Problem Set 5

Due on Friday, November 14, 2014 at 11:55 pm

How to Submit

Create one zip file (.zip) of your code and submit it via `ilearn.ucr.edu`. This zip file should have a single script named `ps5.m` that runs all of the examples. It should also contain the code necessary to run these samples. Finally, it should contain a single pdf file (ps5.pdf) that has your plots.

Do not supply any directories in your zip file. Each file should (in comments) list

- Your name
- Your UCR student ID number
- The date
- The course (CS 229)
- The assignment number (PS 5)

Radial Basis Function Networks or Kernel Linear Least Squares
part a.

Implement a procedure to train the weights for a radial basis function network, a function of the form $f(x) = \sum_i w_i K(c_i, x)$, using squared error as the loss metric. In class, the "centers" (or c_i points) we took just to be the original training set. However, in general, they could be any set of points. **NOTE:** even if the c_i points in the formula for f are not the training set, the loss function is still the squared error over all of the training data.

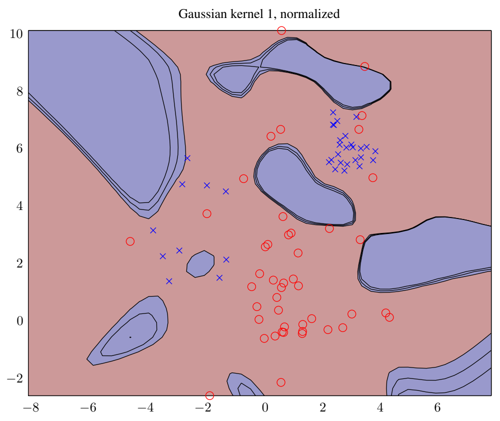
That is, if there are k centers and n training points, we want to minimize

$$\sum_{i=1}^{n}(f(x_i) - y_i)^2 = \sum_{i=1}^{n}(\sum_{j=1}^{k} w_j K(c_j, x_i) - y_i)^2$$

You do not need to add a regularizing term in this case. Notice, this does not exactly match the kernel linear least squares from class, because the set $\{c_i\}_i$ is not necessarily the same as the set $\{x_i\}_i$. However, you have seen a couple of examples of how to derive things like this, and in the end the kernels just serve as the features for a linear classifier, so you should be able to derive the formula for minimizing the loss function above.

You should be able to train either as above (unnormalized basis functions) or for normalized basis functions: $f(x) = \sum_i w_i K(c_i, x) / \sum_j K(c_j, x)$.

First, consider letting $\{c_i\}_i$ be the training set $\{x_i\}_i$ (as discussed in class). Use a "Gaussian" kernel: $K(c_i, x) = e^{-|x-c_i|^2/(2\sigma^2)}$. Train it on the "class2d" dataset from the previous problem set. (Treat the labels as regression values of 0 and 1 and threshold the output at 0.5 to make it a classifier.) Vary the width, σ^2 of the Gaussian kernel for values of 0.2, 1.0, and 5.0. Plot the resulting decision surfaces and the training data together for each of these three widths and for both normalized and unnormalized kernels. Place all 6 plots into a single figure by using `subplot`. For example, before drawing figure number i, call `subplot(2,3,i)`. Label your subplots. For example, one of the plots would look like



Note that the axes are "equal" (same aspect ratio — see the help for `axis`). This will make the results look better. Please do this.

You may find that your matrices for some of the examples as singular or "badly scaled." That's not necessarily a problem.

part b.
Now repeat the same test, but use k-means to select 16 centers from the original training x_i points. k-means was quickly described in class. Algorithm 14.1 in the textbook is more explicit (page 510). There is one tricky case: how to initialize the centers, and what to do with a center that "captures" none of the points. In both cases, set the center to the mean of a random selection of the points (so that it is somewhere reasonable).

Again, plot 6 different conditions on a single figure, using `subplot`. Again, label the plots. For these plots, also add the locations of the found centers as filled in black circles. If you have the centers in a matrix C ($k \times 2$), the following code will plot them as such

```
h = plot(C(:,1),C(:,2),'ko');
set(h,'MarkerFaceColor',[0 0 0 ]);
```

part c.
Can you draw any conclusions about how a good setting of the kernel width depends on the data points? What happens as the normalized kernel's width goes to zero? Demonstrate it by taking the limit of $K(x, x_i)/\sum_j K(x, x_j)$ for the Gaussian kernel as σ^2 approaches zero.

part d.
Perform the same plots as above, but for k-nearest neighbor. Draw 4 plots for differing values of k: 1, 3, 7, 15. Again, use a single figure with subplots and label the plots.