# ELEC 425 - Assignment 1

## 1 – Marginal and Conditional Numeric Distributions

### Calculate and print out (or copy down) the following distributions. (Only show up to 4 decimal places)

1. *The marginal probability vector pC(x1)*

The output of the marginal probability vector can be found below in Figure 1, while it’s code can be found below in Figure 2. The sum of the vector is 1, which validates the calculated values. To calculate it, *x2*, *x3*, and *x4* are summed out.





Figure 2 - Code for calculating marginal probability vector pC(x1)

Figure 1 - Marginal probability vector pC(x1)

1. *The conditional probability table pA(x3, x4 | x1 takes its third value)*

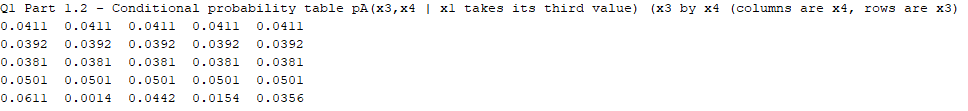
**The conditional probability table of *pA(x3, x4 | x1 = 3)* is stored in the variable *pA\_x3\_x4\_given\_x1\_3rdValue* and can be found below in Figure 3. Its code can be found below in Figure 4. The sum of the table is 1, which validates the calculated values. To calculate it, *x2* was summed out, and *x1* was set to 3.

Figure 4 - Code for calculating conditional probability table

Figure 3 - Conditional probability table pA(x3, x4 | x1 takes its third value)

1. *The conditional probability vector pB(x4 | x2 takes its first value)*

The *conditional probability vector pB(x4 | x2 = 1)* is stored in the variable *pB\_x4\_given\_x2\_1stValue* and can be found below in Figure 5. Its code can be found below in Figure 6. The sum of the table is 1, which validates the calculated values. To calculate it, *x1* and *x3* were summed out and *x2* was set to 1.



Figure 5 - Conditional probability vector pB(x4 | x2 takes its first value)



Figure 6 - Code for calculating conditional probability vector

### For each distribution in {A,B,C} and each of the following statements, say whether

### the statement applies.

1. *x1 is conditionally independent of x2 given x3* for distribution B, but not for distributions A or C. The output of the code for this question can be found below in Figure 7 while the code itself can be found below in Figure 8.

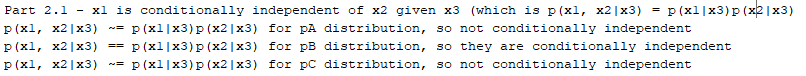


Figure 7 – Output of code for Question 1, part 2.1

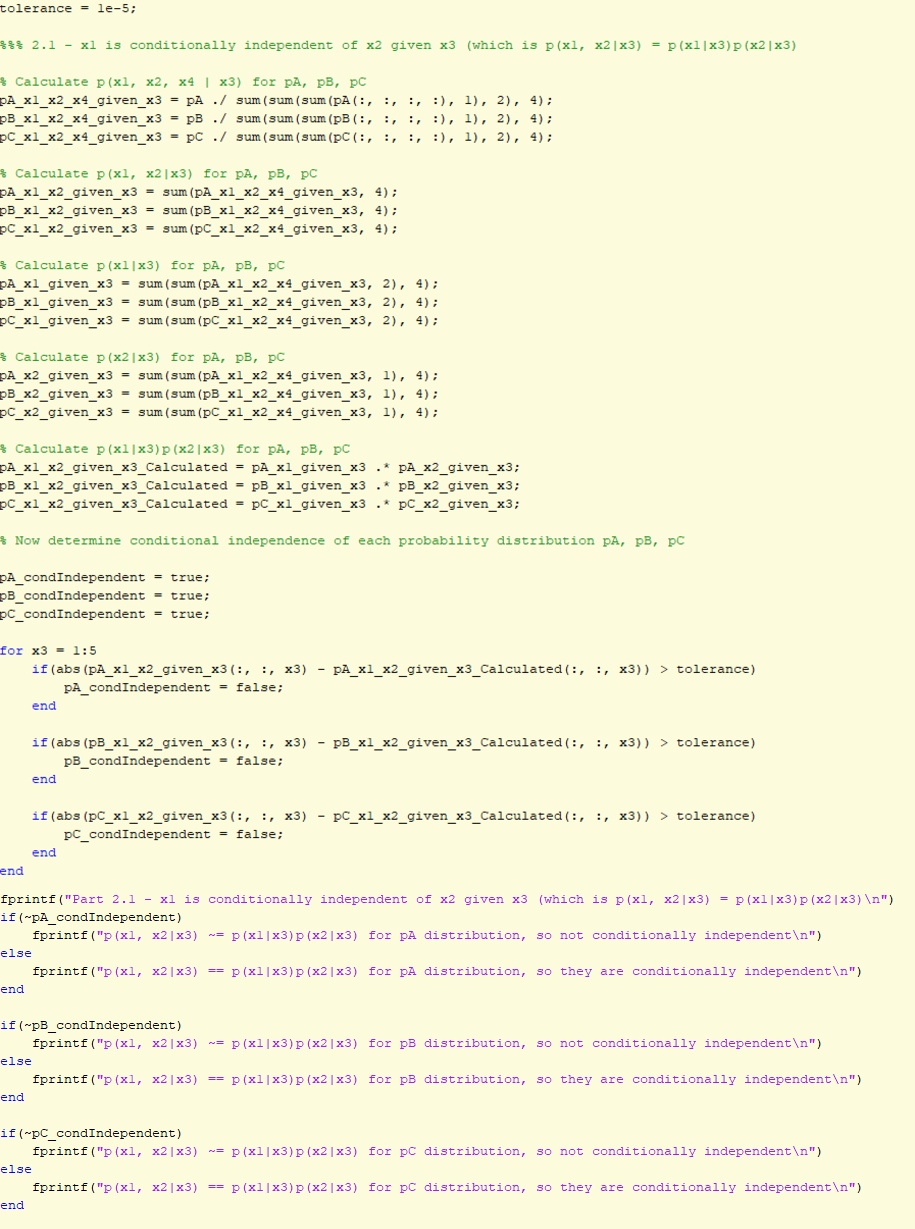


Figure 8 - Code for Question 1, part 2.1

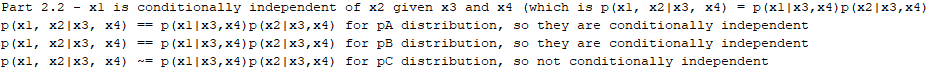
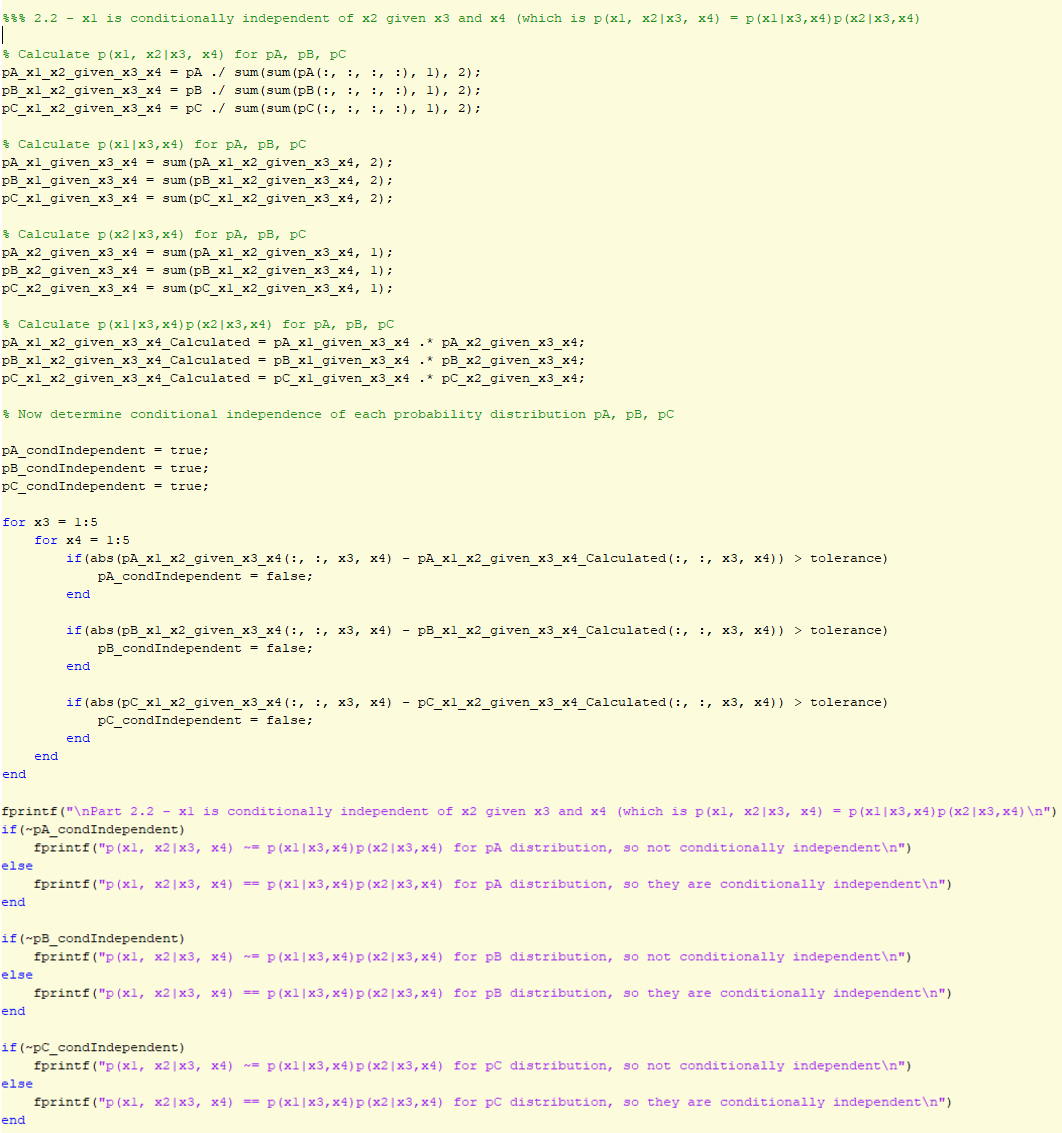
1. *x1 is conditionally independent of x2 given x3 and x4* for distributions A and B, but not for distribution C. The output of the code for this question can be found below in Figure 9 while the code itself can be found below in Figure 10.

Figure 10 - Code for Question 1, part 2.2

Figure 9 - Output of code for Question 1, part 2.2

1. *x1 is marginally independent of x2* for distributions A, B and C. The output of the code for this question can be found below in Figure 11 while the code itself can be found below in Figure 12.

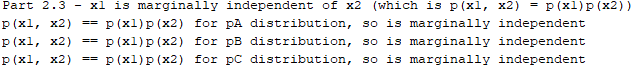
**

Figure 11 - Output of code for Question 1, part 2.3

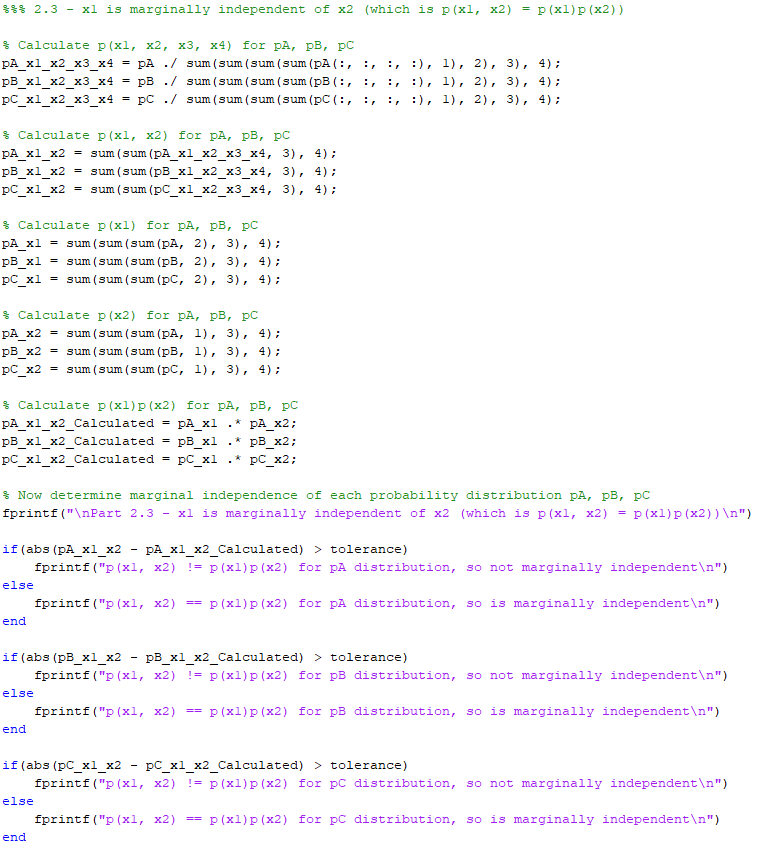
**

Figure 12 - Code for Question 1, part 2.3

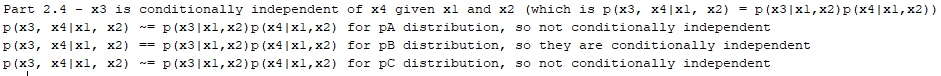
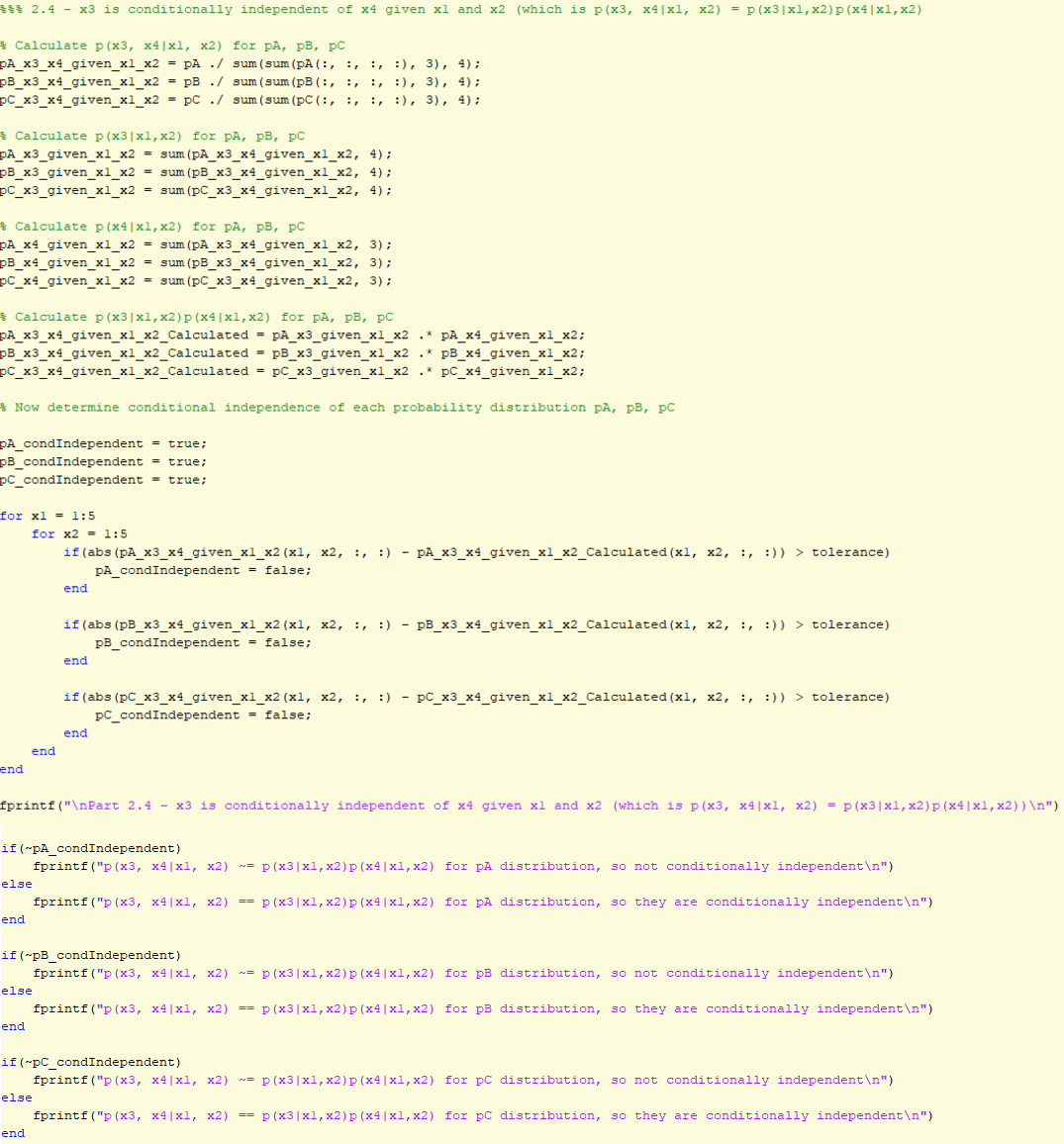
1. *x3 is conditionally independent of x4 given x1 and x2* for distribution B, but not for distributions A or C. The output of the code for this question can be found below in Figure 13 while the code itself can be found below in Figure 14.

Figure 14 - Code for Question 1, part 2.4

Figure 13 - Output of code for Question 1, part 2.4

## 2 – Training Conditional Gaussian Classifiers

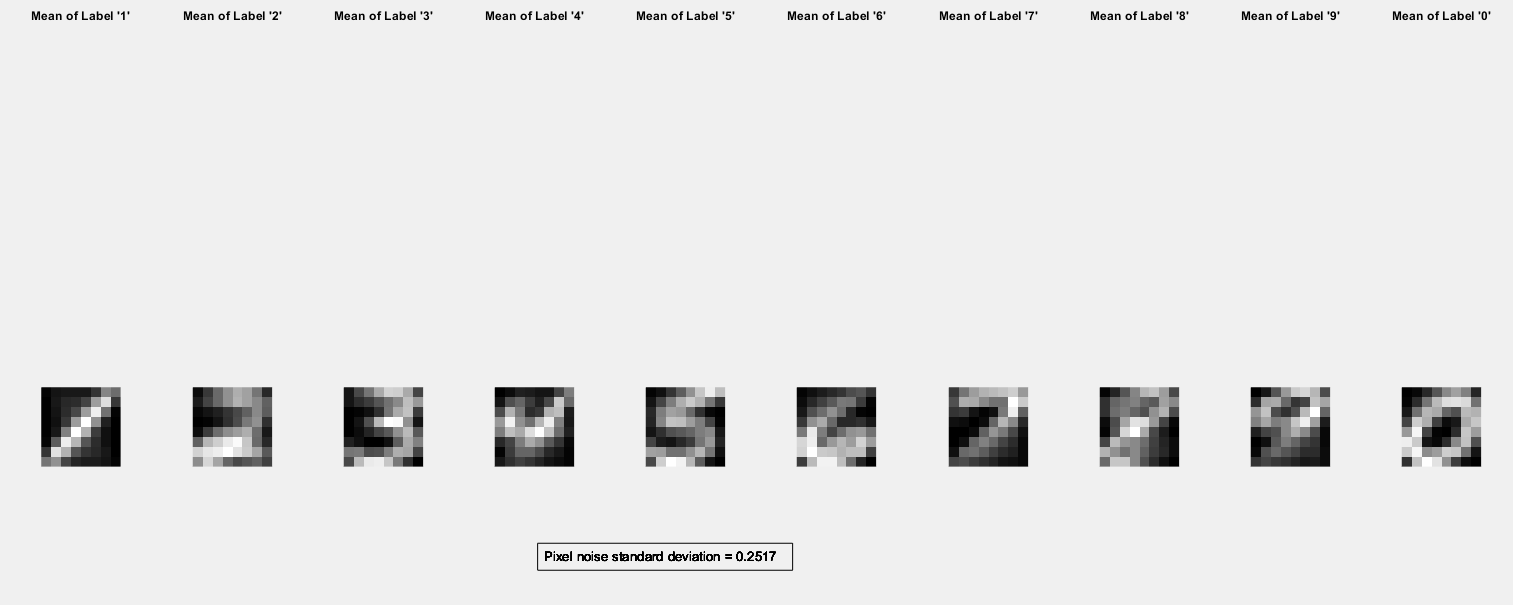
The plotted output of the trained Conditional Gaussian Classifier can be found below in Figure 15. As shown on the plot, the pixel noise standard deviation is roughly 0.2517. The function that trains the Conditional Gaussian Classifier (*trainConditionalGaussian())* can be found below in Figure 16, while the script that plots the output can be found below in Figure 17. The *ûki* parameters are stored in *mean\_featureI\_classK* in both the function and plotting script code.

Figure 15 - Plot of handwritten digits from Conditional Gaussian Classifier

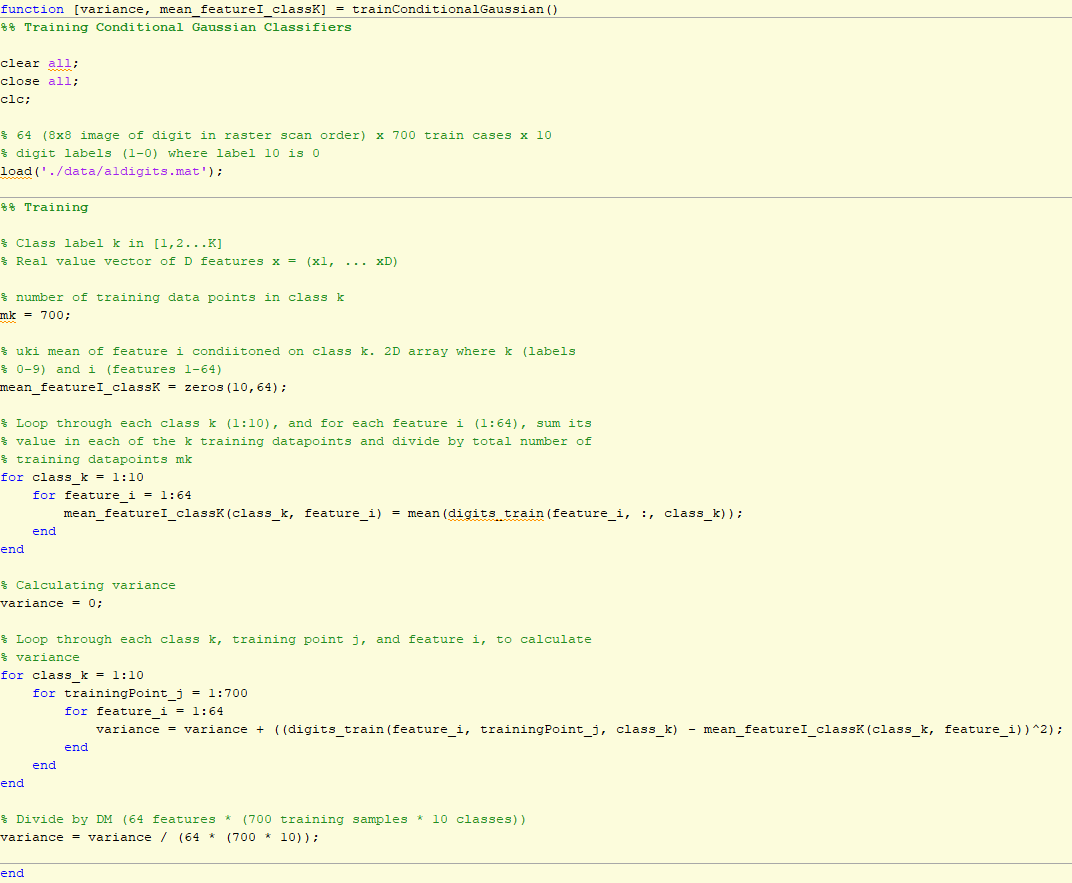


Figure 16 - Training function code for Q2 Conditional Gaussian Classifier

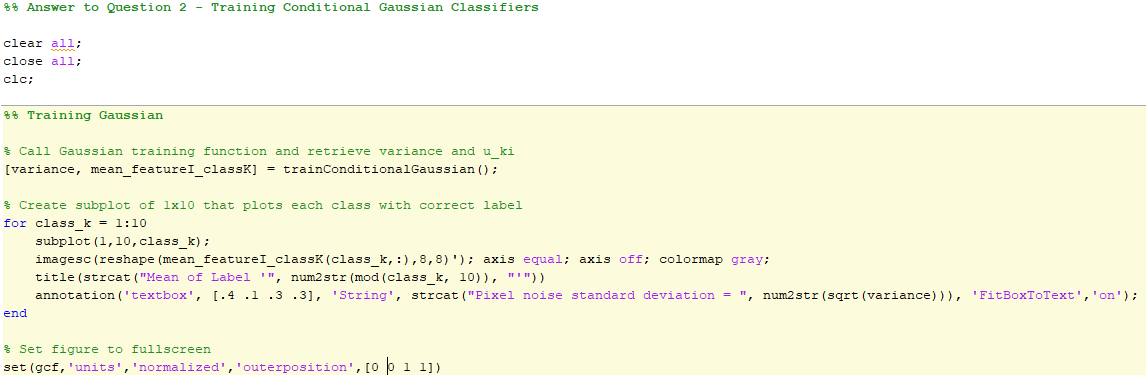
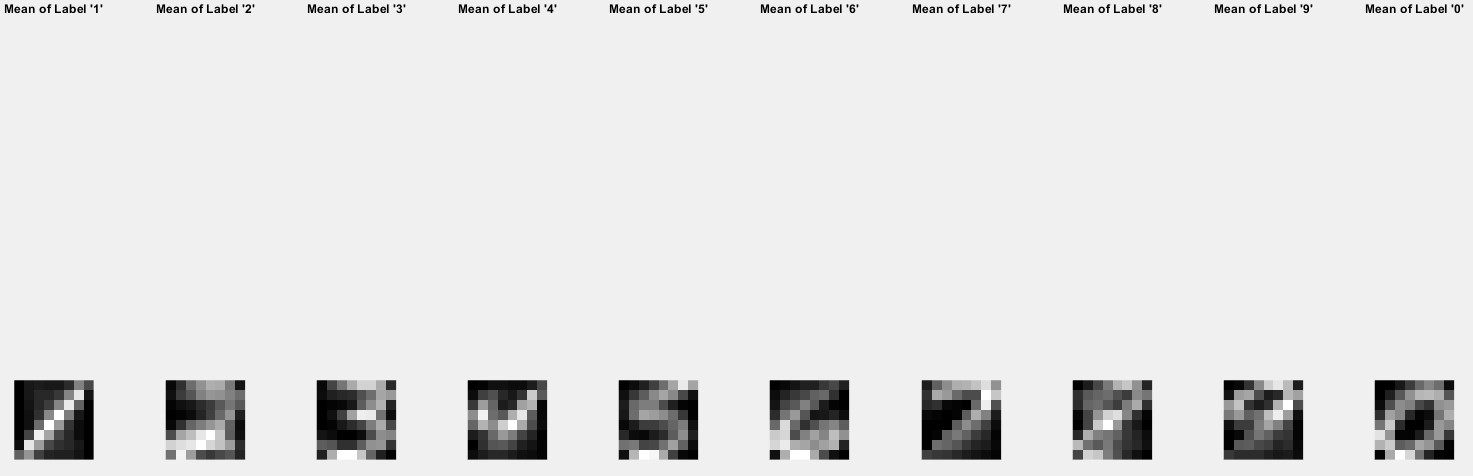


Figure 17 - Plotting code for Q2 Conditional Gaussian Classifier

## 3 – Training Naïve Bayes Classifiers

The plotted output of the trained Naïve Bayes Classifier can be found below in Figure 18. The function that trains the Naïve Bayes Classifier (*trainNaiveBayes()*) can be found below in Figure 19, while the script that plots the output can be found below in Figure 20. The *nki* parameters are stored in *mean\_featureI\_classK* in the function and plotting script code.



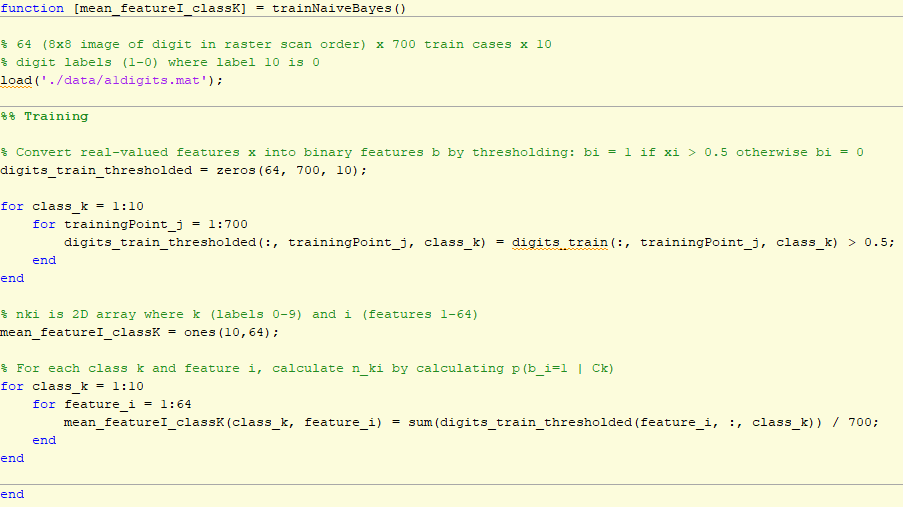


Figure 19 - Training function code for Q3 Naive Bayes Classifier

Figure 18 - Plot of handwritten digits from trained Naive Bayes Classifier

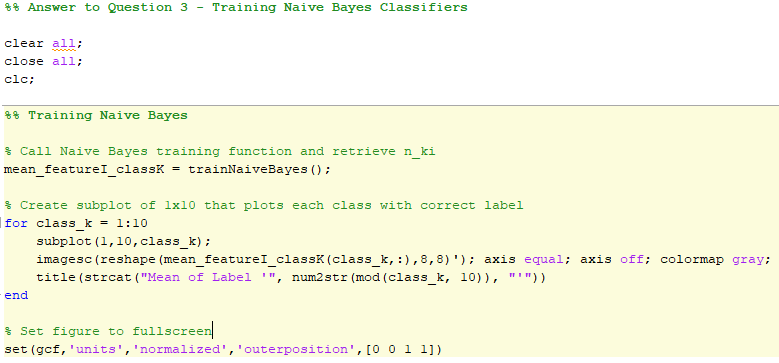


Figure 20 - Plotting code for Q3 Naive Bayes Classifier

## 4 – Test Performance

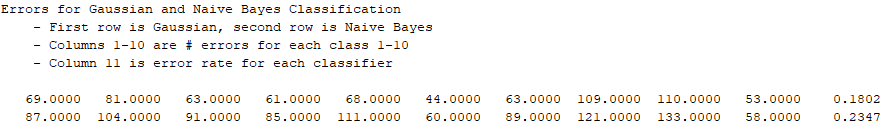
The output of the test performance of both the Conditional Gaussian and Naïve Bayes Classifiers can be found below in Figure 21. From the output, we can see that the Conditional Gaussian Classifier achieves an error rate of ~18.02% while the Naïve Bayes Classifier achieves and error rate of ~23.47%. The codes that tests the Conditional Gaussian Classifier can be found below in Figure 22, while the code that tests the Naïve Bayes Classifier can be found below in Figure 23.

Figure 21 - Error rate for Conditional Gaussian and Naive Bayes Classifiers

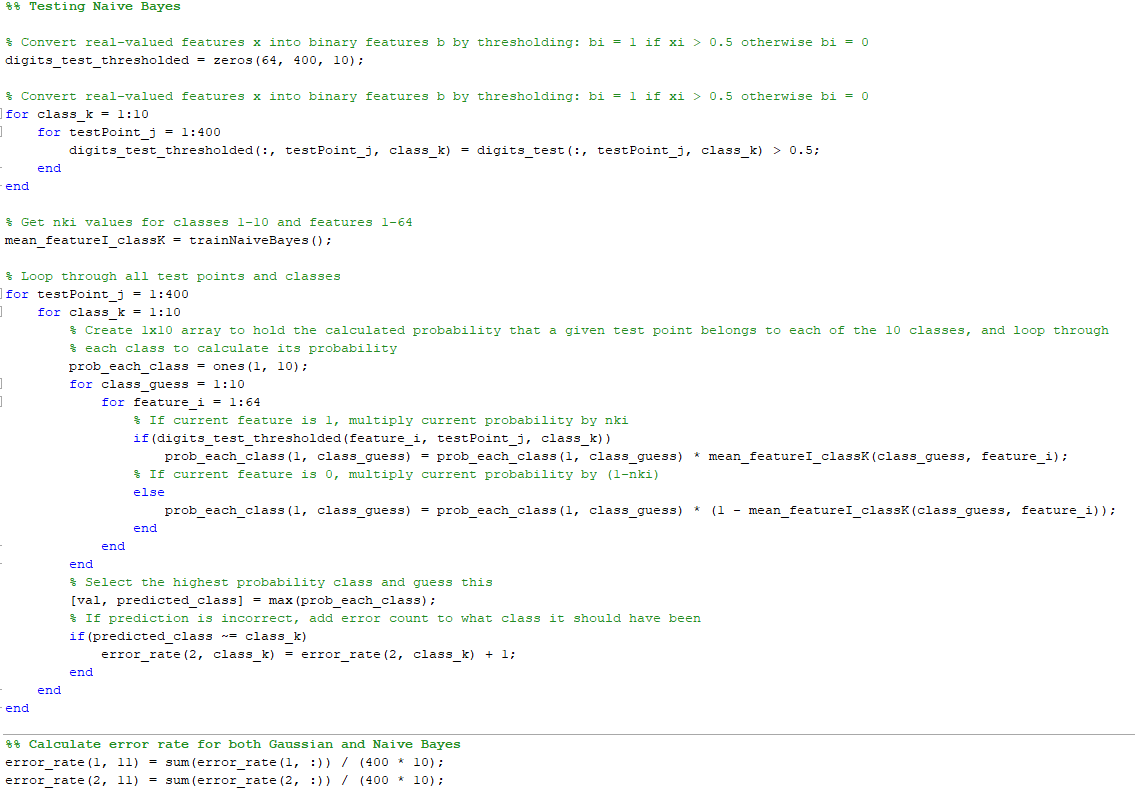


Figure 23 - Code to test Naive Bayes Classifier in Q4

Figure 22 - Code to test Conditional Gaussian Classifier in Q4