9.520 Problem Set 1

Due March 14, 2011

Note: there are six problems total in this set.

Problem 1 One common preprocessing in machine learning is to center the data. In this problem we will see how this can be related to working with an (unpenalized) off-set term in the solution. Consider the usual Tikhonov regularization with a linear kernel, but assume that there is an unpenalized offset term b,

$$\min_{w\in\mathbb{R}^d,\, b\in\mathbb{R}} \left\{ \frac{1}{n}\sum_{i=1}^{n} \big(\langle w, x_i\rangle + b - y_i\big)^2 + \lambda\|w\|^2 \right\}$$

and let (w^*, b^*) be the solution of the above problem.

For $i = 1, \ldots, n$, denote by $x_i^c = x_i - \bar{x}$, $y_i^c = y_i - \bar{y}$ the centered data, where $\bar{y}, \bar{x}$ are the output and input means respectively. Show that w^* also solves

$$\min_{w\in\mathbb{R}^d} \left\{ \frac{1}{n}\sum_{i=1}^{n} \big(\langle w, x_i^c\rangle - y_i^c\big)^2 + \lambda\|w\|^2 \right\}. \tag{1}$$

and determine b^*.

Problem 2 The distance between two elements $\Phi(x), \Phi(s)$ of a feature space induced by some kernel K can be seen as a new distance $d(x, x')$ in the input space. Show that such a distance can always be calculated without knowing the explicit form of the feature map itself.

Problem 3 You are given a dataset of x, y pairs $\{(x_i, y_i)\}_{i=1}^N$, with $x_i \in X$ and $y_i \in \{-1, 1\}$. Assume that n_+, n_- of the x_i have label $+1, -1$, respectively (so $n_+ + n_- = N$), and let's also assume that we are given a kernel K and an associated feature map $\Phi : X \to \mathcal{F}$ satisfying

$$K(x, x') = \langle \Phi(x), \Phi(x')\rangle_{\mathcal{F}}.$$

Derive a classification rule, involving only kernel products (and the sign function), that assigns to a new test point the label of the class whose mean is closest *in the feature space*.

Problem 4 In (binary) classification problems one aims at finding a classification rule (also called the "decision rule") which is a binary valued function on the input space $c : X \to \{1, -1\}$. The quality of a classification rule can be naturally measured by means of the so called misclassification error defined by

$$R(c) = \mathbb{P}\{c(x) \neq y\}.$$

If we introduce the misclassification loss $V(c(x), y) = \theta(-yc(x))$, where $\theta(s) = 1$ if $s > 0$ and $\theta(s) = 0$ otherwise, the misclassification error can be rewritten as

$$R(c) = \int_{X\times Y} \theta(-yc(x))p(x)p(y|x)dxdy.$$

Direct minimization of the misclassification error is not computationally feasible mostly because the misclassification loss is not convex. In practice, one usually looks for real valued (rather than

binary valued) functions $f : X \to \mathbf{R}$ and replaces $\theta(-yc(x))$ with some convex loss $V(-yf(x))$. A classification rule is then obtained by taking the sign, that is $c(x) = \text{sign}(f(x))$. Commonly chosen loss functions are the hinge loss and square loss (see class). Note that in this case the error is measured by the expected error

$$I[f] = \int_{X\times Y} V(-yf(x))p(x)p(y|x)dxdy.$$

However, there is still the problem of relating the convex approximation to the original classification problem.

With the above discussion in mind, and assuming that the distribution $p(x, y)$ is known, answer the following questions:

(a) Check that the square loss can be written as $V(-yf(x))$. Calculate the explicit form of the minimizer of $I[f]$ if V is the square loss.

(b) Calculate the closed-form of the minimizer of $I[f]$ if V is the exponential loss $V(-yf(x)) = \exp\{-yf(x)\}$.

(c) Find the closed-form of the minimizer of $I[f]$ if V for the logistic loss $V(-yf(x)) = \log\left(1 + \exp\{-yf(x)\}\right)$.

(d) The minimizer of $R(c)$ over all possible decision rules is the so called Bayes decision rule $b : X \to \{1, -1\}$. For all the losses considered above, what is their relation to Bayes decision rule?

Problem 5 Consider a bounded loss function $V : \mathbb{R} \times \mathbb{R} \to (0, M]$ and a hypothesis space comprised of N distinct functions, $\mathcal{H} = \{f_1, \ldots, f_N\}$.

(a) Prove that for all $\epsilon > 0$, the following bound holds

$$\Pr\Big(\sup_{f\in\mathcal{H}} \big|I_S[f] - I[f]\big| \geq \epsilon\Big) \leq \frac{CNM^2}{n\epsilon^2} \tag{2}$$

where $C > 0$ is some constant. What is the best C that you can get?
(Hint: use Chebychev's inequality and union bound)

(b) Show that, if f_S is the minimizer of the empirical risk on $\mathcal{H}$, then the above inequality implies that with probability $1 - \eta$ we have

$$I[f_S] \leq I_S[f_S] + \epsilon(n, \eta, N)$$

where $\epsilon(n, \eta, N) = \sqrt{\frac{CNM^2}{\eta n}}$ and $0 < \eta \leq 1$. Discuss the behavior of $I_S[f_S]$, $\epsilon(n, \eta, N)$ and their sum as functions of N.

(c) Denote with f_S and f^* the minimizers on $\mathcal{H}$ of the empirical and expected risks, respectively. Given (2), show that

$$I[f_S] - I[f^*] \leq 2\epsilon(n, \eta, N).$$

(Hint: add and subtract the empirical risks of f_S and f^* in the left hand side of the above inequality. Recall that by definition f_S minimizes the empirical risk.)

Problem 6 *Matlab exercise.*

In this excercise you will implement and use regularized least squares on an artificial classification problem. You will do the following:

- Implement RLS using the linear and polynomial kernels. You should write two functions:
  - `rlsTrain(Ytrain,Xtrain,whichKernel)` takes three inputs
    - `Ytrain` the training labels;
    - `Xtrain` the training inputs;
    - `whichKernel` the kernel to use, e.g. `'linear'`;

    and returns three outputs
    - `coeffs` the optimal RLS coefficients
    - `lambdas` a vector of values tried for the regularization parameter λ
    - `looe` a vector of leave-one-out errors on the training set – errors for the LOO RLS solutions, that is – one for each value of λ in `lambdas`
  - `rlsPredict(Xtest,Xtrain,coeffs,whichKernel)` takes four inputs
    - `Xtest` the test inputs;
    - `Xtrain` the training inputs;
    - `coeffs` the coefficients to use;
    - `whichKernel` the kernel to use, e.g. `'linear'`

    and returns one output
    - `Ytest` the predicted values at the test inputs
- (You might want to write a helper function to construct the kernel matrix, too.)
- (`rlsTrain` picks values for λ automatically. A reasonable value for λ might range up to the maximum eigenvalue of the kernel matrix.)
- Download `ps1-dataset.mat` from the course page. It contains a training set `Xtrain, Ytrain` and a test set `Xtest, Ytest` each one containing 100 samples. The inputs in `Xtrain` and `Xtest` should have two dimensions.
- Use RLS to train a linear classifier on the training set, choosing the regularization parameter λ to minimize the leave-one-out error.
- Do the same thing with a polynomial kernel, using at least 3 different polynomial degrees.
- Compare the obtained classifiers by testing them on the test set and plotting the obtained decision boundaries.

Please include the following in your writeup.

- All of the code you wrote.
- Figures showing:
  - The training set error and leave-one-out error vs. λ for each of the kernels you tried. Plot both kinds of error on the same figure (one figure for each kernel).
  - The decision boundaries overlaid on the training set points (plotted in two dimensions), for each of the kernels you tried.
- A table giving the training and test error and best λ for each kernel you tried. Report the error in terms of the percentage of points correctly classified.

References

[1] T. Evgeniou and M. Pontil and T. Poggio. Regularization Networks and Support Vector Machines. Advances in Computational Mathematics, 2000.

[2] V. N. Vapnik. Statistical Learning Theory. Wiley, 1998.