DMML, 5 Mar 2019

Ensembles of classifiers

Variation in model due to training data

- Variance

Decision tree

Voting across a number of models

- e.g. Cross validation

May produce radically different trees

Bootstrap aggregation ("Bagging")

Repeatedly collect "bootstrap" samples of training data

Training data is $\{x_1, x_2, \ldots, x_N\}$

Uniformly at random pick an x_i

Add x_i to sample

Replace x_i

Repeat

Repeat N times

Sample $y_1, y_2, \ldots, y_N$ with repetitions

What fraction of distinct elements appear in this sample?

Turns out to be approx 63%

Construct bootstrap samples $S_1, S_2, \ldots, S_{2n+1}$

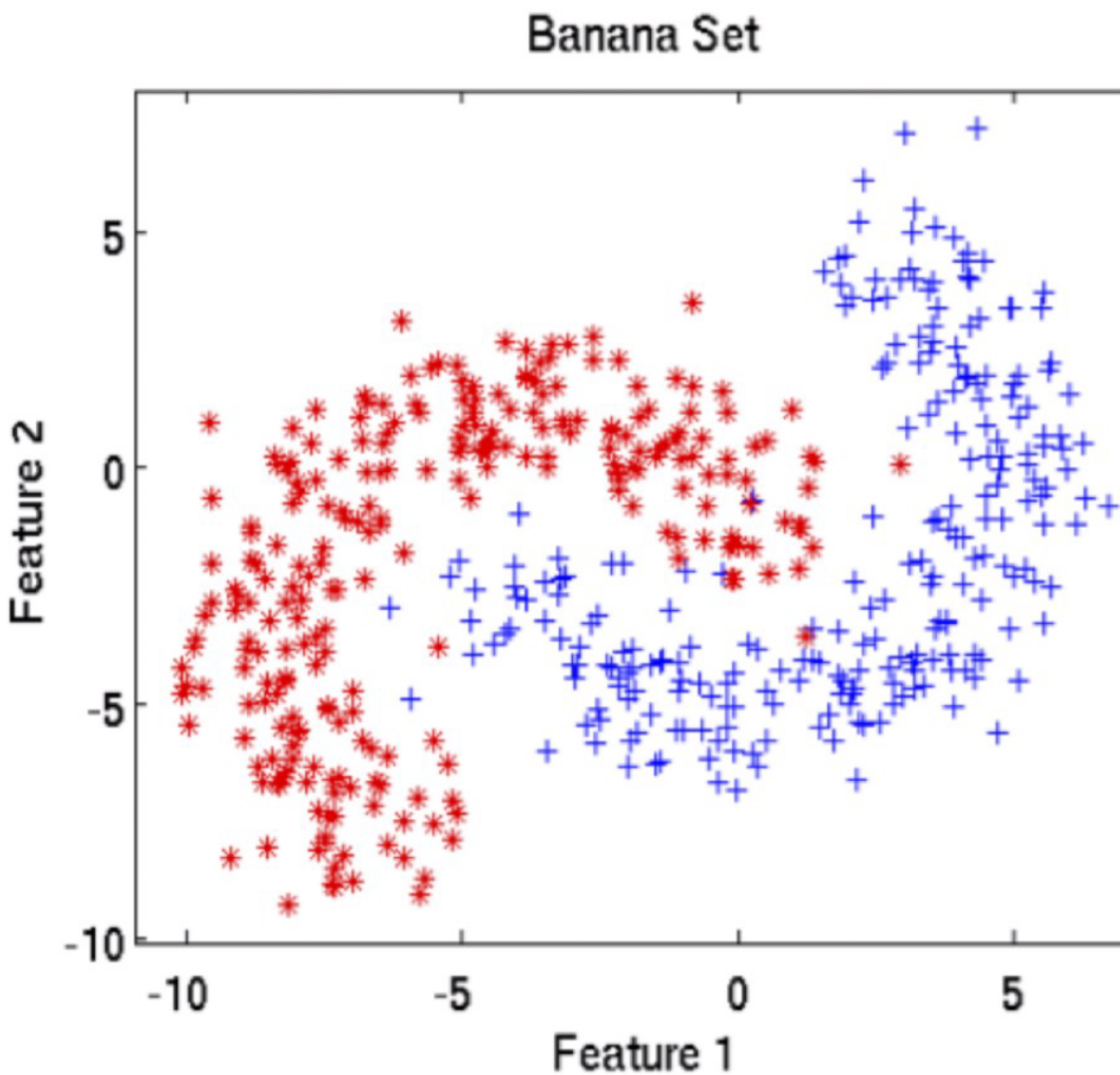
Build classifiers $M_1, M_2, \ldots, M_{2n+1}$

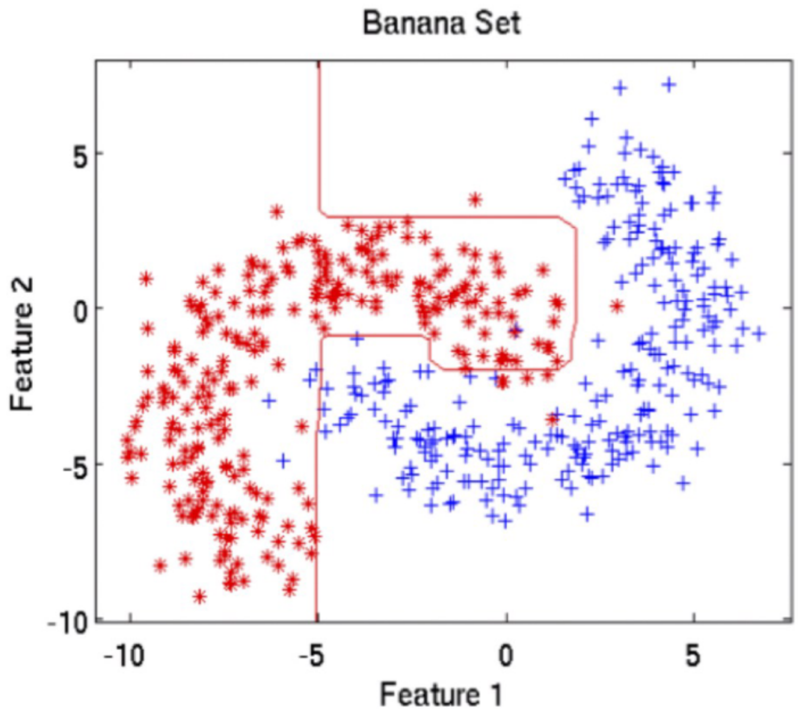
Vote

When does this help?

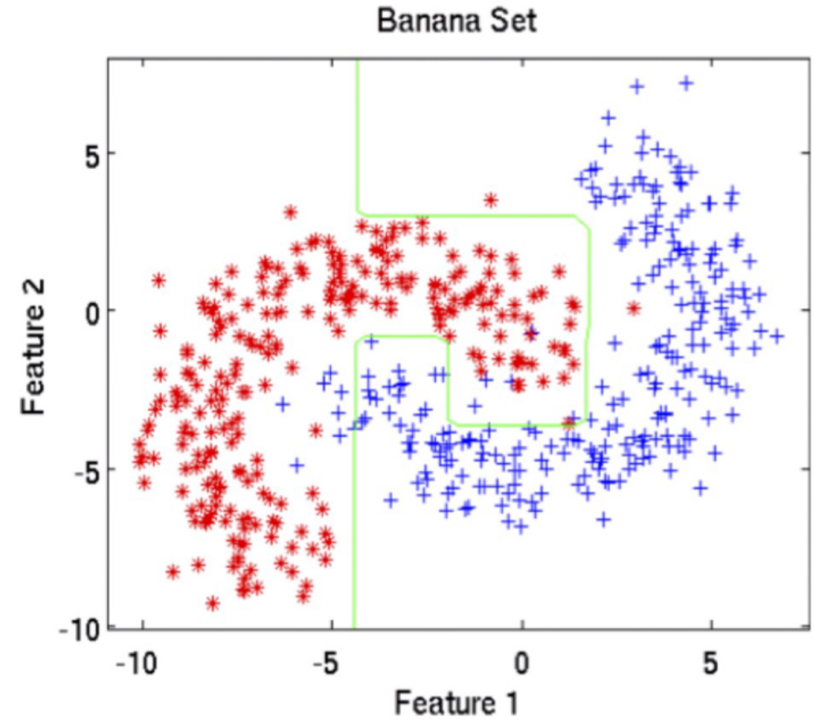
Typically when there is high variance

Bagging in action (decision trees)

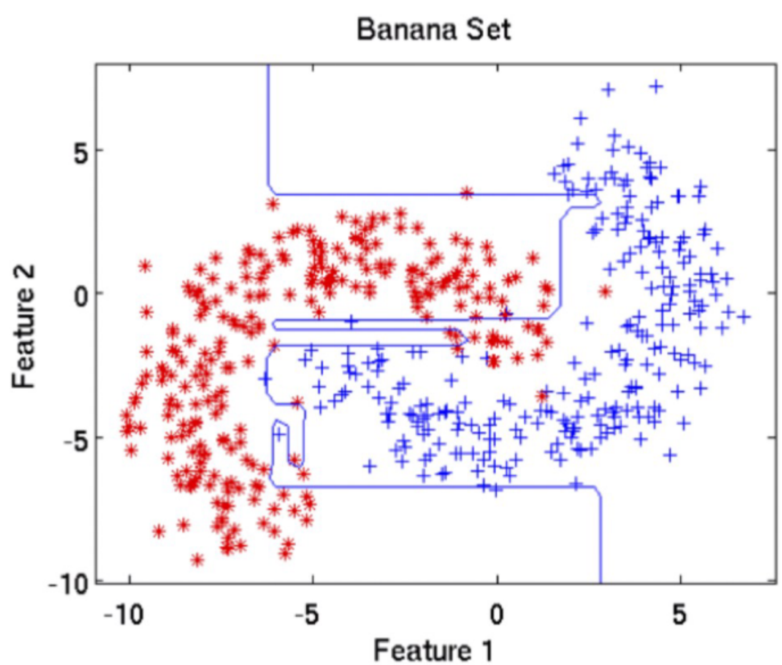




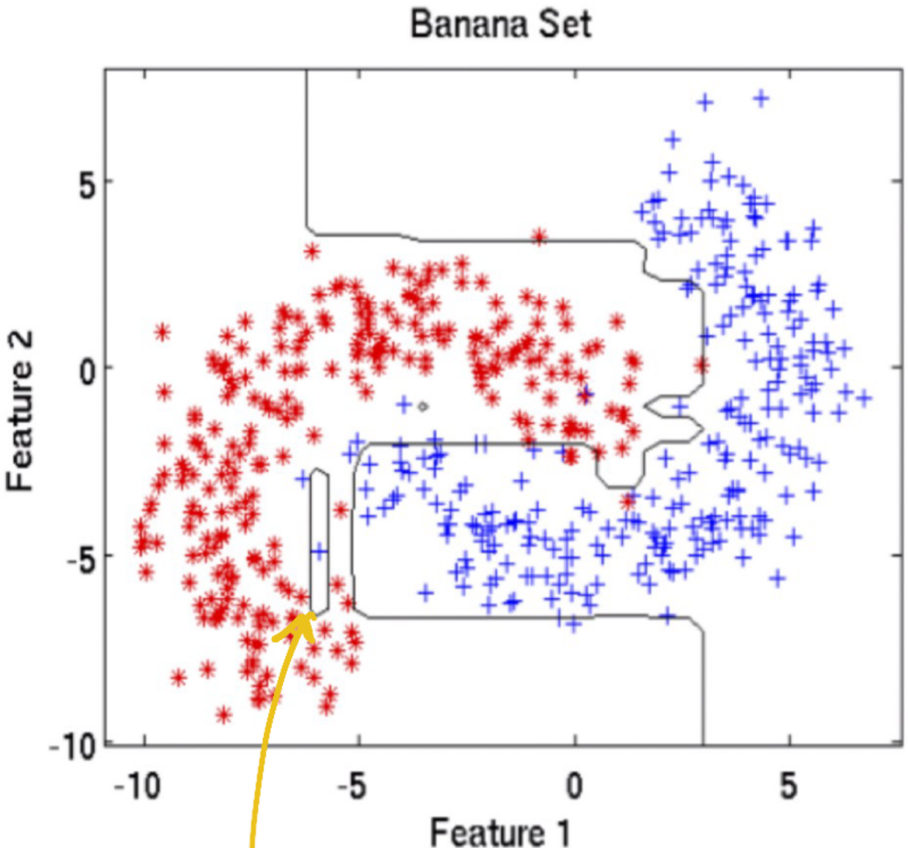
Decision boundary produced by one tree



Decision boundary produced by a second tree

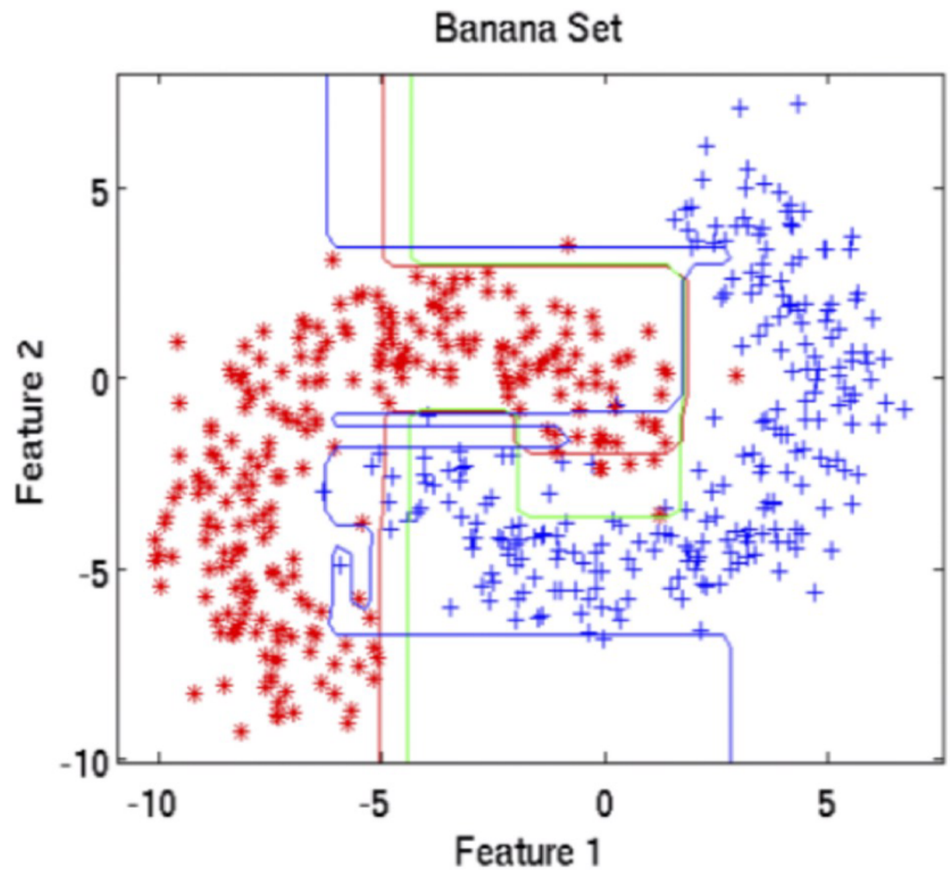


Decision boundary produced by a third tree



Final result from bagging all trees.

Observe "island"



Three trees
overlaid

A variant for decision trees

At each level, choose a small subset of remaining attributes

Among this subset choose the best in terms of impurity reduction

Build a full tree - no pruning

In this way, build $T_1, T_2, \ldots, T_{2n+1}$

— Random Forest

$\downarrow$

Vote

Boosting

Improve the quality of a "weak" classifier

Training set T $\rightarrow$ build a model M

M makes mistakes on $T' \subseteq T$

Compensate for mistakes

Give higher weightage to T'

Training data $T = \{w_1, w_2, \dots, w_n\}$ (x_j, y_j)

Each t_i has a weight: initially $1/n$ –

$W_1 : T \to [0..1]$

$w_j \mapsto 1/n$

Repeat k times

$$M_i = \text{Model}(W_i)$$

$$e_i = \sum_{M_i(w_j) \neq y_j} W_i(w_j)$$

if $e_i \geq 1/2$ quit

$$\beta_i = \frac{e_i}{1-e_i} \leq 1/2$$

< 1

Update weights

$$W_{t+1}(w_j) = W_t(w_j) \times \beta_t \text{ if correct}$$
$$\phantom{W_{t+1}(w_j) = W_t(w_j)} \times 1 \text{ if wrong}$$

$\beta_t < 1$

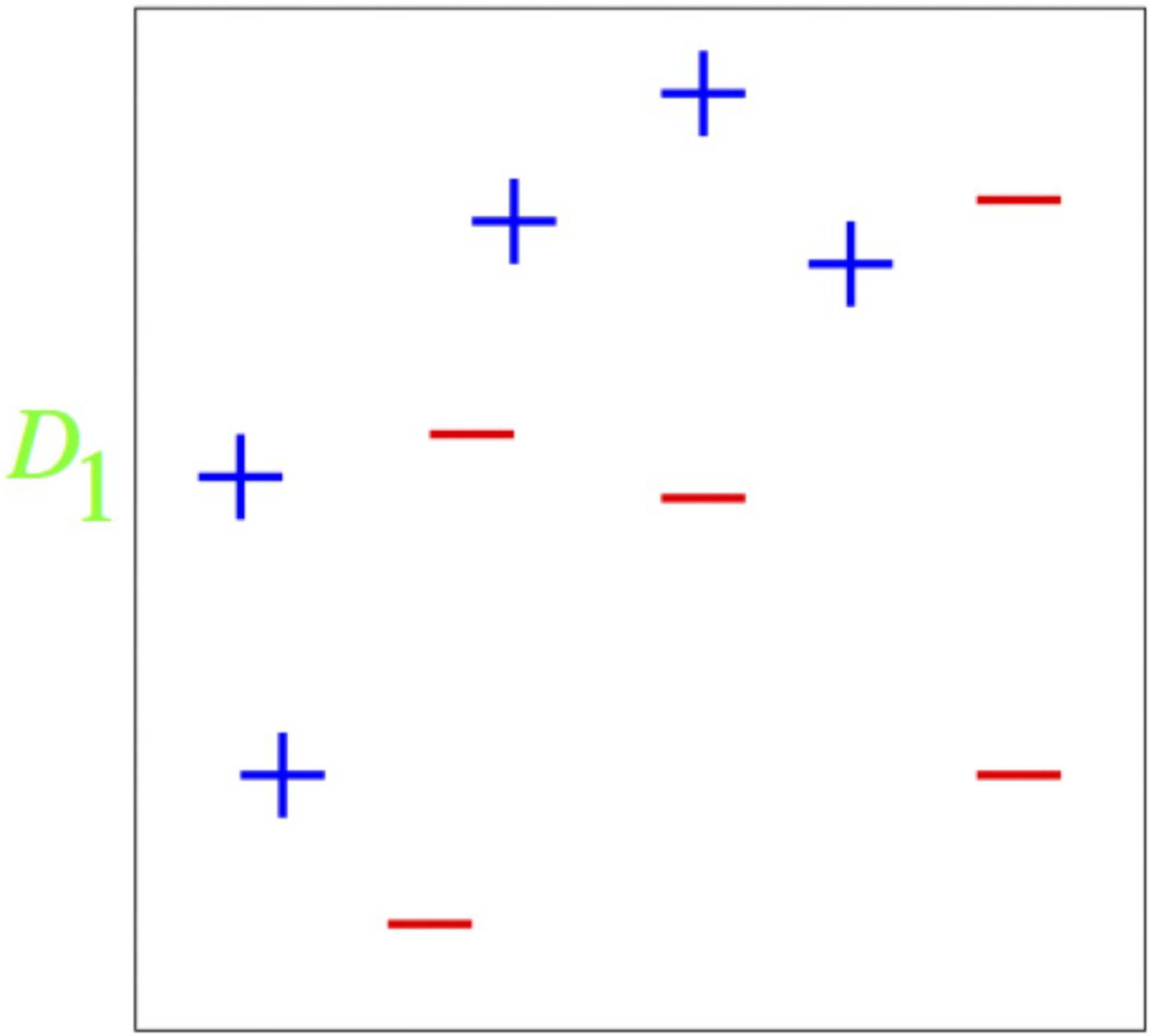
Normalize W_{t+1}

$$\frac{W_t(w_j)}{\sum_k W_t(w_k)}$$

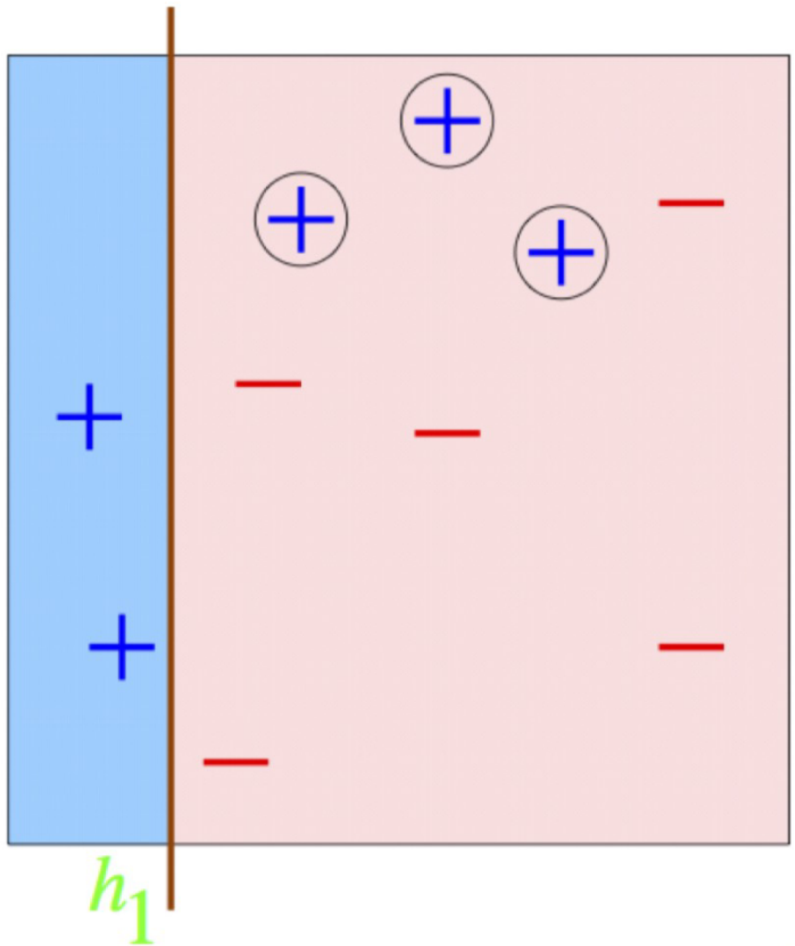
→ build k models $M_1, M_2, \ldots, M_k$

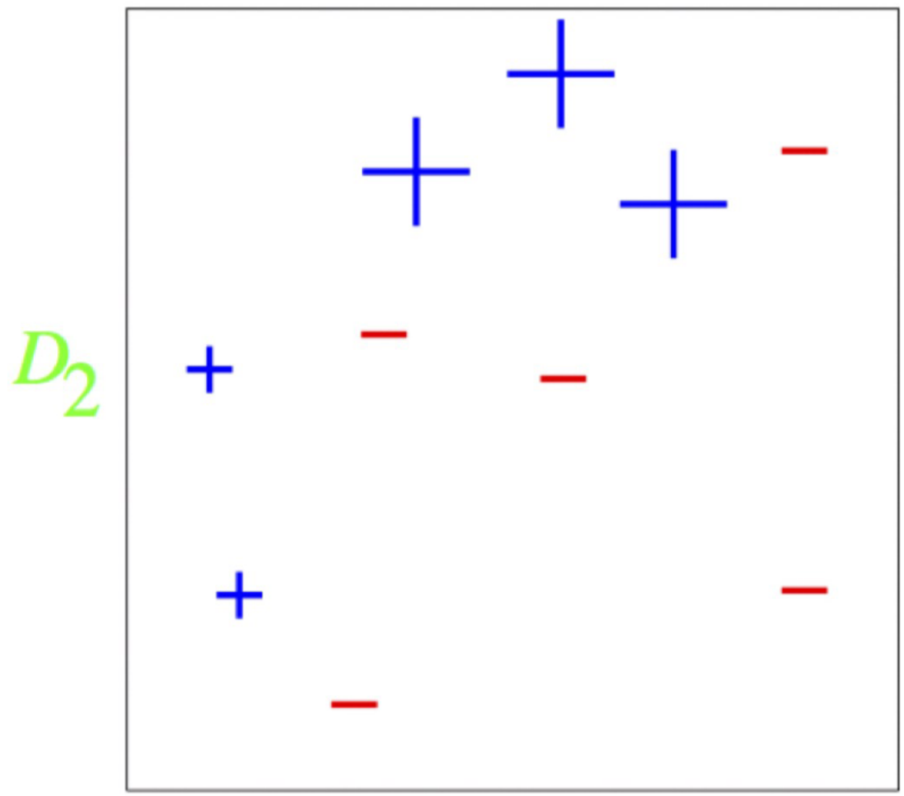
$$f(x) = \underset{y}{\operatorname{argmax}} \sum_{t,\ M_t(x)=y} \log \frac{1}{\beta_t}$$

Boosting at work



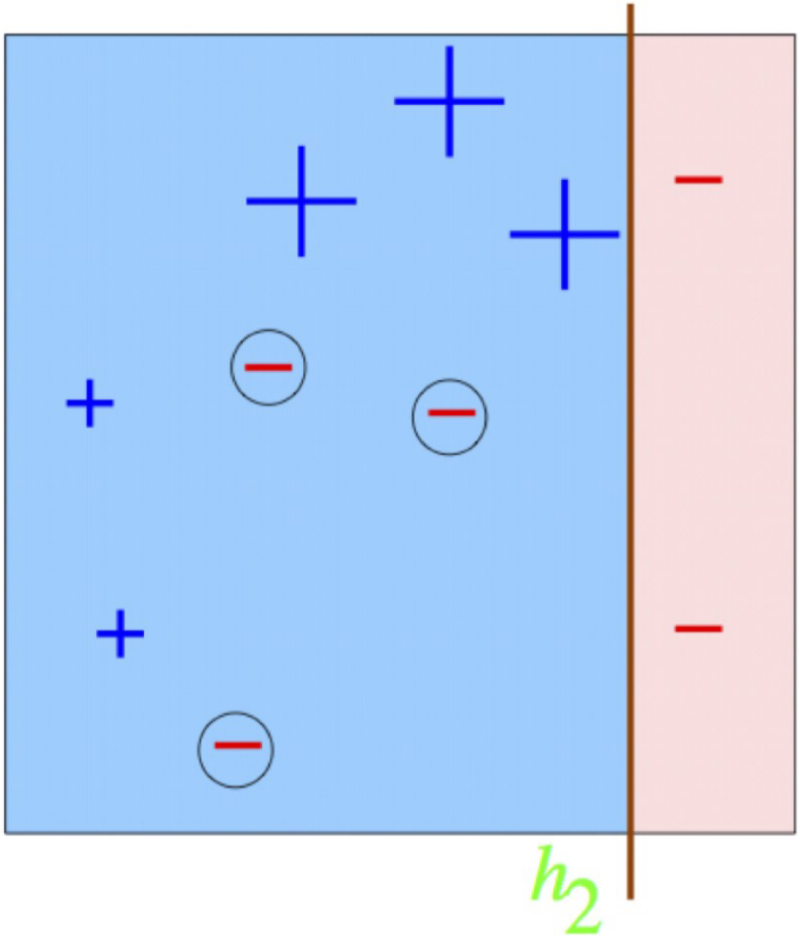
Weak classifiers – horizontal & vertical lines

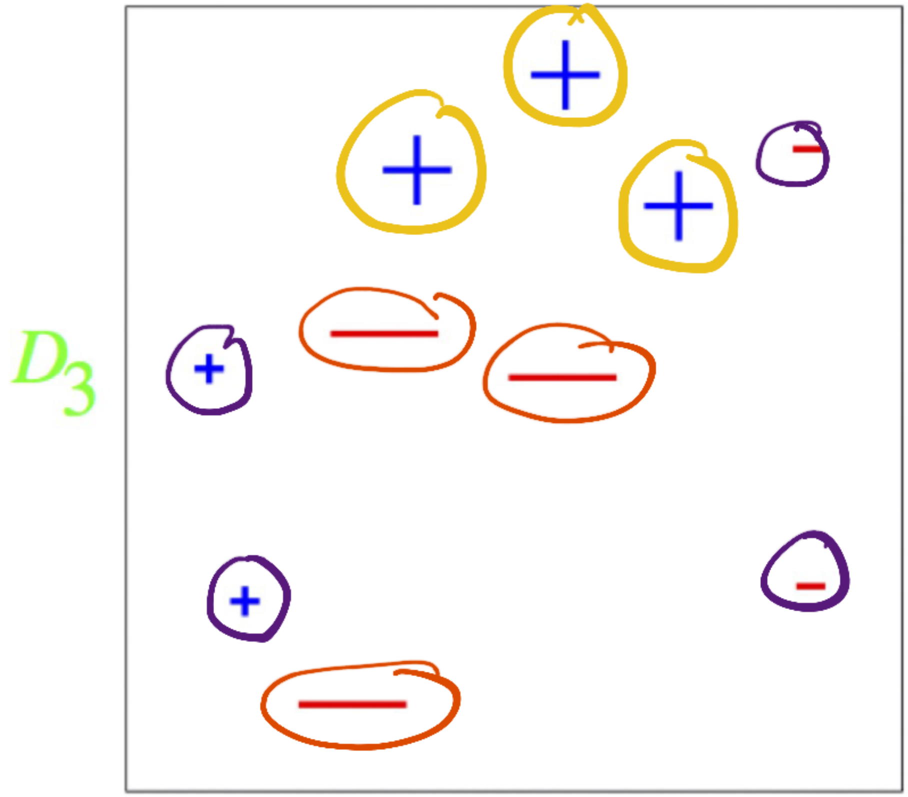




First classifier

Data with updated weights – size proportional to weight



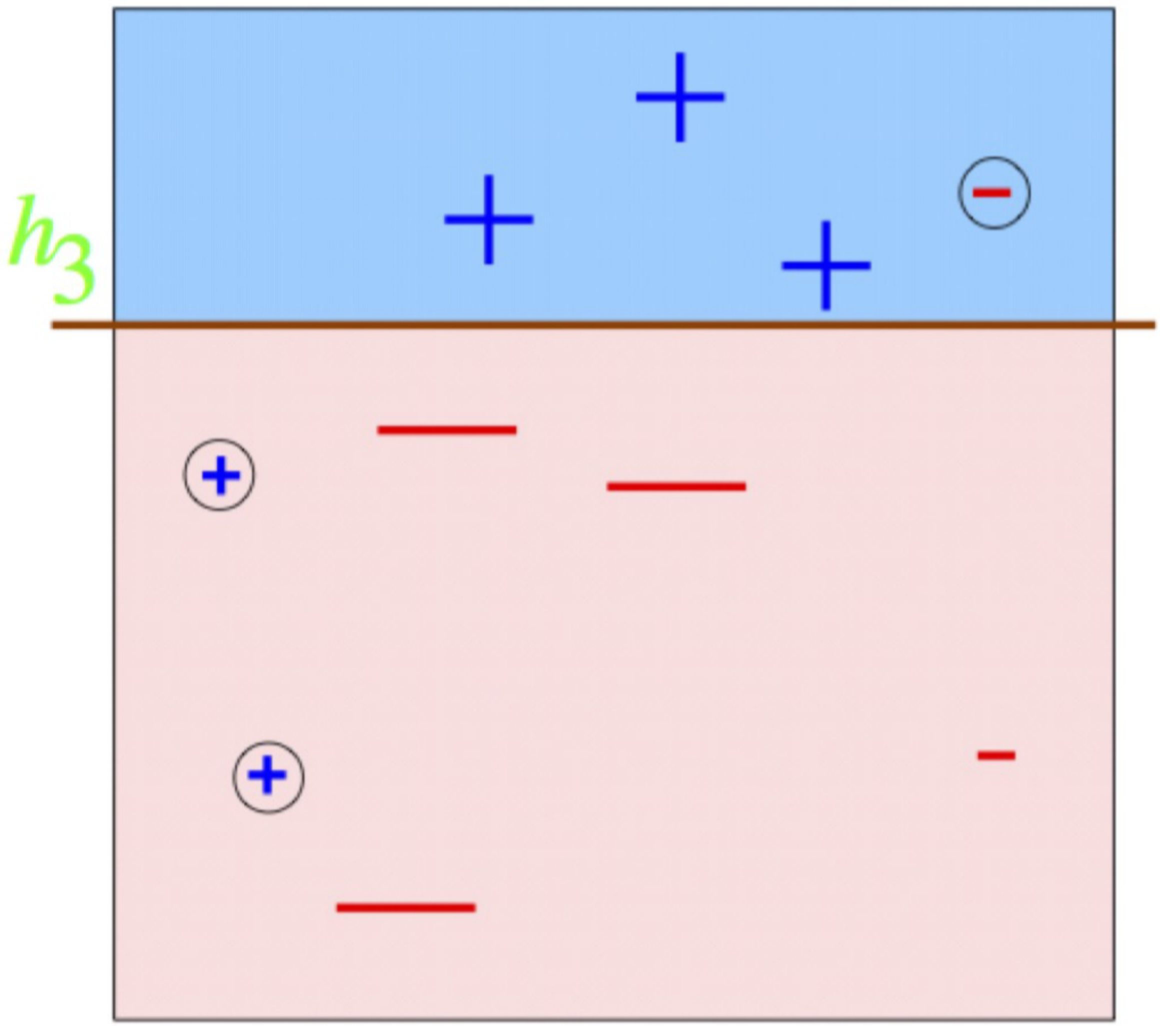


Second weak classifier

- compensates for mistakes of h_1

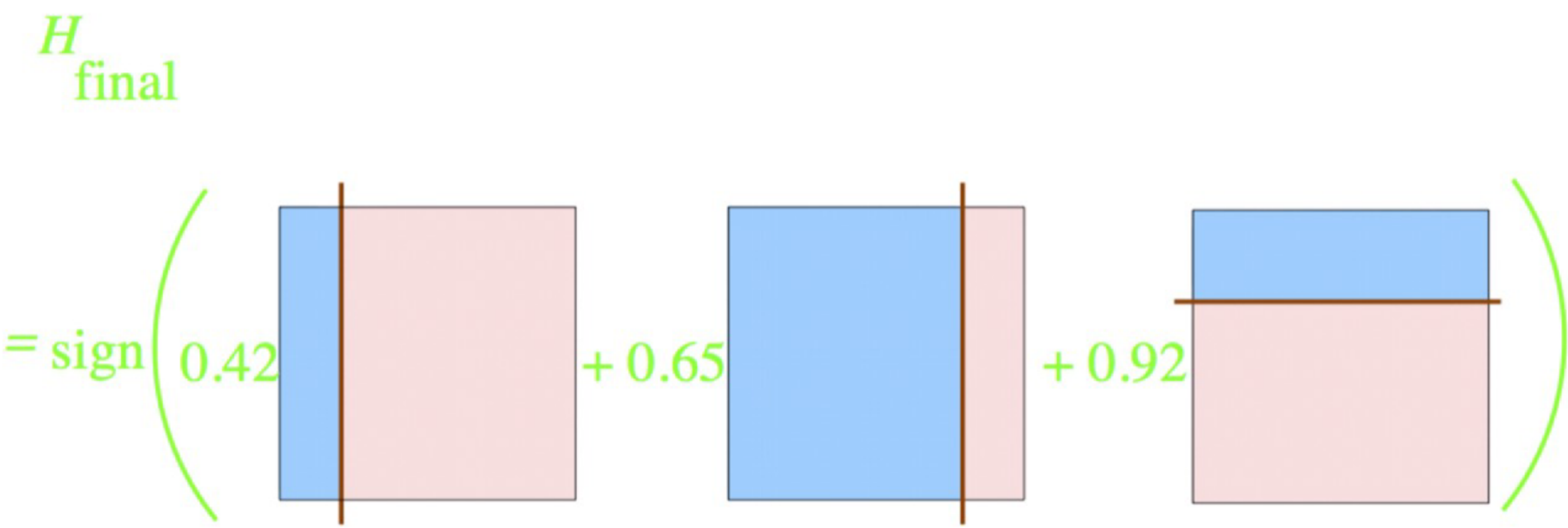
Note 3 weights/size

- Small : h_1 & h_2 correct
- Med : h_1 wrong, h_2 correct
- Large ; h_2 wrong, h_1 correct



3rd classifier - horizontal, not vertical

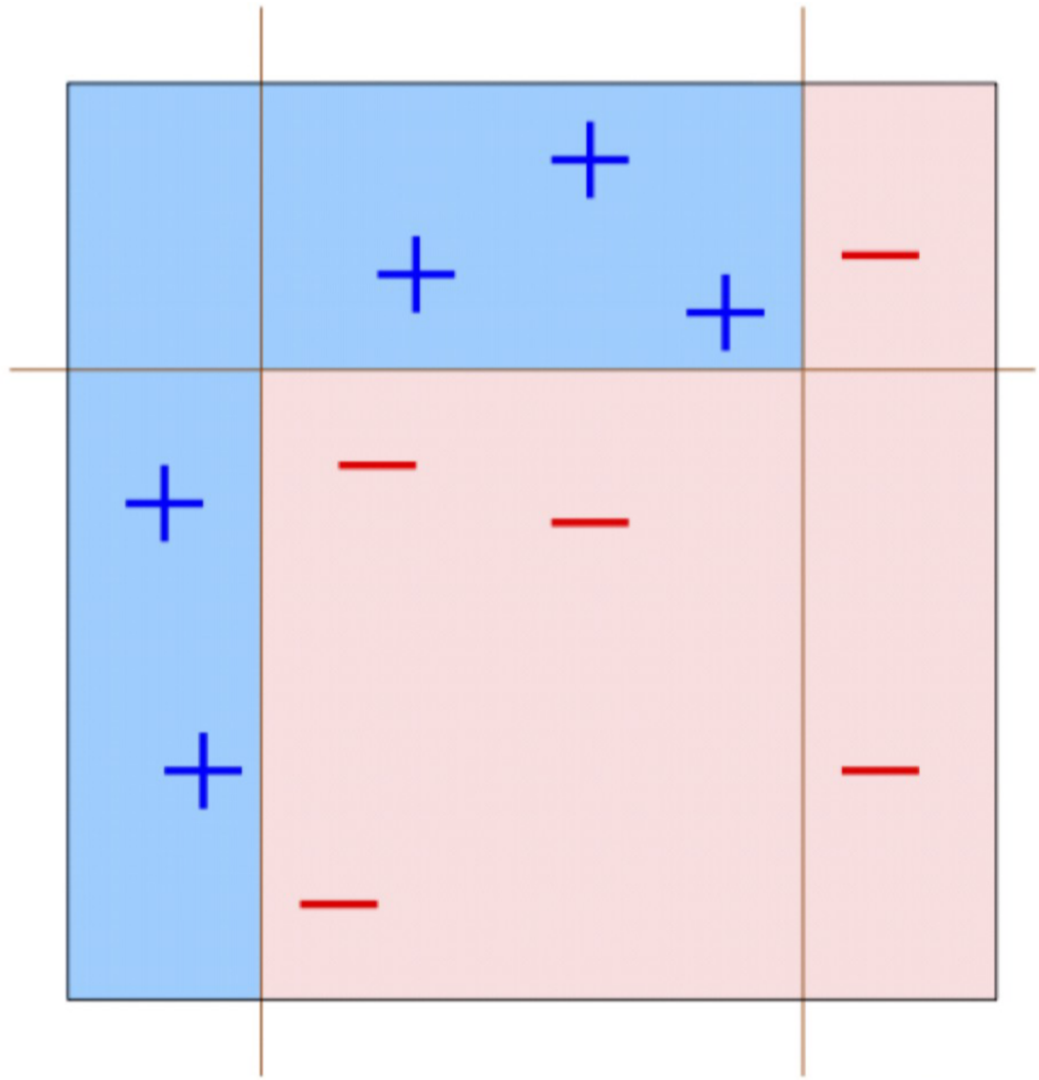
Extracting a single classifier



Weighted combination of weak classifiers

- Recall weights are based on error rates of classifiers

Final classifier



Effectively constructs a piecewise linear boundary from vertical & horizontal lines.

Multiplicative weight update

Have several unreliable "experts"

If India wins the toss, India wins

If xyz bowls well, - - -

If grass is green - - -