CS273a Midterm Exam

Machine Learning & Data Mining: Fall 2012

Thursday November 1st, 2012

Your name: SOLUTIONS

Name of the person in front of you (if any):

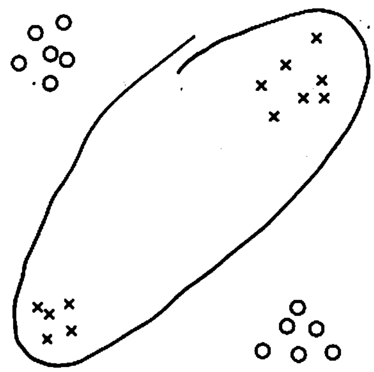
Name of the person to your right (if any):

- Total time is 1:15. READ THE EXAM FIRST and organize your time; don't spend too long on any one problem.
- Please write clearly and show all your work.
- If you need clarification on a problem, please raise your hand and wait for the instructor to come over.
- Turn in any scratch paper with your exam.

(This page intentionally left blank)

Problem 1: (12 points) Separability

For each of the following examples of training data, sketch a classification boundary that separates the data. State whether or not the data are linearly separable, and if not, give a set of features that would allow the data to be separated.

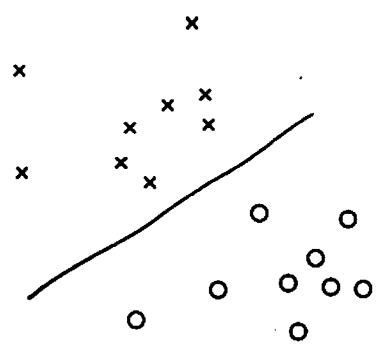


NOT linearly separable

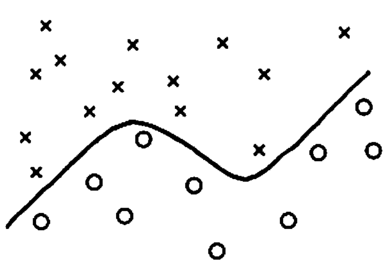
could use eg.

$[x_1 \;\; x_2 \;\; x_1x_2 \;\; x_1^2 \;\; x_2^2]$

to produce the ellipsoidal decision bdry



yes, linearly separable



No, not linearly separable

from the decision bdry, it looks like

$x_2 = a + bx_1 + cx_1^2 + dx_1^3$

so features

$[1 \;\; x_1 \;\; x_2 \;\; x_1^2 \;\; x_1^3]$ are probably sufficient.

Problem 2: (13 points) Under- and Over-fitting

(a) Suppose that we train a classifier, and discover that it achieves zero training error. Are we likely to be over-fitting, under-fitting, neither, or do we need more information? Explain (1-2 sentences).

Need more information – it is impossible to check overfitting from only the training data error; we need validation data or cross-validation.
It could be that we are overfit & have memorized the data; or, it could be that the data are easy to predict & our classifier is very good.

(b) **Circle one answer for each:**

Adding features to a linear classifier will make it **more** equally less likely to overfit the data.

Increasing the regularization parameter for a linear classifier will make it more equally less likely to overfit the data.

Increasing the step size in gradient descent for a linear classifier will make it more equally less likely to overfit the data. (May depend on stopping criteria ...)

Increasing the value of k in a k-nearest neighbor classifier will make it more equally less likely to overfit the data.

Increasing the number of hidden nodes in a neural network will make it more equally less likely to overfit the data.

Problem 3: (10 points) Regression

Suppose that we have training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$ and we wish to predict y using the model:

$$\hat{y}(x) = a \log(x + b) + c$$

(a) Is this a linear or nonlinear regression model. Why?

Nonlinear - it is a non-linear function of parameter b.

(b) Write the mean squared error cost function for our predictor.

$$J(a,b,c) = \frac{1}{N} \sum_i (y^i - \hat{y}^i)^2$$

$$= \frac{1}{N} \sum_i \left(y^i - (a \log(x^i + b) + c)\right)^2$$

(c) Compute its gradient with respect to the parameters a, b, and c.

$$\frac{\partial}{\partial a} J = \frac{2}{N} \sum_i (y^i - \hat{y}^i)(-\log(x^i + b))$$

where $\hat{y}^i = (a \log(x^i + b) + c)$;

$$\frac{\partial}{\partial b} J = \frac{2}{N} \sum_i (y^i - \hat{y}^i)\left(-a \frac{1}{x^i + b}\right)$$

$$\frac{\partial}{\partial c} J = \frac{2}{N} \sum_i (y^i - \hat{y}^i)(-1)$$

$$\nabla J = \left[\frac{\partial}{\partial a} J \quad \frac{\partial}{\partial b} J \quad \frac{\partial}{\partial c} J\right]$$

$$= \frac{-2}{N} \sum_i \underbrace{(y^i - \hat{y}^i)}_{\text{scalar}} \underbrace{\left[\log(x^i + b) \quad \frac{a}{x^i + b} \quad 1\right]}_{\text{vector}}$$

Problem 4: (6 points) Optimization

(a) Give pseudocode for a stochastic (online or incremental) gradient descent algorithm to optimize the model in Problem 2. You do not need to have solved for the gradient $\nabla J(a, b, c)$ to do this; just assume it can be computed. Explain all the parameters used by your algorithm.

Initialize parameters θ to something; $J = \infty$

while (! done) { $\quad J^{old} = J$.

for $i = 1 \dots N$

$J_i = (y^i - \hat{y}^i)^2$

∇J_i is the gradient of J_i

$\theta \leftarrow \theta - \alpha \nabla J_i$ where α is a step size.

end

compute $J = \frac{1}{N} \sum (y^i - \hat{y}^i)^2$

done = true if:

① too many iterations already or

② J hasn't changed from the last iteration. $(|J - J^{old}| < \epsilon)$

}

(b) Explain the difference between batch and stochastic gradient descent (1-3 sentences). Name one advantage for each.

Define $J_i(a,b,c)$ to be the loss on data point i, eg $(y - \hat{y}^i)^2$

$\Rightarrow J(a,b,c) = \frac{1}{N} \sum_i J_i(a,b,c)$

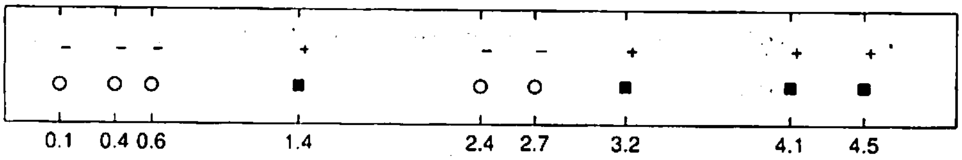
Batch gradient descent uses ∇J, the gradient of the loss on all data, for each update;

SGD uses ∇J_i, the gradient on each point i, to update sequentially.

Batch GD will decrease the overall loss J, so its easier to check convergence and "smoother".

SGD is noisier (more random), but in practice tends to optimize the parameters much faster, especially when N is very large.

Problem 5: (12 points) Cross-validation and Nearest Neighbor



Using the above data with one feature x (whose values are given below each data point) and a class variable $y \in \{-1, +1\}$, with squares indicating $y = +1$ and circles $y = -1$ (the sign is also shown above each data point for redundancy), answer the following:

(a) Compute the leave-one-out cross-validation error of a 1-Nearest-Neighbor classifier. In the case of any ties, select the left-most neighbor at the same distance as the nearest.

✓ ✓ ✓ ✗ ✓ ✓ ✗ ✓ ✓

$\Rightarrow$ $2/9$ error rate.

(b) Compute the leave-one-out cross-validation error for a 3-Nearest-Neighbor classifier.

✓ ✓ ✓ ✗ ✗ ✗ ✗ ✓ ✓

$\Rightarrow$ $4/9$ error rate

(c) Compute the leave-one-out cross-validation error for a 8-NN classifier. In the case of a tie, predict class +1.

tie tie

✗ ✗ ✗ ✗ ✗ ✗ ✗ ✗ ✗

$\Rightarrow$ $9/9$ error rate (!)

Problem 6: (14 points) VC Dimension

(a) Describe VC dimension in your own words, in a few (2-4) sentences.

A classifier can shatter a set of points if it can learn to produce any pattern of class values on those points.

The VC dimension is the largest # of points that can be arranged such that they can be shattered.

(b) Give an example of a model in which the VC dimension is not equal to the number of parameters.

From HW (or similar) $T[a(bx_1 + cx_2 + d)]$

- has four parameters, but "a" doesn't really help in any way.

(c) Circle one answer for each:

Increasing the amount of training data in a linear classifier will increase not change decrease the VC dimension.

Increasing the number of features used in a linear classifier will increase not change decrease the VC dimension.

Increasing the regularization parameter for a linear classifier will increase not change decrease the VC dimension.

Exponentiating feature 1 before training (e.g., `x(:,1) = exp(x(:,1));`) a linear classifier will increase not change decrease the VC dimension.

Problem 7: (12 points) Support Vector Machines

Consider a linear classifier, $T(wx^T + b)$, where $x = [x_1, \dots, x_d]$ is a d-dimensional feature vector, $w = [w_1, \dots, w_d]$ are the coefficients, and b is the constant coefficient.

In class, I described how we could optimize a SVM written in constraint form,

$$\min_w \tfrac{1}{2}\|w\|^2 \qquad s.t.\, y^{(i)}(wx^{(i)} + b) \geq 1$$

by optimizing w along with a set of Lagrange multipliers α:

$$J(w, \alpha) = \min_w \max_{\alpha \geq 0} \tfrac{1}{2}\|w\|^2 + \sum_i \alpha_i (1 - y^{(i)}(wx^{(i)T} + b))$$

(a) By solving $\nabla_w J(w, \alpha) = 0$, show that the optimal value of w is

$$w^* = \sum_i \alpha_i y^{(i)} x^{(i)}$$

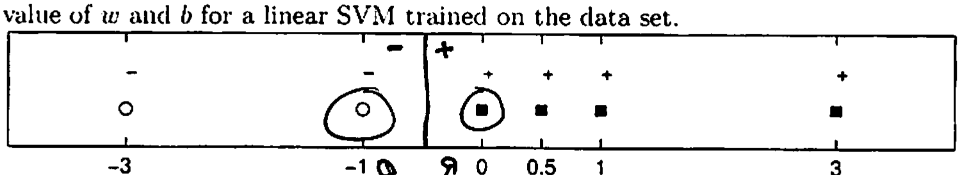
(Hint: just take the derivative and solve for w_1 and argue symmetry.)

$$\frac{\partial}{\partial w_1} J(w, \alpha) = \cancel{\tfrac{2}{2}} w_1 + \sum_i -\alpha_i y^{(i)} x_1^{(i)} = 0$$

$$\Rightarrow \quad w_1 = \sum_i \alpha_i y^{(i)} x_1^{(i)} \qquad\qquad w = [w_1 \; w_2 \; w_3 \; \dots]$$

$$\Rightarrow \quad \underline{w} = \sum_i \alpha_i y^{(i)} \underline{x}^{(i)}$$

(b) For the following data, sketch the decision boundary, identify the support vectors, and give the value of w and b for a linear SVM trained on the data set.



SVs

$$\left.\begin{aligned} wx + b &= 1 \ @\ x = 0 \\ wx + b &= -1 \ @\ x = -1 \end{aligned}\right\} \Rightarrow \begin{aligned} b &= 1 \\ w &= +2 \end{aligned}$$