- **Target:** Implement multi-layer perceptron and analyze the results
- **Dataset:** The dataset (points2d.dat) is attached to the email. The sample includes 400 instances. Each instance is represented by a feature vector of 2 dimensions. The third column corresponds to the class (0/1/2) of the instance.

1. (1 pt) Report how you split the sample for training and test. Report how you designed your neural network to solve this 3-class problem (details of output units, error function, etc.).
2. (2 pt) Derive weight update equations using hyperbolic tangent activation function. Submit a hard-copy of this derivation, drawing the the corresponding neural network with hidden units. All the formulas and the corresponding symbols should be clear. See pg. 24, W09.pdf.
3. (6 pt) Implement multi-layer perceptron with 5 hidden units.
   - Plot the error on training and test sets. (see pg.37 W09.pdf). Describe how error is defined in 3-class problem.
   - After training converges:
     - Plot the validation/test set and show true positives, true negatives, false positives, and false negatives (with different markers, colors, etc.)
     - Plot the decision boundaries (bonus: 1 pt)
4. (0.5 pt) First submission bonus.

- **Submission:**
  - Create an archieve named **student-id.tgz** with the contents (below) placed in a directory named **student-id/**. Send the archieve file through email (subject:**hw3 submission student-id**) to the instructor.
  - Try to put all the source code into a single file named **student-id.x**. Feel free to use matlab/octave/python/R. The source code should be properly commented. Use of built-in functions are limited to the very basic ones. You need to implement the rest yourself. Copy/paste from internet will be considered as cheating.
  - The report should be named as **student-id.pdf**.
  - For simplicity, derive the formulas by hand on a paper, and submit the derivations and other required information (structure of the neural network, symbols correspondences, etc.) at class.