Problem Set 3
Due Monday, May 2, 2016 at 11:55pm

How to Submit

Create one .zip file (**not** .rar or something else) of your code and written answers and submit it via `ilearn.ucr.edu`. Your zip file should contain the following 5 files

- `learnperceptron.m`
- `problem3.m`
- `multiclasslogreg.m`
- `multiclasseval.m`
- `ans.txt` (your written answers to the non-programming parts)

plus any other matlab functions your code depends on that you wrote.

Each file should include at the top (in comments if necessary)

- Your name
- Your UCR student ID number
- The date
- The course (CS 171)
- The assignment number (PS 3)

Note: Do not use any Matlab function that is in a toolbox for this problem set.

Problem 1. [15 pts]

Implement the perceptron learning algorithm. Your function should have the signature `function w = learnperceptron(X,Y,n,eta,w0,animate)` with the following parameter meanings.

1. `w`/`w0`: `w0` is the starting weight vector. `w` is the returned final weight vector. If `X` is m-by-n, then these vectors are n-by-1.
2. `X`: This is the input data. Assume that it has already been augmented by adding any additional features (columns), including the "constant feature" in the first column.
3. `Y`: This is the target output. Elements are either $+1$ or -1.
4. `n`: This is the maximum number of sweeps through the entire dataset the algorithm should do before stopping. The algorithm should, of course, stop earlier if possible.
5. `eta`: This is the learning rate (a scalar), the amount by which you multiply the change to be made to `w` on each iteration.
6. `animate`: This is either 0 or 1. If 1, each time the weight vector changes, plot the data (positive points as blue crosses and negative points as red circles) and the dividing line (you may use the supplied `drawline`). Your code should (if animating) then pause for 0.1 seconds (see the command `pause`) before continuing. You may assume that if `animate` is 1, the `X` input has three columns. The first column of `X` is all 1s. The second column is the first attribute. The third column is the second attribute.

Problem 2. [10 pts]

Try your code from problem 1 on the simple linearly-separable data in `simpleprob.dat`. The last column of this dataset is the class label (y). The first two columns are the attributes.

Try different starting weights (`w0`) and the learning rates (`eta`) and see the effect on the final classifier. Describe the relationship between these parameters of the algorithm and the result. What does this suggest to you about the perceptron learning algorithm?

Problem 3. [15 pts]

The supplied function `learnlogreg` implements logistic regression, including regularization (`lambda` specifies the degree of squared regularization). If `lambda` is non-zero, it assumes that the first feature is the special constant 1.

Write a function called `problem3` that tries logistic regression on the data in the file `phishing.dat`. This data consists of a set of features about websites' URLs (the first 30 columns) and whether the website is a phishing attempt. More information on the features can be found at `https://archive.ics.uci.edu/ml/datasets/Phishing+Websites` Because the features are discretized to 0 or 1 and are thus already standardized, they should not be normalized. Treat the entire dataset as training data.

Your code should load the data and try both linear and quadratic models (by augmenting the feature space). It should try different regularization strengths.

Have your code pick a particular set of features and regularization. Explain why you decided to do it this way.

Problem 4. [10 pts]

For this problem, use the iris dataset again (`iris.dat`). The supplied function `runp4` will load the data and split into training and testing and run your code.

You need to write `multiclasslogreg` to train logistic regression for this multiclass problem using one-versus-all.[1] You need to also write `multiclasseval` to find and return the number of errors on the testing set.

The return value from `multiclasslogreg` (and thus the first argument to `multiclasseval`) should be whatever you feel is necessary to specify this multi-class classifier. For the logistic regression, set `lambda` to 0.01 for the purposes of this problem.

[1] There is a non-binary generalization of logistic regression, but we'll pretend it doesn't exist for this problem set.