Academic honesty and plagiarism

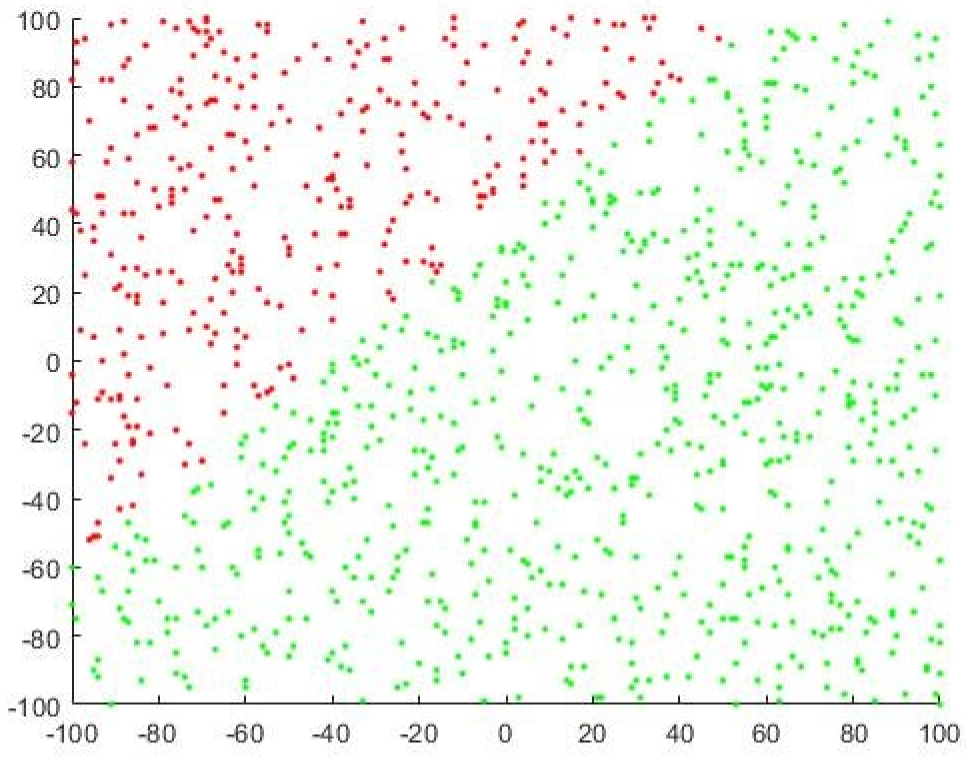
- Plagiarism is unacceptable.
  - Code you submit must be your own. No copying, adapting, or submitting code you did not create is allowed. Working together and presenting variants of the same file is not acceptable. Here are some specific guidelines to make sure you don't cross the line:
    - Do not exchange programs or program fragments in any form on paper, via e-mail, photos, or by other means.
    - Do not copy solutions from any source, including the web or previous quarters' students.
    - Do not discuss code with other students at the level of detail that will lead to identical programs or program fragments.
  - Ask me if you are uncertain.
  - **All violations of the academic honesty will be immediately referred to the dean's office.**
- **Plagiarism detection will be applied to all code. It is very good and will catch you!! (20% of a class referred to dean!!)**

Lab 8: Classification using perceptron, Due June 2, 7pm.

A security researcher wanted to see if they could figure out the password entered in a smart-phone just by looking at the accelerometer data from the phone. They expect that pressing the phone in different positions (corresponding to pressing different numbers on the screen) would cause different signals on the accelerometer. The team gathered training data, where each data point is the x,y values sensed by the accelerometer and the label is the number pressed. Your job is to build a classifier using a perceptron that can classify the button pressed by looking at just the accelerometer data.

Please use python3 and don't import any extra files. Each call of the student function must take under 60 seconds to execute.

Problem 1:
To verify you can build a classifier, the researcher asked that you first build a binary classifier that can classify 2D data that is linearly separable. To complete this task you are to make a function that takes in some labeled training data (data is x,y values and each data is labeled as class 0 or 1), and your function is to classify the test data given. The team wants your classifier to be correct more than 95% of the time on the test data. An example data set is shown below.



For this part you are to complete one function: def part_one_classifier(data_train,data_test):

The function receives:

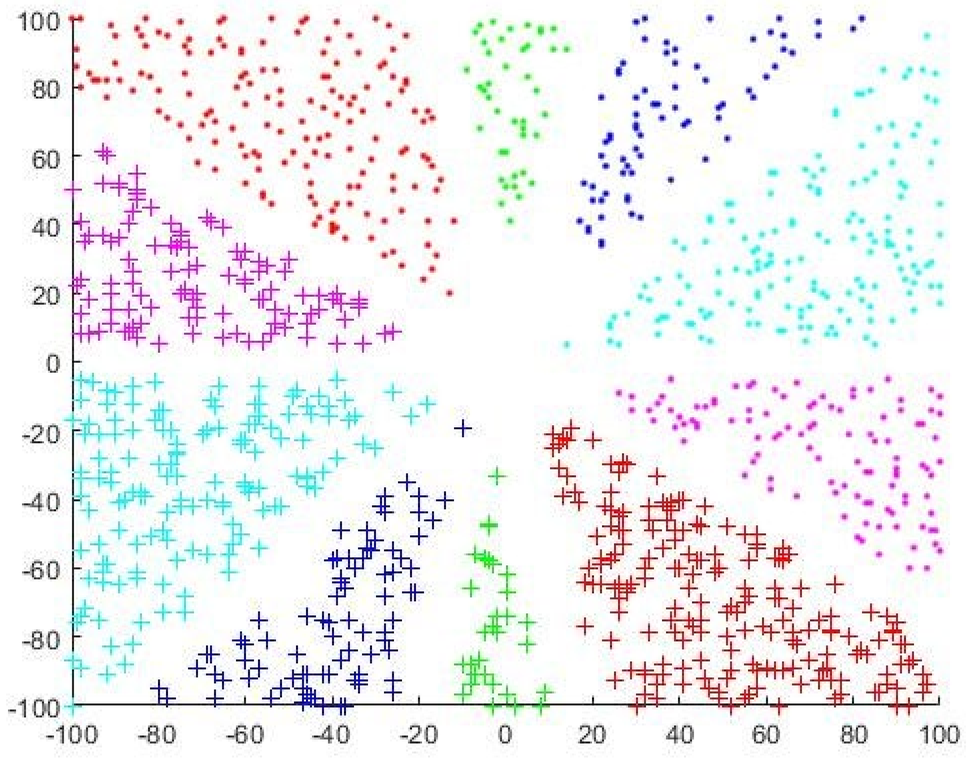
- bidimensional structure data_train of size TRAINING_SIZE x 3. Every row contains a value for X in position 0, a value for for Y in position 1 and a value for the class in position 2.
- bidimensional structure data_test of size TEST_SIZE x 3. Every row contains a value for X in position 0, a value for for Y in position 1 and an empty space for the class in position 2.

The function must modify:

-The third column of the "data_test " structure, by entering the right class of each element. Valid values for classes are 0 or 1.

Hint: for problem 1, you may want to use a bias feature.

Problem 2:
Now the researcher give you the training data collected from the phone, which for each data point it is the x,y accelerometer values, and the button pressed (0-9). Next they give you test data, just the x,y accelerometer values sensed when a button was pressed. You are to write a function that takes in the training data and test data, and correctly classifies the test data. The researcher wants your classifier to be correct more than 95% of the time. Example data is shown below. All data for problem 2 will have a decision boundary that goes through the origin.



For this part you are to complete one function: def part_two_classifier(data_train,data_test):

The function receives:

- bidimensional structure data_train of size TRAINING_SIZE x 3. Every row contains a value for X in position 0, a value for for Y in position 1 and a value for the class in position 2.
- bidimensional structure data_test of size TEST_SIZE x 3. Every row contains a value for X in position 0, a value for for Y in position 1 and an empty space for the class in position 2.

The function must modify:

-The third column of the “data_test ” structure, by entering the right class of each element. Valid values for classes are 0 to 9.