Robotics 2

AdaBoost for People and Place Detection

Kai Arras, Cyrill Stachniss,

Maren Bennewitz, Wolfram Burgard

v.1.1, Kai Arras, Jan 12, including material by Luciano Spinello and Oscar Martinez Mozos

Classification

- Classification algorithms are **supervised algorithms** to predict **categorical labels**
- Differs from **regression** which is a supervised technique to predict **real-valued labels**

Formal problem statement:

- **Produce a function that maps**

  $C : \mathcal{X} \rightarrow \mathcal{Y}$

- **Given a training set**

  $\{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$

  $y \in \mathcal{Y}$ label

  $\mathbf{x} \in \mathcal{X}$ training sample

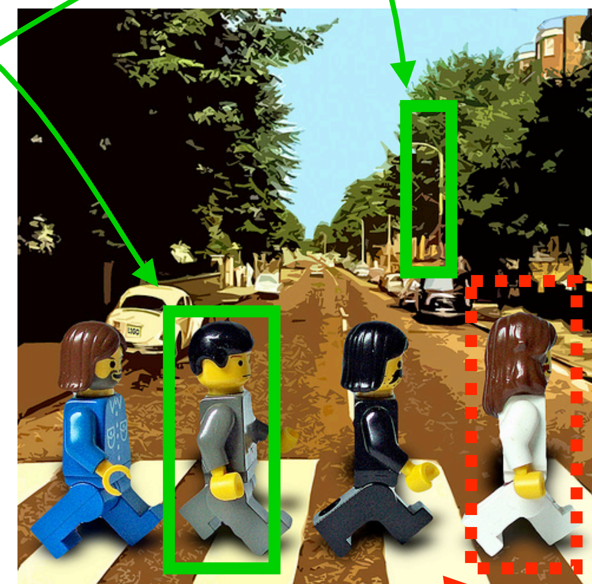
Classification

Error types

True value

Predicted value of the classifier

	T	N
T'	True Positive	False Positive
N'	False Negative	True Negative



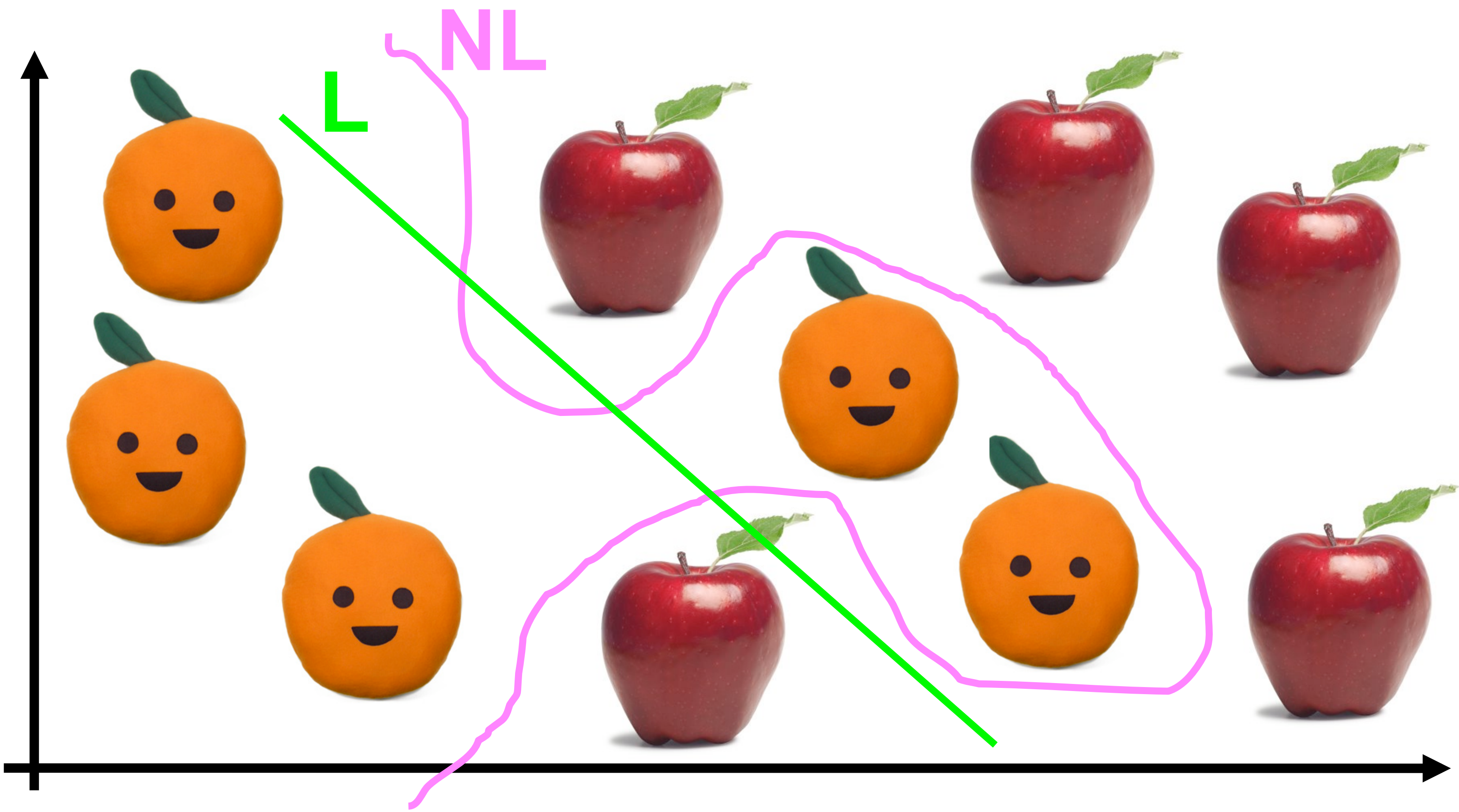
Detected
Not Detected

- **Precision** = TP / (TP + FP)
- **Recall** = TP / (TP + FN)

Many more measures...

Classification

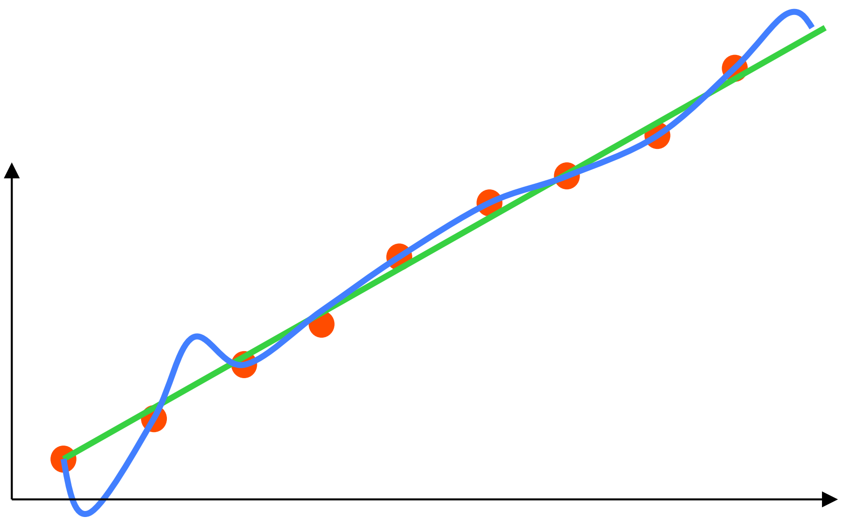
Linear vs. **Non-Linear Classifier, Margin**



Classification

Overfitting

- Overfitting occurs when a model begins to memorize the **training data** rather than learning the underlying relationship
- Occurs typically when fitting a statistical model with too many parameters
- Overfitted models explain training data perfectly but they **do not generalize!**
- There are techniques to avoid overfitting such as regularization or cross-validation



Boosting

- An **ensemble technique** (a.k.a. committee method)
- Supervised learning: given <samples *x*, labels *y*>

- Learns an accurate **strong classifier** by combining an ensemble of inaccurate "rules of thumb"
- **Inaccurate rule** $h(x_i)$: "weak" classifier, weak learner, basis classifier, feature
- **Accurate rule** $H(x_i)$: "strong" classifier, final classifier

- Other **ensemble techniques** exist: Bagging, Voting, Mixture of Experts, etc.

AdaBoost

- Most popular algorithm: **AdaBoost**
  [Freund et al. 95], [Schapire et al. 99]

- Given an ensemble of weak classifiers $h(x_i)$, the combined strong classifier $H(x_i)$ is obtained by a **weighted majority voting scheme**

$$f(x_i) = \sum_{t=1}^{T} \alpha_t h_t(x_i) \qquad H(x_i) = \operatorname{sgn}\left(f(x_i) \right)$$

- AdaBoost in Robotics:
  [Viola et al. 01], [Treptow et al. 04], [Martínez-Mozos et al. 05], [Rottmann et al. 05] , [Monteiro et al. 06] , [Arras et al. 07]

AdaBoost

Why is AdaBoost interesting?

1. It tells you what the **best "features"** are
2. What the **best thresholds** are, and
3. How to **combine them to a classifier**

- AdaBoost can be seen as a **principled feature selection strategy**

- Classifier design becomes **science**, not art

AdaBoost

- AdaBoost is a **non-linear classifier**
- Has **good generalization properties**: can be proven to maximize the margin
- Quite robust to **overfitting**
- Very **simple** to implement

- **Prerequisite:**
  weak classifier must be better than chance:
  error < 0.5 in a binary classification problem

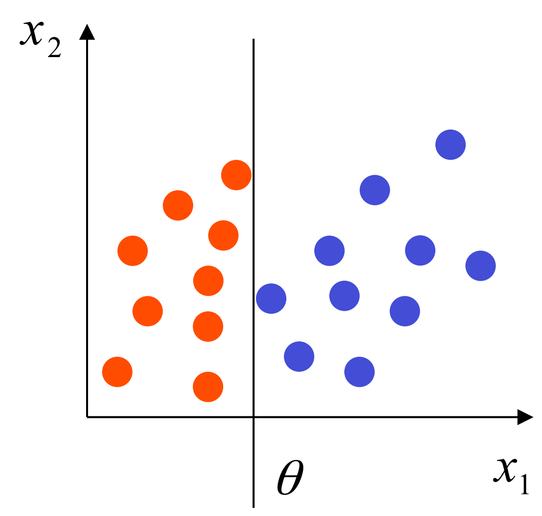
AdaBoost

- **Possible Weak Classifiers:**
  - **Decision stump**:
    Single axis-parallel partition of space
  - **Decision tree**:
    Hierarchical partition of space
  - **Multi-layer perceptron**:
    General non-linear function approximators
  - **Support Vector Machines** (SVM):
    Linear classifier with RBF Kernel
- Trade-off between diversity among weak learners versus their accuracy. Can be complex, see literature
- Decision stumps are **a popular choice**

AdaBoost: Weak Classifier

Decision stump

- Simple-most type of **decision tree**
- Equivalent to linear classifier defined by affine hyperplane
- Hyperplane is orthogonal to axis with which it intersects in threshold θ
- Commonly not used on its own
- Formally,



$$h(x; j, \theta) = \begin{cases} +1 & x_j > \theta \\ -1 & \text{else} \end{cases}$$

where x is (d-dim.) training sample, j is dimension

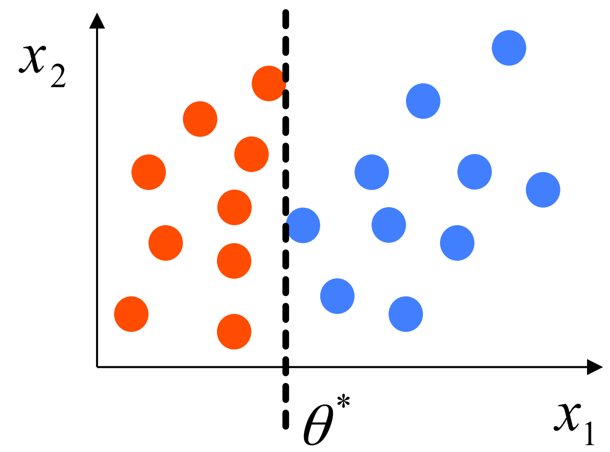
AdaBoost: Weak Classifier

- **Train a decision stump on weighted data**

$$(j^*, \theta^*) = \operatorname{argmin}_{j,\theta} \left\{ \sum_{i=1}^{n} w_t(i)\, \mathrm{I}(y_i \neq h_t(x_i)) \right\}$$

- **This consists in…**

Finding an optimum parameter θ^* for each dimension $j = 1 \ldots d$ and then select the j^* for which the weighted error is minimal.



AdaBoost: Weak Classifier

A simple training algorithm for stumps:

$\forall\ j = 1 \ldots d$

Sort samples x_i in ascending order along dimension j

$\forall\ i = 1 \ldots n$

Compute n cumulative sums $w_{cum}^{j}(i) = \sum_{k=1}^{i} w_k\, y_k$

end

Threshold θ_j is at extremum of w_{cum}^{j}

Sign of extremum gives direction p_j of inequality

end

Global extremum in all d sums w_{cum} gives
threshold θ^* and **dimension** j^*

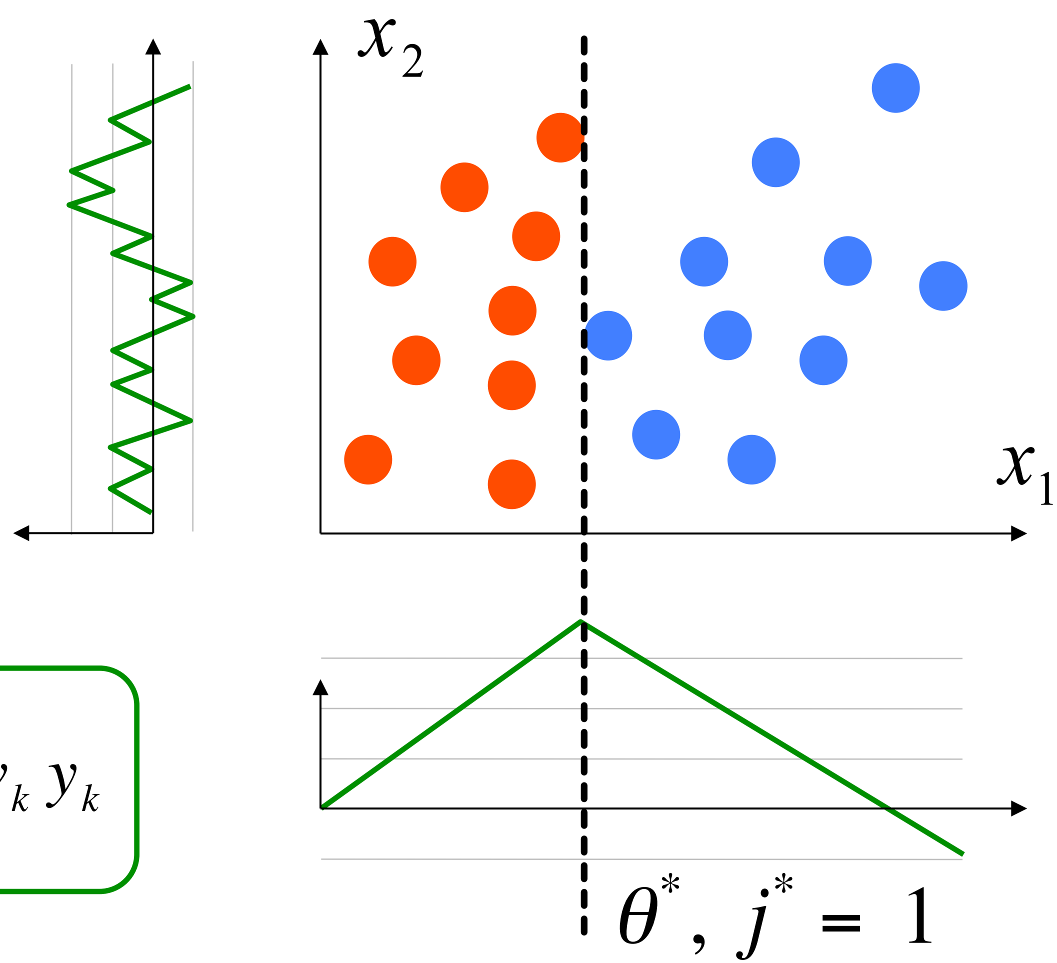
AdaBoost: Weak Classifier

Training algorithm for stumps: Intuition

- **Label** y **:**
  red: +
  blue: –

- Assuming all weights = 1

$$w^{j}_{cum}(i) = \sum_{k=1}^{i} w_k \, y_k$$



AdaBoost: Algorithm

Given the **training data** $\{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$ $\quad \mathbf{x} \in \mathcal{X} \quad y \in \mathcal{Y}$

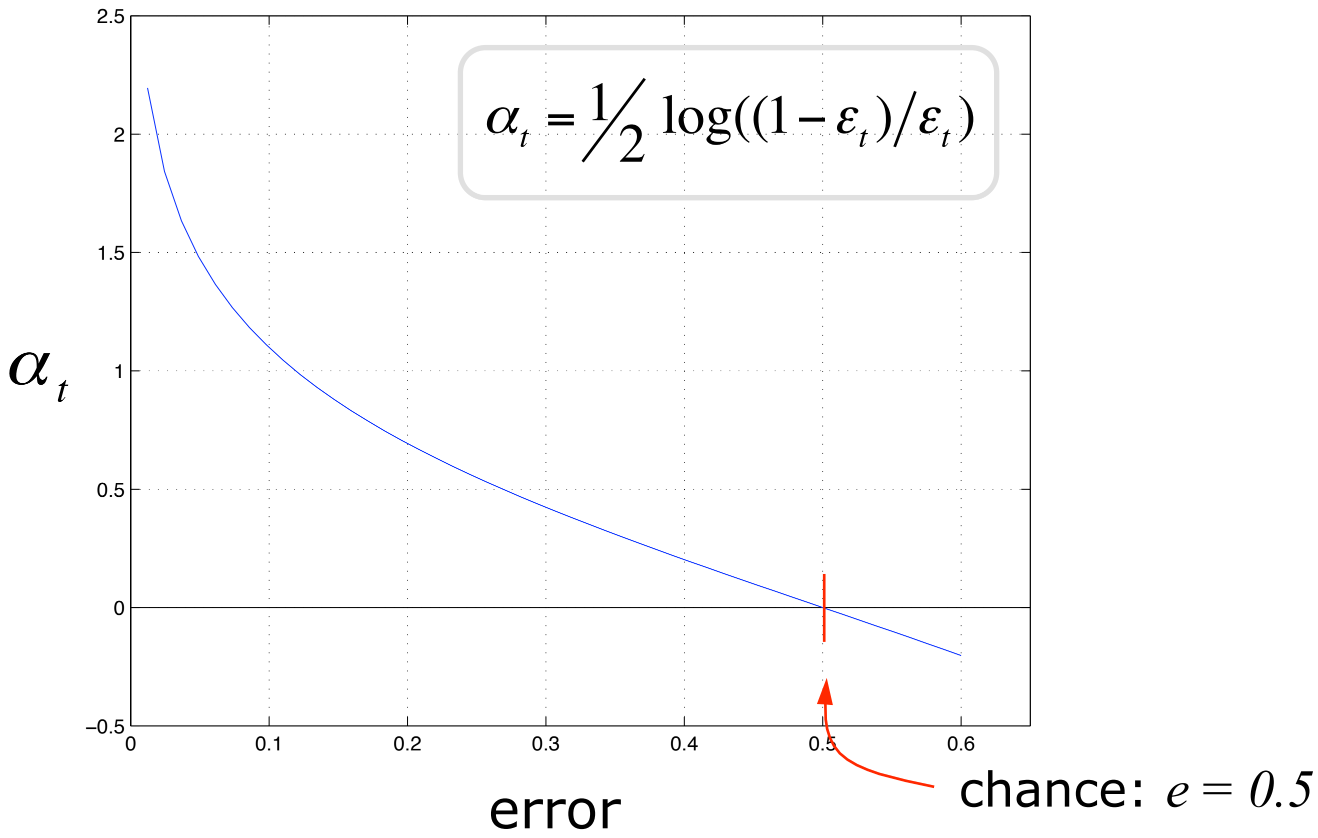
1. Initialize weights $w_t(i) = 1/n$

2. For $t = 1,\ldots,T$
   - Train a **weak classifier** $h_t(x)$ on weighted training data minimizing the error $\varepsilon_t = \sum_{i=1}^{n} w_t(i)\, \mathrm{I}(y_i \neq h_t(x_i))$
   - Compute voting weight of $h_t(x)$: $\alpha_t = \frac{1}{2} \log((1-\varepsilon_t)/\varepsilon_t)$
   - Recompute weights: $w_{t+1}(i) = w_t(i) \exp\{-\alpha_t\, y_i\, h_t(x_i)\} / Z_t$

3. Make predictions using the final **strong classifier**

AdaBoost: Voting Weight

- Computing the **voting weight** α_t of a weak classifier
- α_t measures the **importance** assigned to $h_t(x_i)$



AdaBoost: Weight Update

- Looking at the weight update step:

$$w_{t+1}(i) = w_t(i) \exp\left\{-\alpha_t \, y_i \, h_t(x_i)\right\} / Z_t$$

Z_t : **Normalizer** such that w_{t+1} is a prob. distribution

$$\exp\left\{-\alpha_t \, y_i \, h_t(x_i)\right\} = \begin{cases} <1, & y_i = h_t(x_i) \\ >1, & y_i \neq h_t(x_i) \end{cases}$$

→ Weights of misclassified training samples are **increased**

→ Weights of correctly classified samples are **decreased**

- Algorithm generates weak classifier by training the next learner on the **mistakes of the previous one**
- Now we understand the name: **AdaBoost** comes from **adaptive Boosting**

AdaBoost: Strong Classifier

- **Training is completed...**

  The weak classifiers $h_{1...T}(x)$ and their voting weight $\alpha_{1....T}$ are now fix

- **The resulting strong classifier is**

$$H(x_i) = \mathrm{sgn}\left(\sum_{t=1}^{T} \alpha_t h_t(x_i) \right) \longrightarrow \text{Class Result } \{+1, -1\}$$

Put your data here

Weighted majority voting scheme

AdaBoost: Algorithm

Given the **training data** $\{(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n)\}$ $\quad \mathbf{x} \in \mathcal{X} \quad y \in \mathcal{Y}$

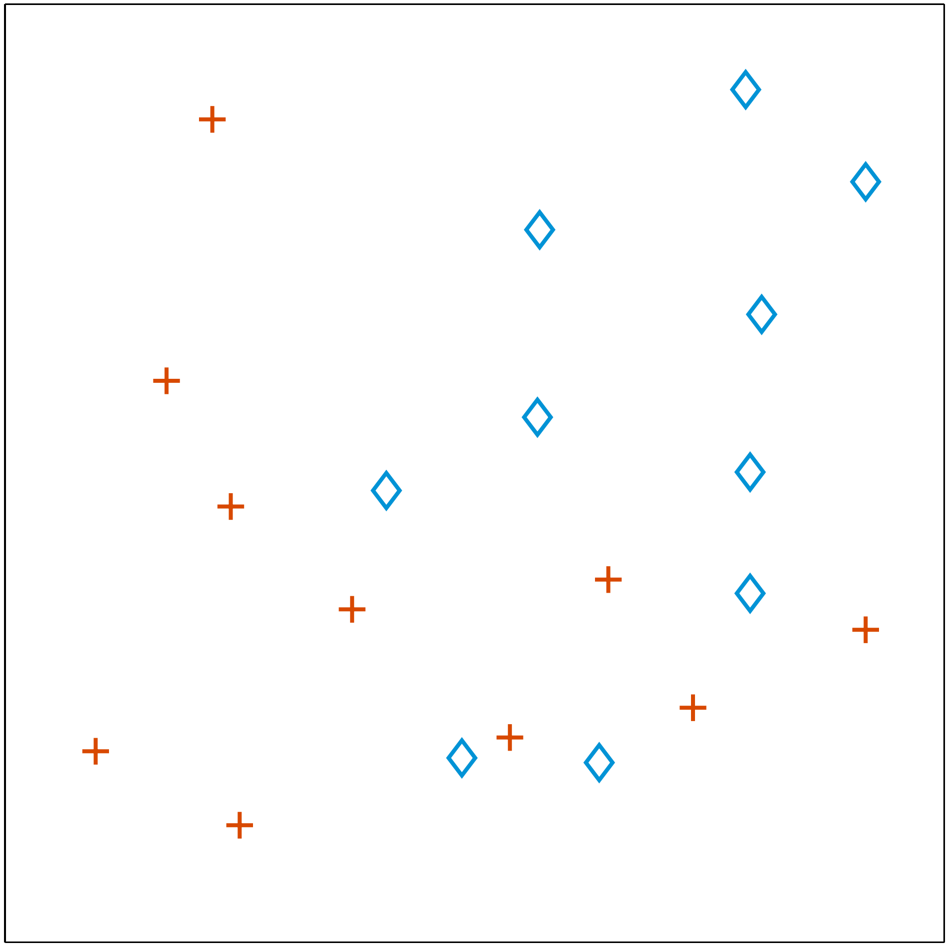
1. Initialize weights $w_t(i) = 1/n$

2. For $t = 1, \ldots, T$
   - Train a **weak classifier** $h_t(x)$ on weighted training data minimizing the error $\varepsilon_t = \sum_{i=1}^{n} w_t(i) \, \mathrm{I}(y_i \neq h_t(x_i))$
   - Compute voting weight of $h_t(x)$: $\alpha_t = \frac{1}{2} \log((1-\varepsilon_t)/\varepsilon_t)$
   - Recompute weights: $w_{t+1}(i) = w_t(i) \exp\{-\alpha_t \, y_i \, h_t(x_i)\} / Z_t$

3. Make predictions using the final **strong classifier**

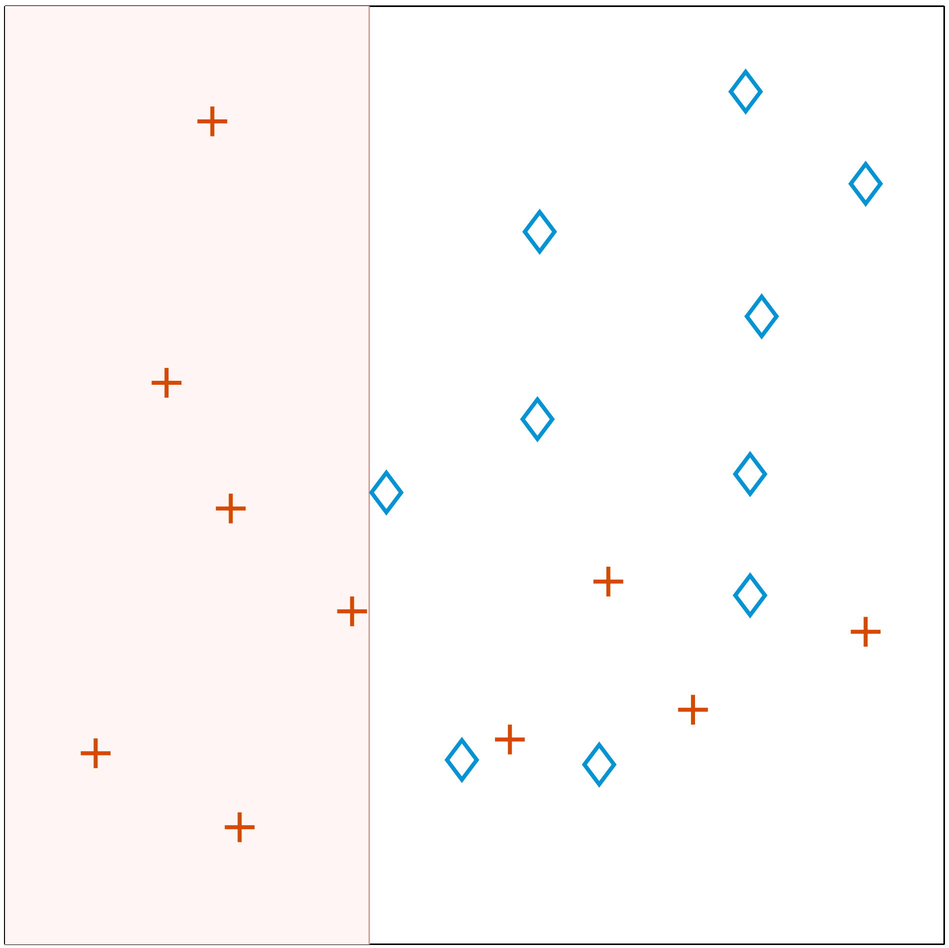
AdaBoost: Step-By-Step

- **Training data**



AdaBoost: Step-By-Step

- **Iteration 1, train weak classifier 1**



Threshold
$\theta^* = 0.37$

Dimension
$j^* = 1$

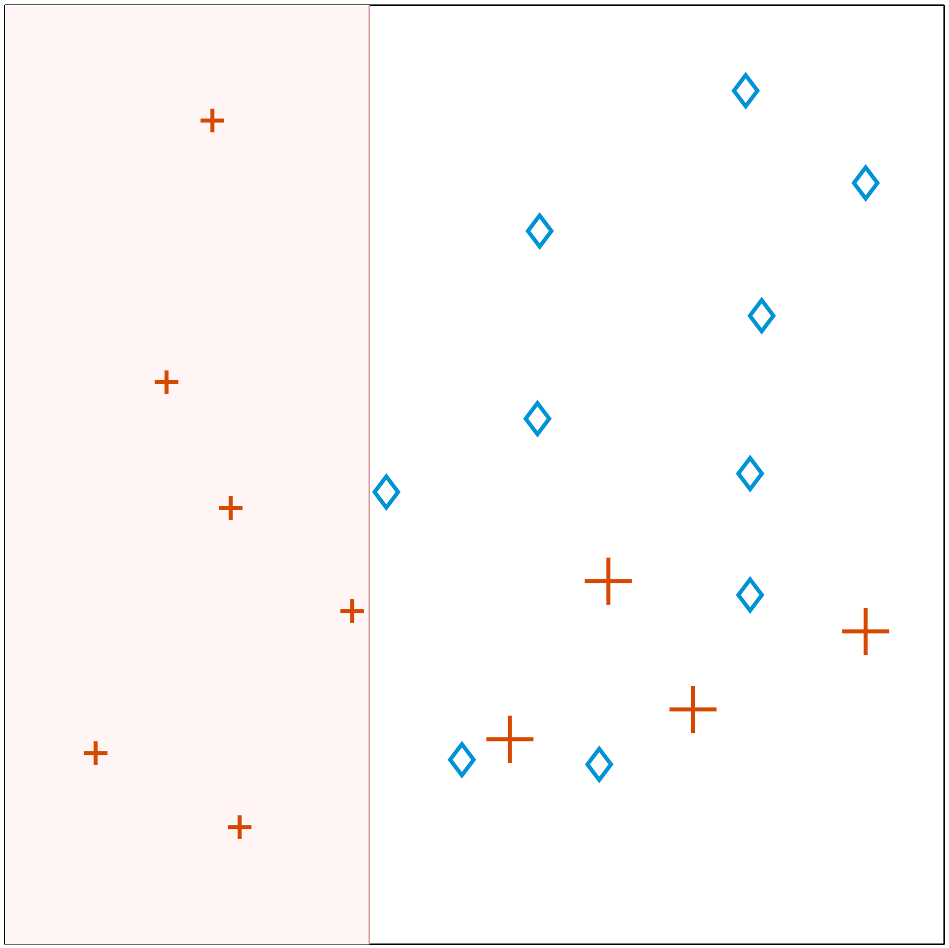
Weighted error
$e_t = 0.2$

Voting weight
$\alpha_t = 1.39$

Total error = 4

AdaBoost: Step-By-Step

- **Iteration 1, recompute weights**



Threshold
$\theta^* = 0.37$

Dimension
$j^* = 1$

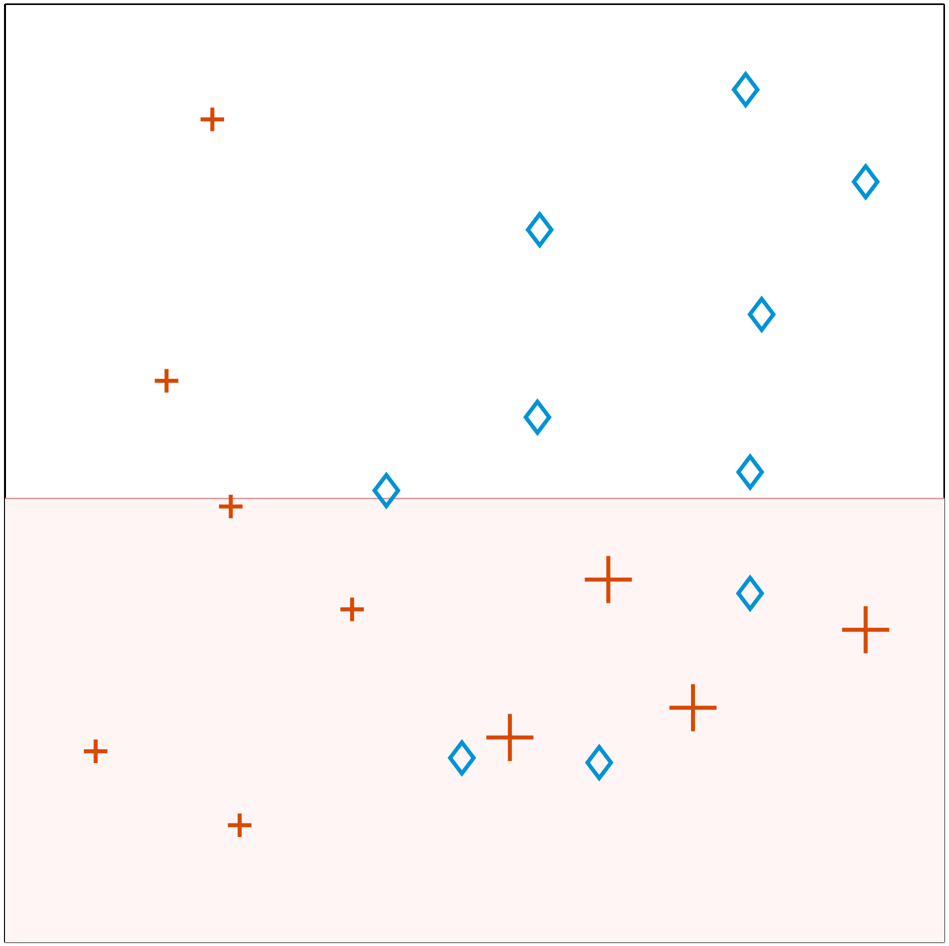
Weighted error
$e_t = 0.2$

Voting weight
$\alpha_t = 1.39$

Total error = 4

AdaBoost: Step-By-Step

- **Iteration 2, train weak classifier 2**



Threshold
$\theta^* = 0.47$

Dimension
$j^* = 2$

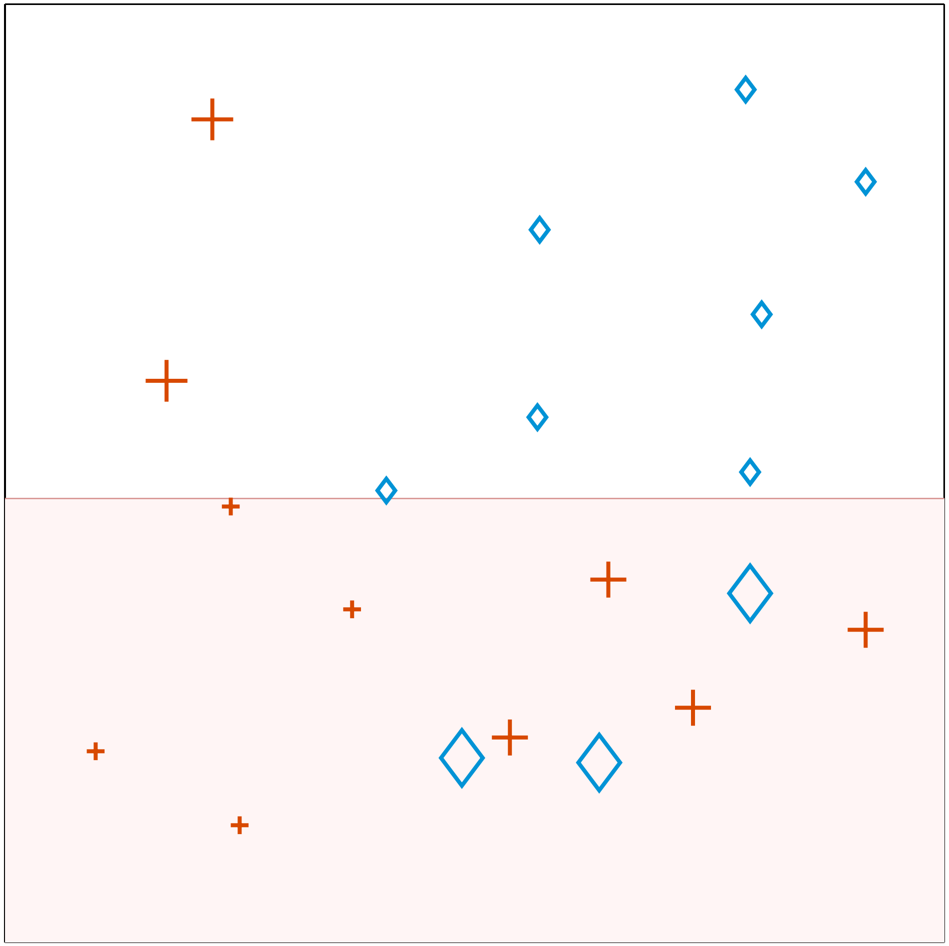
Weighted error
$e_t = 0.16$

Voting weight
$\alpha_t = 1.69$

Total error = 5

AdaBoost: Step-By-Step

- **Iteration 2, recompute weights**



Threshold
$\theta^* = 0.47$

Dimension
$j^* = 2$

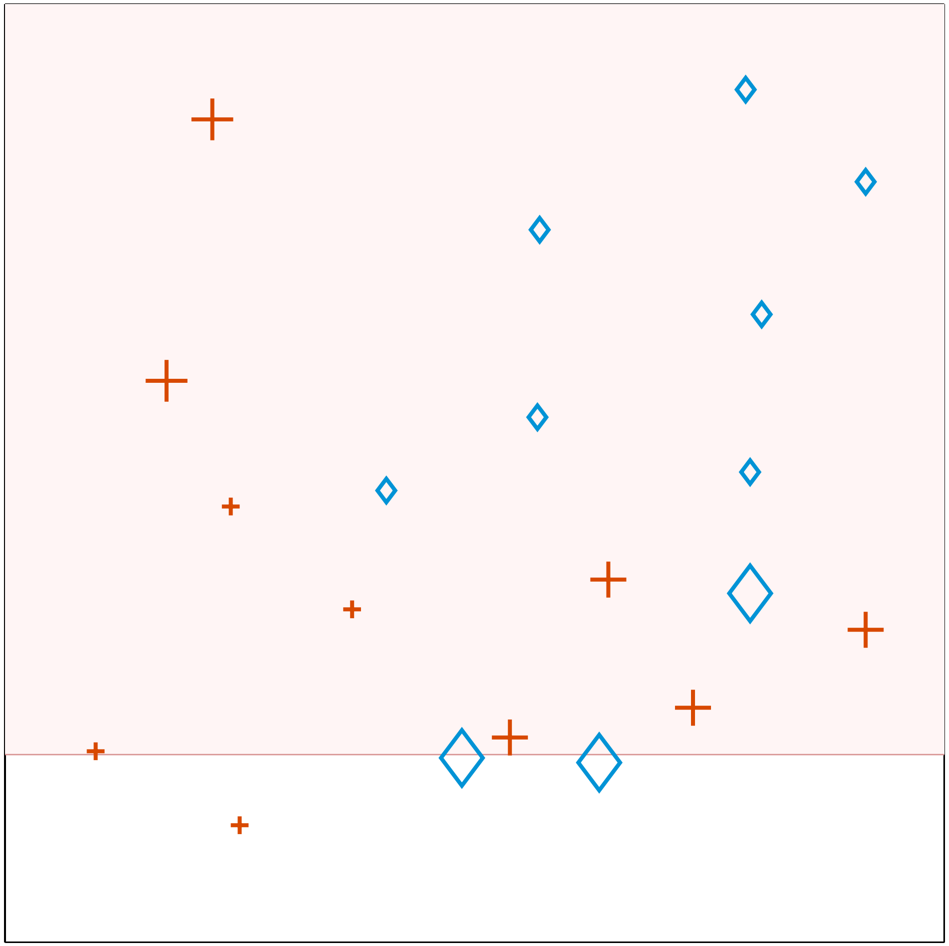
Weighted error
$e_t = 0.16$

Voting weight
$\alpha_t = 1.69$

Total error = 5

AdaBoost: Step-By-Step

- **Iteration 3, train weak classifier 3**



Threshold
$\theta^* = 0.14$

Dimension, sign
$j^* = 2$, neg

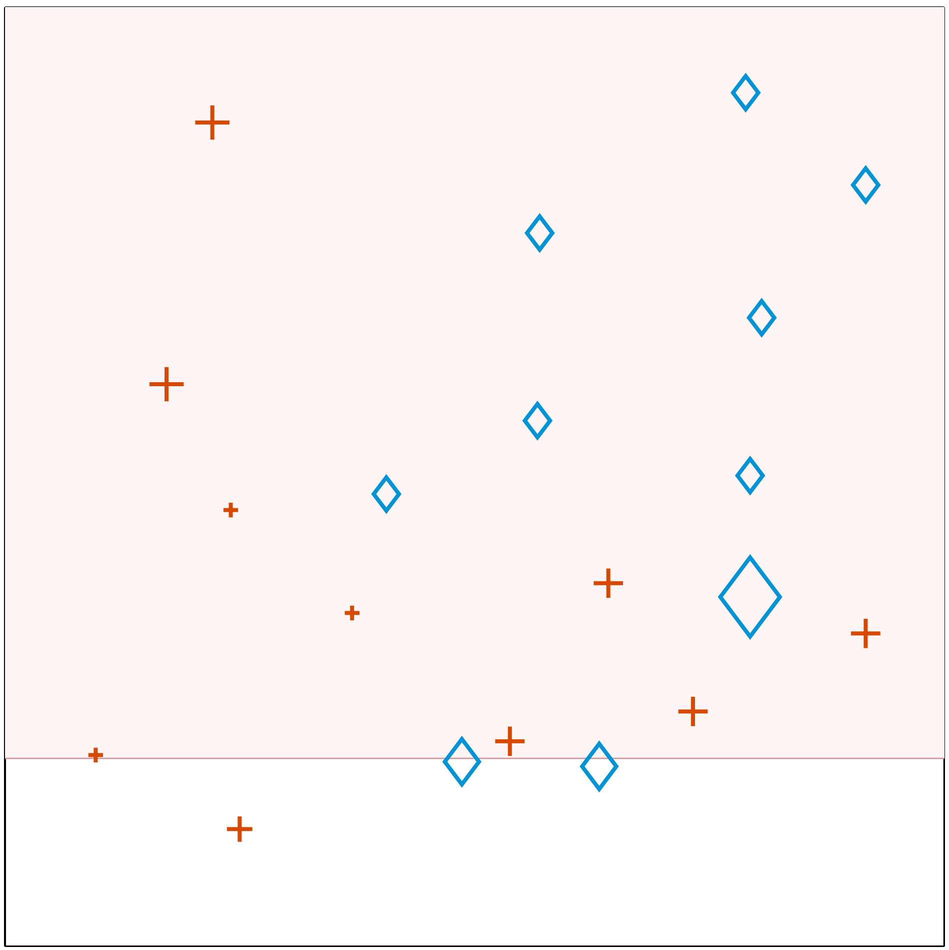
Weighted error
$e_t = 0.25$

Voting weight
$\alpha_t = 1.11$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 3, recompute weights**



Threshold
$\theta^* = 0.14$

Dimension, sign
$j^* = 2$, neg

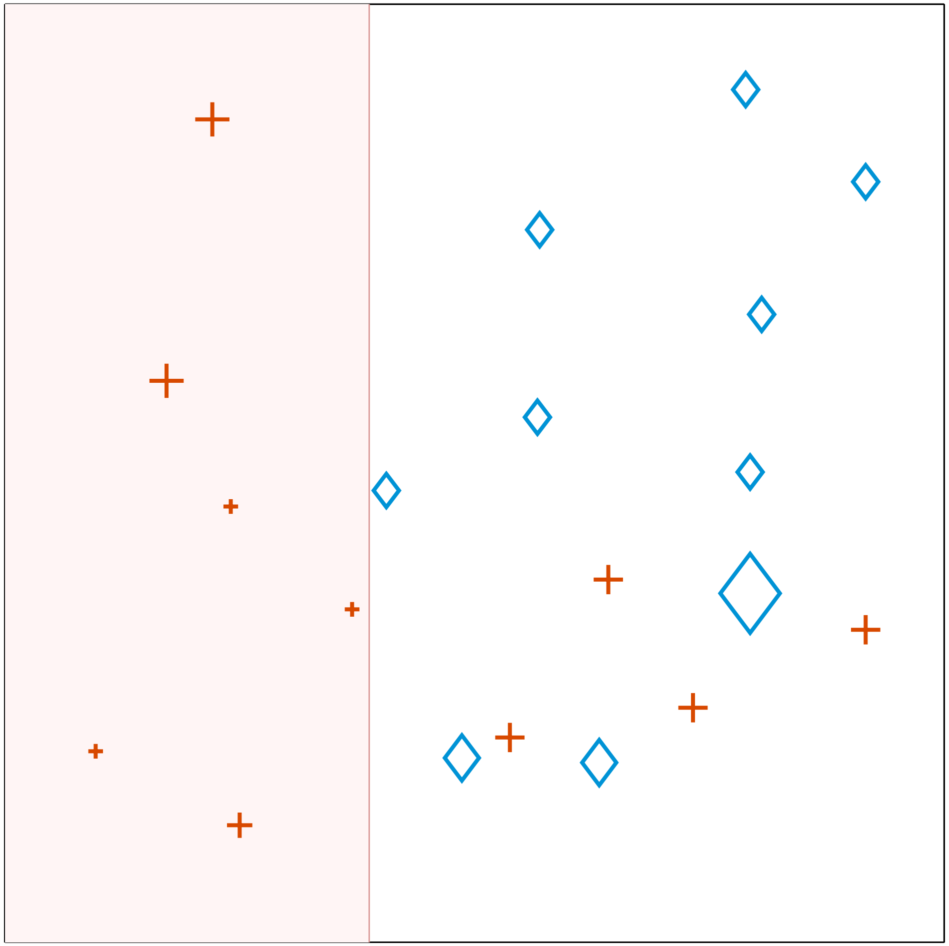
Weighted error
$e_t = 0.25$

Voting weight
$\alpha_t = 1.11$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 4, train weak classifier 4**



Threshold
$\theta^* = 0.37$

Dimension
$j^* = 1$

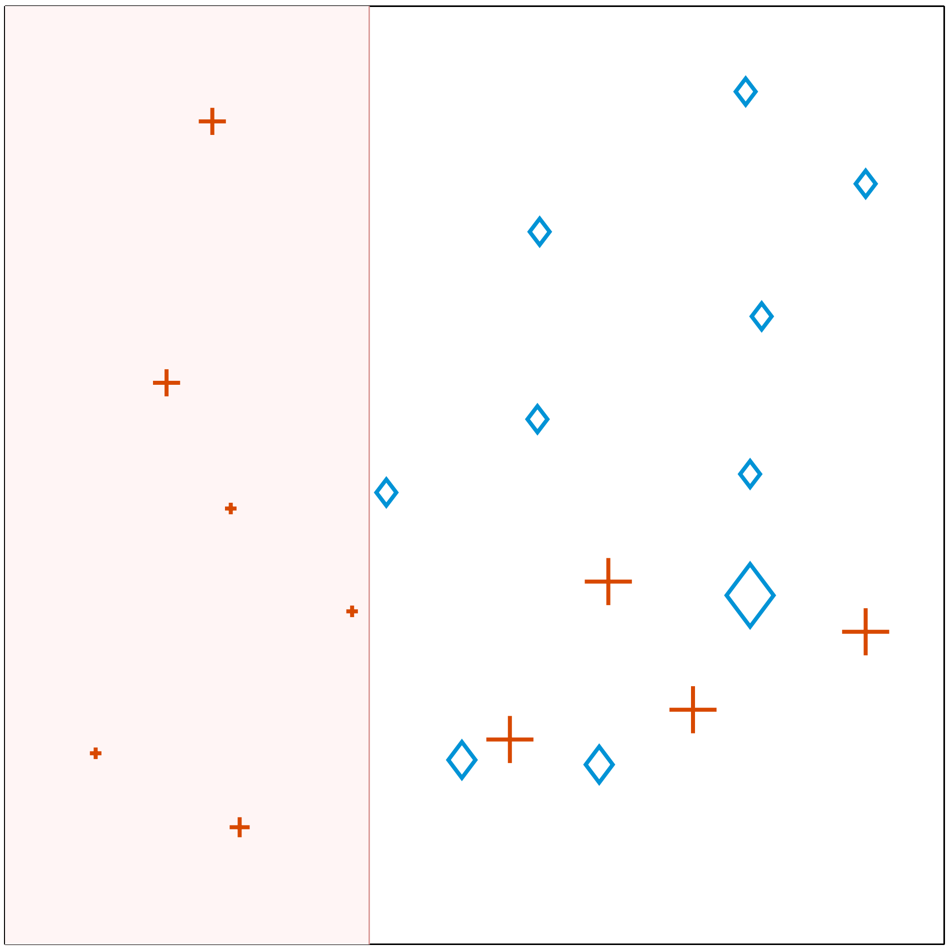
Weighted error
$e_t = 0.20$

Voting weight
$\alpha_t = 1.40$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 4, recompute weights**



Threshold
$\theta^* = 0.37$

Dimension
$j^* = 1$

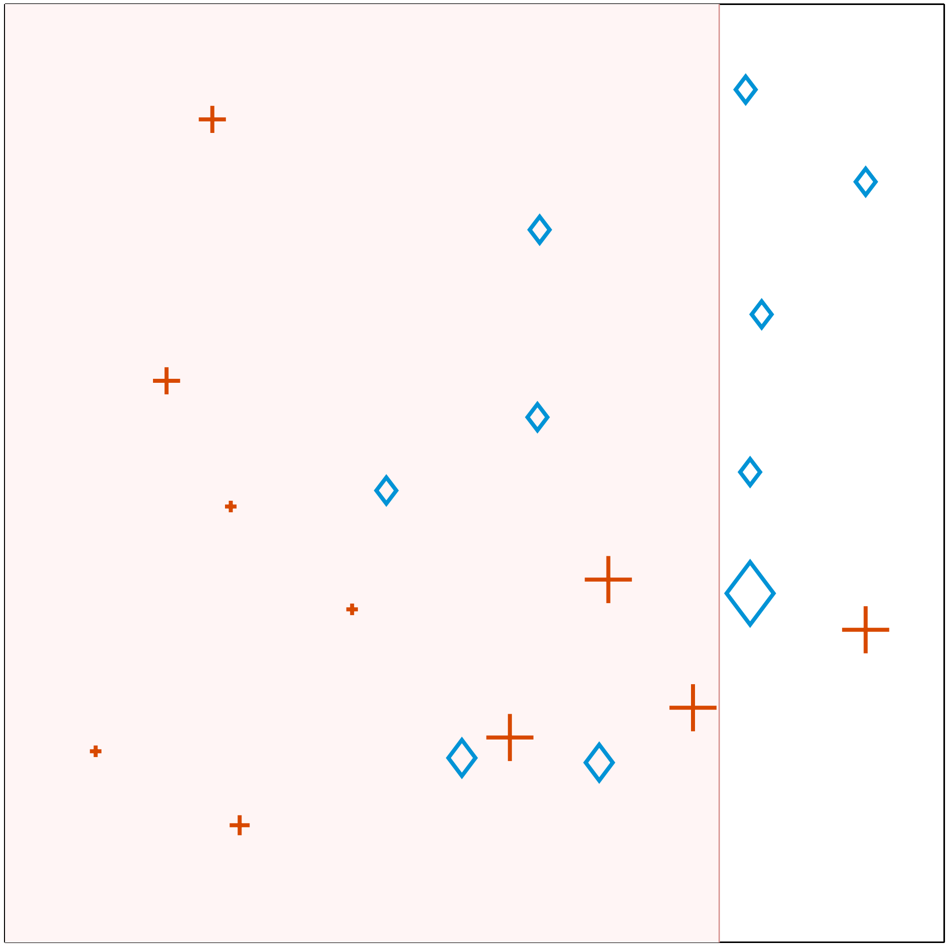
Weighted error
$e_t = 0.20$

Voting weight
$\alpha_t = 1.40$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 5, train weak classifier 5**



Threshold
$\theta^* = 0.81$

Dimension
$j^* = 1$

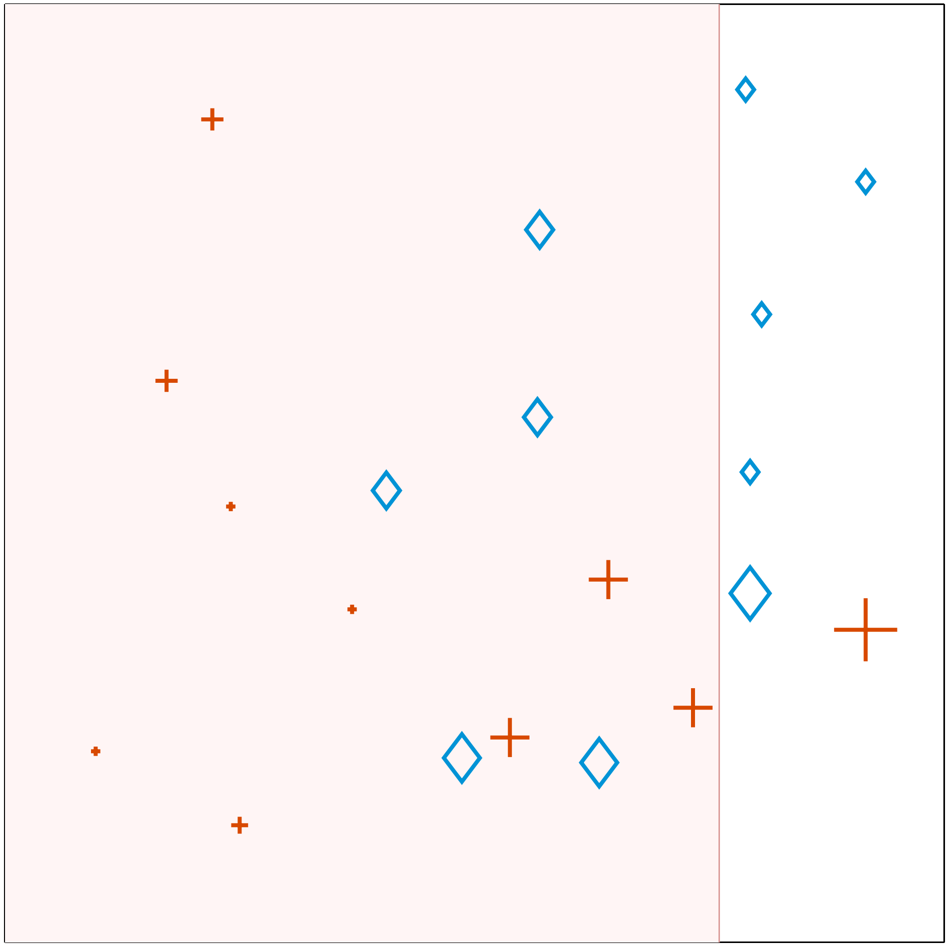
Weighted error
$e_t = 0.28$

Voting weight
$\alpha_t = 0.96$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 5, recompute weights**



Threshold
$\theta^* = 0.81$

Dimension
$j^* = 1$

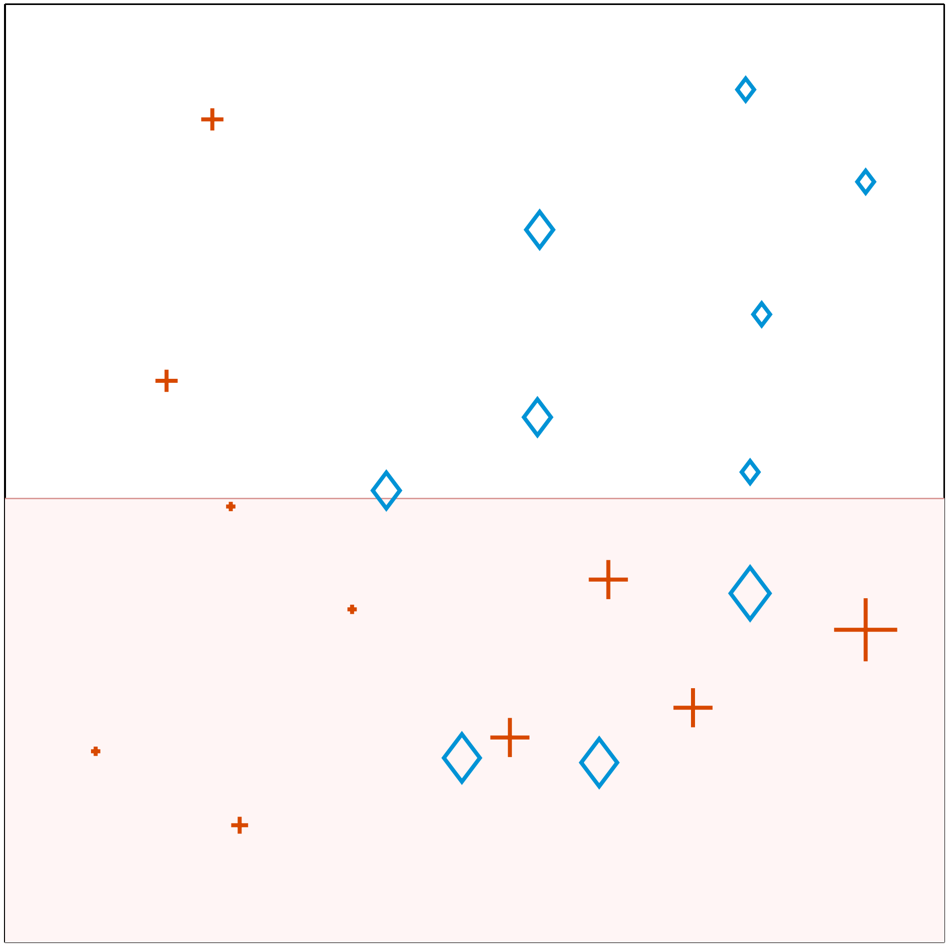
Weighted error
$e_t = 0.28$

Voting weight
$\alpha_t = 0.96$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 6, train weak classifier 6**



Threshold
$\theta^* = 0.47$

Dimension
$j^* = 2$

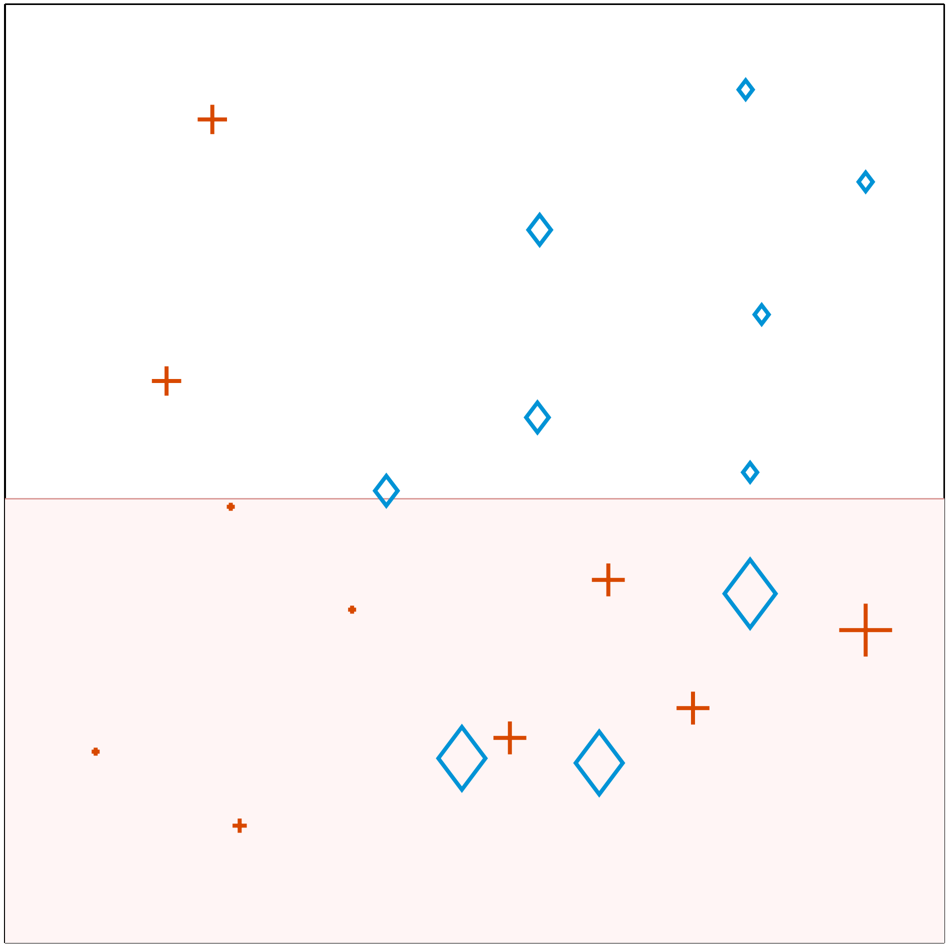
Weighted error
$e_t = 0.29$

Voting weight
$\alpha_t = 0.88$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 6, recompute weights**



Threshold
$\theta^* = 0.47$

Dimension
$j^* = 2$

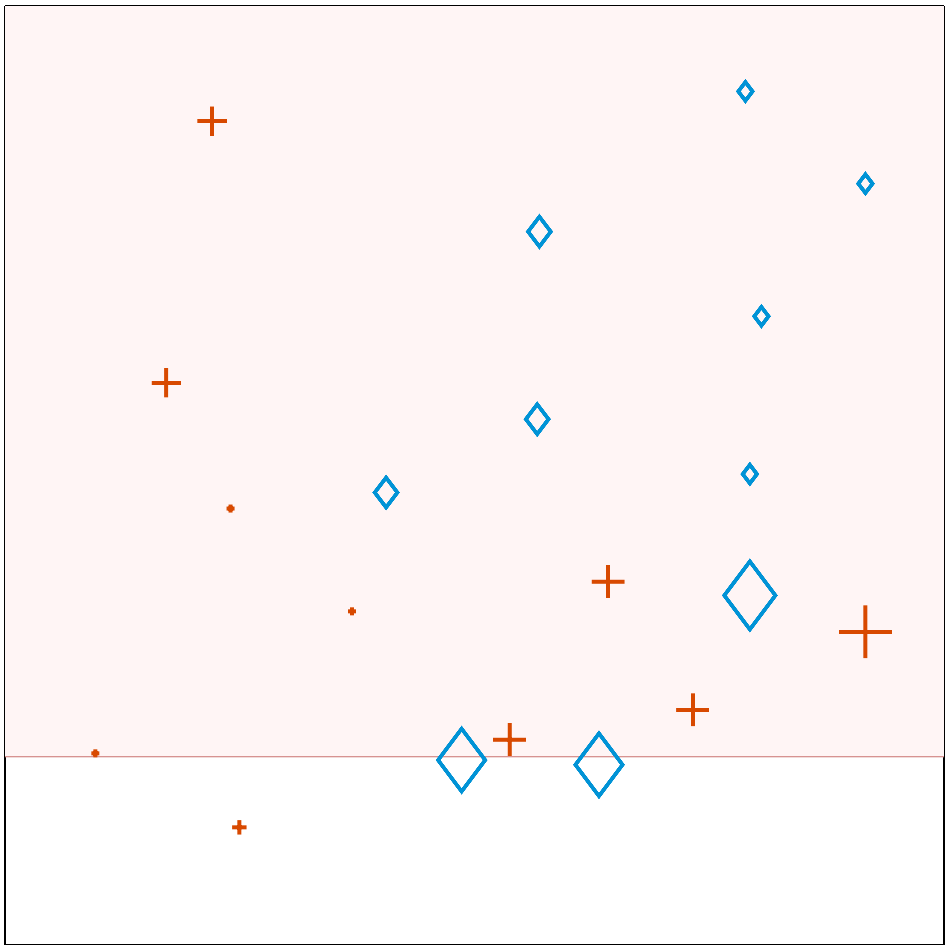
Weighted error
$e_t = 0.29$

Voting weight
$\alpha_t = 0.88$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 7, train weak classifier 7**



Threshold
$\theta^* = 0.14$

Dimension, sign
$j^* = 2$, neg

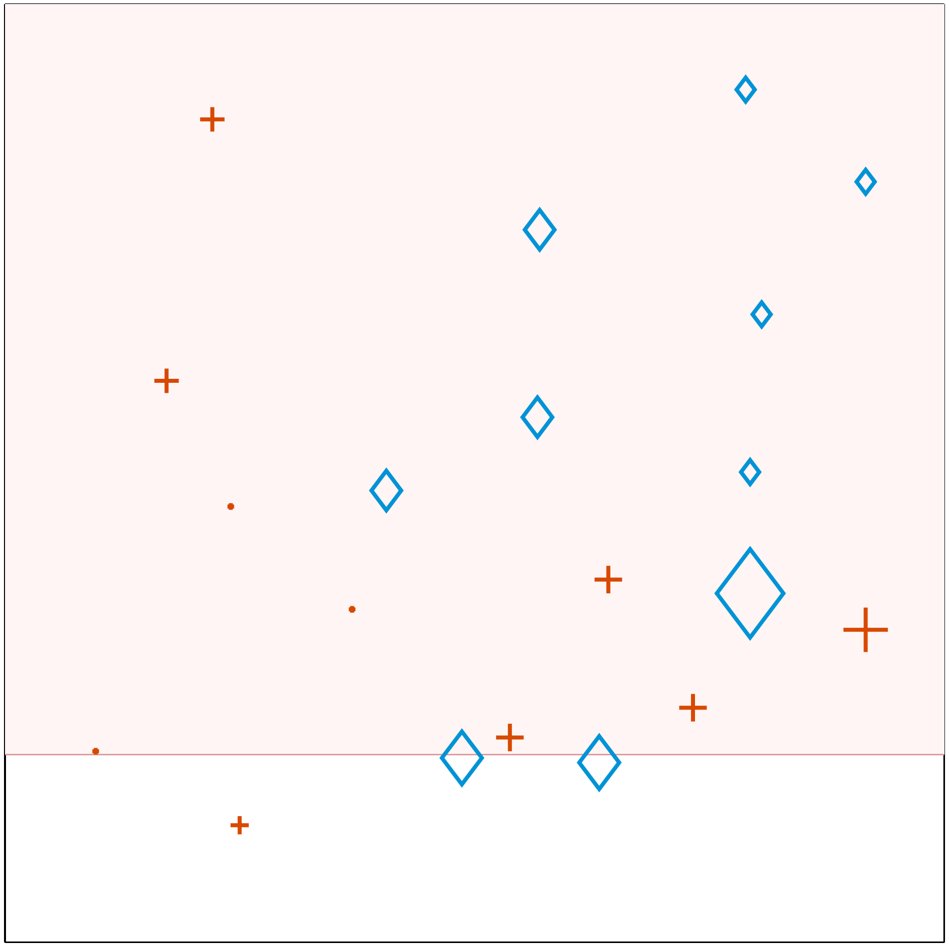
Weighted error
$e_t = 0.29$

Voting weight
$\alpha_t = 0.88$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 7, recompute weights**



Threshold
$\theta^* = 0.14$

Dimension, sign
$j^* = 2$, neg

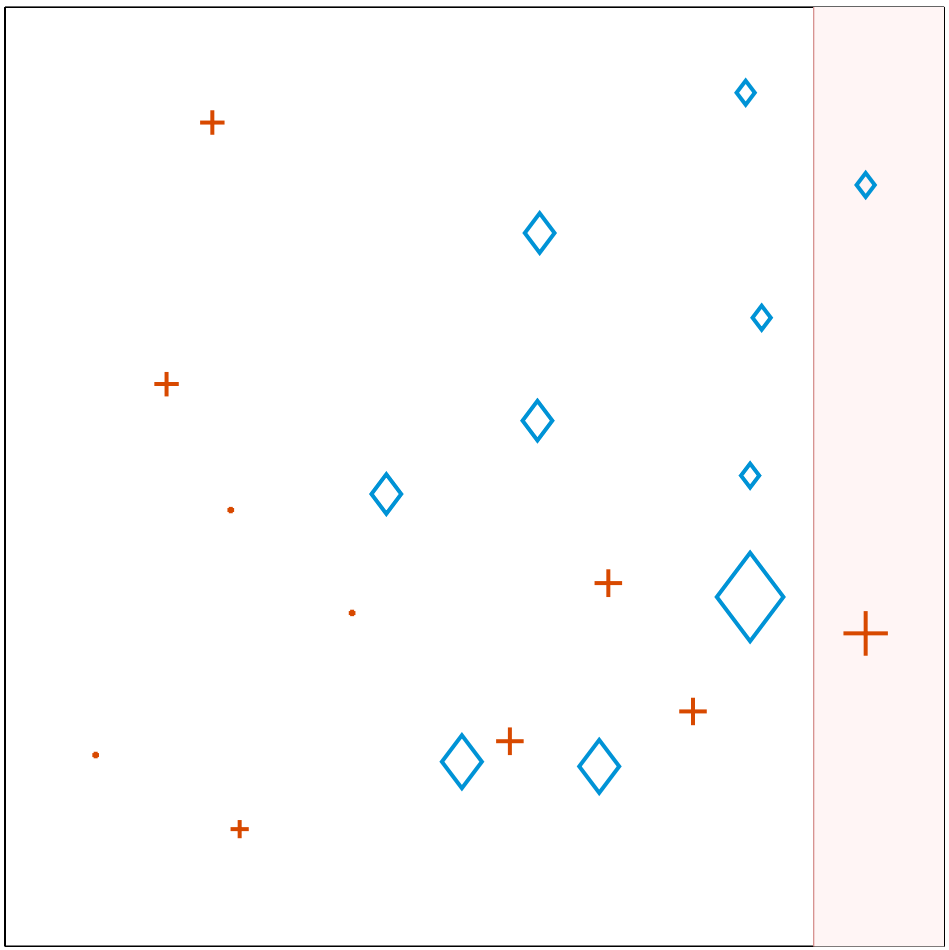
Weighted error
$e_t = 0.29$

Voting weight
$\alpha_t = 0.88$

Total error = 1

AdaBoost: Step-By-Step

- **Iteration 8, train weak classifier 8**



Threshold
$\theta^* = 0.93$

Dimension, sign
$j^* = 1$, neg

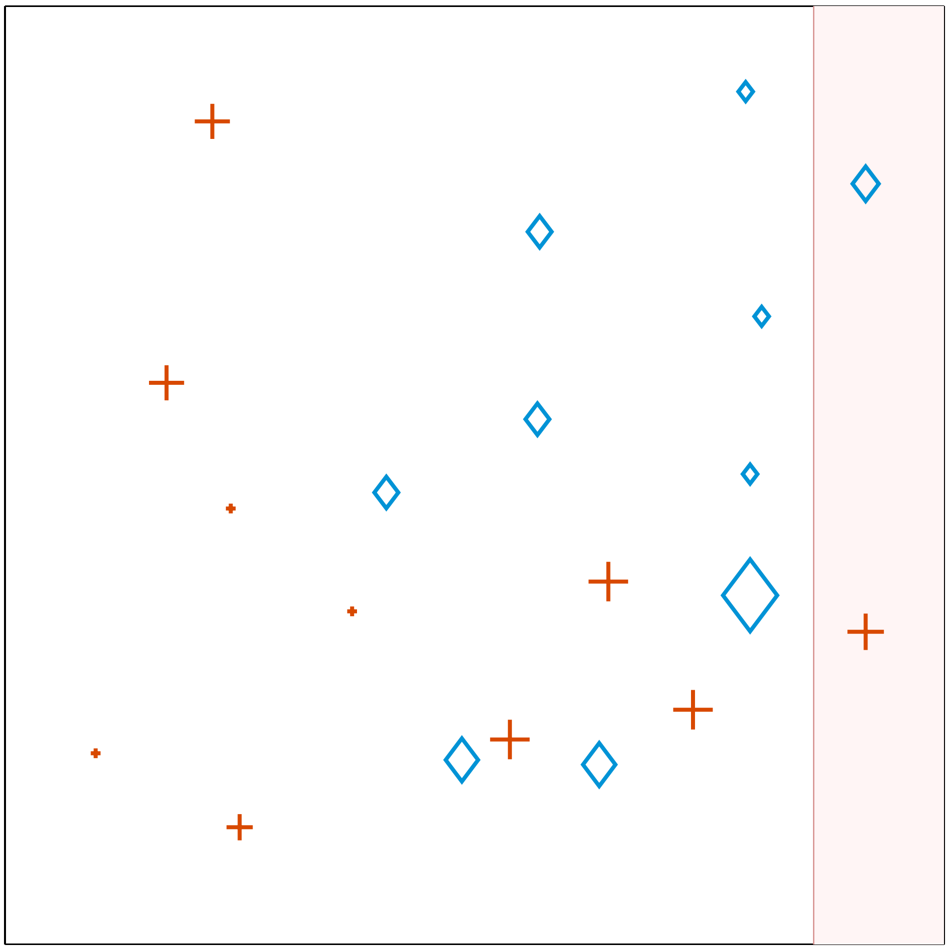
Weighted error
$e_t = 0.25$

Voting weight
$\alpha_t = 1.12$

Total error = 0

AdaBoost: Step-By-Step

- **Iteration 8, recompute weights**



Threshold
$\theta^* = 0.93$

Dimension, sign
$j^* = 1$, neg

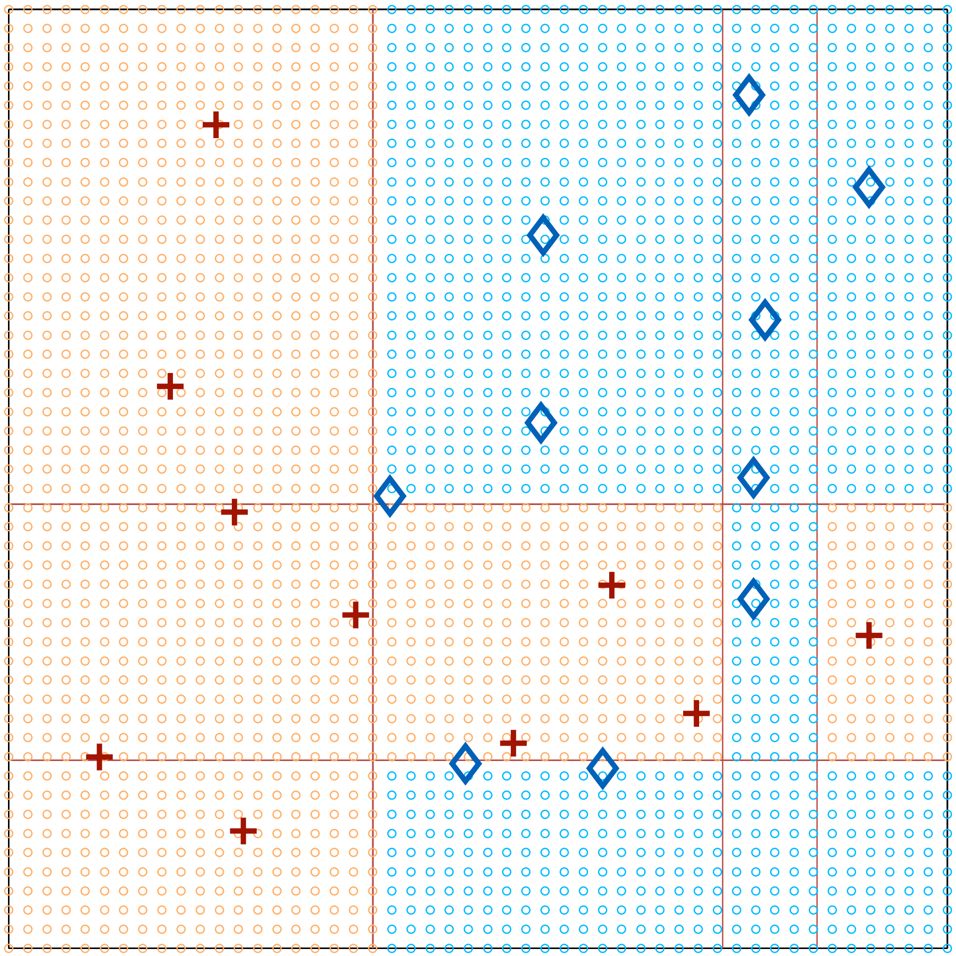
Weighted error
$e_t = 0.25$

Voting weight
$\alpha_t = 1.12$

Total error = 0

AdaBoost: Step-By-Step

- **Final Strong Classifier**



Total training error = 0

(Rare in practice)