Problem Set 2

Due Monday, April 18, 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 four Matlab files

- `cvknn.m`,
- `knntest.m`,
- `cvridge.m`,
- `regerr.m`

plus any other matlab functions your code depends on.

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 2)

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

The supplied script `runps2.m` will run both problems. You may wish to comment out parts of it (or the other supplied scripts) until you have them working.

Problem 1. [30 pts]

Two classification data sets are included with this problem set: `iris.dat` (with which you are already familiar), and `vert.dat`. The latter consists of measurements about the vertebral column. In particular, the first six columns are the attributes (pelvic incidence, pelvic tilt, lumbar, lordosis angle, sacral slope, pelvic radius, and grade of spondylolisthesis). The last column is the category of patient (0 for disk hernia, 1 for spondylolisthesis, and 2 for normal). Just as for the iris data set, the examples have been randomly ordered. Do not further scramble them (so that we can easily check your solutions). For both data sets, you should use all attributes.

The supplied function `runknn` does four things:

1. loads and splits a data set into training and testing;
2. z-normalizes the training (and applies the same normalization to the testing);
3. splits the training into training and validation and runs cross-validation for k-nearest neighbor; and
4. checks the accuracy of the result on the testing set.

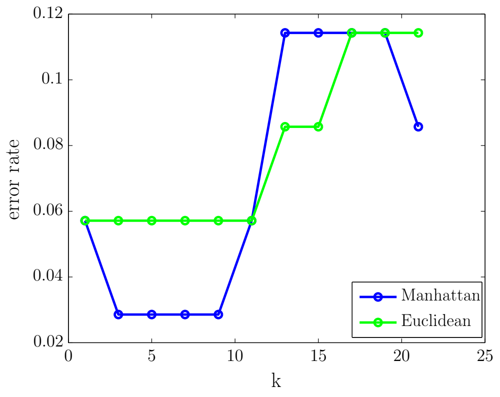
However, it is missing two critical pieces: `cvknn` and `knntest`. Your task is to supply these functions.

The former has signature `[k,lnorm] = cvknn(TrainX,TrainY,ValidX,ValidY,maxk)`. It should take in the training and validation data sets and the maximum k value to check. It should use cross-validation to select the best combination of k (among all of the odd values between 1 and maxk, inclusive) and distance metric (between either Euclidean or Manhattan distance). It should return the best pair. The returned `lnorm` should be 2 for Euclidean and 1 for Manhattan. *Additionally* your code should also draw a plot of the validation-set error rate versus k for both of the distance metrics (an example is shown below).

The latter function has signature `[err,C] = knntest(TrainX,TrainY,TestX,TestY,k,lnorm)` and takes in a training and testing set, along with k and the distance metric (encoded as above). It should return the testing error for k-nearest-neighbor, as well as a confusion matrix, C. A confusion matrix records how often a label of one type was classified as a different type. In particular, the i-j element is the fraction of time the label was reported to be i, when it was actually j.

As an example, on the Iris data set, the solutions report the text and plot below.

```
Iris:
chosen: k=3 and lnorm=1
testing error = 0.04
confusion matrix =
    0.4600         0         0
         0    0.2400         0
         0    0.0400    0.2600
```



Problem 2. [20 pts]

A regression data set is also supplied: `machine.dat`. In this data set the attributes (first six columns) are specifications of a computer: the machine cycle time in nanoseconds, the minimum main memory size in kilobytes, the maximum main memory size in kilobytes, the cache memory size in kilobytes, the minimum number of channels, and the maximum number of channels. The goal is to predict the last column, the relative performance of the computer.

For this problem you need to implement `cvridge.m` and `regerr.m`.

The former has signature `[w,lambda,besterr] = cvridge(X,Y,nfold,lambdas)`. It takes a training set, the number of folds for n-fold cross-validation, and vector of the lambda values to try. It uses n-fold cross-validation to select the best lambda for ridge regression, and it returns the weights associated with this lambda, the lambda value, and the cross-validation error for this best lambda value. The weight vector's length with be one greater than the number of attributes (the first term will be the bias term). *Additionally*, it plots the n-fold cross validation error as a function of lambda, on a semi-log plot (use `semilogx` instead of `plot`).

The latter has signature `err = regerr(w,X,Y)`. It takes a weight vector for linear regression and a testing set. It returns the average squared error using the weight vector `w` on the testing set.

As a check, you should be getting a testing error of about 8460.