6.867 Machine learning

Mid-term exam

October 22, 2002

(2 points) Your name and MIT ID:

Problem 1

We are interested here in a particular 1-dimensional linear regression problem. The dataset corresponding to this problem has n examples $(x_1, y_1), \ldots, (x_n, y_n)$, where x_i and y_i are real numbers for all i. Part of the difficulty here is that we don't have access to the inputs or outputs directly. We don't even know the number of examples in the dataset. We are, however, able to get a few numbers computed from the data.

Let $\mathbf{w}^* = [w_0^*, w_1^*]^T$ be the least squares solution we are after. In other words, $\mathbf{w}^*$ minimizes

$$J(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)^2$$

You can assume for our purposes here that the solution is unique.

1. **(4 points)** Check each statement that must be true if $\mathbf{w}^* = [w_0^*, w_1^*]^T$ is indeed the least squares solution

   ( ) $(1/n)\sum_{i=1}^{n}(y_i - w_0^* - w_1^* x_i)y_i = 0$
   ( ) $(1/n)\sum_{i=1}^{n}(y_i - w_0^* - w_1^* x_i)(y_i - \bar{y}) = 0$
   (x) $(1/n)\sum_{i=1}^{n}(y_i - w_0^* - w_1^* x_i)(x_i - \bar{x}) = 0$
   (x) $(1/n)\sum_{i=1}^{n}(y_i - w_0^* - w_1^* x_i)(w_0^* + w_1^* x_i) = 0$

   where $\bar{x}$ and $\bar{y}$ are the sample means based on the same dataset.

Taking the derivative with respect to w_1 and w_0 gives us the following conditions of optimality (as in the lectures)

$$\begin{aligned}\frac{\partial}{\partial w_0} J(\mathbf{w}) &= \frac{2}{n}\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i) = 0\\ \frac{\partial}{\partial w_1} J(\mathbf{w}) &= \frac{2}{n}\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)x_i = 0\end{aligned}$$

This means that the prediction error $(y_i - w_0 - w_1 x_i)$ does not co-vary with any linear function of the inputs (has a zero mean and does not co-vary with the inputs). $(x_i - \bar{x})$ and $(w_0^* + w_1^* x_i)$ are both linear functions of inputs.

2. **(4 points)** There are several numbers (statistics) computed from the data that we can use to infer $\mathbf{w}^*$. These are

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i,\quad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i,\quad C_{xx} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$
$$C_{xy} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}),\quad C_{yy} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2$$

Suppose we only care about the value of w_1^*. We'd like to determine w_1^* on the basis of only two numbers (statistics) listed above. Which two numbers do we need for this?

We need C_{xx} (spread of x) and C_{xy} (linear dependence between x and y). No justification was necessary as these basic points have appeared repeatedly in the course. If we want to derive these more mathematically, we can, for example, look at one of the answers to the previous question:

$$\frac{1}{n}\sum_{i=1}^{n}(y_i - w_0^* - w_1^* x_i)(x_i - \bar{x}) = 0, \text{ which we can rewrite as}$$
$$\left[\frac{1}{n}\sum_{i=1}^{n} y_i(x_i - \bar{x})\right] - w_0^*\left[\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})\right] - w_1^*\left[\frac{1}{n}\sum_{i=1}^{n} x_i(x_i - \bar{x})\right] = 0$$

By using the fact that $(1/n)\sum_i (x_i - \bar{x}) = 0$ we see that

$$\begin{aligned}\frac{1}{n}\sum_{i=1}^{n} y_i(x_i - \bar{x}) &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})(x_i - \bar{x}) = C_{xy}\\ \frac{1}{n}\sum_{i=1}^{n} x_i(x_i - \bar{x}) &= \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x}) = C_{xx}\end{aligned}$$

Substituting these back into our equation above gives $C_{xy} - w_1^* C_{xx} = 0$.

3. Here we change the rules governing our access to the data. Instead of simply getting the statistics we want, we have to reconstruct these from examples that we query. There are two types of queries we can make. We can either request additional randomly chosen examples from the training set, or we can query the output corresponding to a specific input that we specify. (We assume that the dataset is large enough that there is always an example whose input x is close enough to our query).

   The active learning scenario here is somewhat different from the typical one. Normally we would assume that the data is governed by a linear model and choose the input points so as to best recover this assumed model. Here the task is to recover the best fitting linear model to the data but we make no assumptions about whether the linear model is appropriate in the first place.

   (2 points) Suppose in our case the input points are constrained to lie in the interval $[0, 1]$. If we followed the typical active learning approach, where we assume that the true model is linear, what are the input points we would query?

   We would query the extreme points $x = 0$ and $x = 1$ as they constrain the linear function the most.

   (3 points) In the new setting, where we try to recover the best fitting linear model or parameters $\mathbf{w}^*$, we should (choose only one):

   ( ) Query inputs as you have answered above

   (x) Draw inputs and corresponding outputs at random from the dataset

   ( ) Use another strategy since neither of the above choices would yield satisfactory results

(4 points) Briefly justify your answer to the previous question

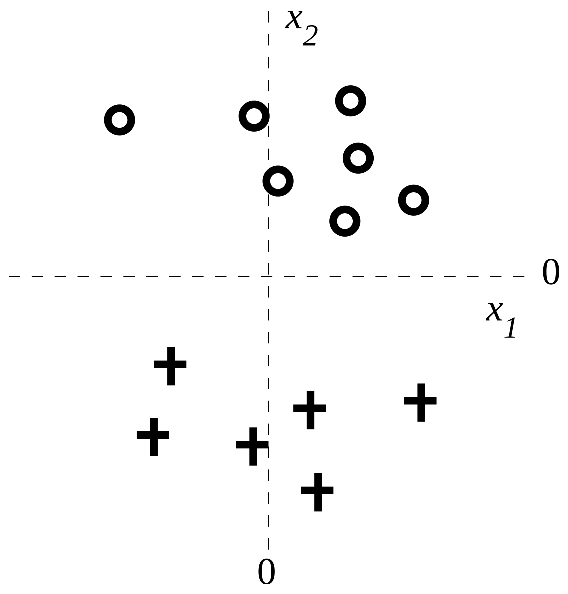
The objective is to recover the least squares solution. This solution depends on the frequency of inputs and outputs in the dataset. Without additional assumptions, the best thing to do is to draw a representative set of examples from the dataset so that the resulting least squares solution would approximate the solution based on the full dataset.

Problem 2

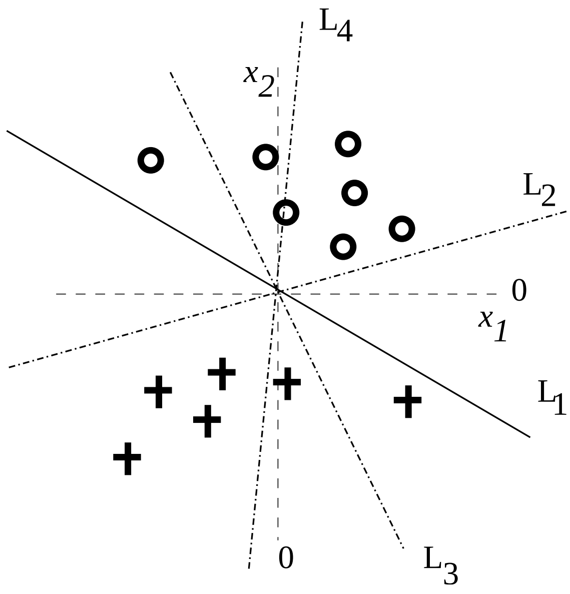
In this problem we will refer to the binary classification task depicted in Figure 1(a), which we attempt to solve with the simple linear logistic regression model

$$\hat{P}(y = 1|\mathbf{x}, w_1, w_2) = g(w_1 x_1 + w_2 x_2) = \frac{1}{1 + \exp(-w_1 x_1 - w_2 x_2)}$$

(for simplicity we do not use the bias parameter w_0). The training data can be separated with zero training error - see line L_1 in Figure 1(b) for instance.



(a) The 2-dimensional data set used in Problem 1



(b) The points can be separated by L_1 (solid line). Possible other decision boundaries are shown by L_2, L_3, L_4.

1. **(6 points)** Consider a regularization approach where we try to maximize

$$\sum_{i=1}^{n} \log p(y_i|\mathbf{x}_i, w_1, w_2) - \frac{C}{2} w_2^2$$

for large C. Note that **only** w_2 is penalized. We'd like to know which of the four lines in Figure 1(b) could arise as a result of such regularization. For each potential line L_2, L_3 or L_4 determine whether it can result from regularizing w_2. If not, explain very briefly why not.

- *L2*

  No. *When we regularize w_2, the resulting boundary can rely less on the value of x_2 and therefore becomes more vertical. L_2 here seems to be more horizontal than the unregularized solution so it cannot come as a result of penalizing w_2*

- *L3*

  Yes. *Here w_2^2 is small relative to w_1^2 (as evidenced by high slope), and even though it would assign a rather low log-probability to the observed labels, it could be forced by a large regularization parameter C.*

- *L4*

  No. *For very large C, we get a boundary that is entirely vertical (line $x_1 = 0$ or the x_2 axis). L_4 here is reflected across the x_2 axis and represents a poorer solution than it's counter part on the other side. For moderate regularization we have to get the best solution that we can construct while keeping w_2 small. L_4 is not the best and thus cannot come as a result of regularizing w_2.*

2. **(4 points)** If we change the form of regularization to one-norm (absolute value) and also regularize w_1 we get the following penalized log-likelihood

$$\sum_{i=1}^{n} \log p(y_i|\mathbf{x}_i, w_1, w_2) - \frac{C}{2} \left(|w_1| + |w_2|\right).$$

Consider again the problem in Figure 1(a) and the same linear logistic regression model $\hat{P}(y = 1|\mathbf{x}, w_1, w_2) = g(w_1 x_1 + w_2 x_2)$. As we increase the regularization parameter C which of the following scenarios do you expect to observe (choose only one):

(x) First w_1 will become 0, then w_2.

(   ) w_1 and w_2 will become zero simultaneously

( ) First w_2 will become 0, then w_1.

( ) None of the weights will become exactly zero, only smaller as C increases

The data can be classified with zero training error and therefore also with high log-probability by looking at the value of x_2 alone, i.e. making $w_1 = 0$. Initially we might prefer to have a non-zero value for w_1 but it will go to zero rather quickly as we increase regularization. Note that we pay a regularization penalty for a non-zero value of w_1 and if it doesn't help classification why would we pay the penalty? The absolute value regularization ensures that w_1 will indeed go to exactly zero.
As C increases further, even w_2 will eventually become zero. We pay higher and higher cost for setting w_2 to a non-zero value. Eventually this cost overwhelms the gain from the log-probability of labels that we can achieve with a non-zero w_2. Note that when $w_1 = w_2 = 0$, the log-probability of labels is a finite value $n \cdot \log(0.5)$.

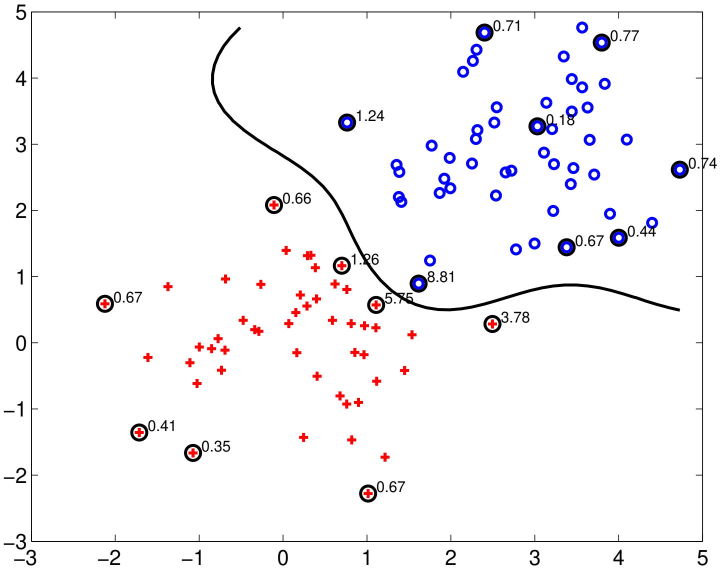


Figure 1: A 2-dim classification problem, the resulting SVM decision boundary with a radial basis kernel, as well as the support vectors (indicated by larger circles around them). The numbers next to the support vectors are the corresponding coefficients $\hat{\alpha}$.

Problem 3

Figure 1 illustrates a binary classification problem along with our solution using support vector machines (SVMs). We have used a radial basis kernel function given by

$$K(\mathbf{x}, \mathbf{x}') = \exp\{ -\|\mathbf{x} - \mathbf{x}'\|^2/2 \}$$

where $\|\cdot\|$ is a Euclidean distance and $\mathbf{x} = [x_1, x_2]^T$. The classification decision for any $\mathbf{x}$ is made on the basis of the sign of

$$\hat{\mathbf{w}}^T \phi(\mathbf{x}) + \hat{w}_0 = \sum_{j \in \mathrm{SV}} y_j \hat{\alpha}_j K(\mathbf{x}_j, \mathbf{x}) + \hat{w}_0 = f(\mathbf{x}; \hat{\alpha}, \hat{w}_0)$$

where $\hat{\mathbf{w}}$, $\hat{w}_0$, $\hat{\alpha}_i$ are all coefficients estimated from the available data displayed in the figure and SV is the set of support vectors. $\phi(\mathbf{x})$ is the feature vector derived from $\mathbf{x}$ corresponding to the radial basis kernel. In other words, $K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x})^T \phi(\mathbf{x}')$. While technically $\phi(\mathbf{x})$ is an infinite dimensional vector in this case, this fact plays no role in the questions below. You can assume and treat it as a finite dimensional vector if you like.

The support vectors we obtain for this classification problem (indicated with larger circles in the figure) seem a bit curious. Some of the support vectors appear to be far away from the decision boundary and yet be support vectors. Some of our questions below try to resolve this issue.

1. **(3 points)** What happens to our SVM predictions $f(\mathbf{x}; \hat{\alpha}, \hat{w}_0)$ with the radial basis kernel if we choose a test point $\mathbf{x}_{far}$ far away from any of the training points $\mathbf{x}_j$ (distances here measured in the space of the original points)?

   The radial basis kernel $K(\mathbf{x}_{far}, \mathbf{x}_j)$ vanishes as the distance $\|\mathbf{x}_{far} - \mathbf{x}_j\|$ to the training point increases. The value of $f(\mathbf{x}; \hat{\alpha}, \hat{w}_0)$ therefore approaches $\hat{w}_0$ for any point $\mathbf{x}_{far}$ sufficiently far from any of the training points.

2. **(3 points)** Let's assume for simplicity that $\hat{w}_0 = 0$. What equation do all the training points $\mathbf{x}_j$ have to satisfy? Would $\mathbf{x}_{far}$ satisfy the same equation?

   If $\hat{w}_0 = 0$, then all the training points will satisfy

   $$y_i \hat{\mathbf{w}}^T \phi(\mathbf{x}_i) - 1 \geq 0$$

   since the problem is separable. $\mathbf{x}_{far}$ cannot satisfy this equation regardless of the label associated with this point since $\hat{\mathbf{w}}^T \phi(\mathbf{x}_{far}) = f(\mathbf{x}_{far}; \hat{\alpha}, 0) \approx 0$.

3. **(4 points)** If we included $\mathbf{x}_{far}$ in the training set, would it become a support vector? Briefly justify your answer.

   $\mathbf{x}_{far}$ would have to become a support vector. Our answers to the above questions indicate that this point could not satisfy the margin constraints without being included in the solution $f(\mathbf{x}; \hat{\alpha}, \hat{w}_0)$.

4. **(T/F – 2 points)** Leave-one-out cross-validation error is always small for support vector machines. F

   The claim is roughly the same as saying that the SVM always has a low generalization error – which is false. The question is admittedly a little ambiguous since you could have been thinking about a different notion of "small".

   Note that the number of support vectors is only partially related to the cross-validation error. We know that cross-validation error has to be smaller than the relative number of support vectors. However, we can have a large number of support vectors and yet very small cross-validation error. This happens when many of the support vectors are not "essential".

5. **(T/F – 2 points)** The maximum margin decision boundaries that support vector machines construct have the lowest generalization error among all linear classifiers

   F

   The maximum margin hyperplane is often a reasonable choice but it is by no means optimal in all cases.

6. **(T/F – 2 points)** Any decision boundary that we get from a generative model with class-conditional Gaussian distributions could in principle be reproduced with an SVM and a polynomial kernel of degree less than or equal to three

   T

   A polynomial kernel of degree two suffices to represent any quadratic decision boundary such as the one from the generative model in question.

7. **(T/F – 2 points)** The decision boundary implied by a generative model (with parameterized class-conditional densities) can be optimal only if the class-conditional densities are correct for the problem at hand

   F

   The decision boundary may not depend on all aspects of the class-conditional densities. For example, in the trivial case where the class-conditional densities are the same for the two classes, the optimal decision boundary is based only on the prior class frequencies. We can easily reproduce this with any identical class-conditional densities.

Problem 4

Consider the following set of 3-dimensional points, sampled from two classes:

labeled '1':

x_1	x_2	x_3
1,	1,	-1
0,	2,	-2
0,	-1,	1
0,	-2,	2

labeled '0':

x_1	x_2	x_3
1,	1,	2
0,	2,	1
1,	-1,	-1
1,	-2,	-2

We have included 2-dimensional plots of pairs of features in the "Additional set of figures" section (figure 3).

1. **(4 points)** Explain briefly why features with higher mutual information with the label are likely to be more useful for classification task (in general, not necessarily in the given example).

A high value of I(feature;label) means that we can substantially reduce our uncertainty about the label by knowing the value of the feature. In other words, we have a good idea about the label if we know the value of the feature. For classification we want features that tell us most about the label.

2. **(3 points)** In the example above, which feature (x_1, x_2 or x_3) has the highest mutual information with the class label, based on the training set?

   x_1

   Clearly, both x_2 and x_3 have zero mutual information with the label - their values do not appear to depend on the label. The value of x_1 does seem to provide some information about the label since, for example, class 0 has a higher chance of $x_1 = 1$ than class 1.

3. **(4 points)** Assume that the learning is done with quadratic logistic regression, where

   x_2, x_3

$$P(y = 1|\mathbf{x}, \mathbf{w}) = g(w_0 + w_1 x_i + w_2 x_j + w_3 x_i x_j + w_4 x_i^2 + w_5 x_j^2)$$

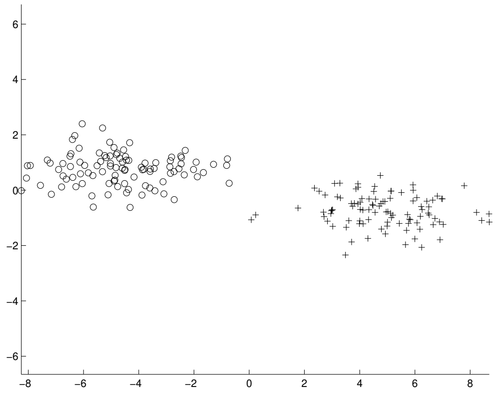
   for some pair of features (x_i, x_j). Based on the training set given above, which pair of features would result in the lowest training error for the logistic regression model?

   One could refer to plots in Figure 3, or simply analyze the values of the features. The values of x_1, x_2 are the same for two pairs examples that belong to different classes: (1,1,-1)/(1,1,2) and (0,2,-2)/(0,2,1). We cannot classify all of these correctly no matter what kind of decision boundary we would have. The same is true for x_1, x_3 - consider, (1,1,-1)/(1,-1,-1) and (0,-1,1)/(0,2,1). However, including x_2, x_3 as features, all the training points appear distinct (see the plot). They can also be separated with a quadratic decision boundary. This is clear from the figure but you can also check that thresholding $x_2 \cdot x_3$ is sufficient for correct classification (set all the weights to zero except w_3).

4. **(T/F – 2 points)** From the point of view of classification it is always beneficial to remove features that have very high variance in the data

   F

The feature with the highest variance can still have the highest mutual information with the label (as in the figure below).



5. **(T/F – 2 points)** A feature which has zero mutual information with the class label might be selected by a greedy selection method, if it happens to improve classifier's performance on the training set

   T

   The mutual information of a single feature and the label measures how good the feature is alone. Greedy selection picks features that are useful in conjunction with those already selected. It is possible that a feature which would be useless alone proves to be useful when combined with another.

Problem 5

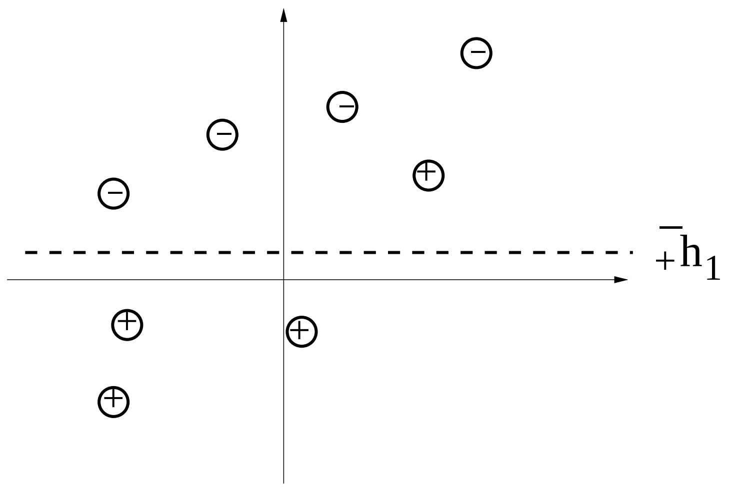


Figure 2: h_1 is chosen at the first iteration of boosting; what is the weight α_1 assigned to it?

1. **(3 points)** Figure 2 shows a dataset of 8 points, equally divided among the two classes (positive and negative). The figure also shows a particular choice of decision stump h_1 picked by AdaBoost in the first iteration. What is the weight α_1 that will be assigned to h_1 by AdaBoost? (Initial weights of all the data points are equal, or 1/8.)

   $\boxed{\log_2 \sqrt{7}}$

   The weighted training error ϵ *is 1/8 - thus* $\alpha = \frac{1}{2}\log_2 \frac{1-\epsilon}{\epsilon} = \frac{1}{2}\log_2 \frac{7/8}{1/8}$.

2. **(T/F – 2 points)** AdaBoost will eventually reach zero training error, regardless of the type of weak classifier it uses, provided enough weak classifiers have been combined.

   $\boxed{\text{F}}$

   Not if the data in the training set cannot be separated by a linear combination of the specific type of weak classifiers we are using.

3. **(T/F – 2 points)** The votes α_i assigned to the weak classifiers in boosting generally go down as the algorithm proceeds, because the weighted training error of the weak classifiers tends to go up

   $\boxed{\text{T}}$

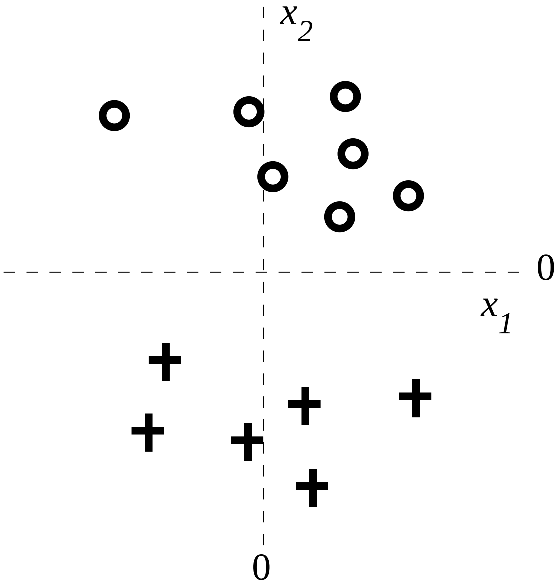
   In the course of boosting iterations the weak classifiers are forced to try to classify more difficult examples. The weights will increase for examples that are repeatedly misclassified by the weak component classifiers. The weighted training error of the components therefore tends to go up and, as a result, their votes go down.

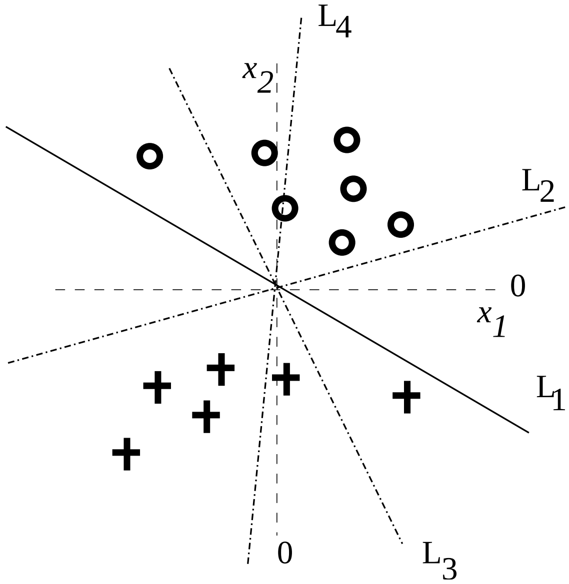
4. **(T/F – 2 points)** The votes α assigned to the classifiers assembled by AdaBoost are always non-negative

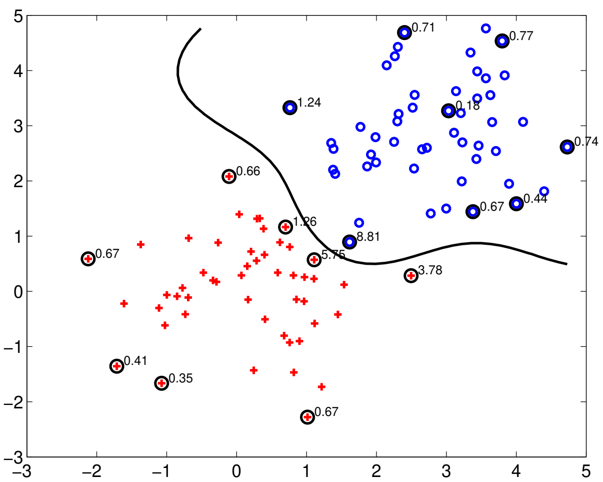
   $\boxed{\text{T}}$

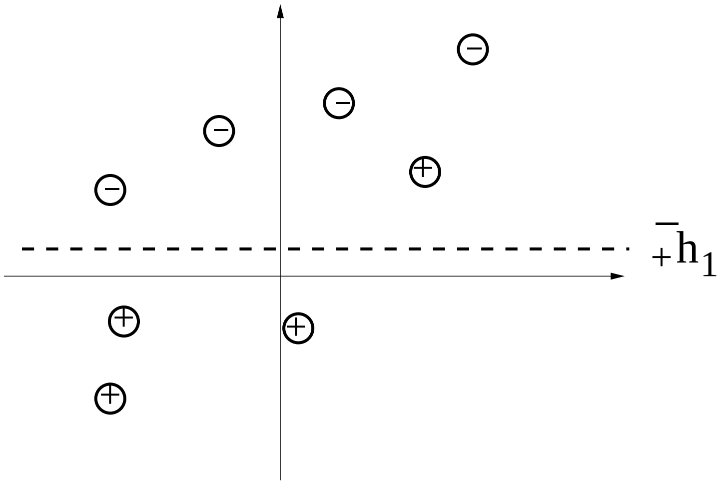
   As defined in class, AdaBoost will choose classifiers with training error above 1/2. This will ensure that $\log_2(1-\epsilon/\epsilon)$*, and therefore the vote, is positive. Note that if the classifier does worse than* $1/2$ *we can always "flip" the sign of its predictions and therefore get a classifier that does slightly better than* $1/2$*. The vote assigned to the "flipped" classifier would be non-negative.*

Additional set of figures









there's more ...

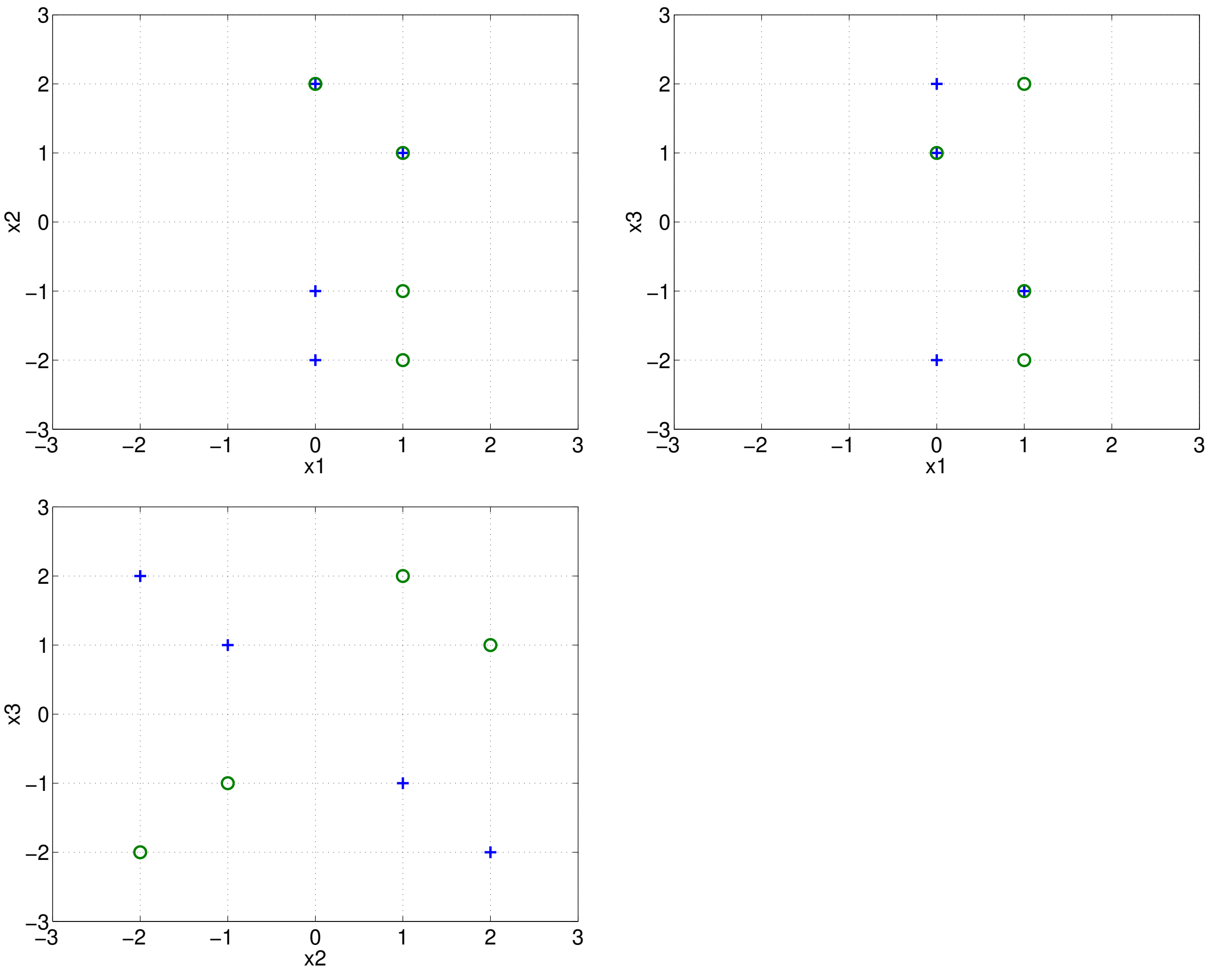


Figure 3: 2-dimensional plots of pairs of features for problem 4. Here '+' corresponds to class label '1' and 'o' to class label '0'.

.