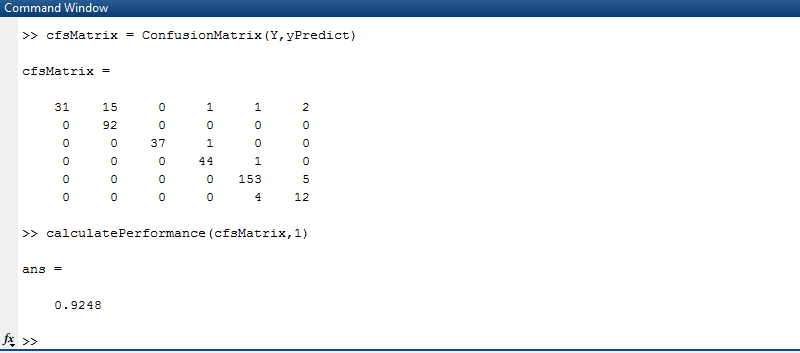
Terlihat ada warna plot yang berbeda dari yang hasil scatter data asli. Tetapi secara keseluruhan JST yang ditrain dapat mengkelompokkan data tersebut sesuai kelasnya dalam visual ini adalah warna.

1. Performance Analysis

* ConfusionMatrix.m



* Output

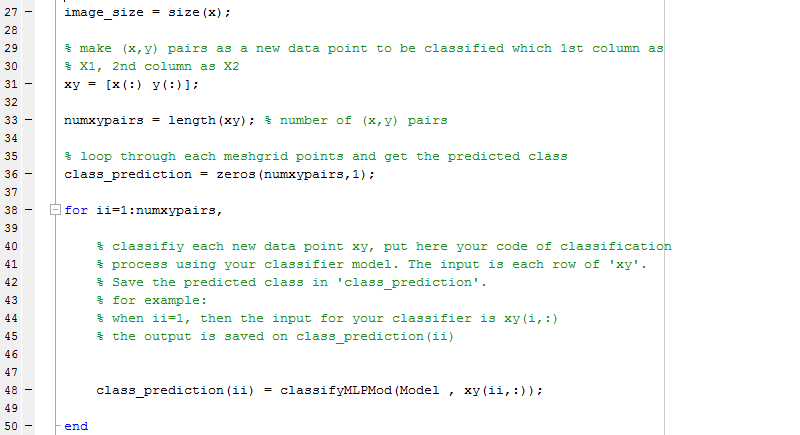
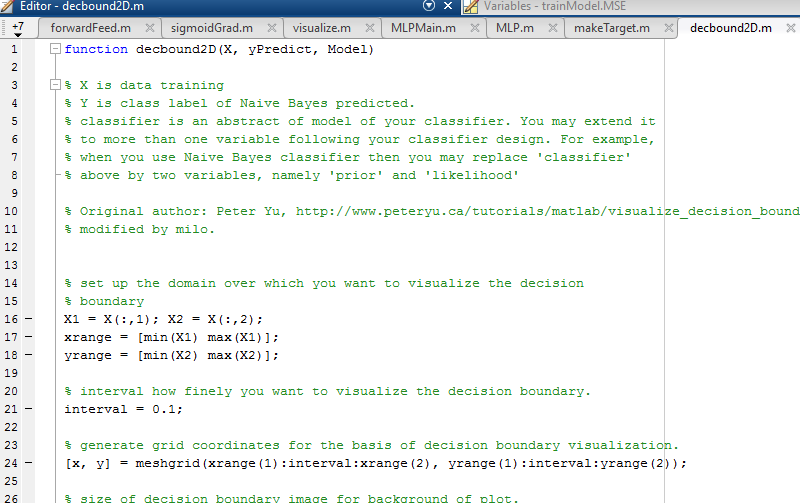


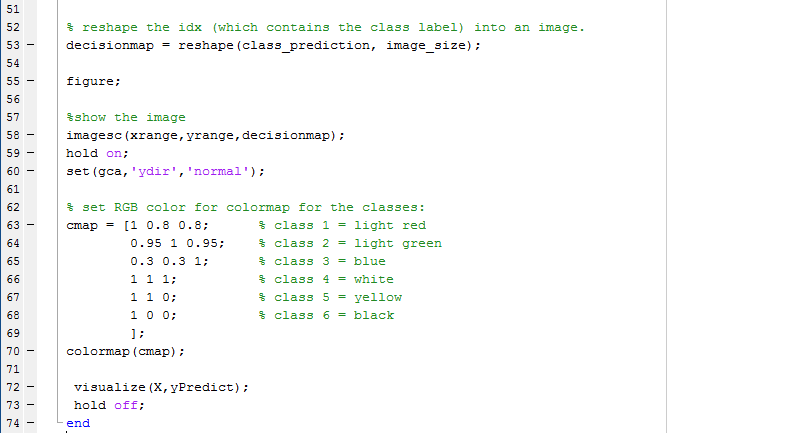
Analisis Program :

Untuk menghitung performance terlebih dahulu membuat confusion matrix berdasarkan label data train dan label data testing. Kemudian untuk menghitung performasi memanggil fungsi calculatePerformance(param1, param2). Dengan param1 adalah confusion matrix dan param2 adalah mode atau jenis performasi, 1 untuk F1 Micro 2 untuk F2 Macro dan 3 untuk Accuracy. Berdasarkan hasil perhitungan didapat hasil klasifikasi naïve bayes adalah 0.9248. atau dalam persen adalah 92.48%.

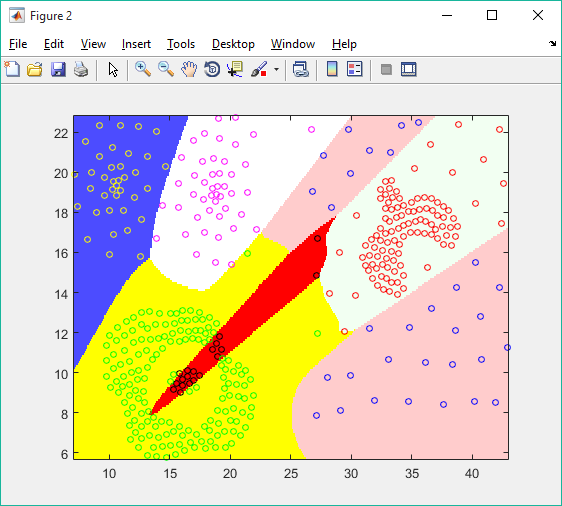
1. Decision Boundaries

* Decbound2D.m





* Output



Analisis Program :

Fungsi decbound2d(param1,param2,param3) akan menampilkan decision boundaries dengan param1 adalah data testing , param2 adalah class label hasil dari model JST yang sudah di latih, dan param3 adalah model yang akan digunakan untuk memberi label kelas pada new data point yang dipetakan pada imagesec.