DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Homework Assignment No. 01:

HW No. 01: Title of the Assignment

submitted to:

Professor Joseph Picone
ECE 8527: Introduction to Pattern Recognition and Machine Learning
Temple University
College of Engineering
1947 North 12th Street
Philadelphia, Pennsylvania 19122

January 22, 2024

prepared by:

Leo Berman
Email: leo.berman@temple.edu

A. DESCRIPTION OF THE TASK

The point of the first task is to generate points in a Guassian distribution around the four coordinates (1,1), (1,-1), (-1,-1), and (-1,1) and classify them into classes one, two, three, and four respectively. These points are generated using a Guassian distribution with a covariance matrix that indicates the points are independently correlated with a deviation of 0.1.

After these points were generated, the JMP analysis software was used to debug this and verify that these points didn't have any correlation to each other and that they were able to be put on a decision-making plane that would have a misclassification rate of 0% when using the Naive Bayes predictive modelling in JMP.

In order to generate this data I used the NumPy python library's built-in function for generating multidimensional Gaussian coordinate points using covariance matrixes:

```
# Have to execute command for each class changing the mean for the
# different coordinates mentioned above
```

cov_1 = $\begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$ typed as a numpy array

```
# number_elements = 100 # (global variable that changes all execution
# of the command for each class and covariance matrix
numpy.random.multivariate_normal(mean, cov_1, number_elements)
```

B. DESCRIPTION OF THE TASK

The goal of the second task was to introduce error to the same type of JMP analysis using the Naive Bayes predictive modeling. This is done by repeating all the steps above, but changing the covariance matrix's deviation numbers from 0.1 to 1.

```
# Have to execute command for each class changing the mean for the
# different coordinates mentioned above
```

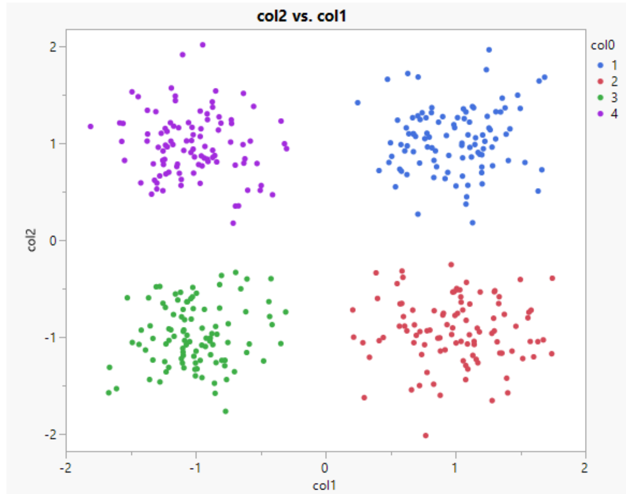
cov_2 = $\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ typed to a NumPy array

```
# number_elements = 100 # (global variable that changes all execution
# of the command for each class and covariance matrix
numpy.random.multivariate_normal(mean, cov_2, number_elements)
```

C. SUMMARY

Task 1:

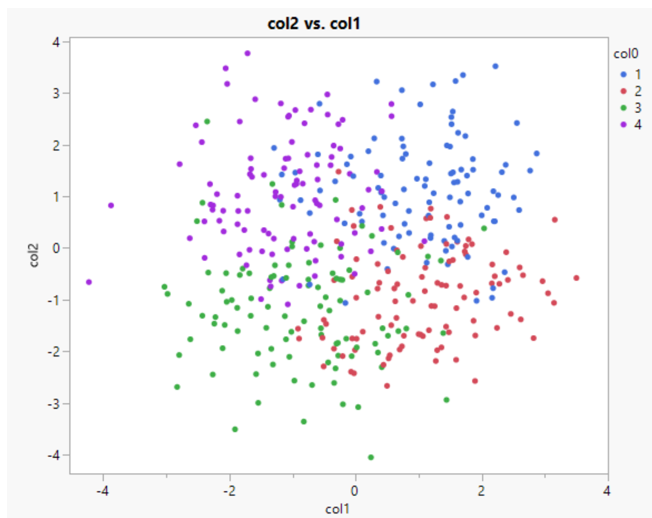
When plotted, the points that had a covariance matrix with less deviation (.1) looked clumped together and clear boundaries would have been easily drawn:



The Naive Bayes predictive algorithm results were what was expected. A 0 % misclassification rate. This means that JMP was successfully able to identify all the points without fail.

Task 2:

When plotted, the points that had a covariance matrix with more deviation (1) had significantly more spread out data that was intersecting across where the previous decision plane's boundaries would have been drawn:



The Naive Bayes predictive algorithm had a misclassification rate of 28.25% and falsely correlated points with each other which can be observed by looking at the confusion matrix from JMP assessing the training data as evaluation data.

This assignment gave me a basic understanding of how decision-making surfaces are drawn up using the Naive Bayes predictive classification algorithm in JMP and also a basic understand of confusion matrix's and misclassification rates. In order to get a more accurate assessment of how the algorithm performs, it would be essential to have training data that is completely separate from the evaluation data. This would give us a more neutral idea on how the Naive Bayes predicitive classification algorithm actually performs.