SPS CW1 - CLUSTERING AND CLASSIFYING DATA

ALEX ROBINSON (AR15247/1424755)

1. INTRODUCTION

1.1. **Introduction**. The coursework explores an unsupervised clustering and classification problem in python with two data sets, train/test - n=150/15. Demonstrating use and understanding of k-means as a clustering method, and nearest-centroid and maximum likelihood classification methods. This coursework was carried out individually.

2. FEATURE SELECTION

2.1. **Feature Selection**. The data contains 5 features, the task states that we must find two features which best separate the classes. In order to do this, we plot a matrix (figure 1) of every feature plotted against every other. Note the diagram is mirrored in the diagonal. By visual inspection the diagram at (2,1) separates the data into three distinct classes. This is chosen over other combinations of features as it has three dense centres while the other graphs are much sparser and also the separation between classes is larger and clearer than other plots. We go forward with only these features.

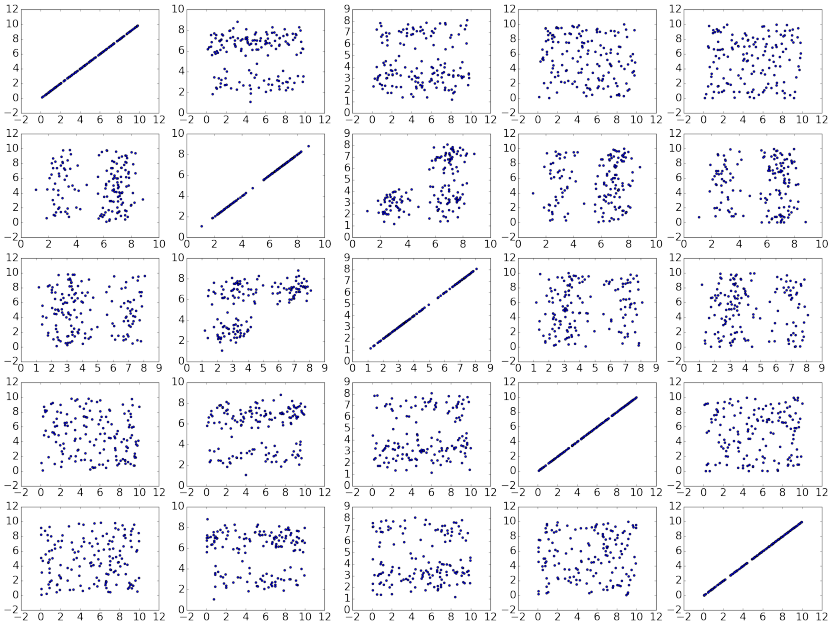

Figure 1 Feature Matrix

3. IDENTIFYING THE CLASSES

3.1. **Apply K-means**. By visual inspection above we can see that there are three distinct classes, therefore by applying K-means with K=3 we label the data into the three classes. Figure 3 shows the labelled clusters. K-means initially picks random observations as centroids and calculates the distortion – sum of squared differences between observations and closest centroid. A new centroid is computed

by averaging the position of all points associated with a cluster. This process is reported until it converges. In order to avoid local minima multiple initialisations are run and the best used.

4. Nearest-centroid classification

4.1. **Apply Nearest-Centroid Classification**. We now demonstrate the Nearest-centroid classifier (NCC) trained on the labels produced by K-means to classify our testing data. It works by calculating the euclidean distance from the test point to each cluster centroid and adopting the label of the cluster centroid with the smallest distance. The classified points as well as the decision boundaries are shown on Figure 3.

4.2. **Non-optimal K-means**. In order to demonstrate a non-optimal clustering we apply K-means 100 times and plot the clusters as well as their decision boundaries which had the worst within-cluster sums. In order to increase likelihood of a non-optimal clustering we altered K-means initialisation parameters to start with only a single random centroid seed rather than producing multiple random centroid seeds and picking the best. Figure 3 shows an example of a non-optimal clustering as well as its decision boundaries. The plot is clearly non-optimal as it cuts clusters 0 and 2 while not cutting cluster 1, furthermore it creates three sparse clusters which have no obvious centre.

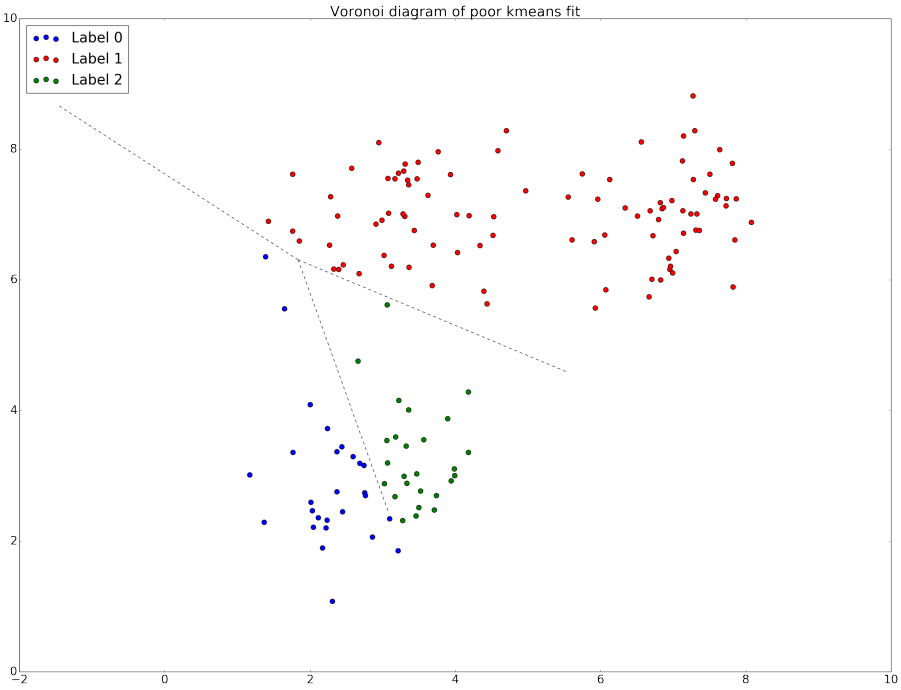

Figure 2 Poor Clustering

5. Maximum-likelihood classification

5.1. **Apply Maximum-Likelihood Classification**. Assuming the data is normally distributed, taking a mesh we now label each cell by using the variances and covariances of the classes to calculate the probability the cell lies in each class and adopting the label of the class with the highest probability. Using this mesh, the decision boundaries can be plotted as the contours of the boundaries where cells are classified as one cluster over the other. Figure 2 shows Maximum-Likelihood classification (MLC)

against NCC decision boundaries. MLC boundaries are not linear and therefore can fit the data better, however in this case both MLC and NCC classifiers classify the test data points in the same way.

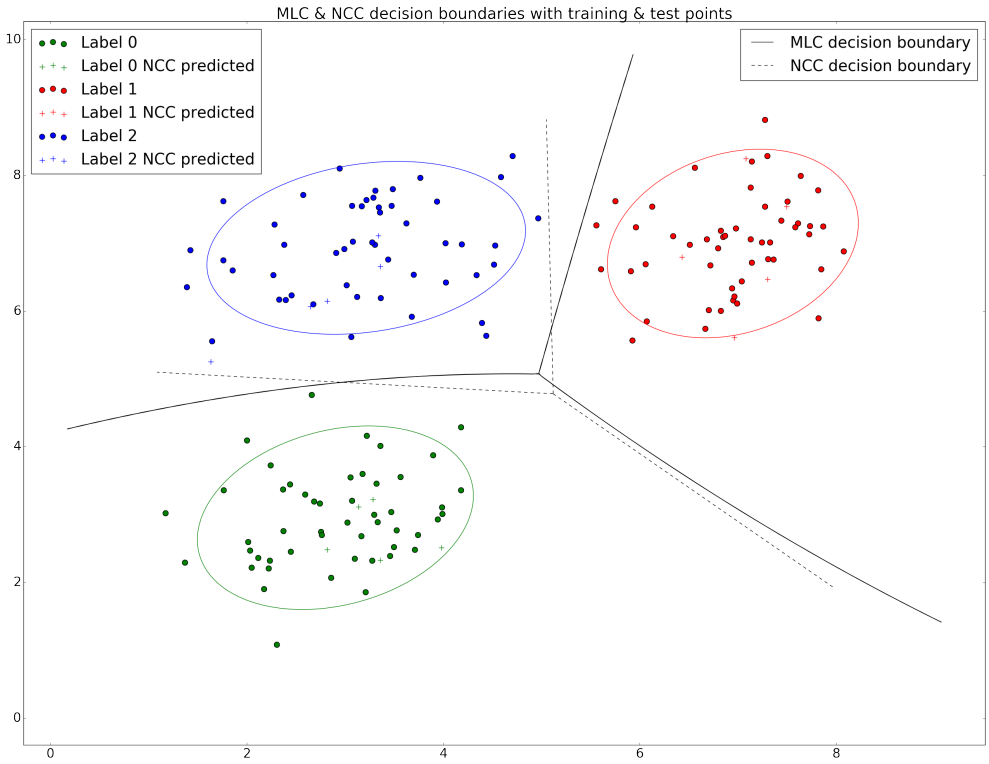

Figure 3 MLC & NLC decision boundaries with 95% probability mass

5.2. **95% Probability mass**. In order to plot a contour which contains 95% of the probability mass we simple create a mesh and calculate the pdf of each cell for each cluster, then plot the contour where p = 0.05.

5.3**. MLC emulate NCC**. The task asks to make the MLC decision boundaries the same as the NCC decision boundaries, this can be achieved by setting the variance-covariance matrix equal to the identity matrix and such the covariance matrix is now symmetric and therefore the likelihood is the same as the Euclidean distance.

5.4. **MLC twice as likely**. The task asks how the decision boundaries would change if we knew one of the classes was twice as likely as the other two. To do this we assumed class 1 was twice as likely and multiplied its covariance matrix element-wise by 2, reapplying MLC and plotting the decision boundaries is shown in Figure 4.

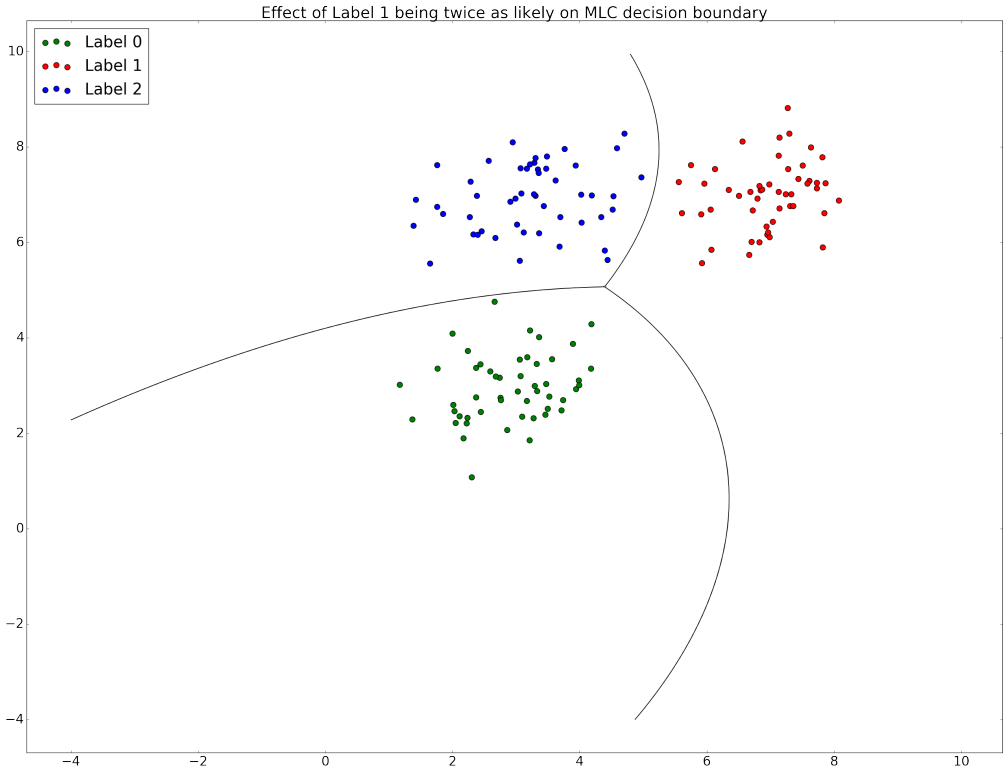

Figure 4 Label 1 twice as likely

6. Discussion of results

Not applicable as I completed this coursework individually and therefore have no other results to compare and contrast.