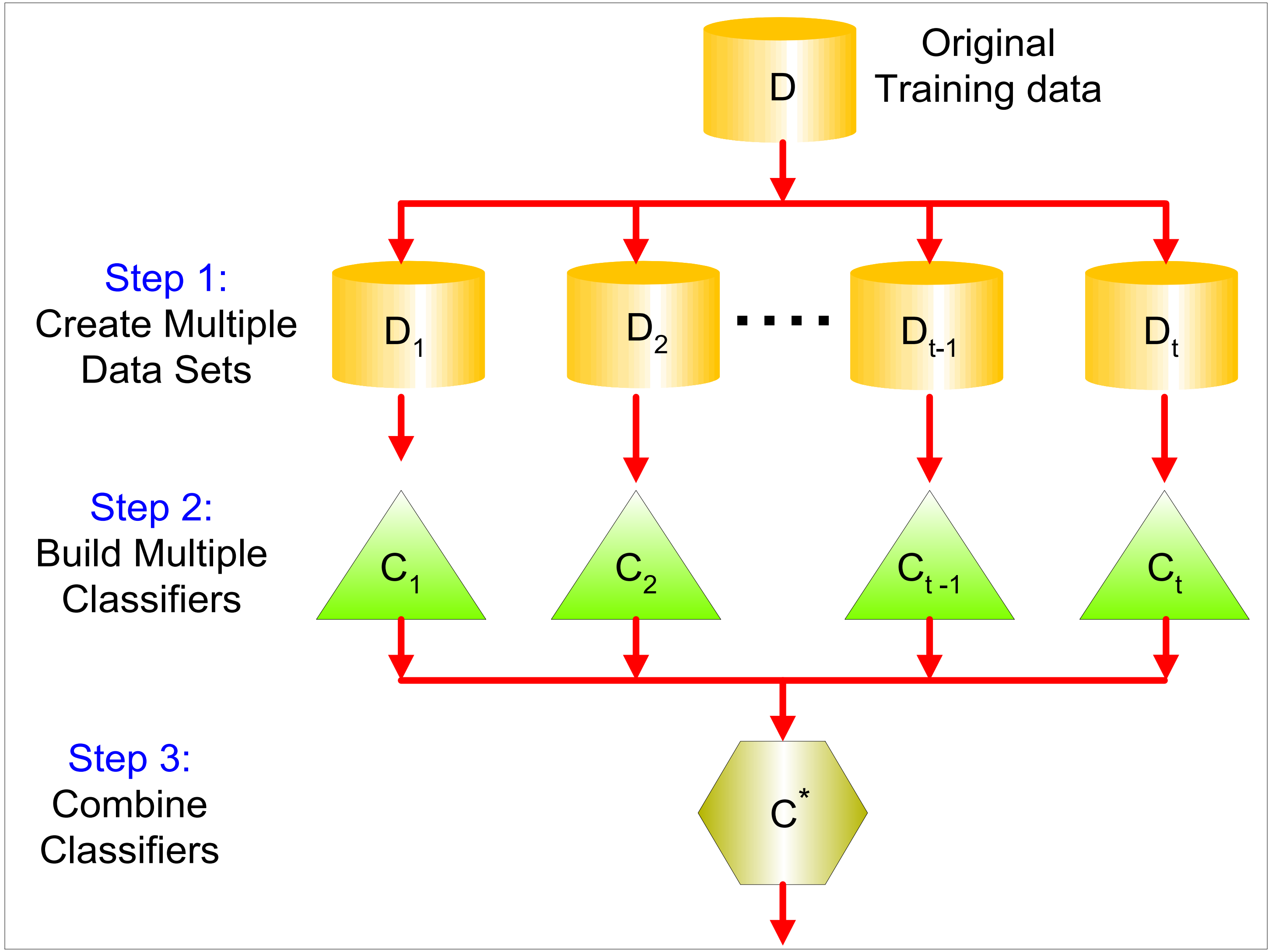
Ensemble Learning

Motivation

- **No Free Lunch Theorem**: There is **no** algorithm that is **always** the most accurate
- Generate a group of base-learners which when combined has higher accuracy
- Each algorithm makes assumptions which might be or not be valid for the problem at hand or not.
- Different learners use different
  - Algorithms
  - Hyperparameters
  - Representations (Modalities)
  - Training sets
  - Subproblems

Bias-Variance Tradeoff

General Idea



Fixed Combination Rules

Rule	Fusion function $f(\cdot)$
Sum	$y_i = \frac{1}{L}\sum_{j=1}^{L} d_{ji}$
Weighted sum	$y_i = \sum_j w_j d_{ji}, w_j \geq 0, \sum_j w_j = 1$
Median	$y_i = \text{median}_j d_{ji}$
Minimum	$y_i = \min_j d_{ji}$
Maximum	$y_i = \max_j d_{ji}$
Product	$y_i = \prod_j d_{ji}$

	C_1	C_2	C_3
d_1	0.2	0.5	0.3
d_2	0.0	0.6	0.4
d_3	0.4	0.4	0.2
Sum	0.2	**0.5**	0.3
Median	0.2	**0.5**	0.4
Minimum	0.0	**0.4**	0.2
Maximum	0.4	**0.6**	0.4
Product	0.0	**0.12**	0.032

Why does it work?

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume classifiers are independent
  - Probability that the ensemble classifier makes a wrong prediction:

$$\sum_{\boxed{i=13}}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.082$$

Why are ensembles successful?

- Bayesian perspective: $P(C_i \mid x) = \sum_{\text{all models } \mathcal{M}_j} P(C_i \mid x, \mathcal{M}_j) P(\mathcal{M}_j)$

- If d_j are independent

$$\text{Var}(y) = \text{Var}\left(\sum_j \frac{1}{L} d_j\right) = \frac{1}{L^2} \text{Var}\left(\sum_j d_j\right) = \frac{1}{L^2} L \cdot \text{Var}(d_j) = \frac{1}{L} \text{Var}(d_j)$$

- Bias does not change, variance decreases by L
- If dependent, error increase with positive correlation

$$\text{Var}(y) = \frac{1}{L^2} \text{Var}\left(\sum_j d_j\right) = \frac{1}{L^2}\left[\sum_j \text{Var}(d_j) + 2 \sum_j \sum_{i<j} Cov(d_i, d_j)\right]$$

Main challenges to develop ensemble models

- The main challenge is not to obtain highly accurate base models, but rather to obtain base models which make different kinds of errors.
- For example, if ensembles are used for classification, high accuracies can be accomplished if different base models misclassify different training examples, even if the base classifier accuracy is low.
  - Independence between two base classifiers can be assessed in this case by measuring the degree of overlap in training examples they misclassify ($|A \cap B|/|A \cup B|$)—more overlap means less independence between two models.

Ensemble Methods

- Bagging
- Boosting
- Stacking / blending
- Misc

Bagging

- Use bootstrapping to generate *L* training sets and train one base-learner with each (Breiman, 1996)
- Use voting (Average or median with regression)
- Unstable algorithms can profit from bagging

Original Data	1	2	3	4	5	6	7	8	9	10

E[X] = 5.5
Pr(E(X) > 7) =

Bagging Example

- Sampling with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

E[X] = 7.4
E[X] = 3.4
E[X] = 5.9

- Build classifier on each bootstrap sample

- Each sample has probability $(1 – 1/n)^n$ of being selected

Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records
  - Initially, all N records are assigned equal weights
  - Unlike bagging, weights may change at the end of boosting round

Boosting

- Records that are wrongly classified will have their weights increased
- Records that are classified correctly will have their weights decreased

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

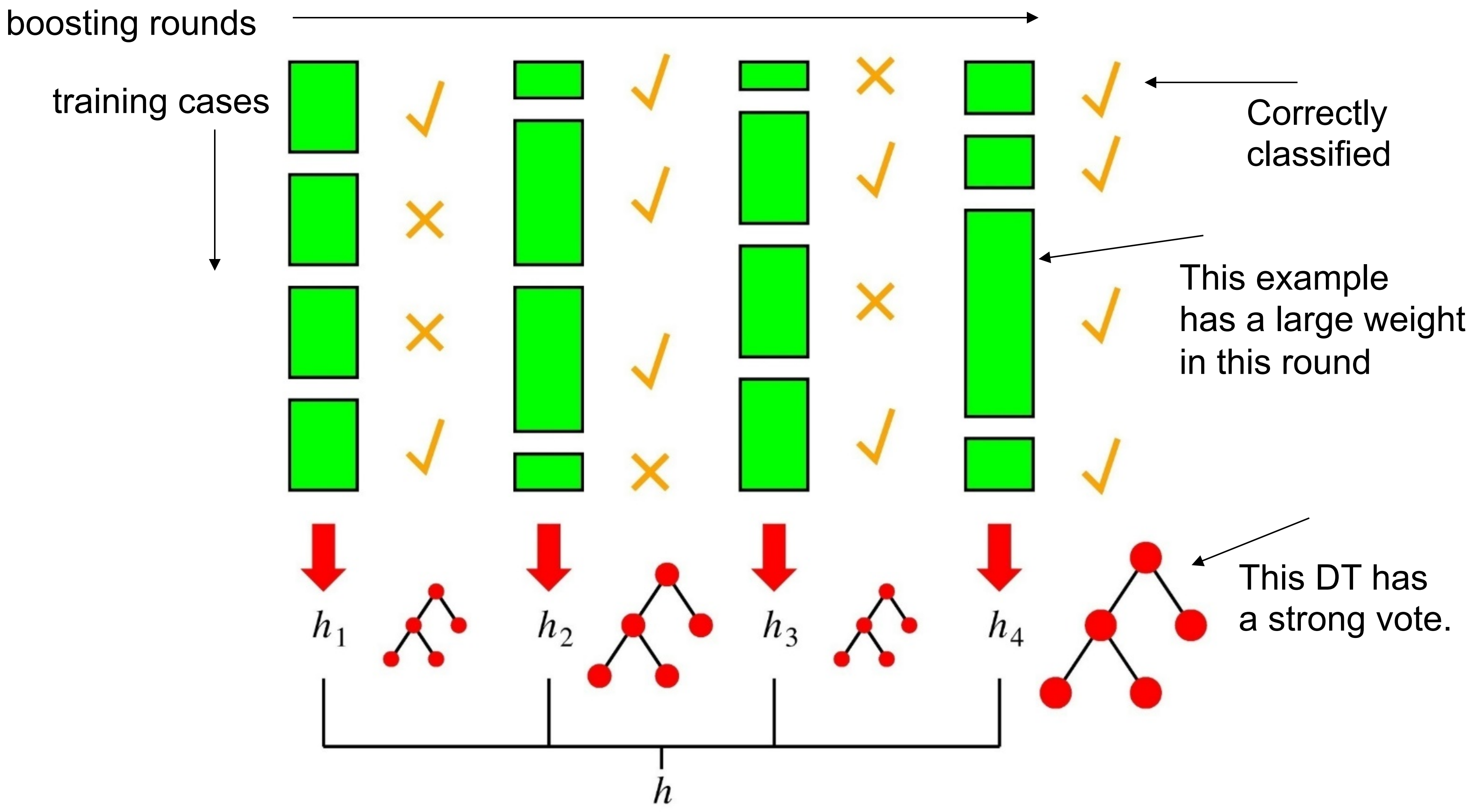
- Example 4 is hard to classify
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

Adaboost - Overview

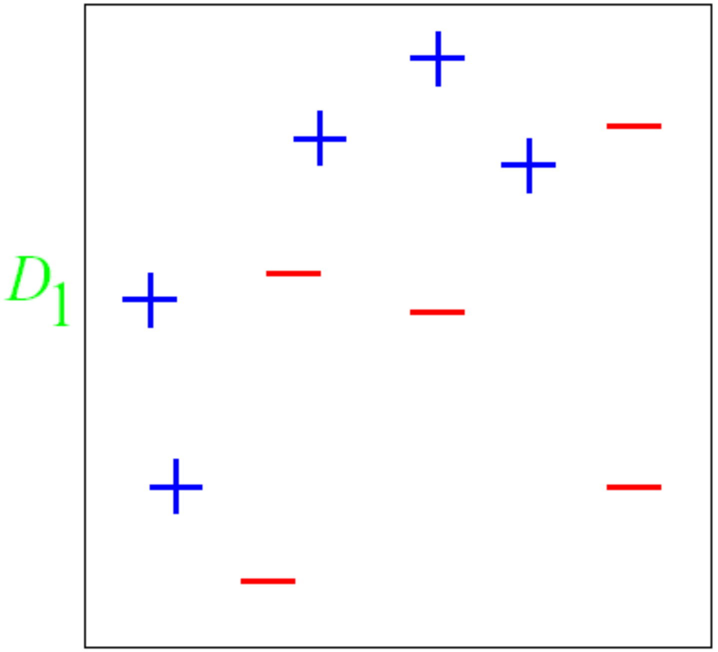
- Using decision trees as weak learner
- One of the best out-of-box boosting algorithm
- Exponential loss

D_1= initial dataset with equal weights
FOR i = 1 to k
    Learn new classifier $C_{i;}$
    Compute α_i (classifier's importance);
    Update example weights;
    Create new training set D_{i+1} (using weighted sampling)
END
Construct Ensemble which uses **all** C_i weighted by α_i (i=1,k)

Boosting in a Picture



Boosting in animation /1

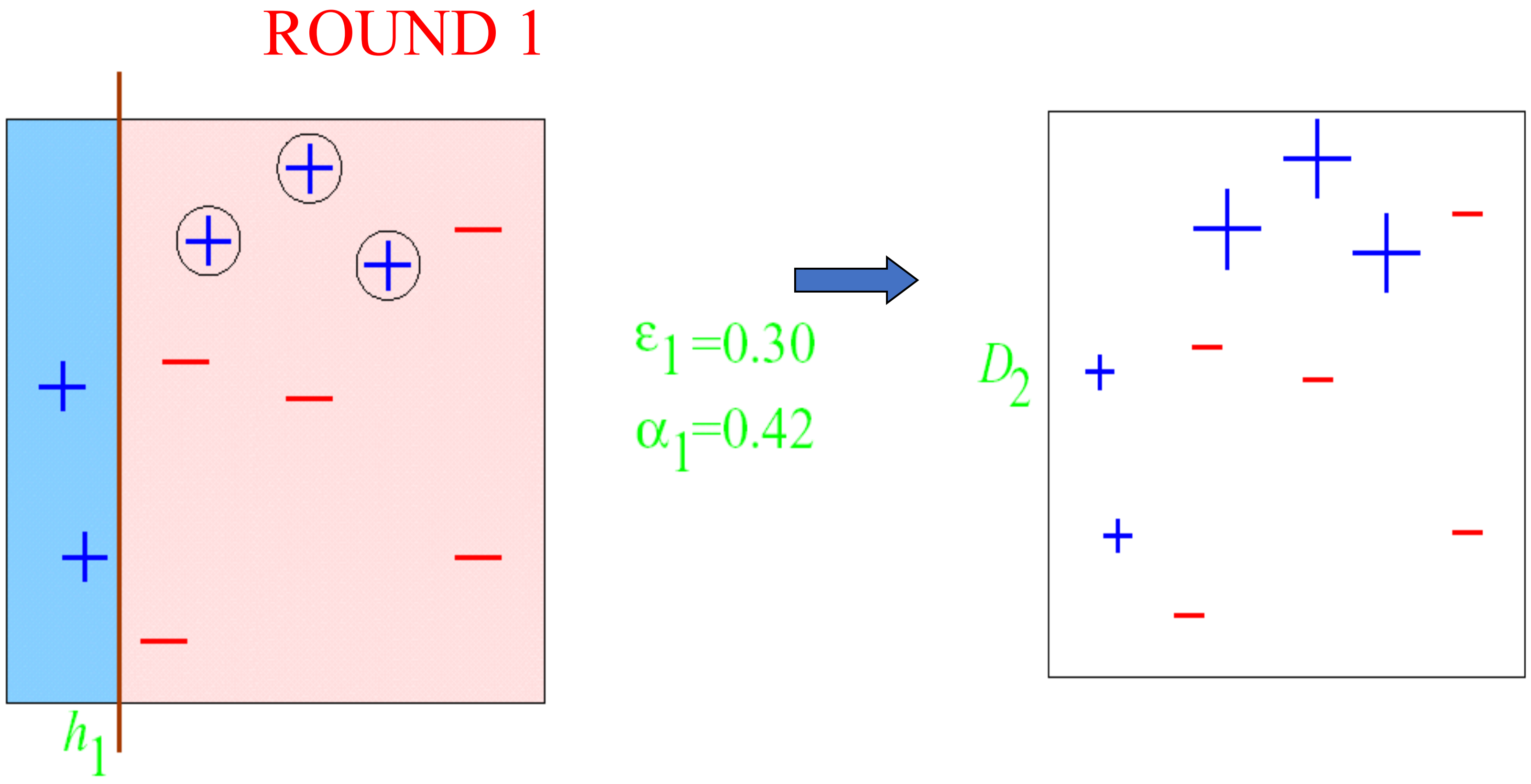


Original training set: equal weights to all training samples

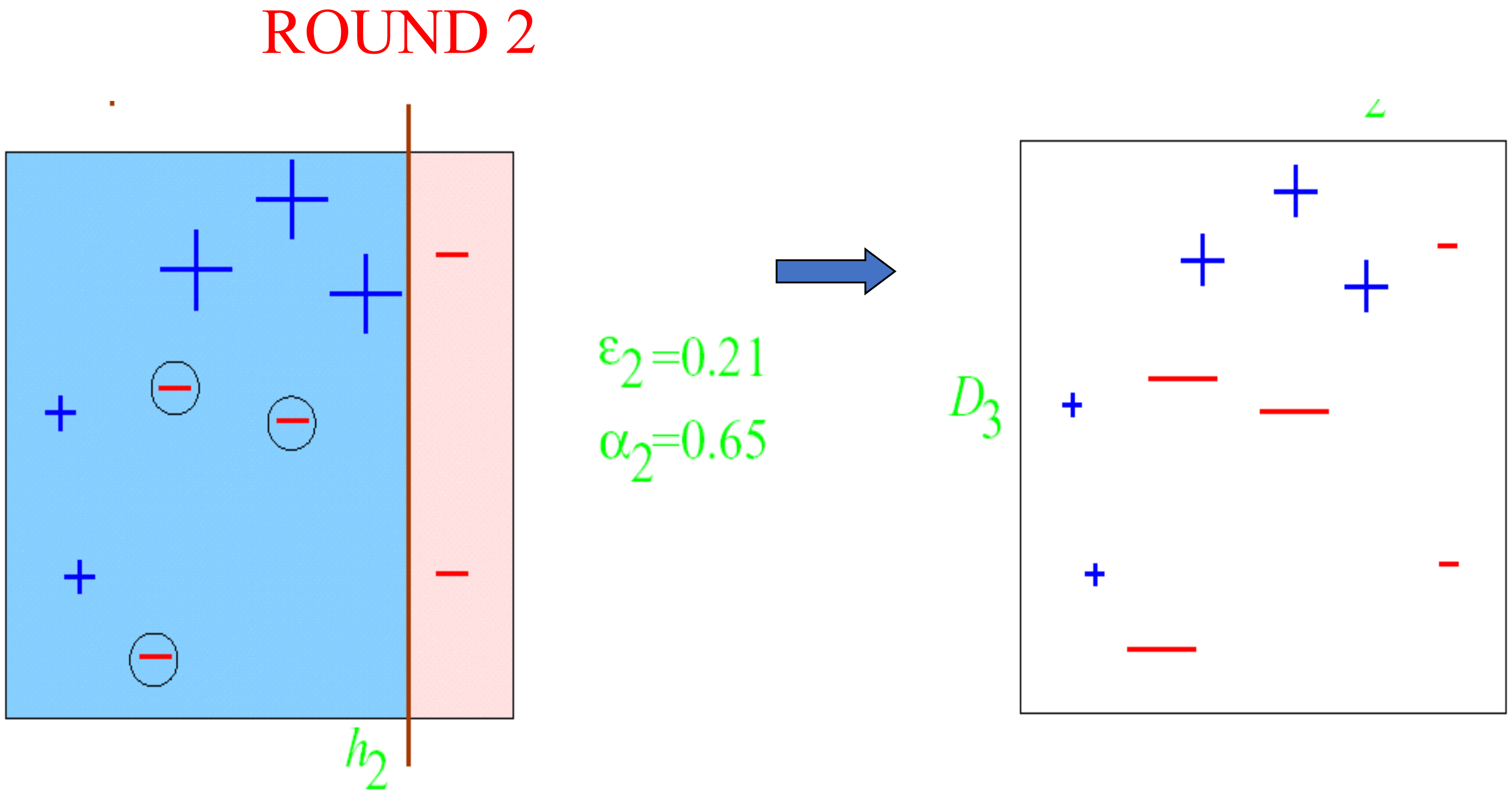
Taken from "**A Tutorial on Boosting**" **by** Yoav Freund and Rob Schapire

Boosting in animation /2

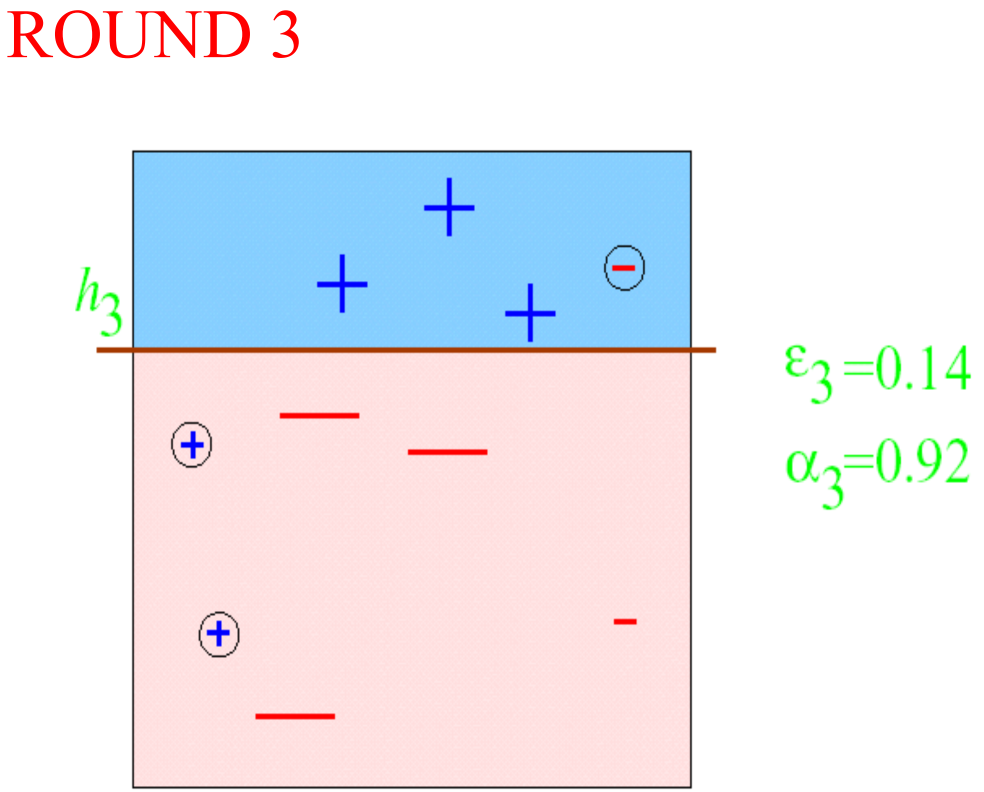
ε = error rate of classifier
α = weight of classifier



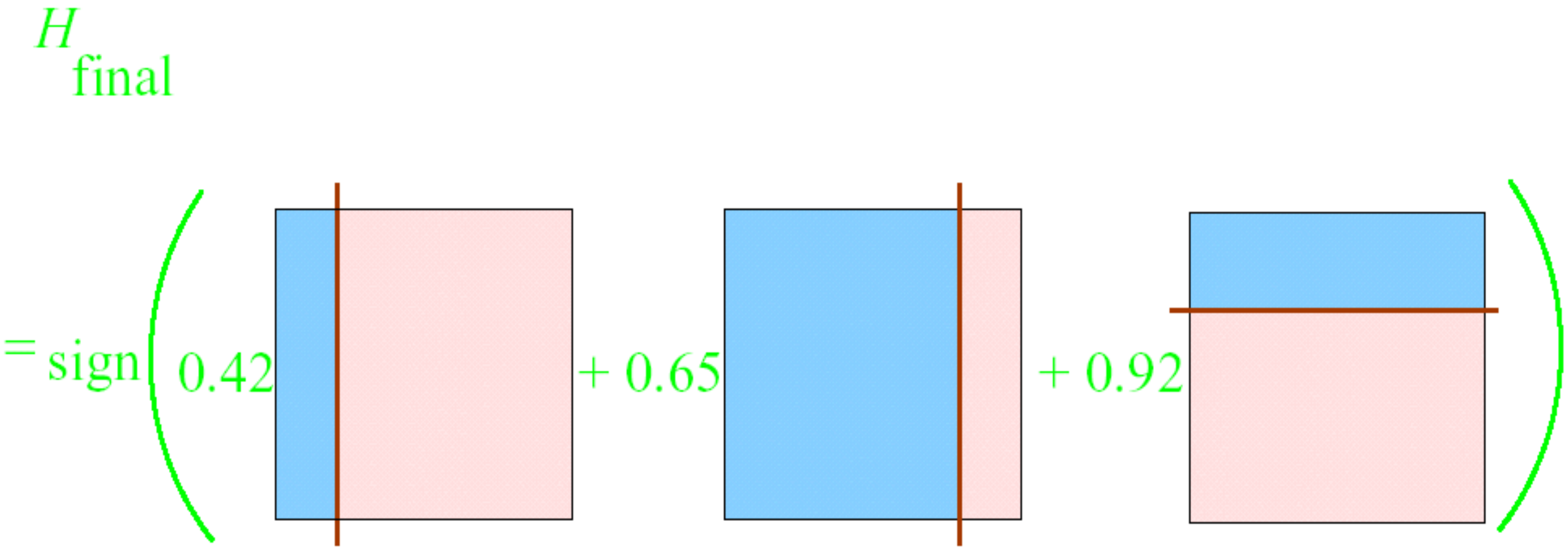
Boosting in animation /3



Boosting in animation /4



Boosting in animation /5

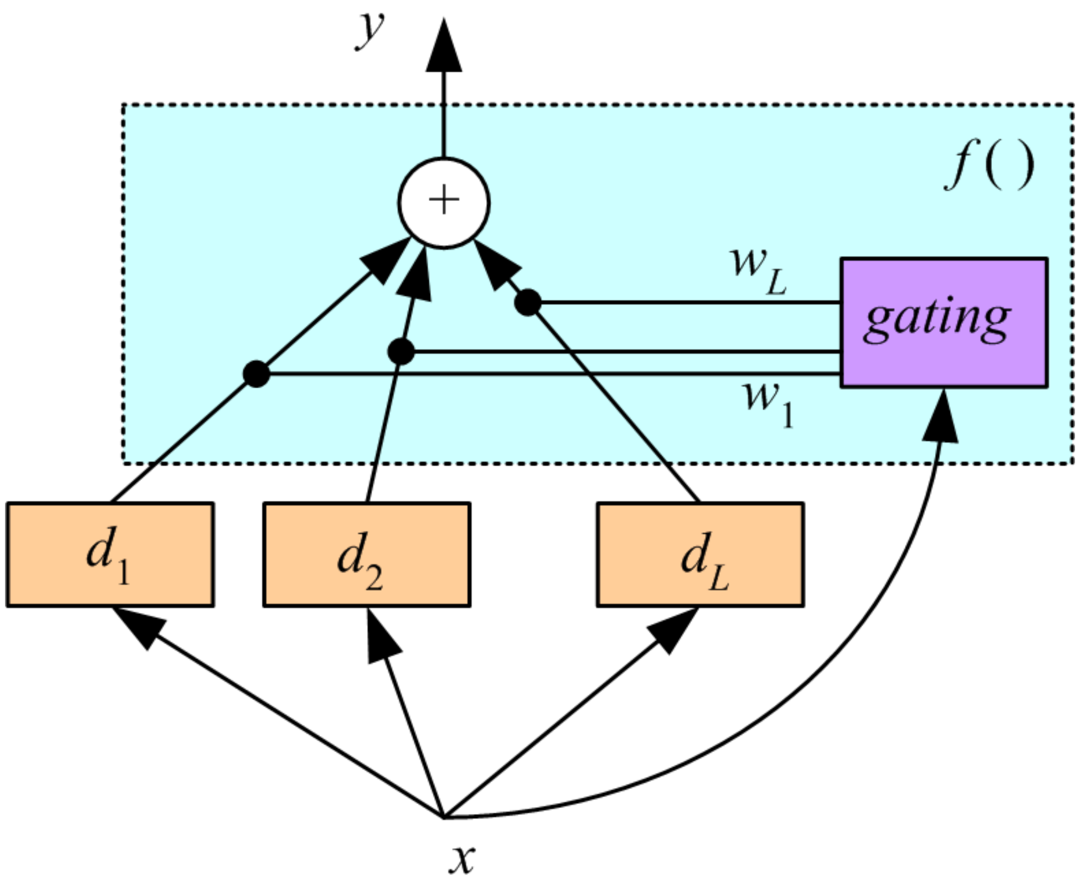


Mixture of Experts

Voting where weights are input-dependent (gating, which can be non-linear)

$$y = \sum_{j=1}^{L} w_j d_j$$

(Jacobs et al., 1991)



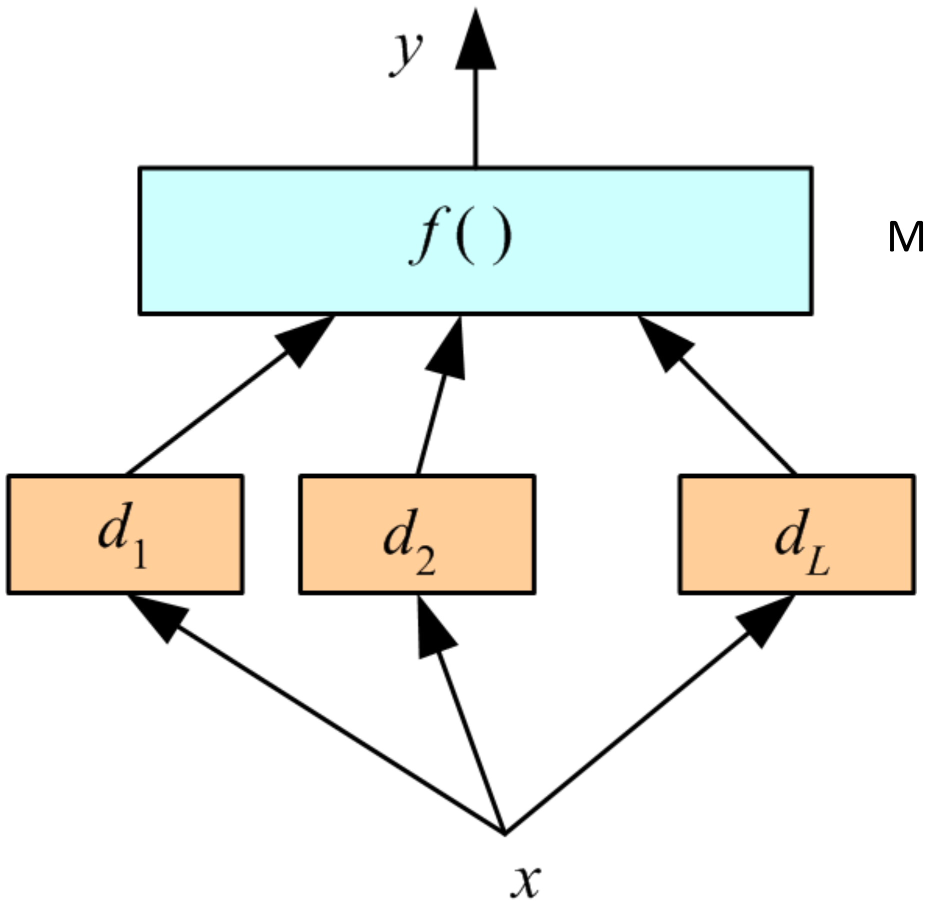
- Gating decides which expert to use
- Need to learn the individual experts as

well as the gating functions $w_i(x)$:

$\Sigma w_j(x) = 1$, for all x

Stacking

- Combiner *f* () is another learner (Wolpert, 1992)



Mixture of Experts

Cascading

Use d_j only if preceding ones are not confident

Cascade learners in order of complexity