Assignment 1
Perceptron

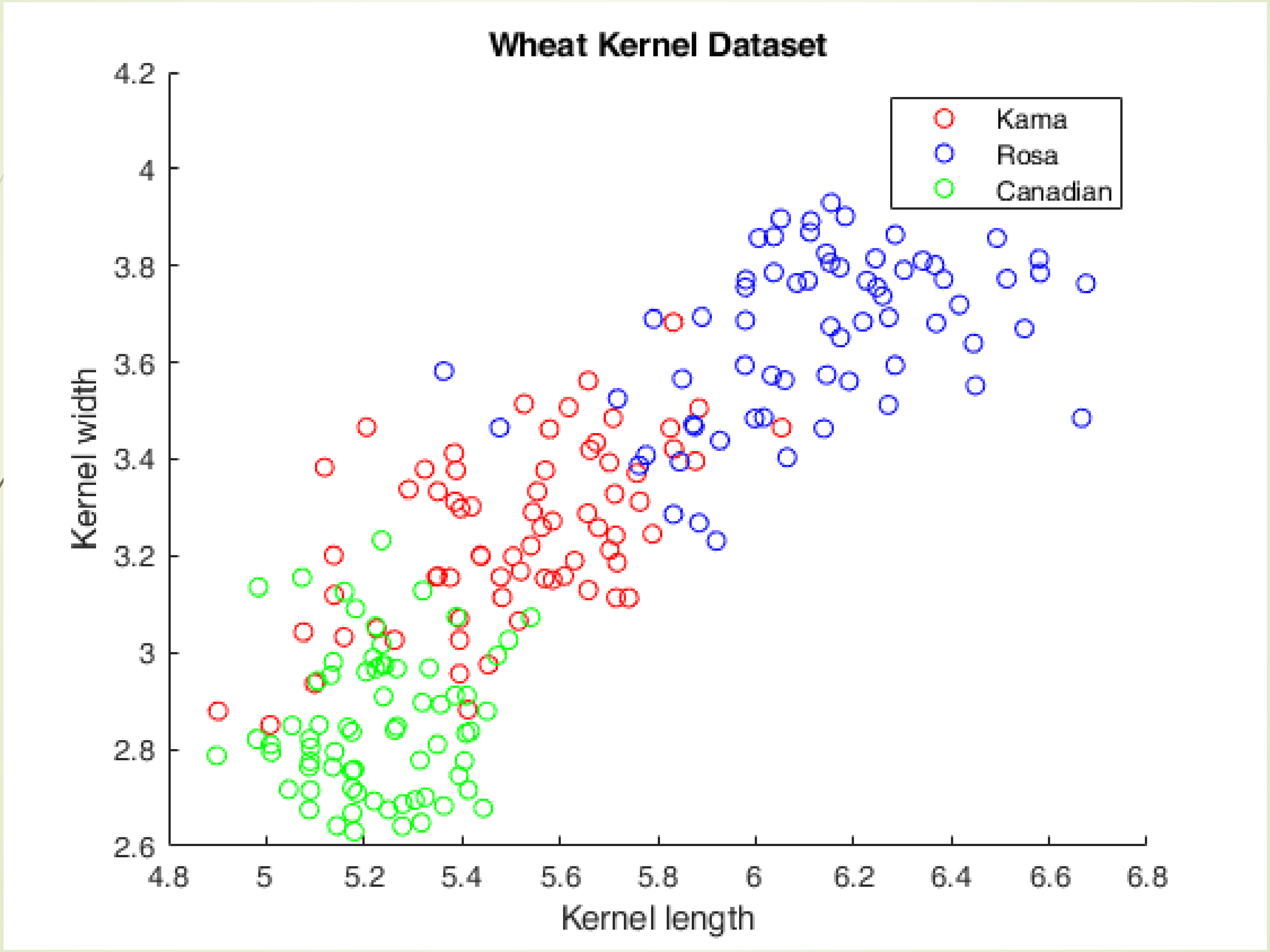
Farhana Zulkernine

General Instructions for Code and Submission (for all assignments)

1. You can use any programming language (preferred python, C, C++, Java, or Matlab)
2. Make **one zip file named as Asgn1_studentID**.
3. **Upload zip file** to the OnQ site.

Assignment 1: The Data

- The dataset for the first assignment is comprised of 3 different wheat varieties
- The data used is real world data, collected through measurements of the wheat kernels using a soft x-ray technique
- The measurements(attributes) included in this dataset with respect to each individual kernel are: area, perimeter, compactness, length, width, asymmetry coefficient, and length of kernel groove.
- The three types of wheat (classes) are: 1 = Kama, 2 = Rosa and 3 = Canadian
- The following slide shows the relationship between the kernel length and width
  - Observe that the data points are not linearly separable.



Wheat Kernel Dataset
Kama
Rosa
Canadian
Kernel width
Kernel length
4.2
4
3.8
3.6
3.4
3.2
3
2.8
2.6
4.8
5
5.2
5.4
5.6
5.8
6
6.2
6.4
6.6
6.8

Assignment 1 – Data and Assumptions

- Each line in the CSV files represents a separate data point where each data point includes the following comma separated values as dimensions and the last column shows the class label:

area, perimeter, compactness, length, width, asymmetry coefficient, length of kernel groove, class

Example: 15.03, 14.77, 0.8658, 5.702, 3.212, 1.933, 5.439, 1

- The data is provided as two files: trainSeeds.csv and testSeeds.csv.
- TrainSeeds.csv contains 55 data points for each class. Use this to train your perceptron.
- TestSeeds.csv contains 15 data points for each class. Use this to test that your perceptron works correctly.

Assignment 1 – Your Task
Total 10 marks

1. Design and train a perceptron using the data in TrainSeeds.csv to predict the three classes of the wheat data using simple feedback learning. Use TestSeeds.csv to test the performance of your ANN in terms of precision and recall.
2. Submit the program code **with comments**, associated libraries (and/or executables and instructions for execution) and data so that the TAs can execute it (otherwise you lose points). (**3 marks** for executable fully functional code with correct learning algorithm)
   - Once your ANN has been trained and tested, classify all data points in the given order using your ANN. Implement a function in your program to create an output text file to log the original and the predicted class values for all data points. Submit the text file. (**1 mark**)
   - Add the following to the text file (manually or programmatically):
   1. The initial and final weight vectors. (**0.5 marks**)
   2. Total number of iterations used for training and the terminating criteria used. (**0.5 marks**)
   3. Calculate precision and recall of your classification and the confusion matrix. (**2 marks**) (see the last slide)

Use a tool - 3 marks

- Use a tool of your choice that has built in libraries to design and train the same perceptron e.g. Matlab.
  - Feed the data (may need to combine the data first).
  - Generate the results in terms of precision and recall.
  - Calculate precision and recall from the ANN you coded.
- Submit:
  - Compare the precision and recall values of the two ANNs (tool-based and your code) in the text file. (**1 mark**)
  - Proof of your implementation: Add screen shots (for GUI based tools) or codes as applicable of the tool-based ANN and the results in a doc or pdf file and add that to the final zip file. (**2 marks**)

Precision/Recall/Confusion Matrix for Separate Classes

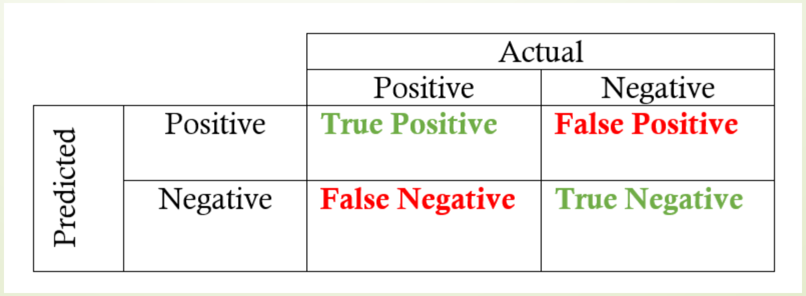
- Precision $= \dfrac{\text{True Positive}}{\text{True Positive + False Positive}} =$

$$\frac{\text{\# of points correctly classified as class N}}{\text{\# of points correctly classified as N + \# of points incorrectly classified as class N}}$$

- Recall $= \dfrac{\text{True Positive}}{\text{True Positive + False Negative}} =$

$$\frac{\text{\# of points correctly classified as class N}}{\text{\# of points correctly classified as N + \# of class N points incorrectly classified}}$$

- Confusion matrix



		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative