Assignment 1 – Linear Discriminant Analysis

Question 1.

Using the comma separated multivariate data in the file fld.txt:

(a) determine the discriminant line found by Fishers Linear Discriminant.
(b) Plot both the data and the discriminant line on a scatter plot
(c) Using this line, determine the class of each of the data points in the training dataset, assuming that the threshold is 0 (i.e. positive values are in one class and negative values in the other).
(d) Determine what percentage of data points are incorrectly classified.

NOTE: The first 2 columns in fld.txt are data columns. The third column is the class to which each data point belongs.

Solve this problem in python using the implementation that I gave you in class AND the scikit learn (sklearn) library routine called discriminant_analysis.LinearDiscriminantAnalyis. In your code, display the parameters for the resulting discriminant line (i.e. the slope and the intercept) for both my code implementation and the sklearn implementation. To do this you must reconcile the difference in the parameters of the discriminant line for my representation and the parameters of the discriminant line as determined by sklearn. They are NOT the same – however when interpreted correctly the mean exactly the same thing. The discriminant line position and the classification results for both implementations should be almost identical, minus small differences in precision.

Question 2.

Using the multivariate data in the file spam.xlsx, determine the discriminant line found by Fishers Linear Discriminant. Using this line, determine the class of each of the data points in the training dataset, assuming that the threshold is 0.

NOTE: The first 57 columns in spam.xlsx are data columns. The 58th column is the class to which each data point belongs.

Determine what percentage of the training data points are incorrectly classified by your classifier. You will notice that most of the data in the first class is classified correctly while the data in the second class is not. Therefore, it makes sense to adjust the threshold. Try the classification again a few times while adjusting the threshold so that it is a small negative number to see if you can improve the overall classification error rate (percentage of errors in BOTH classes).

FYI, information about the dataset is given in the folder with the spam dataset. This is a real dataset, from the Machine Learning Repository at UCI (University of California Irvine). I simply made it a bit smaller by only using the first 500 data points from the first class and the first 500 data points from the second. The original database has over 4000 data points and it was awkward to work with that in Excel. I have not, however, reduced the number of attributes. For those that are interested in a real-life example, this is one. It is a dataset with attributes used to try to detect spam from email.

Perform the above analysis using only my implementation in Python and report the minimum error you could obtain by adjusting the threshold (you could use a loop to search for the optimal threshold value if you like but this doesn’t need to be the exact “minimum” error for the purposes of this assignment). For that minimum error value of the threshold, report the confusion matrix when classifying the training data.