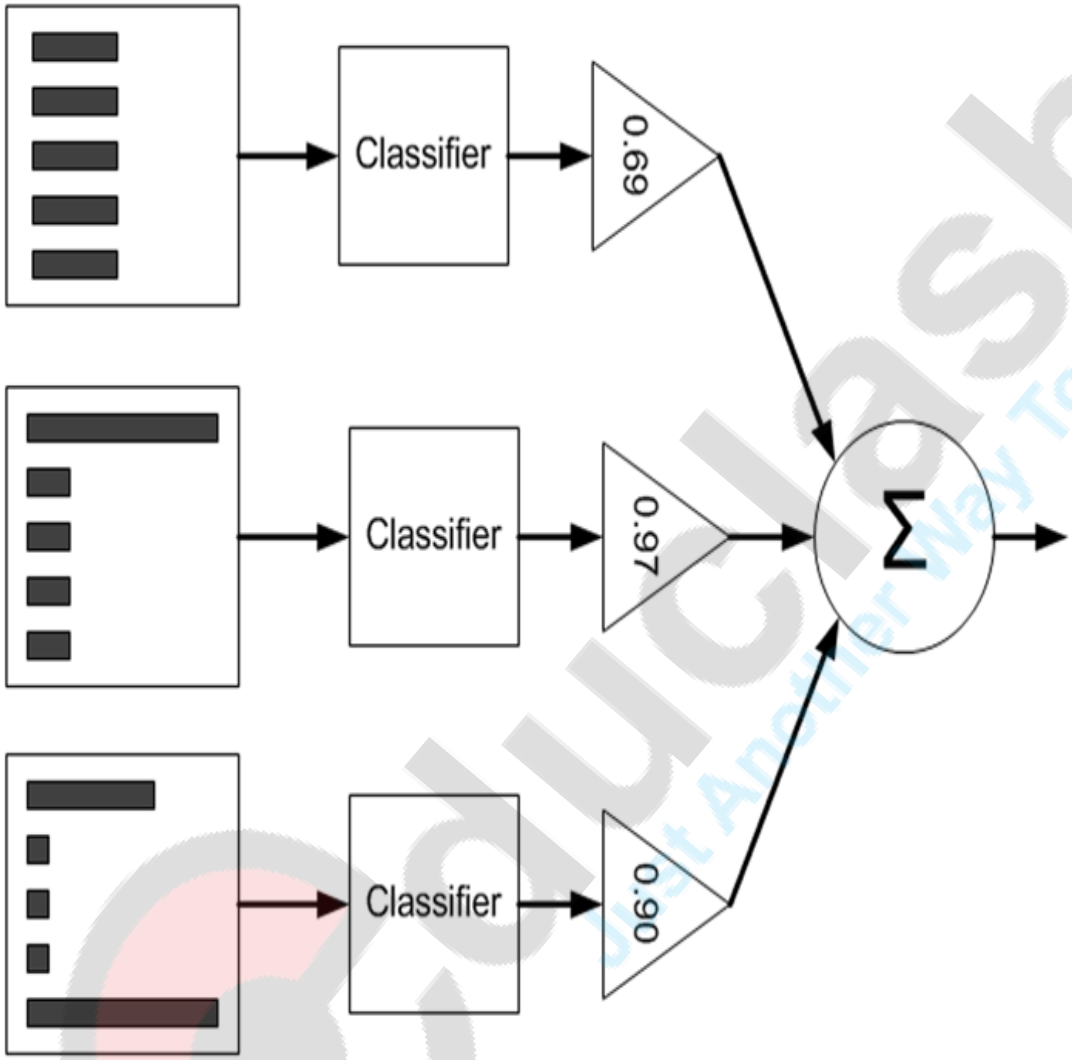
Adaboost

AdaBoost is short for *adaptive boosting. AdaBoost works this way: A weight is applied* to every example in the training data. We'll call the weight vector D. Initially, these weights are all equal. A weak classifier is first trained on the training data. The errors from the weak classifier are calculated, and the weak classifier is trained a second time with the same dataset. This second time the weak classifier is trained, the weights of the training set are adjusted so the examples properly classified the first time are weighted less and the examples incorrectly classified in the first iteration are weighted more. To get one answer from all of these weak classifiers, AdaBoost assigns α values to each of the classifiers. The α values are based on the error of each weak classifier.

The error Ɛ is given by

$$\varepsilon = \frac{\textit{number of incorrectly classified examples}}{\textit{total number of examples}}$$

- $\alpha = \frac{1}{2} \ln(1-\varepsilon)/\varepsilon$



Schematic representation of AdaBoost; with the dataset on the left side, the different widths of the bars represent weights applied to each instance. The weighted predictions
pass through a classifier, which is then weighted by the triangles (values). The weightedoutput of each triangle is summed up in the circle, which produces the final

- After you calculate □, you can update the weight vector D so that the examples that are
- correctly classified will decrease in weight and the misclassified examples will increase
- in weight. D Is given by

- $D_i^{(t+1)} = \frac{D_i^t * e^{-\alpha}}{Sum(D)}$ *if correctly predicted*
- $D_i^{(t+1)} = \frac{D_i^t * e^{\alpha}}{Sum(D)}$ *if incorrectly predicted*

- After D is calculated, AdaBoost starts on the next iteration. The AdaBoost algorithm
- repeats the training and weight-adjusting iterations until the training error is 0
- or until the number of weak classifiers reaches a user-defined value.

Boosting

- Definition of Boosting[1]:

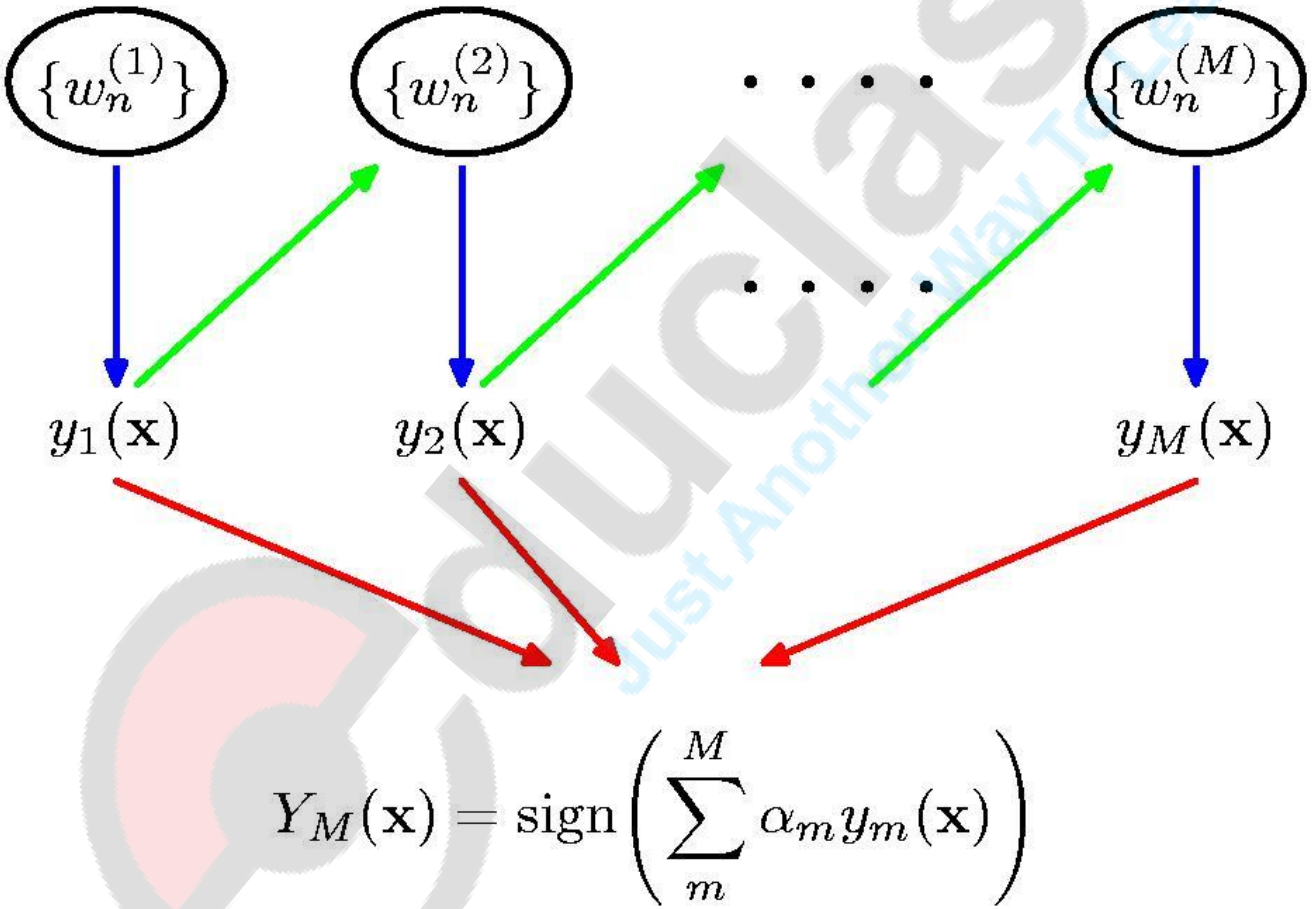
  Boosting refers to a general method of producing a very accurate prediction rule by combining rough and moderately inaccurate rules-of-thumb.

- Intuition:

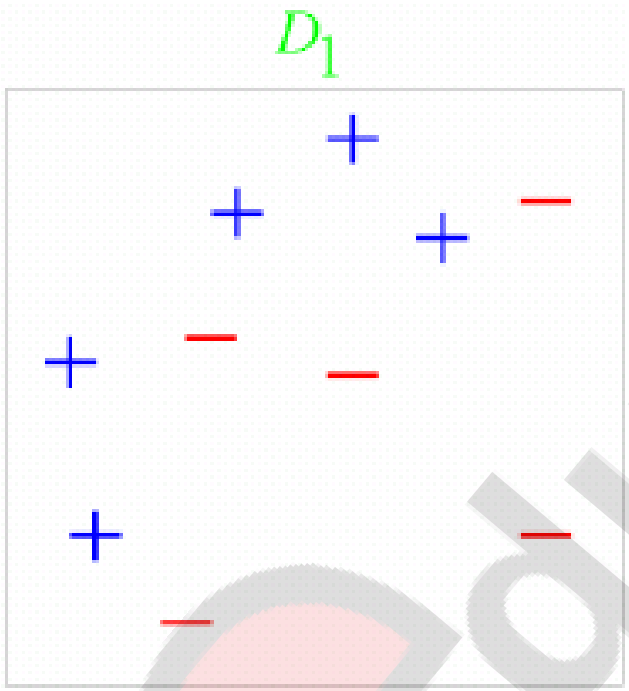
  1) No learner is always the best;

  2) Construct a set of base-learners which when combined achieves higher accuracy

Schematic illustration of the boosting Classifier

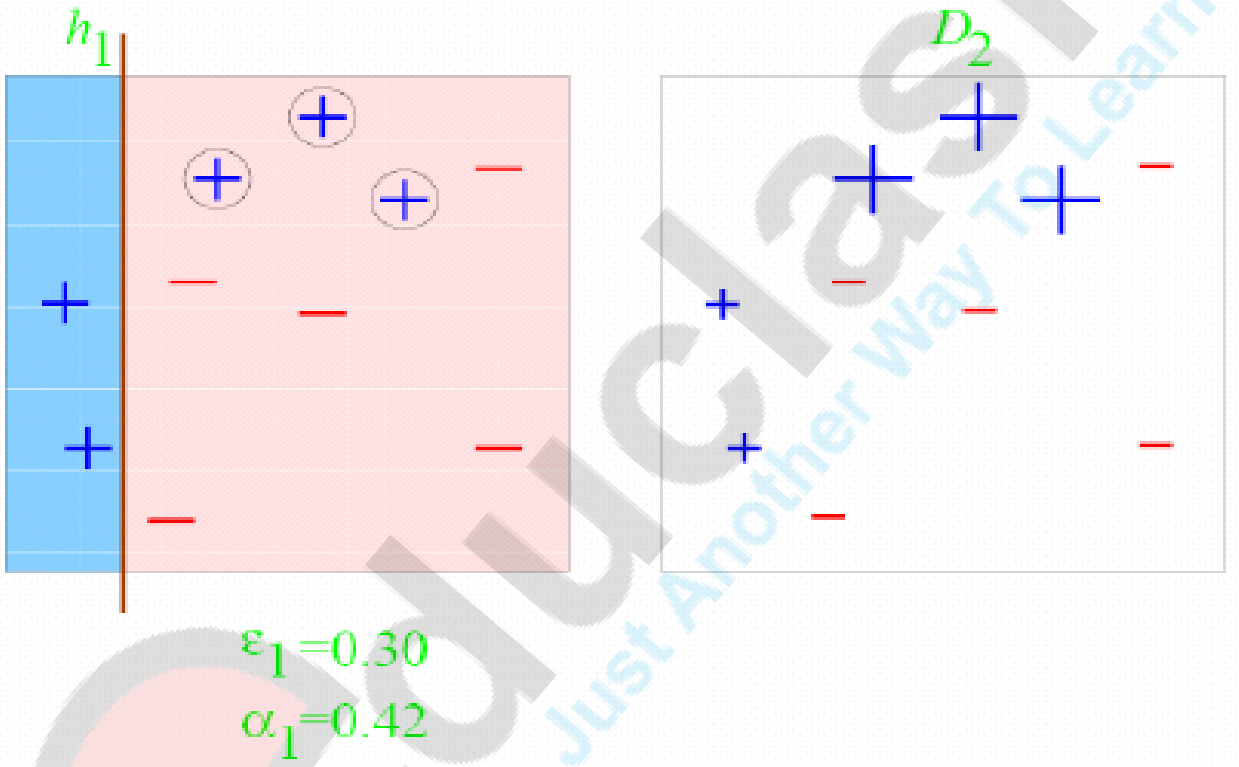


A toy example[2]



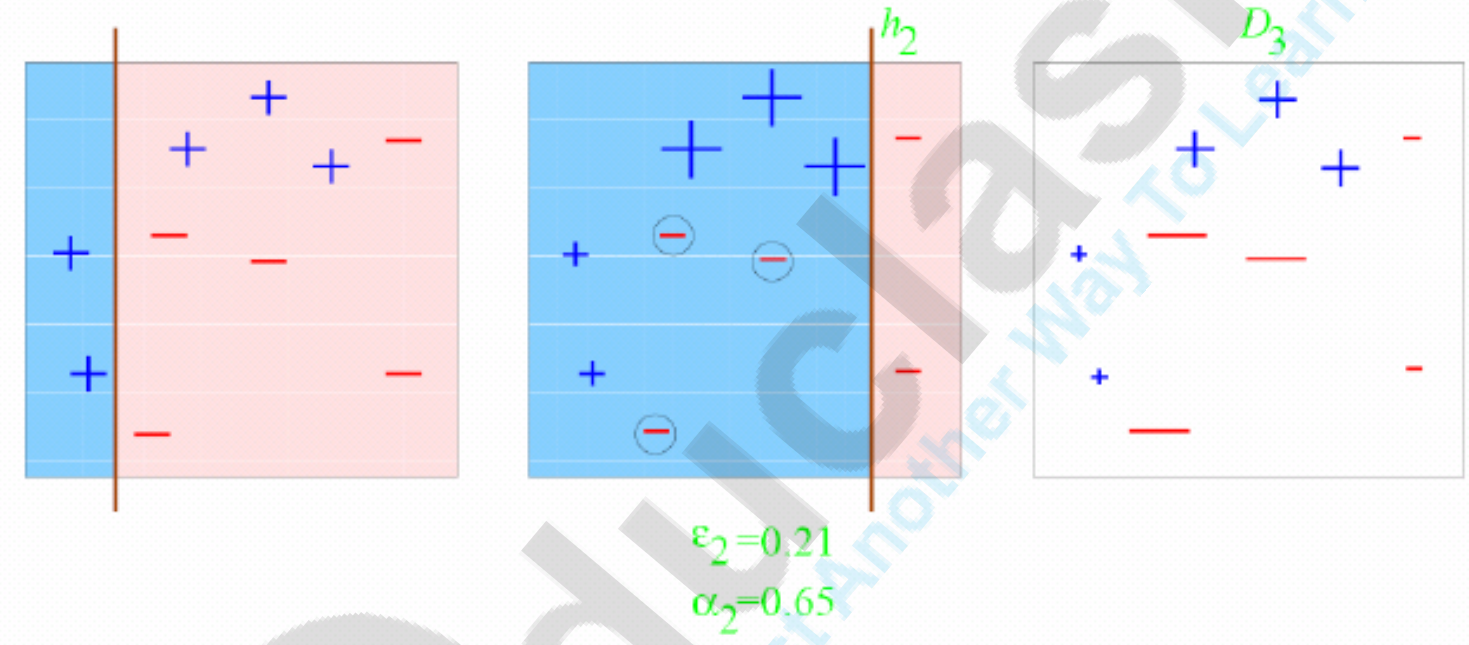
Training set: 10 points
(represented by plus or minus)
Original Status: Equal Weights
for all training samples

A toy example(cont'd)



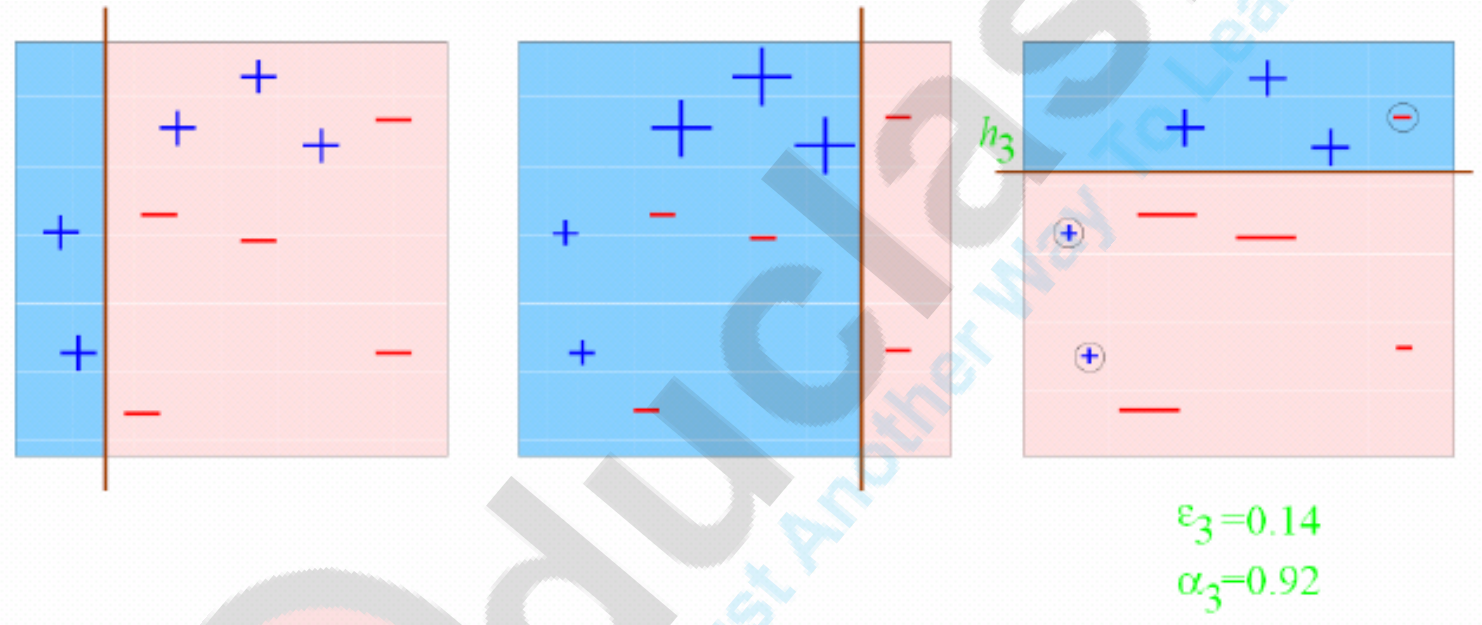
Round 1: Three "plus" points are not correctly classified;
They are given higher weights.

A toy example(cont'd)



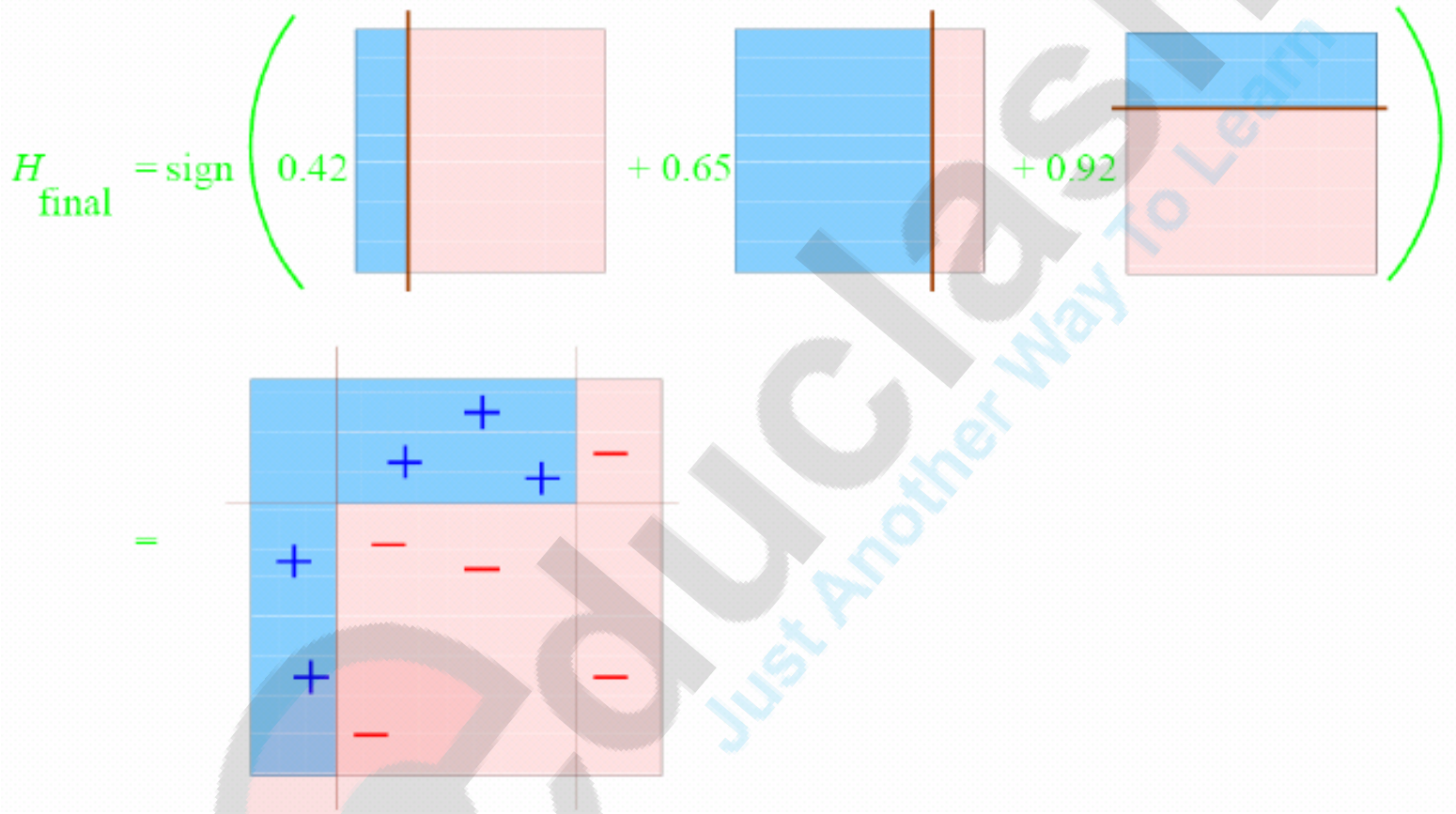
Round 2: Three "minuse" points are not correctly classified;
They are given higher weights.

A toy example(cont'd)



Round 3: One "minuse" and two "plus" points are not correctly classified;
They are given higher weights.

A toy example(cont'd)



Final Classifier: integrate the three "weak" classifiers and obtain a final strong classifier.