Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.S064 Introduction to Machine Learning

Midterm exam (March 21, 2013)

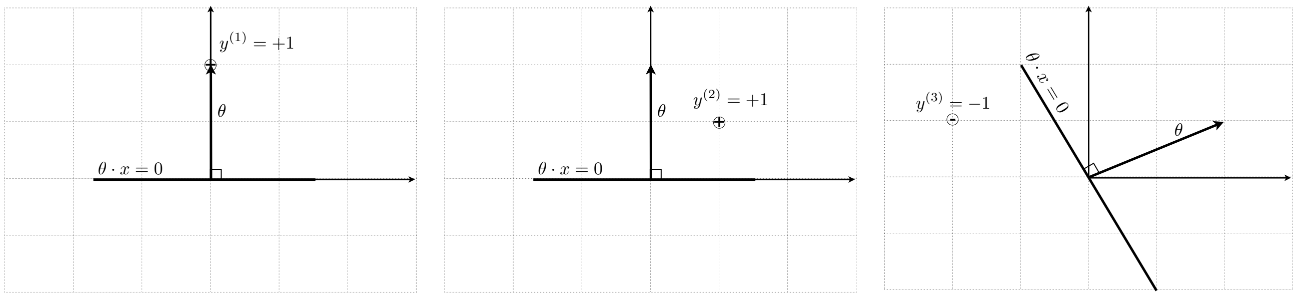
Your name & ID: ______________________________

- This is a closed book exam
- You do not need nor are permitted to use calculators
- The value of each question – number of points awarded for full credit – is shown in parenthesis
- The problems are not necessarily in any order of difficulty. We recommend that you read through all the problems first, then do the problems in whatever order suits you best.
- Record all your answers in the places provided

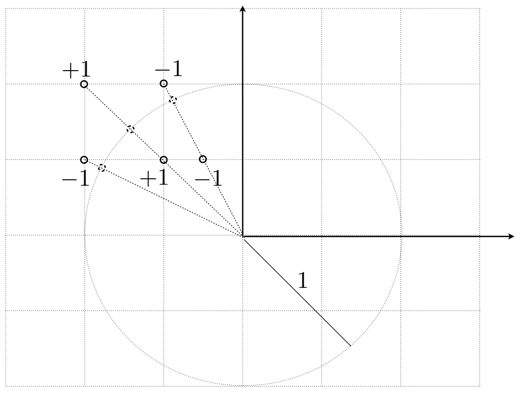
Problem 1.1	Problem 2.1	Problem 2.2	Problem 3.1	Problem 4.1	Total
12	12	10	14	9	57

(1.1) The perceptron algorithm remains surprisingly popular for a 50+ year old (algorithm). Hard to find a simpler algorithm for training linear or non-linear classifiers. Worth knowing this algorithm by heart, yes?

(a) **(6 points)** Let $\theta = 0$ (vector) initially. We run the perceptron algorithm to train $y = \text{sign}(\theta \cdot x)$, where $x \in \mathcal{R}^2$, using the three labeled 2-dimensional points in the figure below. In each figure, approximately draw both θ and the decision boundary *after* updating the parameters (if needed) based on the corresponding point.



(b) **(6 points)** We were wondering what would happen if we normalized all the input examples. In other words, instead of running the algorithm using x, we would run it with feature vectors $\phi(x) = x/\|x\|$. Let's explore this with linear classifiers that include offset, i.e., we use $y = \text{sign}(\theta \cdot x + \theta_0)$ or $y = \text{sign}(\theta \cdot \phi(x) + \theta_0)$ after feature mapping. In the figure below, label *all the points* such that 1) the perceptron algorithm wouldn't converge if they are given as original points, 2) the algorithm would converge if run with $\phi(x) = x/\|x\|$.



The points are mapped radially to the unit circle. The initial set of points are not linearly separable. The resulting three points on the unit circle are.

(2.1) Support vector machines (SVMs) are so popular that we decided to try them out a bit in a simple setting where $x \in \mathcal{R}$. We used both linear and non-linear SVMs:

$$\min\ w^2/2 \quad \text{subject to} \quad y^{(t)}(wx^{(t)} + w_0) \geq 1, \quad t = 1, \ldots, n \tag{1}$$
$$\min\ \|\theta\|^2/2 \quad \text{subject to} \quad y^{(t)}(\theta \cdot \phi(x^{(t)}) + \theta_0) \geq 1, \quad t = 1, \ldots, n \tag{2}$$

where $\phi(x)$ is a feature vector constructed from the real valued input x. We wish to compare the resulting classifiers when $\phi(x) = [x, x^2]^T$.

(a) **(3 points)** Provide three input points $x^{(1)}$, $x^{(2)}$, and $x^{(3)}$, where $x^{(i)} \in [0, 4]$, and their associated ± 1 labels such that 1) they cannot be separated with the linear classifier, but 2) are separable by the non-linear classifier with $\phi(x) = [x, x^2]^T$. You may find Figure 2.1 helpful in answering this question.
$(x^{(1)}, y^{(1)}) = ($ 0 ,+1 $)$
$(x^{(2)}, y^{(2)}) = ($ 2 , -1 $)$
$(x^{(3)}, y^{(3)}) = ($ 4 ,+1 $)$

(b) **(4 points)** Map your three labeled points as labeled feature vectors in Figure 2.1. Approximately draw the resulting *decision boundary* in the feature space of the non-linear SVM classifier with $\phi(x) = [x, x^2]^T$.

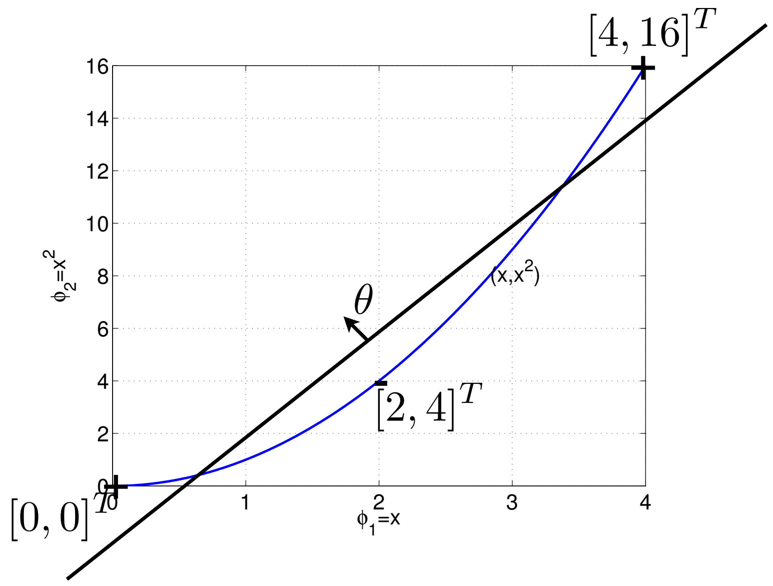


Figure 2.1. Feature space.

(c) **(3 points)** Consider two labeled points $(x = 1, y = 1)$ and $(x = 3, y = -1)$. Is the resulting geometric margin we attain in the feature space using feature vectors $\phi(x) = [x, x^2]^T$

(X) greater, ( ) equal, or ( ) smaller

than the geometric margin resulting from using the input x directly?

Answer: we could simply omit the 2nd coordinate in the feature space and get the same margin as in the original input space. Including the 2nd coordinate can only further increase the margin.

(d) **(2 points)** In general, is the geometric margin we would attain using scaled feature vectors $\phi(x) = 2[x, x^2]^T$

(X) greater, ( ) equal, ( ) smaller, or ( ) could be any of these

in comparison to the geometric margin resulting from using $\phi(x) = [x, x^2]^T$?

Answer: the geometric margin scales with the feature vectors.

(2.2) **(10 points)** We were given a dataset with $n = 10$ labeled two-dimensional examples. Having learned about the support vector machine, we wanted to try it out on this data. However, we were unsure which kernel function to use. So we asked students in the class for possible kernel functions, and tried each one of them:

$$\begin{aligned}
&(1)\ K(x, x') = x \cdot x' \\
&(2)\ K(x, x') = (x \cdot x') + 2(x \cdot x')^2 \\
&(3)\ K(x, x') = 1 - 2(x \cdot x')^2 \\
&(4)\ K(x, x') = (1 + x \cdot x')^5
\end{aligned}$$

(a) For any kernel $K(x, x') = \phi(x) \cdot \phi(x')$. What can you say about $K(x, x)$ in general?
( ) $K(x, x)$ is never 0, (X) $K(x, x) < 0$ never, ( ) $K(x, x) > 1$ always

Answer: $K(x, x) = \phi(x) \cdot \phi(x) = \|\phi(x)\|^2 \geq 0$.

(b) The optimization routine for the dual SVM problem complained about one of the kernels, and it couldn't be used further. Which one (1-4)? (3)

Answer: from part (a) we know that $K(x, x) \geq 0$. Use $\|x\| = 1$ for "kernel" (3) to see that it is negative so cannot be kernel.

(c) Only one of the kernels resulted in zero training error. Which one (1-4)? (4)

Answer: kernel (4) is the highest order polynomial kernel. If any of them achieves zero training error, it is this one.

(d) As far as the training error is concerned, one of the kernels was by far the worst. Which one (1-4)? **(** 1 **)**

Answer: the linear kernel is the simplest one (the weakest classifier).

(e) After training the three classifiers (one we couldn't use), we obtained additional data to assess their test errors. Which one would you expect to have the lowest test error (1-4)? **(** 2 **)**

Answer: because we only have $n = 10$ training examples, we cannot reliably use 5th order polynomial terms. On the other hand, the simple linear classifier didn't perform well even on the training set. So must be (2).

(3.1) We are faced with a content filtering problem where the idea is to rank new songs by trying to predict how they might be rated by a particular user. Each song x is represented by a feature vector $\phi(x)$ whose coordinates capture specific acoustical properties. The ratings are binary valued $y \in \{0, 1\}$ ("need earplugs" or "more like this"). Given n already rated songs, we decided to use regularized linear regression to predict the binary ratings. The training criterion is

$$J(\theta) = \frac{\lambda}{2}\|\theta\|^2 + \frac{1}{n}\sum_{t=1}^{n}(y^{(t)} - \theta \cdot \phi(x^{(t)}))^2/2 \tag{3}$$

(a) **(8 points)** Let $\hat{\theta}$ be the optimal setting of the parameters with respect to the above criterion, which of the following conditions must be true (check all that apply)

(X **)** $\lambda\hat{\theta} - \frac{1}{n}\sum_{t=1}^{n}(y^{(t)} - \hat{\theta} \cdot \phi(x^{(t)}))\phi(x^{(t)}) = 0$
Answer: this is the optimality condition

(**)** $J(\hat{\theta}) \geq J(\theta)$, for all $\theta \in \mathcal{R}^d$
Answer: we minimize the objective so $\hat{\theta}$ is the minimum, not the maximum

(X **)** If we increase λ, the resulting $\|\hat{\theta}\|$ will decrease
Answer: the more we emphasize the norm $\|\theta\|^2$ in the optimization problem, the smaller its value will be.

(X **)** If we add features to $\phi(x)$ (whatever they may be), the resulting squared training error will NOT increase
Answer: we could ignore the additional coordinates, setting the corresponding θ values to zero. As a result, we can always attain the same error. Optimizing the parameters for the additional coordinates can only minimize the error further.

(b) **(3 points)** Once we have the estimated parameters $\hat{\theta}$, we must decide how to predict ratings for new songs. Note that the possible rating values are 0 or 1. When do we choose rating $y = 1$ for a new song x? Please write the corresponding expression.

$y = 1$ if $\hat{\theta} \cdot \phi(x) \geq 0.5$. Note that since the target ratings are 0 or 1, the resulting regression function values (predictions) are also mostly between 0 and 1. So, for example, $\hat{\theta} \cdot \phi(x) \geq 0$ would be a poor decision rule (most if not all instances would be rated $y = 1$).

(c) **(3 points)** If we change λ, we obtain different $\hat{\theta}$, and therefore different rating predictions according to your rule above. What will happen to your predicted ratings when we increase the regularization parameter λ?

When we increase λ, $\|\hat{\theta}\|$ will decrease. As a result, the regression function values, i.e., $\hat{\theta} \cdot \phi(x)$, will also tend towards zero. Given the decision rule above, the predictions are going to be biased towards $y = 0$.

(4.1) Consider the simple K-means algorithm for clustering. Each iteration of the algorithm consists of two steps, assigning points to the centroids, and updating the centroids based on the points assigned to them. We will assume that $k = 2$.

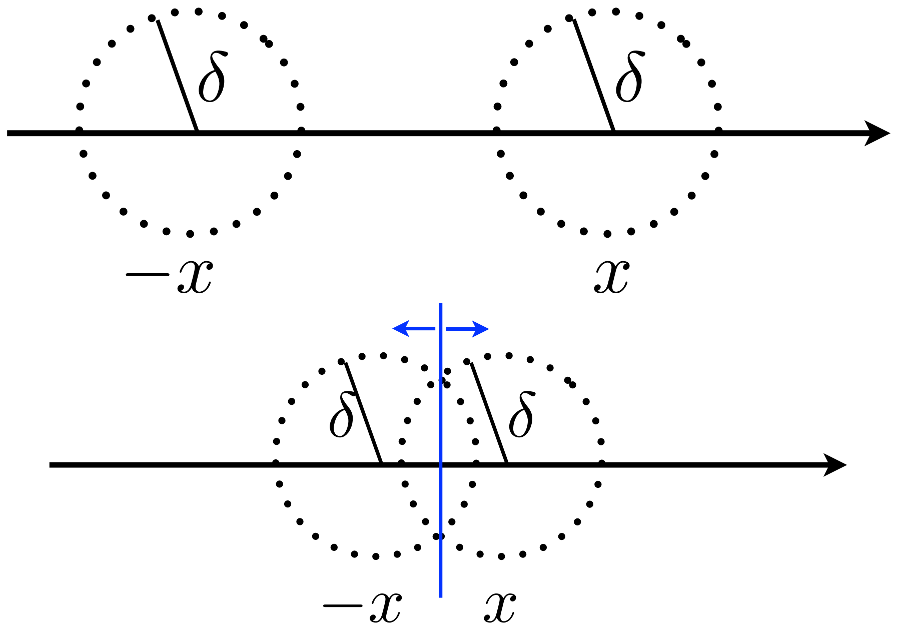
(a) **(2 points)** If we initialize the centroids to be the means of the two well-separated clusters, will the centroids change after the first iteration? (Y/N) **(** N **)**

(b) **(3 points)** If we initialize the centroids by drawing a random point from each of the two well-separated clusters, how many iterations does it take for the k-means to converge? **(** 1 **)**

Answer: since the clusters are well-separated, the initialization will not change assignments of points to centroids. The centroids become the cluster means (the fixed points) after the first iteration.

(c) **(4 points)** Consider two spherical (circle-like) clusters of radius δ as shown below. The clusters are centered at locations $-x$ and x. For which values of x would the k-means algorithm fail to find the centers of the two clusters regardless of the initialization?

$|x| < \delta$. The best initialization would be to start with the centers of the two spherical clusters. When $|x| < \delta$, as in the figure below, the clusters overlap. Since points are assigned to their closest centroids, divided according to the blue line, the resulting cluster means no longer lie at the center of the original clusters.



δ
δ
$-x$
x
δ
δ
$-x$
x