Homework #6

TA email: `ml2011ta@csie.ntu.edu.tw`

RELEASE DATE: 12/7/2011

DUE DATE: **EXTENDED TO 12/30/2011 4PM IN R536**

Unless granted by the instructor in advance, you must turn in a printed/written copy of your solutions (without the source code) for all problems. For problems marked with (*), please follow the guidelines on the course website and upload your source code and predictions to designated places.

Any form of cheating, lying, or plagiarism will not be tolerated. Students can get zero scores and/or fail the class and/or be kicked out of school and/or receive other punishments for those kinds of misconducts.

Discussions on course materials and homework solutions are encouraged. But you should write the final solutions alone and understand them fully. Books, notes, and Internet resources can be consulted, but not copied from.

Since everyone needs to write the final solutions alone, there is absolutely no need to lend your homework solutions and/or source codes to your classmates at any time. In order to maximize the level of fairness in this class, lending and borrowing homework solutions are both regarded as dishonest behaviors and will be punished according to the honesty policy.

You should write your solutions in English with the common math notations introduced in class or in the problems. We do not accept solutions written in any other languages.

6.1 Transforms: Explicit versus Implicit

Consider the following training set:

$$\mathbf{x}_1 = (1,0), y_1 = -1 \qquad \mathbf{x}_2 = (0,1), y_2 = -1 \qquad \mathbf{x}_3 = (0,-1), y_3 = -1$$

$$\mathbf{x}_4 = (-1,0), y_4 = +1 \qquad \mathbf{x}_5 = (0,2), y_5 = +1 \qquad \mathbf{x}_6 = (0,-2), y_6 = +1$$

$$\mathbf{x}_7 = (-2,0), y_7 = +1$$

(1) Use following nonlinear transformation of the input vector $\mathbf{x}$ to the transformed vector $\mathbf{z} = (\phi_1(\mathbf{x}), \phi_2(\mathbf{x}))$:

$$\phi_1(\mathbf{x}) = (\mathbf{x}[2])^2 - 2\mathbf{x}[1] + 3 \qquad \phi_2(\mathbf{x}) = (\mathbf{x}[1])^2 - 2\mathbf{x}[2] - 3$$

(a) (10%) Write down the equation of the optimal separating "hyperplane" in the $\mathcal{Z}$ space. Then, plot the transformed training points in the $\mathcal{Z}$ space as well as the boundary between the $+1$ and -1 regions, and mark the on-the-boundary vectors (the potential support vectors).

(b) (10%) Write down the equation of the corresponding nonlinear curve in the $\mathcal{X}$ space. Then, plot the original training points on the $\mathcal{X}$ plane as well as the boundary between the $+1$ and -1 regions, and mark the on-the-boundary vectors (the potential support vectors).

(2) Consider the same training set, but instead of explicitly transforming the input space $\mathcal{X}$, apply the (hard-margin) SVM algorithm with the kernel function

$$K(\mathbf{x}, \mathbf{x}') = (2 + \mathbf{x}^T\mathbf{x}')^2,$$

which corresponds to a second-order polynomial transformation.

(a) (10%) Set up the optimization problem using $(\alpha_1, \cdots, \alpha_7)$ and numerically solve for them (you can use any package you want). What is the optimal $\boldsymbol{\alpha}$?

(b) (10%) Write down the equation of the corresponding nonlinear curve in the $\mathcal{X}$ space. Then, plot the original training points on the $\mathcal{X}$ plane as well as the boundary between the $+1$ and -1 regions, and mark the on-the-boundary vectors (the potential support vectors).

(3) (10%) Should the two nonlinear curves (and potential support vectors) found in (1) and (2) be the same? Why or why not? Make a comparison and briefly describe your findings.

6.2 A Leave-One-Out Bound of Support Vector Machine

Consider the soft-margin SVM

$$\begin{aligned}(D) \quad \min_{\boldsymbol{\alpha}} \quad & E_N(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} \alpha_n\alpha_m y_n y_m K(\mathbf{x}_n,\mathbf{x}_m) - \sum_{n=1}^{N}\alpha_n \\ \text{subject to} \quad & \sum_{n=1}^{N} y_n\alpha_n = 0 \\ & 0 \le \alpha_n \le C\end{aligned}$$

and a soft-margin SVM without the N-th example

$$\begin{aligned}(D_{-N}) \quad \min_{\boldsymbol{\beta}} \quad & E_{N-1}(\boldsymbol{\beta}) = \frac{1}{2}\sum_{n=1}^{N-1}\sum_{m=1}^{N-1} \beta_n\beta_m y_n y_m K(\mathbf{x}_n,\mathbf{x}_m) - \sum_{n=1}^{N-1}\beta_n \\ \text{subject to} \quad & \sum_{n=1}^{N-1} y_n\beta_n = 0 \\ & 0 \le \beta_n \le C\end{aligned}$$

(1) (10%) Assume that $\boldsymbol{\alpha}^*$ is an optimal solution for (D) with $\alpha_N^* = 0$. Let $\hat{\boldsymbol{\beta}} = \left(\alpha_1^*, \alpha_2^*, \ldots, \alpha_{N-1}^*\right)$. Argue that $\hat{\boldsymbol{\beta}}$ is a feasible vector for (D_{-N}). That is, check that $\hat{\boldsymbol{\beta}}$ satisfies all constraints of (D_{-N}).

(2) (10%) Assume that $\boldsymbol{\beta}^*$ is an optimal solution for (D_{-N}). That is, $E_{N-1}(\boldsymbol{\beta}) \ge E_{N-1}(\boldsymbol{\beta}^*)$ for all feasible vectors $\boldsymbol{\beta}$. Prove that the $\hat{\boldsymbol{\beta}}$ above satisfies

$$E_{N-1}(\hat{\boldsymbol{\beta}}) = E_{N-1}(\boldsymbol{\beta}^*).$$

In other words, $\hat{\boldsymbol{\beta}}$ is also optimal for (D_{-N}).

(3) (10%) Recall that #SV = (# of nonzero α_n^*). With the results in (1) and (2), prove that the leave-one-out cross-validation error of SVM is upper bounded by the percentage of support vectors. You can use the fact that

$$\alpha_n^* = 0 \Longrightarrow y_n\Big((\mathbf{w}^*)^T\phi(\mathbf{x}_n) + b^*\Big) \ge 1.$$

6.3 Dual Problem of L2-Loss Soft-Margin Support Vector Machines

In class, we taught the soft-margin support vector machine as follows.

$$\begin{aligned}(P_1) \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n \\ \text{s.t.} \quad & y_n\Big(\mathbf{w}^T\mathbf{x}_n + b\Big) \ge 1 - \xi_n \\ & \xi_n \ge 0.\end{aligned}$$

The support vector machine penalizes the margin violation linearly. Another popular formulation penalizes the margin violation quadratically. In this problem, we derive the dual of such a formulation. The formulation is as follows.

$$\begin{aligned}(P_2') \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n^2 \\ \text{s.t.} \quad & y_n\Big(\mathbf{w}^T\mathbf{x}_n + b\Big) \ge 1 - \xi_n, \text{ for } n = 1, 2, \cdots, N; \\ & \xi_n \ge 0, \text{ for } n = 1, 2, \cdots, N.\end{aligned}$$

(1) (5%) Argue that the constraints $\xi_n \geq 0$ are not necessary for the new formulation. In other words, the formulation $(P2')$ is equivalent to the following optimization problem.

$$
\begin{aligned}
(P_2) \min_{\mathbf{w},b,\xi} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n^2 \\
\text{s.t.} \quad & y_n\Big(\mathbf{w}^T\mathbf{x}_n + b\Big) \geq 1 - \xi_n, \text{ for } n = 1, 2, \cdots, N.
\end{aligned}
$$

(2) (10%) Let α_n be the Lagrange multipliers for the n-th constraint in (P_2). Following the derivation of the dual SVM in class, write down (P_2) as an equivalent optimization problem

$$\min_{(b,\mathbf{w},\boldsymbol{\xi})} \max_{\alpha_n \geq 0} \mathcal{L}((b, \mathbf{w}, \boldsymbol{\xi}), \boldsymbol{\alpha}).$$

What is $\mathcal{L}((b, \mathbf{w}, \boldsymbol{\xi}), \boldsymbol{\alpha})$?

(3) (10%) Using (assuming) strong duality, the solution to (P_2) would be the same as the Lagrange dual problem

$$\max_{\alpha_n \geq 0} \min_{(b,\mathbf{w},\boldsymbol{\xi})} \mathcal{L}((b, \mathbf{w}, \boldsymbol{\xi}), \boldsymbol{\alpha}).$$

Use the KKT conditions to simplify the Lagrange dual problem, and obtain a dual problem that involves only α_n.

(4) (5%) Explain what would happen when we use $\mathbf{z}_n = \boldsymbol{\phi}(\mathbf{x}_n)$ instead of $\mathbf{x}_n$, and write down the optimization problem that uses $K(\mathbf{x}_n, \mathbf{x}_m)$ to replace $\boldsymbol{\phi}(\mathbf{x}_n)^T\boldsymbol{\phi}(\mathbf{x}_m)$—that is, the kernel trick.

6.4 Large-Margin Perceptron Classification (*)

(1) (15%) Implement the perceptron learning algorithm in Problem 1.5. Run the algorithm on the following data set for training (until E_{in} reaches 0):

`http://www.csie.ntu.edu.tw/~htlin/course/ml11fall/doc/hw6_4_train.dat`

and the following set for testing

`http://www.csie.ntu.edu.tw/~htlin/course/ml11fall/doc/hw6_4_test.dat`

Let $g(\mathbf{x}) = \text{sign}\big(\mathbf{w}^T\mathbf{x} + b\big)$ with $(b, \mathbf{w})$ coming from PLA. Record the following two items:

- the margin of the hyperplane
- the out-of-sample error E_{out} of g

Repeat the experiment over 100 runs. Plot a histogram of the margin and another histogram of the out-of-sample error. Briefly state your findings.

(2) (15%) Implement the large-margin perceptron (linear hard-margin SVM) formulation below:

$$
\begin{aligned}
\min_{\mathbf{w},b} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\
\text{subject to} \quad & y_n\left(\mathbf{w}^T\mathbf{x}_n + b\right) \geq 1 \text{ for } n = 1, 2, \ldots, N.
\end{aligned}
$$

Run the algorithm on the following data set for training:

`http://www.csie.ntu.edu.tw/~htlin/course/ml11fall/doc/hw6_4_train.dat`

and the following set for testing

`http://www.csie.ntu.edu.tw/~htlin/course/ml11fall/doc/hw6_4_test.dat`

Let $g(\mathbf{x}) = \text{sign}\big(\mathbf{w}^T\mathbf{x} + b\big)$ with $(b, \mathbf{w})$ coming from SVM. Record the following two items:

- the margin of the hyperplane

- the out-of-sample error E_{out} of g

Compare the numbers with the histograms that you get from PLA. Briefly state your findings.

(*Note: You can use any general-purpose packages for quadratic programming to solve this problem, but you* ***cannot*** *use any SVM-specific packages.*)

6.5 Experiments with Nonlinear Support Vector Machine (*)

Write a program to implement the nonlinear soft-margin Support Vector Machine by solving

$$\begin{aligned}\min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i,\mathbf{x}_j) - \sum_{i=1}^{N}\alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{N} y_i\alpha_i = 0 \\ & 0 \le \alpha_i \le C\end{aligned}$$

(1) (15%) Use the following set for training:

`http://www.csie.ntu.edu.tw/~htlin/course/ml10fall/data/hw6_5_train.dat`

Consider the polynomial kernel $(1+\mathbf{x}^T\mathbf{x}')^d$ with $d = 3, 6, 9$, and $C = 0.001, 1, 1000$. For each (d, C) combination, show E_{in}, E_{cv} with 5-fold cross validation, and $\frac{\#SV}{N}$ (an upper-bound of leave-one-out cross validation error). Briefly describe your findings.

(2) (15%) Use the following set for training:

`http://www.csie.ntu.edu.tw/~htlin/course/ml10fall/data/hw6_5_train.dat`

Consider the Gaussian-RBF kernel $\exp\left(\frac{-\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$ with $\sigma = 0.125, 0.5, 2$ and $C = 0.001, 1, 1000$. For each (σ, C) combination, show E_{in}, E_{cv} with 5-fold cross validation, and $\frac{\#SV}{N}$ (an upper-bound of leave-one-out cross validation error). Briefly describe your findings.

(3) (30%) Use any SVM kernel, any SVM parameters on the following set for training to produce one SVM hypothesis g:

`http://www.csie.ntu.edu.tw/~htlin/course/ml10fall/data/hw6_5_train.dat`

Then, use the returned SVM hypothesis to predict the label of each example within the following test set:

`http://www.csie.ntu.edu.tw/~htlin/course/ml11fall/data/hw6_5_test.dat`

Describe the procedure you use to choose the kernel and the parameters, and write down the final kernel/parameters you choose. Submit your predictions to the designated place (to be announced on the course website)—*note that you can only submit* ***once***. **This time your score would be partially partially based on how reasonable your procedure is and how well your predictions perform. So please validate your hypothesis and write down the procedure carefully.**

(*Note: For this problem, you CAN use any package you want. A recommended choice is LIBSVM developed by Prof. Chih-Jen Lin in our department*)

6.6 L2-Loss and Hard-Margin

(1) (Bonus 10%) The L2-Loss soft-margin SVM (P_2) in Problem 6.3 is actually equivalent to a hard-margin SVM that takes examples $(\tilde{\mathbf{x}}_n, \tilde{y}_n)$ instead of $(\mathbf{x}_n, y_n)$. Write down $(\tilde{\mathbf{x}}_n, \tilde{y}_n)$ and prove the equivalence.

(*Note: You can actually use the equivalence to make the derivations in Problem 6.3 simpler.*)