Informatics 2b Coursework 1 Report

s1006260

21/03/2012

Part I
Explantion of Code

1 Functions

Functions are herein listed and described in roughly chronological order. Some functions are ommitted, as they were only used for testing and/or the plotting of graphs.

1.1 distanceMatrix

distanceMatrix is a function which take no arguments, and returns a matrix of all the Euclidean Distances between every point in the file data_90.mat. It requires the data_90 file to be present in the same file directory.

1.2 findmeans

findmeans is a function which takes a single argument, (designed to be a distance matrix), and returns the two points furthest from each other. It requires the data_90 file to be present in the same file directory.

1.3 kMeans

kMeans is a funtion which takes 2 arguments, (designed to be an integer and a matrix containing a list of means),

and returns additional means based on the value of the integer. For example, if the integer were 5, kMeans would return the initial list of means it was given, plus additional means calculated by kMeans, until 5 means were returned. Each additional mean returned is determined by finiding the point with the greatest product of distances from the current list of means. The means returned are contained within a matrix. It requires the data_90 file to be present in the same file directory.

1.4 kmeanscluster

kmeanscluster is a function which takes 2 arguments, (designed to be an integer and a matrix containing a list of means). It returns 2 items; a matrix containing the set of points and the cluster value of each point; and matrix containing recalculated means. The integer is the number of clusters to cluster the data into. The function clusters the data in data_90.mat based on the means given, and then recalculates the means based on the new clusters. This process loops until every time newmeans are calculated, they are no different from the previous set of means. For example,

[a,b] = kmeanscluster(3, kmeans(3, findmeans(distanceMatrix)))

Returns:

a =

-46.8961 -75.9698 -11.2257 2.0000

3.8562 -2.9817 7.7743 3.0000

36.2124 206.7241 -13.9085 1.0000

24.5805 -57.3156 7.6623 2.0000

-59.7804 -84.4784 -11.1064 2.0000

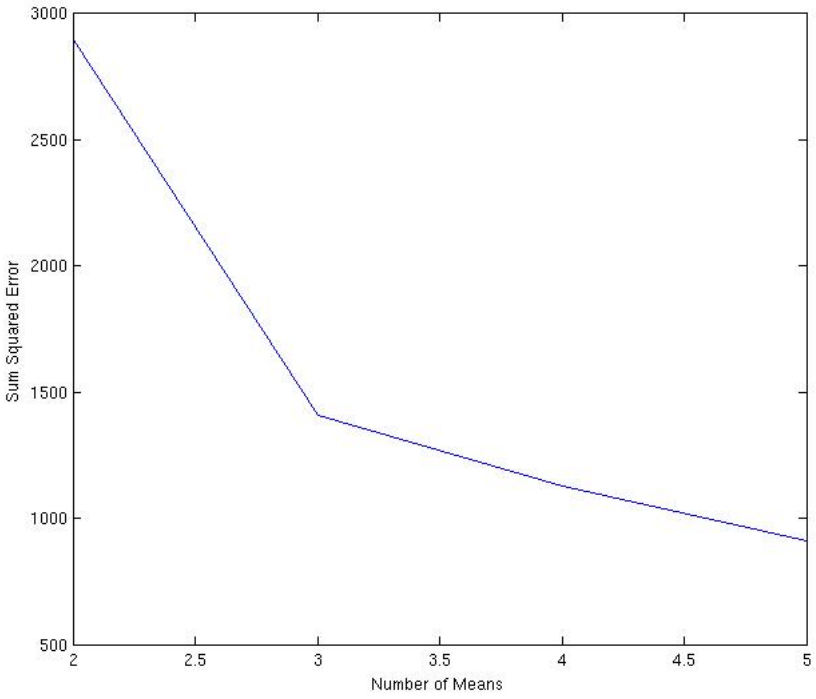
-28.0477 35.7853 9.9302 3.0000

35.8269 224.5214 -25.8797 1.0000
-48.7897 -87.8380 -5.9678 2.0000
24.0046 14.2970 12.7559 3.0000
...
b =
22.3244 203.5259 -13.2099
-14.8388 -81.5841 -9.3343
0.9240 9.5330 10.9567

The clusters for 2,3,4 & 5 means can all be determined by running the command [a,b] = kmeanscluster(i, kmeans(i, findmeans(distanceMatrix))) from the matlab terminal, replacing i with the number of means.
kmeanscluster requires the data_90 file to be present in the same file directory.

1.5 sumsquarederrors

sumsquarederrors is function which takes no arguments and returns a vector of error values. Rather than pass in variables, I hard coded them into the function instead, (bad practice, but it would make function calls easier). It obtains the outputs of kmeanscluster for 2,3,4 and 5 means, (using the same amount of clusters), and uses these to return the error vector. A plot of the error vector (the sum-squared error plot) is shown below:



sumsquarederrors requires the distanceMatrix, findmeans, kMeans and kmeanscluster files to be present in the same file directory.

1.6 covar

covar is a function which takes 1 argument, (designed to be an integer giving the number of means) and returns the covariance matrices for each cluster in a single matrix (immediately below each other). For example, covar(3) returns:

ans =
1.0e+03 *
0.7758 -0.2949 -0.0017
-0.2949 1.0332 -0.3522

-0.0017 -0.3522 0.1518
0.7508 -0.1381 0.0361
-0.1381 0.2969 -0.0181
0.0361 -0.0181 0.0175
0.6612 -0.3331 0.0648

The 9 by 3 matrix represents 3 3 by 3 matrices stacked on top of one another.

The covariance matrices were calculated using the formula

$$\frac{1}{N}\sum((x-\mu)\times(x-\mu)')$$

where N was the total number of points, x a point and $\mu$the mean.

It requires the distanceMatrix, findmeans, kMeans and kmeanscluster files to be present in the same file directory.

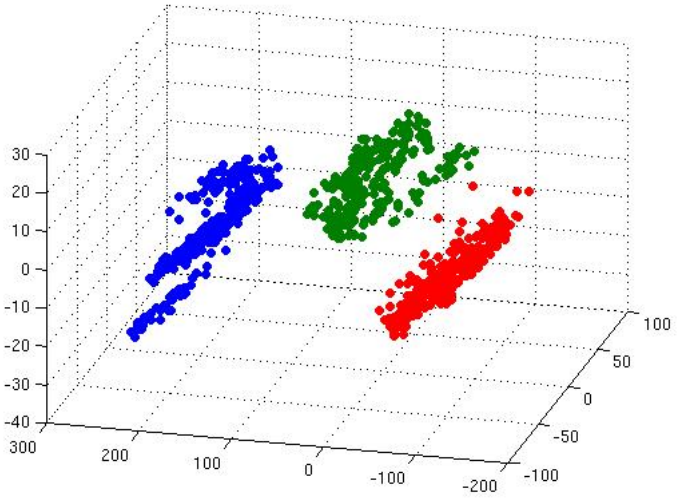
1.7 classify900

classify900 is a function which takes no arguments and returns a 900 by 1 matrix containing the value of the cluster that the point with the same index in data_900.mat should be assigned to according to the Gaussians. It determines this by calculating the probability density function for each cluster, using the formula:

$$\frac{1}{2\pi^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}}exp(-0.5(x-\mu)'\Sigma^{-1}(x-\mu))$$

There were then 3 values for the pdf. Taking the max of these values yielded which cluster the point should be assigned to.

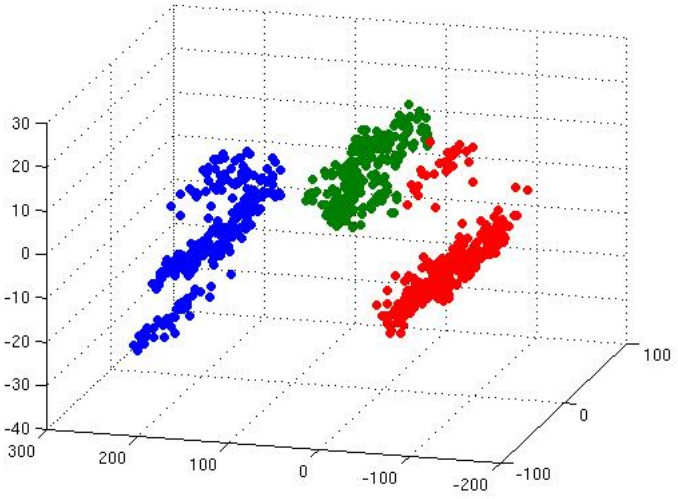
A graph of the 900 points with these cluster values is shown below:



The function requires the data_900.mat, distanceMatrix, findmeans, kMeans and kmeanscluster files to be present in the same file directory.

1.8 kmeanscluster900

Identical to the kmeanscluster function, except works with values from data_900 rather than data_90. A graph of the 900 points clustered according to kmeans is shown below:



This function requires the data_90 file to be present in the same file directory.

1.9 trueclusters

When comparing my clusters to the 'true' clusters, I saw that my clusters had been labelled differently from the ones given, namely, the 2s and 3s had been allocated differently. For ease of computing the confusion matrices, I wrote the function trueclusters, which takes the two sets of clusters given by cluster900 and kmeanscluster900, and swaps the 2s and 3s. The clusters themselves are preserved, just the labels are switched. truecluster returns the two 'switched' clusters. It requires kmeanscluster900, kMeans, findmeans and distanceMatrix to be in the same file directory.

1.10 confusion

confusion is a function which takes no arguments, and returns the confusion matrices for both the Gaussian Classi-

fication and K Means Clustering (In both cases assuming 3 clusters). The confusion matrices are given below:

a =
300 0 0
0 300 0
0 0 300
b =
300 0 0
0 268 0
0 32 300

where a is the confusion matrix for the Gaussian Classification and b the confusion matrix for the K Means Clustering.
Confusion requires trueclusters and the true_900 file to be in the same file directory.

1.11 main

main is a function I designed to try and give outputs to verify that I had completed all the programming subtasks. When executed from the Matlab terminal it produces 4 graphs, (the sum squared error plot, the clustered data_90 by 3 means, the gaussian classified data_900 and the 3 means clustered data_900), and also outputs the covariance matrices with their means, and the confusion matrices. It requires all functions, data_900.mat and data_90.mat to be present to run.

Part II
Comparison

2 Higher versus Lower number of means for K Means

From the sum squared error chart it is clear to see that a higher number of means produces a lower error. This is intuitively obvious, as the more means(cluster centres) you have, the less likely you are to have large distances between those cluster centres and points belonging to that cluster.
It can also be seen from the graph that, whilst having more means does initially reduce your error drastically, the error is influenced less and less with each extra mean you add. Again this is intuitively obvious, as initially adding means will greatly reduce the potential distance between points and their cluster centre, since said cluster centres will be very far apart. However as more means are created they will draw closer and closer together, not reducing the distance between points and means by nearly as much.

3 K Means Clustering versus Gaussian Classification

In this case it is clear that clustering by Gaussian Classification was more effective - it can be clearly seen from the graphs and the lack of errors in the confusion matrix. K Means, however, did make some errors - 32/900 values were clustered incorrectly, giving a 3.5556% error. Still, this is quite a low percentage, and in systems where accuracy is not critical, or where there are clear divisions between different clusters, it is still a viable method.