exping hw5 solution

3 Adaboost

3.1 True or False

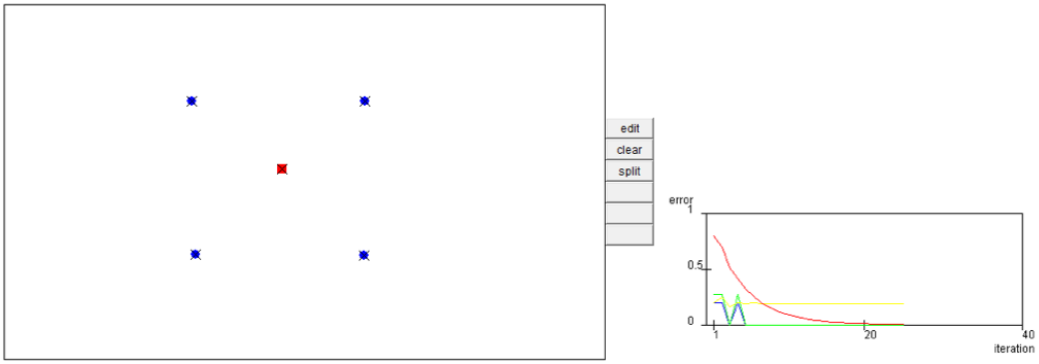
- (a) False. Adaboost guarantees that an upper bound on the training error never increases, but does not guarantee that the training error itself never increases.
- (b) False. If the weak learners h_t do not perform better than random guessing, i.e. $\epsilon_t = 0.5$, then Adaboost will assign them zero weight:

$$\begin{aligned} \alpha_t &= \frac{1}{2} \ln \left(\frac{1-\epsilon_t}{\epsilon_t} \right) \\ &= \frac{1}{2} \ln 1 \\ &= 0. \end{aligned}$$

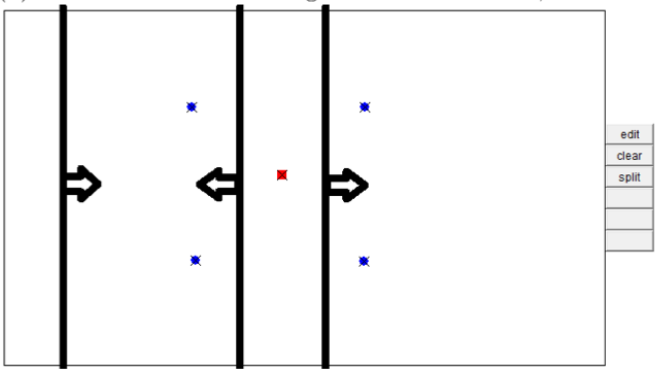
  Thus, the weak learner contributes nothing to the learnt function $f(x)$, so the true error does not decrease.

3.2 Boosting

- (a) Training samples on the left, error graph (after adding additional test samples) on the right:

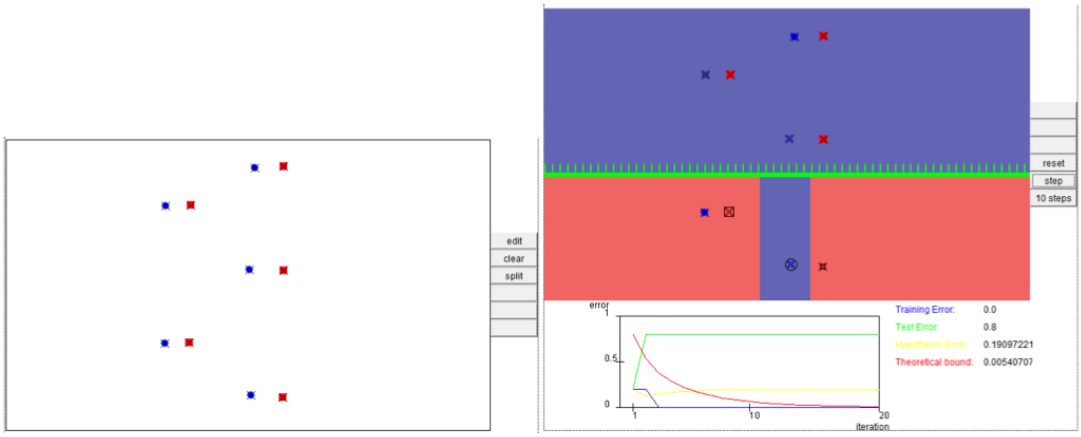


- (b) We can achieve 0 training error in 3 iterations, i.e. 3 weak learners (shown below).



Proof: Clearly, 1 weak learner cannot achieve 0 training error. Moreover, if we only use the 2 weak learners in the middle, then we cannot get the learnt Adaboost function $f(x)$ to be positive on all 4 blue points; the 3rd weak learner on the left is required for this to happen.

- (c) Adaboost overfits on the following dataset (left). The train/test split and error graph are shown on the right. Test points are solid blue/red shapes, while training points are dithered blue/red shapes. Only 0.2 of the test points are classified correctly.



- (d) Terminate Adaboost early, before the training error reaches zero. This produces a lower-complexity classifier (i.e. fewer weak learners), which has more bias but less variance — and hence is less prone to overfitting. For the dataset in part (c), just one iteration would have given 0.8 test accuracy, but two or more iterations will give only 0.2 test accuracy.

aarti hw6

3 Boosting [Shing-hon Lau, 25 points]

1. [4 pts] Suppose I have the 2-dimensional dataset depicted in Figure 1. Will Adaboost (with Decision Stumps as the weak classifier) ever achieve better than 50% classification accuracy on this dataset? Why or why not? Briefly justify your answer.

   ★ **SOLUTION:** Adaboost will never achieve better than 50% classification accuracy on this dataset since every single split produces an error of 50%.

2. [4 pts] Suppose AdaBoost is run on m training examples, and suppose on each round that the weighted training error ϵ_t of the t^{th} weak hypothesis is at most γ, for some number $0.5 > \gamma > 0$. 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 γ. (Hint: What is the training error when 1 example is misclassified?)

   ★ **SOLUTION:** We have that the training error of the final classifier, H, is bounded by:
   $\frac{1}{m}\sum_{i=1}^{m}\delta(H(x_i) \neq y_i) \leq exp(-2\sum_{t=1}^{T}(0.5-\epsilon_t)^2)$.
   We can ensure that H is consistent with the m training points if we bound the training error by $\frac{1}{m}$. Thus, we would like to choose T such that:
   $\frac{1}{m} \geq exp(-2\sum_{t=1}^{T}(0.5-\epsilon_t)^2)$
   However, it is difficult to operate with the expression on the right-hand side, so we instead choose T such that:
   $\frac{1}{m} \geq exp(-2T(0.5-\gamma_t)^2)$
   Note that this is OK since $\frac{1}{m} \geq exp(-2T(0.5-\gamma_t)^2) \geq exp(-2\sum_{t=1}^{T}(0.5-\epsilon_t)^2)$
   Solving gives us that:
   $\frac{1}{m} \geq exp(-2T(0.5-\gamma_t)^2)$

$\Rightarrow ln(m) \leq 2T(0.5-\gamma_t)^2$
$\Rightarrow \frac{ln(m)}{2(0.5-\gamma_t)^2} \leq T$

For the next six questions, consider the 1-dimensional dataset depicted in Figure 2. Suppose that we are using Adaboost with Decision Stumps as the weak classifier.

3. [3 pts] Draw the decision boundary of the first classifier, h_1. Indicate which side is classified as the + class.

   ★ **SOLUTION:** The boundary goes between the 3 '+' points and the 3 '-' points, with the '+' side on the left.

4. [3 pts] Compute and report ϵ_1 and α_1. What is the classification accuracy if we stop Adaboost here?

   ★ **SOLUTION:** $\epsilon_1 = \frac{1}{7}$ and $\alpha_1 = \frac{1}{2}ln\frac{6/7}{1/7} = \frac{1}{2}ln(6) = 0.8959$
   The classification accuracy would be $\frac{6}{7}$.

5. [3 pts] What are the new weights, $D_2(i)$, for the seven points? (Hint: Remember to normalize the weights by Z.)

   ★ **SOLUTION:** $Z_1 = \frac{6}{7}exp(-0.8959) + \frac{1}{7}exp(0.8959) = 0.6998542$
   $D_2(i) = \frac{(1/7)exp(-0.8959)}{Z_1} = 0.083333 \quad \forall i = 1, ..., 6$
   $D_2(7) = \frac{(1/7)exp(0.8959)}{Z_1} = 0.5$

6. [3 pts] Draw the decision boundary of the second classifier, h_2. Again, indicate which side is classified as the + class.

   ★ **SOLUTION:** The boundary goes between the single '+' point and the 3 '-' points, with the '+' side on the right.

7. [2 pts] Which point(s) will have the lowest weight after the second iteration of Adaboost is finished?

   ★ **SOLUTION:** The 3 '-' points will have the lowest weight after the second iteration since they were classified correctly by both h_1 and h_2.

8. [3 pts] Does the classification accuracy improve between first and second iterations of Adaboost? Explain briefly why the accuracy does (or does not) improve.

SOLUTION: No, the classification accuracy does not improve. In fact, all of the points are classified the same after 1 and 2 iterations.

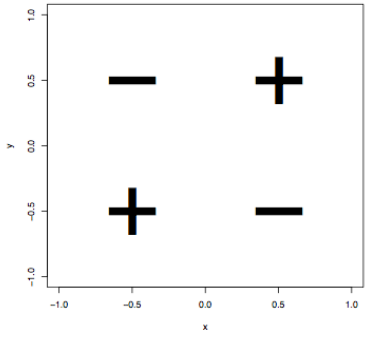


Figure 1: XOR dataset

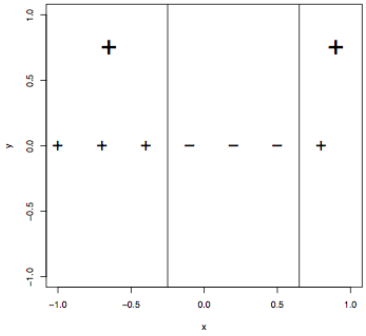


Figure 2: Seven point dataset

2.1 Loss functions [5pts]

Match each predictor to the loss function it typically minimizes:

boosting	⇔	exponential loss
logistic regression	⇔	log loss
k-NN classifier	⇔	0/1 loss
linear regression	⇔	squared loss
SVM classification	⇔	hinge loss

2.2 Decision boundaries [10 pts]

For each classifier, circle the type(s) of decision boundary it can yield for a binary classification problem. In some cases, more than 1 option may be correct. Circle all options that you think are correct.

decision trees:	linear, piecewise linear
(non-kernel) SVM:	linear
logistic regression	linear
Gaussian Naive Bayes	linear, quadratic
boosting	linear, piecewise linear, quadratic

1 [10 points] Big Picture

Following the example given, add 10 edges to Figure 1 relating the pair of algorithms. Each edge should be labeled with one characteristic the methods share, and one difference. These labels should be short and address basic concepts, such as types of learning problems, loss functions, and hypothesis spaces.

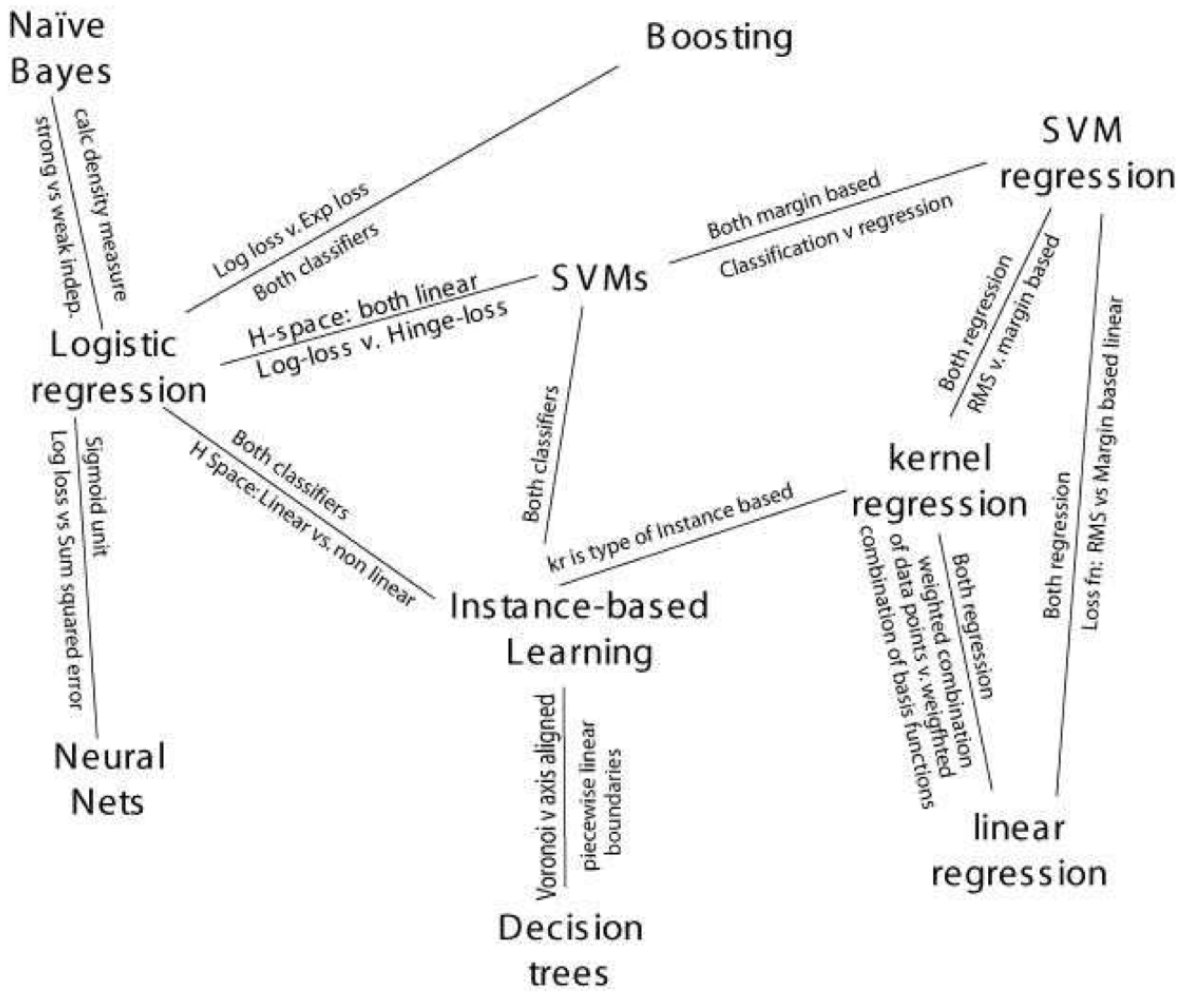


Figure 2: Big picture solutions.

6. [2 points] **true/false** In AdaBoost weights of the misclassified examples go up by the same multiplicative factor.

★ **SOLUTION:** True, follows from the update equation.

7. [2 points] **true/false** In AdaBoost, weighted training error ϵ_t of the t^{th} weak classifier on training data with weights D_t tends to increase as a function of t.

★ **SOLUTION:** True. 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 classifiers. The weighted training error ϵ_t of the t^{th} weak classifier on the training data therefore tends to increase.

8. [2 points] **true/false** AdaBoost will eventually give zero training error regardless of the type of weak classifier it uses, provided enough iterations are performed.

★ **SOLUTION:** 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. For example consider the EXOR example(In hw2 we worked with a *rotated* EXOR toy dataset) with decision stumps as weak classifiers. No matter how many iterations are performed zero training error will not be achieved.

3 [10 points] Feature selection with boosting

Consider a text classification task, such that the document X can be expressed as a binary feature vector of the words. More formally $X = [X_1, X_2, X_3, ..., X_m]$, where $X_j = 1$ if word j is present in document X, and zero otherwise. Consider using the AdaBoost algorithm with a simple weak learner, namely

$$h(X;\theta) = yX_j$$
$$\theta = \{j, y\}\ j \text{ is the word selector ; } y \text{ is the associated class}$$
$$y \in \{-1, 1\}$$

More intuitively, each weak learner is a word associated with a class label. For example if we had a word `football`, and classes {`sports`,`non-sports`}, then we will have two weak learners from this word, namely

- *Predict* **sports** *if document has word* **football**
- *Predict* **non-sports** *if document has word* **football**.

1. [2 points] How many weak learners are there ?

★ **SOLUTION:** Two weak learners for each word, i.e. 2m weak learners.

2. This boosting algorithm can be used for feature selection. We run the algorithm and select the features in the *order in which they were identified* by the algorithm.

   (a) [4 points] Can this boosting algorithm select the same weak classifier more than once? Explain.

   ★ **SOLUTION:** The boosting algorithm optimizes each new α by assuming that all the previous votes remain fixed. It therefore does not optimize these coefficients jointly. The only way to correct the votes assigned to a weak learner later on is to introduce the same weak learner again. Since we only have a discrete set of possible weak learners here, it also makes sense to talk about selecting the exact same weak learner again.

   (b) [4 points] Consider ranking the features based on their individual mutual information with the class variable y, i.e. $\hat{I}(y; X_j)$. Will this ranking be more informative than the ranking returned by AdaBoost ? Explain.

   ★ **SOLUTION:** The boosting algorithm generates a linear combination of weak classifiers (here features). The algorithm therefore evaluates each new weak classifier (feature) relative to a linear prediction based on those already included. The mutual information criterion considers each feature individually and is therefore unable to recognize how multiple features might interact to benefit linear prediction.

1. Boosting decision stumps can result in a quadratic decision boundary.

   False. The sign of a finite linear combination of decision stumps always results in a piecewise linear decision boundary.

   (T) The coefficients α assigned to the classifiers assembled by AdaBoost are always non-negative.