Machine Learning: 10701 and 15781, 2003

Assignment 4

Primary contact for questions and clarifications: Rong Zhang (rongz@cs.cmu.edu)

Due Date: 10.30am Tuesday November 4th (one week)
Group Work: You may work in a group of two people if you wish.

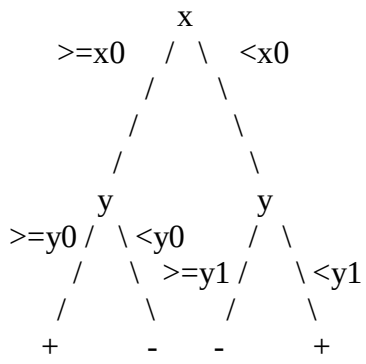
1. VC Dimension (30 Points)

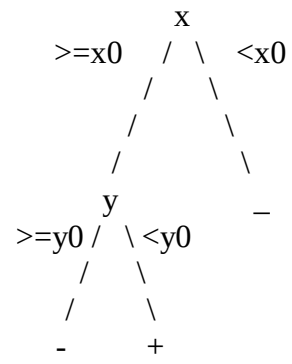
Consider the space of instance X corresponding to all points in the 2D x, y plane. Give the VC dimension of the following hypothesis spaces. No explanation required.

(a) H_r = the set of all axes-parallel rectangles in the x, y plane. That is, $H_r = \{((a < x < b) \wedge (c < y < d)) \mid a, b, c, d \in \mathbf{R}\}$. Points inside the rectangle are classified as positive.

(b) H_a = like (a), but including all rectangles (not just the ones parallel to the axes of coordinate system).

(c) H_d = real-valued, depth-2 decision trees. For example, the following trees are in H_d.





(e) (**Zero Points**) H_f = hypotheses with the form $f(x) = \theta(\sin(\alpha \cdot x))$, $x, \alpha \in \mathbf{R}$ where $\theta(z) = 1$ iff $z > 0$ and 0 otherwise. You can consider this question in 1D space. (**Zero Points: Only Do This For Fun If You Would Like To And If You Have Time**.)

2. Support Vector Machine (45 Points)

The following question requires you to use **MatLab**, but it has been designed to be just as easy for a MatLab novice as for a MatLab expert. Please read the appendix for how to use MatLab.

We will investigate Support Vector Machine with two toy datasets. The file "1d.clean" contains 100 examples each of which has a real-valued input attribute x and a class label y. The data set is generated in the following way:

$x \sim p(x)$ where $p(x) = 0.5$ iff $-1 \le x \le 1$, otherwise $p(x) = 0$.

$$y(x) = \begin{cases} 1 & x \geq 0 \\ -1 & x < 0 \end{cases}$$

In addition, we add noise to this dataset by negating the class of some examples that produces the noisy dataset "1d.noise".

Training a SVM involves setting up a convex quadratic optimization problem and solving it. MatLab is a common mathematical programming language that facilitates this by providing quadratic programming functions, **qp** or **quadprog**. A MatLab program, "svm.m", has been prepared for you that trains the SVM on each of the datasets and outputs results including margin width, training error, number of support vectors and number of misclassifications.

In this assignment, you are asked to investigate the impact of the trade-off weight C on margin width and training error. Considering the object function for non-separable case:

$$L = \tfrac{1}{2}\|\mathbf{w}\|^2 + C(\sum_i \xi_i)$$

the margin width is defined as: Margin = $2/\|\mathbf{w}\|$,

and the training set error is defined as: Error = $\sum_i \xi_i$.

(a) Read the program and complete the setting up of Hessian matrix H, and the lower and upper bounds for Lagrange multiplier α_i (Alpha). Note that the Hessian matrix H is exactly the Q matrix in our slides.

$H(i, j) =$

$LB =$

$UB =$

(b) Measure the impact of trade-off weight C on the margin width and training error with the clean dataset "1d.clean". Turn in plots, showing how margin width and training error vary with C, including the values 0.01, 0.1, 1, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000 and inf.

(c) Measure the impact of trade-off weight C on the margin width and training error with the noise dataset "1d.noise". Turn in plots, showing how margin width and training error vary with C, including the values 0.01, 0.1, 1, 5, 10, 20, 50, 100, 200, 500, 1000, 2000, 5000, 10000 and inf.

(d) Briefly explain your findings.

3. K Nearest Neighbor in Regression (25 Points)

Suppose we have a real-valued training dataset $\{(x_i, y_i) \mid 1 \leq i \leq M, x_i \in \mathbf{R}, y_i \in \mathbf{R}\}$ which is generated using the following distribution:

$y_i \sim N(c, \sigma^2)$ where c is unknown to us. (Note that we assume the variance σ^2 is known.)

$x_i \sim P(x)$ where $P(x) = 1$ for $0 \leq x \leq 1$, otherwise $P(x) = 0$.

Our task is to compare the performance of the following two regression algorithms.

Alg1: Use Maximum Likelihood Estimation (MLE) to learn c from the dataset. The MLE assumption results in $\hat{c} = \frac{1}{M}\sum_{i=1}^{M} y_i$. For any input x, the output is simply $\hat{y}(x) = \hat{c}$.

Alg2: Use 1- Nearest Neighbor to predict y. Namely, $\hat{y}(x) = y_i$, where y_i is the output value associated with the training set datapoint x_i that is the nearest neighbor of x. If there is a tie among multiple training datapoints for being the nearest neighbor of x, then we just randomly select one of them.

(a) Assume $M \to \infty$. What is the expected squared error of Alg1 and Alg2 on the training set?

For Alg1, $\lim_{M\to\infty} \frac{1}{M}\sum_{i=1}^{M} (\hat{y}(x_i) - y_i)^2 = ?$

For Alg2, $\lim_{M\to\infty} \frac{1}{M}\sum_{i=1}^{M} (\hat{y}(x_i) - y_i)^2 = ?$

(b) Assume $M \to \infty$. What is the expected squared error of Alg1 and Alg2 for predicting the output y of a future data point (x, y) generated in the same way as training data.

For Alg1, $E((\hat{y}(x) - y)^2) = ?$

For Alg2, $E((\hat{y}(x) - y)^2) = ?$

(c) (**Zero Points**) If M is finite, what is the expected squared error of Alg1 and Alg2 for predicting a future data point (x, y) generated in the same way as training data. (**Zero Points: Only Do This For Fun If You Would Like To And If You Have Time**.)

For Alg1, $E((\hat{y}(x) - y)^2) = ?$

For Alg2, $E((\hat{y}(x) - y)^2) = ?$

Appendix

MatLab is a powerful matrix oriented programming language that is similar to C with matrix operators added. Tutorials can be found at Engineering & Science Library or a lot of websites. (Ask *google* for the name of these websites. A brief one would suffice to this assignment.)

MatLab can be found as a standard part of all the CS Linux/Unix machines. If you are a CS student and use Linux/Unix machine, just type in 'matlab' to start the software. If you are a CS student but use Windows, you can download and install MatLab from the machine \\monolith\PC_DIST\Matlab. (You may need to contact CS help center for the permission of access.)

For the students who come from other departments and can't find MatLab from his or her computer, please go to the Computer Clusters in the 5th floor of Wean Hall and choose any machine running Linux/Unix. You should be able to login into the computer using your Andrew ID and password. After login, just type 'matlab' to verify the software is available. This may take several minutes due to the computer speed.

Supposing you are going to use a Linux/Unix machine but not familiar with the OS and MatLab, the following instructions may help.

(1) Login to a Linux/Unix machine using your CS or Andrew ID and password.
(2) Create a new directory for our assignment by typing ‘mkdir svm’.
(3) Enter the directory you just created by typing ‘cd svm’.
(4) Download the program ‘svm.m’ and data files ‘1d.clean’ and ‘1d.noise’ to the local directory. You can do this by starting the browser ‘netscape’ and go to our course website.
(5) Start MatLab by typing ‘matlab’.
(6) Read the program svm.m and complete it.
(7) In the MatLab window, type in ‘svm’. If your answer is correct, you should be able to see four figures appearing. The first two are the plots that we need.
(8) If something wrong, use your tutorial or MatLab online help for debugging. For example, typing ‘help qp’ in the MatLab window will show you the information of the quadratic programming function ‘qp’.
(9) Contact TA for help if you still have troubles.