# 10-701/15-781 Machine Learning - Midterm Exam, Fall 2010

Aarti Singh
Carnegie Mellon University

1. Personal info:
   - Name:
   - Andrew account:
   - E-mail address:
2. There should be **15** numbered pages in this exam (including this cover sheet).
3. You can use any material you brought: any book, class notes, your print outs of class materials that are on the class website, including annotated slides and relevant readings, and Andrew Moore's tutorials. You cannot use materials brought by other students. Calculators are not necessary. Laptops, PDAs, phones and Internet access are not allowed.
4. If you need more room to work out your answer to a question, use the back of the page and clearly mark on the front of the page if we are to look at what's on the back.
5. Work efficiently. Some questions are easier, some more difficult. Be sure to give yourself time to answer all of the easy ones, and avoid getting bogged down in the more difficult ones before you have answered the easier ones.
6. You have **90** minutes.
7. Good luck!

| Question | Topic | Max. score | Score |
|---|---|---|---|
| 1 | Short questions | 20 | |
| 2 | Bayes Optimal Classification | 15 | |
| 3 | Logistic Regression | 18 | |
| 4 | Regression | 16 | |
| 5 | SVM | 16 | |
| 6 | Boosting | 15 | |
| | **Total** | 100 | |

# 1 Short Questions [20 pts]

**Are the following statements True/False? Explain your reasoning in only 1 sentence.**

1. Density estimation (using say, the kernel density estimator) can be used to perform classification.

   **True:** Estimate the joint density $P(Y, X)$, then use it to calculate $P(Y|X)$.

2. The correspondence between logistic regression and Gaussian Naïve Bayes (with identity class covariances) means that there is a one-to-one correspondence between the parameters of the two classifiers.

   **False:** Each LR model parameter corresponds to a whole set of possible GNB classifier parameters, there is no one-to-one correspondence because logistic regression is discriminative and therefore doesn't model $P(X)$, while GNB does model $P(X)$.

3. The training error of 1-NN classifier is 0.

   **True:** Each point is its own neighbor, so 1-NN classifier achieves perfect classification on training data.

4. As the number of data points grows to infinity, the MAP estimate approaches the MLE estimate for all possible priors. In other words, given enough data, the choice of prior is irrelevant.

   **False:** A simple counterexample is the prior which assigns probability 1 to a single choice of parameter $\theta$.

5. Cross validation can be used to select the number of iterations in boosting; this procedure may help reduce overfitting.

   **True:** The number of iterations in boosting controls the complexity of the model, therefore, a model selection procedure like cross validation can be used to select the appropriate model complexity and reduce the possibility of overfitting.

6. The kernel density estimator is equivalent to performing kernel regression with the value $Y_i = \frac{1}{n}$ at each point $X_i$ in the original data set.

   **False:** Kernel regression predicts the value of a point as the weighted average of the values at nearby points, therefore if all of the points have the same value, then kernel regression will predict a constant (in this case, $\frac{1}{n}$) for all values.

7. We learn a classifier $f$ by boosting weak learners $h$. The functional form of $f$'s decision boundary is the same as $h$'s, but with different parameters. (e.g., if $h$ was a linear classifier, then $f$ is also a linear classifier).

   **False:** For example, the functional form of a decision stump is a single axis-aligned split of the input space, but the functional form of the boosted classifier is linear combinations of decision stumps which can form a more complex (piecewise linear) decision boundary.

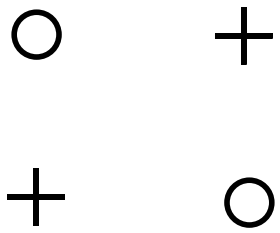
8. The depth of a learned decision tree can be larger than the number of training examples used to create the tree.

   **False:** Each split of the tree must correspond to at least one training example, therefore, if there are $n$ training examples, a path in the tree can have length at most $n$.

   **Note:** There is a pathological situation in which the depth of a learned decision tree can be larger than number of training examples $n$ - if the number of features is larger than $n$ and there exist training examples which have same feature values but different labels. Points have been given if you answered true and provided this explanation.

**For the following problems, circle the correct answers:**

1. Consider the following data set:



   Circle all of the classifiers that will achieve zero training error on this data set. (You may circle more than one.)

   (a) Logistic regression

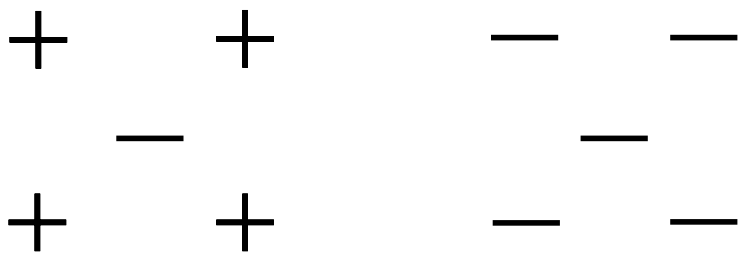
   (b) SVM (quadratic kernel)

   (c) Depth-2 ID3 decision trees

   (d) 3-NN classifier

   **Solution:** SVM (quad kernel) and Depth-2 ID3 decision trees

2. For the following dataset, circle the classifier which has larger Leave-One-Out Cross-validation error.



a) 1-NN

b) 3-NN

**Solution:** 1-NN since 1-NN CV err: 5/10, 3-NN CV err: 1/10

# 2 Bayes Optimal Classification [15 pts]

In classification, the loss function we usually want to minimize is the 0/1 loss:

$$\ell(f(x), y) = \mathbf{1}\{f(x) \neq y\}$$

where $f(x), y \in \{0, 1\}$ (i.e., binary classification). In this problem we will consider the effect of using an asymmetric loss function:

$$\ell_{\alpha,\beta}(f(x), y) = \alpha\mathbf{1}\{f(x) = 1, y = 0\} + \beta\mathbf{1}\{f(x) = 0, y = 1\}$$

Under this loss function, the two types of errors receive different weights, determined by $\alpha, \beta > 0$.

1. **[4 pts]** Determine the Bayes optimal classifier, i.e. the classifier that achieves minimum risk assuming $P(x, y)$ is known, for the loss $\ell_{\alpha,\beta}$ where $\alpha, \beta > 0$.

   **Solution:** We can write

$$\begin{aligned}
\arg\min_f \mathbb{E}\ell_{\alpha,\beta}(f(x), y) &= \arg\min_f \mathbb{E}_{X,Y}[\alpha\mathbf{1}\{f(X) = 1, Y = 0\} + \beta\mathbf{1}\{f(X) = 0, Y = 1\}] \\
&= \arg\min_f \mathbb{E}_X[\mathbb{E}_{Y|X}[\alpha\mathbf{1}\{f(X) = 1, Y = 0\} + \beta\mathbf{1}\{f(X) = 0, Y = 1\}]] \\
&= \arg\min_f \mathbb{E}_X[\int_y \alpha\mathbf{1}\{f(X) = 1, y = 0\} + \beta\mathbf{1}\{f(X) = 0, y = 1\} dP(y|x)] \\
&= \arg\min_f \int_x [\alpha\mathbf{1}\{f(x) = 1\}P(y = 0|x) + \beta\mathbf{1}\{f(x) = 0\}P(y = 1|x)] dP(x)
\end{aligned}$$

   We may minimize the integrand at each $x$ by taking:

$$f(x) = \begin{cases} 1 & \beta P(y = 1|x) \geq \alpha P(y = 0|x) \\ 0 & \alpha P(y = 0|x) > \beta P(y = 1|x). \end{cases}$$

2. **[3 pts]** Suppose that the class $y = 0$ is extremely uncommon (i.e., $P(y = 0)$ is small). This means that the classifier $f(x) = 1$ for all $x$ will have good risk. We may try to put the two classes on even footing by considering the risk:

$$R = P(f(x) = 1|y = 0) + P(f(x) = 0|y = 1)$$

   Show how this risk is equivalent to choosing a certain $\alpha, \beta$ and minimizing the risk where the loss function is $\ell_{\alpha,\beta}$.

   **Solution:** Notice that

$$\begin{aligned}
E\ell_{\alpha,\beta}(f(x), y) &= \alpha P(f(x) = 1, y = 0) + \beta P(f(x) = 0, y = 1) \\
&= \alpha P(f(x) = 1|y = 0)P(y = 0) + \beta P(f(x) = 0|y = 1)P(y = 1)
\end{aligned}$$

   which is same as the minimizer of the given risk $R$ if $\alpha = \frac{1}{P(y=0)}$ and $\beta = \frac{1}{P(y=1)}$.

3. [**4 pts**] Consider the following classification problem. I first choose the label $Y \sim$ Bernoulli($\frac{1}{2}$), which is 1 with probability $\frac{1}{2}$. If $Y = 1$, then $X \sim$ Bernoulli($p$); otherwise, $X \sim$ Bernoulli($q$). Assume that $p > q$. What is the Bayes optimal classifier, and what is its risk?

**Solution:** Since label is equally likely to be 1 or 0, to minimize prob of error simply predict the label for which feature value $X$ is most likely. Since $p > q$, $X = 1$ is most likely for $Y = 1$ and $X = 0$ is most likely for $Y = 0$. Hence $f^*(X) = X$. Baye's risk $= P(X \neq Y) = 1/2 \cdot (1 - p) + 1/2 \cdot q$.

**Formally:** Notice that since $Y \sim$ Bernoulli($\frac{1}{2}$), we have $P(Y = 1) = P(Y = 0) = 1/2$.

$$\begin{aligned} f^*(x) = \arg\max_y P(Y = y|X = x) &= \arg\max_y P(X = x|Y = y)P(Y = y) \\ &= \arg\max_y P(X = x|Y = y) \end{aligned}$$

Therefore, $f^*(1) = 1$ since $p = P(X = 1|Y = 1) > P(X = 1|Y = 0) = q$, and $f^*(0) = 0$ since $1 - p = P(X = 0|Y = 1) < P(X = 0|Y = 0) = 1 - q$. Hence $f^*(X) = X$. The risk is $R^* = P(f^*(X) \neq Y) = P(X \neq Y)$.

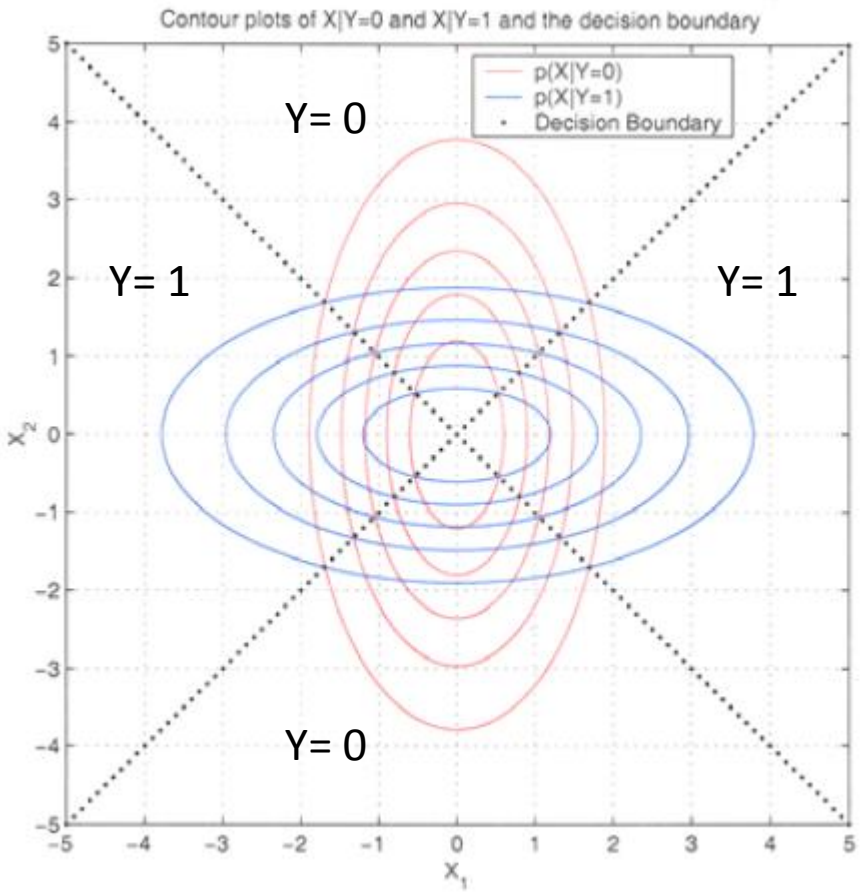
$$R^* = P(Y = 1)P(X = 0|Y = 1) + P(Y = 0)P(X = 1|Y = 0) = \frac{1}{2} \cdot (1 - p) + \frac{1}{2} \cdot q.$$

4. [**4 pts**] Now consider the regular 0/1 loss $\ell$, and assume that $P(y = 0) = P(y = 1) = 1/2$. Also, assume that the class-conditional densities are Gaussian with mean $\mu_0$ and co-variance $\Sigma_0$ under class 0, and mean $\mu_1$ and co-variance $\Sigma_1$ under class 1. Further, assume that $\mu_0 = \mu_1$.

For the following case, draw contours of the level sets of the class conditional densities and label them with $p(x|y = 0)$ and $p(x|y = 1)$. Also, draw the decision boundaries obtained using the Bayes optimal classifier in each case and indicate the regions where the classifier will predict class 0 and where it will predict class 1.

$$\Sigma_0 = \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix}, \Sigma_1 = \begin{bmatrix} 4 & 0 \\ 0 & 1 \end{bmatrix}$$

**Solution:** next page


Contour plots of X|Y=0 and X|Y=1 and the decision boundary
p(X|Y=0)
p(X|Y=1)
Decision Boundary
Y= 0
Y= 1
Y= 1
Y= 0
X2
X1

# 3 Logistic Regression [18 pts]

We consider here a discriminative approach for solving the classification problem illustrated in Figure 1.

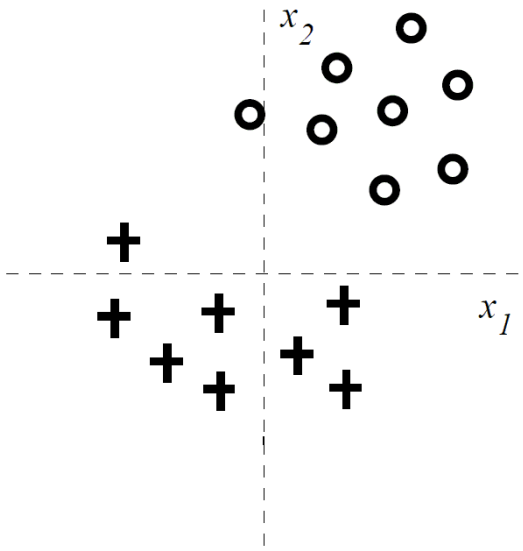



Figure 1: The 2-dimensional labeled training set, where '+' corresponds to class $y$=1 and 'O' corresponds to class $y = 0$.

1. We attempt to solve the binary classification task depicted in Figure 1 with the simple linear logistic regression model

$$P(y = 1|\vec{x}, \vec{w}) = g(w_0 + w_1x_1 + w_2x_2) = \frac{1}{1 + exp(-w_0 - w_1x_1 - w_2x_2)}.$$

Notice that the training data can be separated with *zero* training error with a linear separator.

Consider training *regularized* linear logistic regression models where we try to maximize

$$\sum_{i=1}^{n} \log\ (P(y_i|x_i, w_0, w_1, w_2)) - Cw_j^2$$

for very large $C$. The regularization penalties used in penalized conditional log-likelihood estimation are $-Cw_j^2$, where $j = \{0, 1, 2\}$. In other words, only one of the parameters is regularized in each case. Given the training data in Figure 1, how does the training error change with regularization of each parameter $w_j$? State whether the training error increases or stays the same (zero) for each $w_j$ for very large $C$. Provide a brief justification for each of your answers.

(a) By regularizing $w_2$ **[2 pts]**

**SOLUTION:** Increases. When we regularize $w_2$, the resulting boundary can rely less and less on the value of $x_2$ and therefore becomes more vertical. For very large $C$, the training error increases as there is no good linear vertical separator of the training data.

(b) By regularizing $w_1$ **[2 pts]**

**SOLUTION:** Remains the same. When we regularize $w_1$, the resulting boundary can rely less and less on the value of $x_1$ and therefore becomes more horizontal and the training data can be separated with *zero* training error with a horizontal linear separator.

(c) By regularizing $w_0$ **[2 pts]**

**SOLUTION:** Increases. When we regularize $w_0$, then the boundary will eventually go through the origin (bias term set to zero). Based on the figure, we can *not* find a linear boundary through the origin with *zero* error. The best we can get is one error.

2. If we change the form of regularization to L1-norm (absolute value) and regularize $w_1$ and $w_2$ only (but not $w_0$), we get the following penalized log-likelihood

$$\sum_{i=1}^{n} \log P(y_i|x_i, w_0, w_1, w_2) - C(|w_1| + |w_2|).$$

Consider again the problem in Figure 1 and the same linear logistic regression model $P(y = 1|\vec{x}, \vec{w}) = g(w_0 + w_1x_1 + w_2x_2)$.

(a) **[3 pts]** As we increase the regularization parameter $C$ which of the following scenarios do you expect to observe? (Choose only one) Briefly explain your choice:

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

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

( ) $w_1$ and $w_2$ will become zero simultaneously.

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

**SOLUTION:** First $w_1$ will become 0, then $w_2$.

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 does not help classification why would we pay the penalty? Also, 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 $w_2$.

(b) **[3 pts]** For very large $C$, with the same L1-norm regularization for $w_1$ and $w_2$ as above, which value(s) do you expect $w_0$ to take? Explain briefly. (Note that the number of points from each class is the same.) (You can give a range of values for $w_0$ if you deem necessary).

**SOLUTION:** For very large $C$, we argued that both $w_1$ and $w_2$ will go to zero. Note that when $w_1 = w_2 = 0$, the log-probability of labels becomes a finite value, which is equal to n log(0.5), i.e. $w_0 = 0$. In other words, $P(y = 1|\vec{x}, \vec{w})$=$P(y = 0|\vec{x}, \vec{w})$=0.5. We expect so because the number of elements in each class is the same and so we would like to predict each one with the same probability, and $w_0$=0 makes $P(y = 1|\vec{x}, \vec{w})$=0.5.

(c) **[3 pts]** Assume that we obtain more data points from the '+' class that corresponds to $y$=1 so that the class labels become unbalanced. Again for very large $C$, with the same L1-norm regularization for $w_1$ and $w_2$ as above, which value(s) do you expect $w_0$ to take? Explain briefly. (You can give a range of values for $w_0$ if you deem necessary).

**SOLUTION:** For very large $C$, we argued that both $w_1$ and $w_2$ will go to zero. With unbalanced classes where the number of '+' labels are greater than that of 'o' labels, we want to have $P(y = 1|\vec{x}, \vec{w}) > P(y = 0|\vec{x}, \vec{w})$. For that to happen the value of $w_0$ should be greater than zero which makes $P(y = 1|\vec{x}, \vec{w}) > 0.5$.

# 4 Kernel regression [16 pts]

Now lets consider the non-parametric kernel regression setting. In this problem, you will investigate univariate locally linear regression where the estimator is of the form:

$$\widehat{f}(x) = \beta_1 + \beta_2 x$$

and the solution for parameter vector $\beta = [\beta_1 \ \beta_2]$ is obtained by minimizing the weighted least square error:

$$J(\beta_1, \beta_2) = \sum_{i=1}^{n} W_i(x)(Y_i - \beta_1 - \beta_2 X_i)^2 \quad \text{where} \quad W_i(x) = \frac{K\left(\frac{X_i - x}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{X_i - x}{h}\right)},$$

where $K$ is a kernel with bandwidth $h$. Observe that the weighted least squares error can be expressed in matrix form as

$$J(\beta_1, \beta_2) = (Y - A\beta)^T W (Y - A\beta),$$

where $Y$ is a vector of $n$ labels in the training example, $W$ is a $n \times n$ diagonal matrix with weight of each training example on the diagonal, and

$$A = \begin{bmatrix} 1 \ X_1 \\ 1 \ X_2 \\ \cdots \\ 1 \ X_n \end{bmatrix}$$

1. [**4 pts**] Derive an expression in matrix form for the solution vector $\hat{\beta}$ that minimizes the weighted least square.

   **Solution:** Differentiating the objective function wrt $\beta$, we have:

   $$\frac{\partial J(\beta)}{\beta} = 2A^T W A\beta - 2A^T W^T Y.$$

   Therefore, the solution $\hat{\beta}$ satisfies the following normal equations:

   $$A^T W A\beta = A^T W^T Y$$

   And if $A^T W A$ is invertible, then the solution is $\hat{\beta} = (A^T W A)^{-1} A^T W^T Y$. (Note that $W = W^T$, so the solution can be written in terms of either).

2. [**3 pts**] When is the above solution unique?

   **Solution:** When $A^T W A$ is invertible. Since $W$ is a diagonal matrix, $A^T W A = (W^{1/2}A)^T (W^{1/2}A)$ and hence rank$(A^T W A) = \min(n, 2)$ - Refer TK's recitation notes. Since a matrix is invertible if it is full rank, a unique solution exists if $n \geq 2$.

3. [**3 pts**] If the solution is not unique, one approach is to optimize the objective function $J$ using gradient descent. Write the update equation for gradient descent in this case. Note: Your answer must be expressed in terms of the matrices defined above.

**Solution:** Let $\alpha > 0$ denote the step-size.

$$\begin{aligned}\beta^{(t+1)} &= \beta^{(t)} - \frac{\alpha}{2}\frac{\partial J(\beta)}{\partial \beta} \\ &= \beta^{(t)} - \alpha A^T W(A\beta - Y)\end{aligned}$$

4. [**3 pts**] Can you identify the signal plus noise model under which maximizing the likelihood (MLE) corresponds to the weighted least squares formulation mentioned above?

   **Solution:** $Y = \beta_1 + \beta_2 X + \epsilon$, where $\epsilon_i \overset{iid}{\sim} \mathcal{N}(0, \sigma_i^2)$ for $i = 1, \ldots, n$. Here $\sigma_i^2 \propto 1/W_i(x)$.

5. [**3 pts**] Why is the above setting non-parametric? Mention one advantage and one disadvantage of nonparametric techniques over parametric techniques.

   **Solution:** The above setting is non-parametric since it performs locally linear fits, therefore number of parameters scale with data. Notice that $W_i(x)$, and hence the solution $\hat{\beta}$, depends on $x$. Thus we are fitting the parameters to every point $x$ - therefore total number of parameters can be larger than $n$.

   Nonparametric techniques do not place very strict assumptions on the form of the underlying distribution or regression function, but are typically computationally expensive and require large number of training examples.

# 5 SVM [16 pts]

## 5.1 L2 SVM

Let $\{(\mathbf{x}_i, y_i)\}_{i=1}^l$ be a set of $l$ training pairs of feature vectors and labels. We consider binary classification, and assume $y_i \in \{-1, +1\}\ \forall i$. The following is the primal formulation of L2 SVM, a variant of the standard SVM obtained by squaring the hinge loss:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^\top\mathbf{w} + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top\mathbf{x}_i + b) \ \geq\ 1 - \xi_i, \quad i \in \{1, \ldots, l\}, \\ & \xi_i \ \geq\ 0, \quad i \in \{1, \ldots, l\}. \end{aligned}$$

1. **[4 pts]** Show that removing the last set of constraints $\{\xi_i \geq 0\ \forall i\}$ does not change the optimal solution to the primal problem.

   **Solution:** Let $(\mathbf{w}^*, b^*, \boldsymbol{\xi}^*)$ be the optimal solution to the problem without the last set of constraints. It suffices to show that $\xi_i^* \geq 0\ \forall i$. Suppose it is not the case, then there exists some $\xi_j^* < 0$. Then we have

   $$y_j((\mathbf{w}^*)^\top\mathbf{x}_j + b^*) \ \geq\ 1 - \xi_j^* \ >\ 1,$$

   implying that $\xi_j' = 0$ is a feasible solution and yet gives a smaller objective value since $(\xi_j')^2 = 0 < (\xi_j^*)^2$, a contradiction to the assumption that $\xi_j^*$ is optimal.

2. **[3 pts]** After removing the last set of constraints, we get a simpler problem:

   $$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \frac{1}{2}\mathbf{w}^\top\mathbf{w} + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top\mathbf{x}_i + b) \ \geq\ 1 - \xi_i, \quad i \in \{1, \ldots, l\}. \end{aligned} \tag{1}$$

   Give the Lagrangian of (1).

   **Solution:** The Lagrangian is

   $$L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) \ :=\ \frac{1}{2}\mathbf{w}^\top\mathbf{w} + \frac{C}{2}\sum_{i=1}^{l}\xi_i^2 - \sum_{i=1}^{l}\alpha_i(y_i(\mathbf{w}^\top\mathbf{x}_i + b) - 1 + \xi_i),$$

   where $\alpha_i \geq 0,\ \forall i$ are the Lagrange multipliers.

3. **[6 pts]** Derive the dual of (1). How is it different from the dual of the standard SVM with the hinge loss?

**Solution:** Taking partial derivatives of the Lagrangian wrt $\mathbf{w}$, $b$ and $\xi_i$,

$$\nabla_{\mathbf{w}} L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) \;=\; 0 \iff \mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i \;,$$

$$\partial_b L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) \;=\; 0 \iff \sum_{i=1}^{l} \alpha_i y_i \;=\; 0,$$

$$\partial_{\xi_i} L(\mathbf{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}) \;=\; 0 \iff \xi_i = \alpha_i / C \;.$$

Plugging these back to the Lagrangian, rearranging terms and keeping constraints on the Lagrange multipliers we obtain the dual

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & -\frac{1}{2}\boldsymbol{\alpha}^\top (Q + I/C)\boldsymbol{\alpha} + \mathbf{1}^\top \boldsymbol{\alpha} \\ \text{s.t.} \quad & \mathbf{y}^\top \boldsymbol{\alpha} \;=\; 0, \quad \alpha_i \;\geq\; 0 \;\forall i, \end{aligned}$$

where $\mathbf{1}$ is a vector of ones, $I$ is the identity matrix, $\mathbf{y}$ is the vector of labels $y_i$'s, and $Q$ is the $l$-by-$l$ kernel matrix such that $Q_{ij} = y_i y_j \mathbf{x}_i^\top \mathbf{x}_j$. Compared with the dual of the standard SVM, the quadratic term is regularized by an additional positive diagonal matrix, and thus has stronger convexity leading to faster convergence. The other difference is that the dual variables here are only bounded from below, but in the standard SVM the dual variables are bounded both from above (by $C$) and from below. In fact, for L2 svms the solution does not depend on the tradeoff parameter $C$.

## 5.2 Leave-one-out Error and Support Vectors

**[3 pts]** Consider the standard two-class SVM with the hinge loss. Argue that under a given value of $C$,

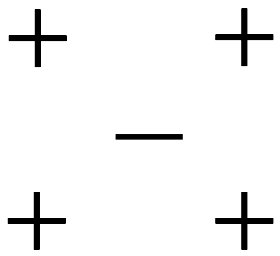
$$\text{LOO error} \;\leq\; \frac{\#\text{SVs}}{l},$$

where $l$ is the size of the training data and #SVs is the number of support vectors obtained by training SVM on the entire set of training data.

**Solution:** Since the decision function only depends on the support vectors, removing a non-support vector from the training data and then re-training an SVM would lead to the same decision function. Also, non-support vectors must be classified correctly. As a result, errors found in the leave-one-out validation must be caused by removing the support vectors, proving the desired result.

# 6 Boosting [15 pts]

1. Consider training a boosting classifier using decision stumps on the following data set:

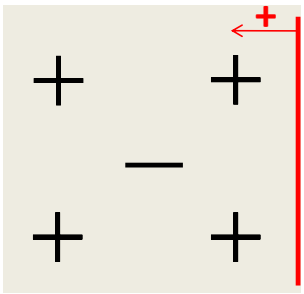


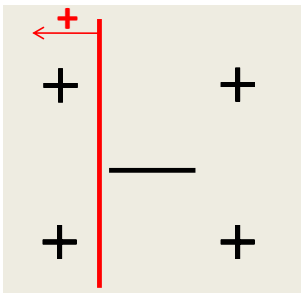
(a) **[3 pts]** Which examples will have their weights increased at the end of the first iteration? Circle them.

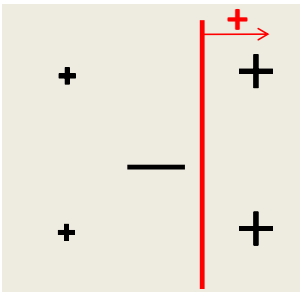
**Solution:** The negative example since the decision stump with least error in first iteration is constant over the whole domain. Notice this decision stump only predicts incorrectly on the negative example, whereas any other decision stump predicts incorrectly on at least two training examples.

(b) **[3 pts]** How many iterations will it take to achieve zero training error? Explain.

**Solution:** At least three iterations. The first iteration misclassifies the negative example, the second iteration misclassifies two of the positive examples as the negative one has large weight. The third iteration is needed since a weighted sum of the first two decision stumps can't yield zero training error, and misclassifies the other two positive examples. See Figures below.







(c) **[3 pts]** Can you add one more example to the training set so that boosting will achieve zero training error in two steps? If not, explain why.

**Solution:** No. Notice that the simplest case is adding one more negative example in center or one more positive example between any two positive examples, as it still yields three decision regions with axis-aligned boundaries. If only two steps were enough, then a linear combination of only two decision stumps $sign(\alpha_1 h_1(x) + \alpha_2 h_2(x))$

should be able to yield three decision regions. Also notice that at least one of $h_1$ or $h_2$ misclassifies two positive examples. If only $h_2$ misclassifies two positive examples, the possible decisions are (1) $sign(\alpha_1 - \alpha_2)$ on those two positive examples, (2) $sign(\alpha_1 + \alpha_2)$ on the remaining positive examples and (3) $sign(\alpha_1 - \alpha_2)$ on the negative examples - which don't yield zero training error since signs on (1) and (3) agree. If both $h_1$ and $h_2$ misclassify two positive examples, we have (1) $sign(\alpha_1 - \alpha_2)$ on two positive examples, (2) $sign(-\alpha_1 + \alpha_2)$ on the remaining positive examples and (3) $sign(-\alpha_1 - \alpha_2)$ on the negative - which again don't yield zero training error since signs on (1) and (2) don't agree.

2. [**2 pts**] Why do we want to use "weak" learners when boosting?

   **Solution:** To prevent overfitting, since the complexity of the overall learner increases at each step. Starting with weak learners implies the final classifier will be less likely to overfit.

3. [**4 pts**] Suppose AdaBoost is run on $m$ training examples, and suppose on each round that the weighted training error $\epsilon_t$ of the $t^{th}$ weak hypothesis is at most $1/2 - \gamma$, for some number $\gamma > 0$. After how many iterations, $T$, will the combined hypothesis $H$ be consistent with the $m$ training examples, i.e., achieves zero training error? Your answer should only be expressed in terms of $m$ and $\gamma$. (Hint: What is the training error when 1 example is misclassified?)

   **Solution:** Training error when 1 example is misclassified = $1/m$. Therefore, we need to guarantee that training error is $< 1/m$. Since $\epsilon_t \leq 1/2 - \gamma$, from class notes we know that

   $$\text{Training err of the combined hypothesis } H \leq \exp(-2T\gamma^2)$$

   The upper bound is $< 1/m$ if $T > \ln m/2\gamma^2$.