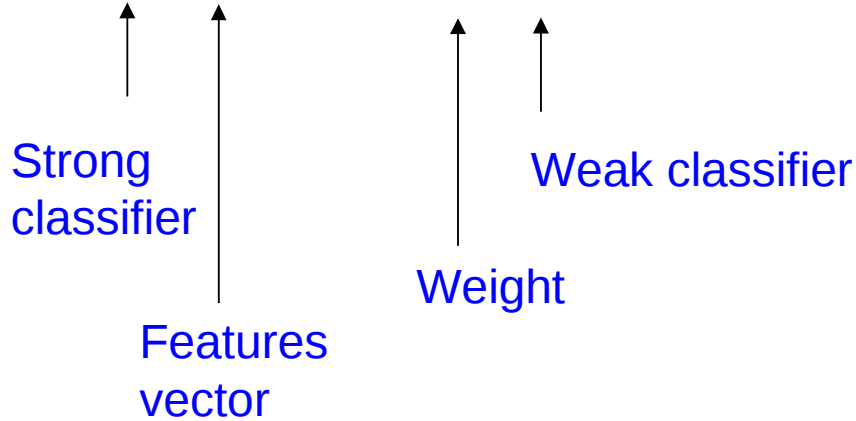


Boosting

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \dots$$



Boosting

- Defines a classifier using an additive model:

$$F(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x) + \alpha_3 f_3(x) + \dots$$

Strong classifier

Weak classifier

Weight

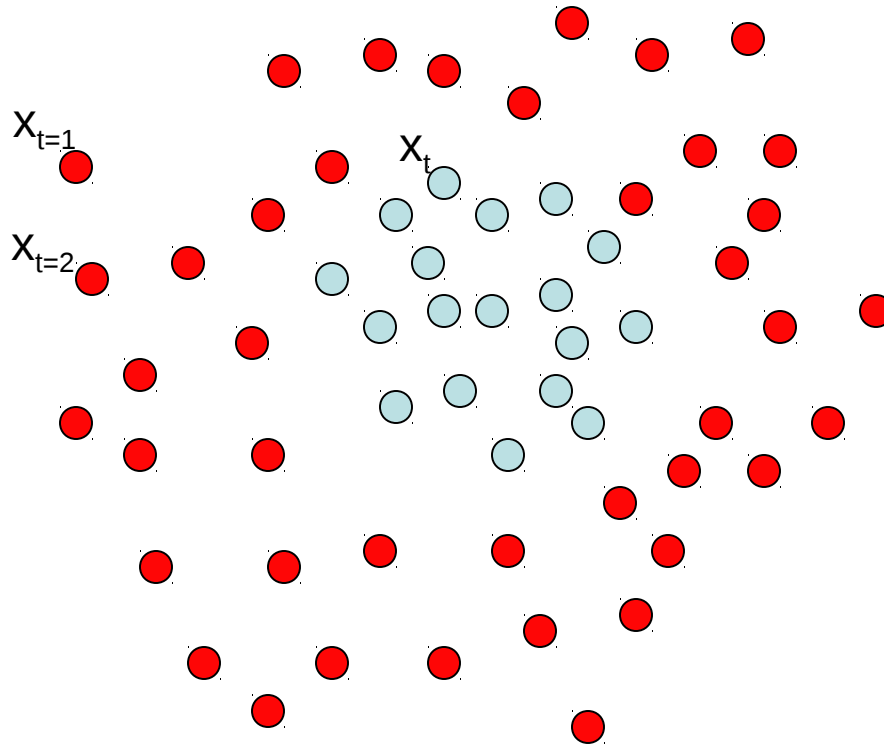
Features vector

- We need to define a family of weak classifiers

$f_k(x)$ from a family of weak classifiers

Boosting

- It is a sequential procedure:



Each data point has
a class label:

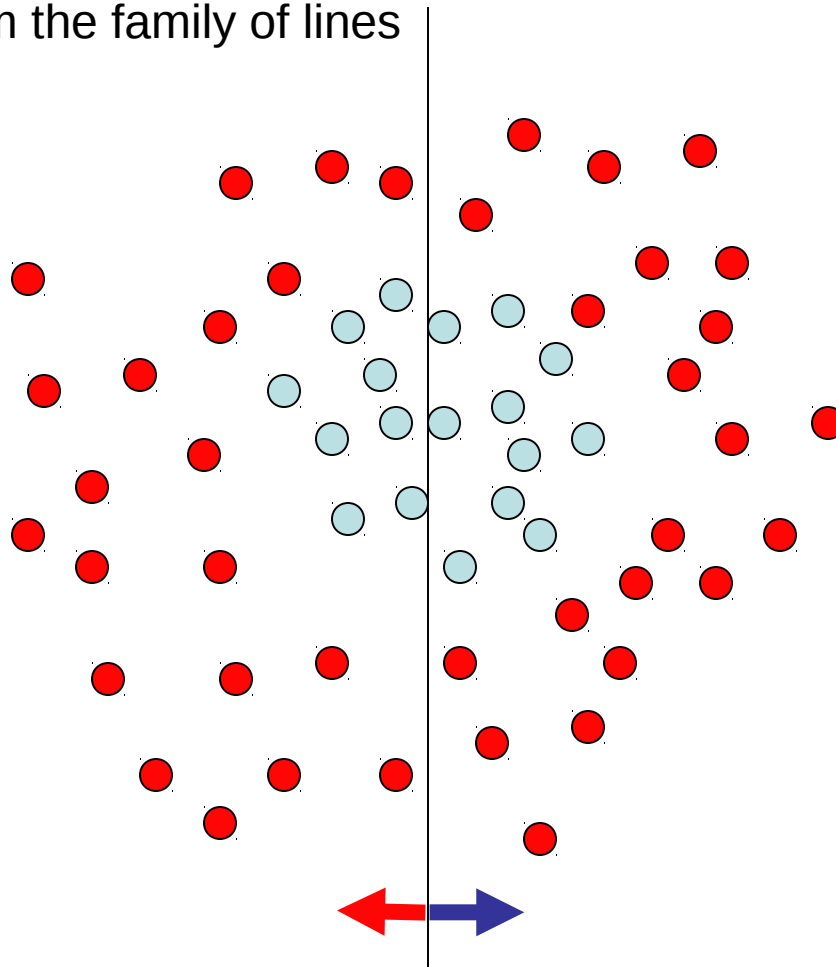
$$y_t = \begin{cases} +1 (\bullet) \\ -1 (\circ) \end{cases}$$

and a weight:

$$w_t = 1$$

Toy example

Weak learners from the family of lines



Each data point has
a class label:

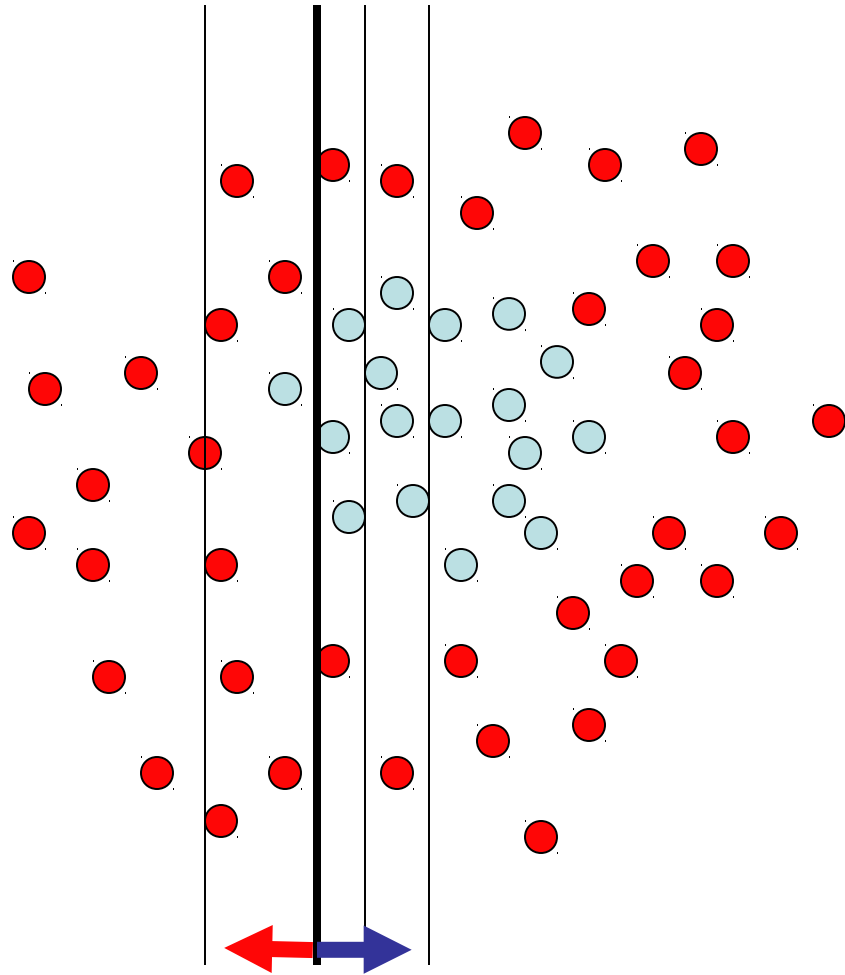
$$y_t = \begin{cases} +1 (\bullet) \\ -1 (\circ) \end{cases}$$

and a weight:

$$w_t = 1$$

$h \Rightarrow p(\text{error}) = 0.5$ it is at chance

Toy example



Each data point has
a class label:

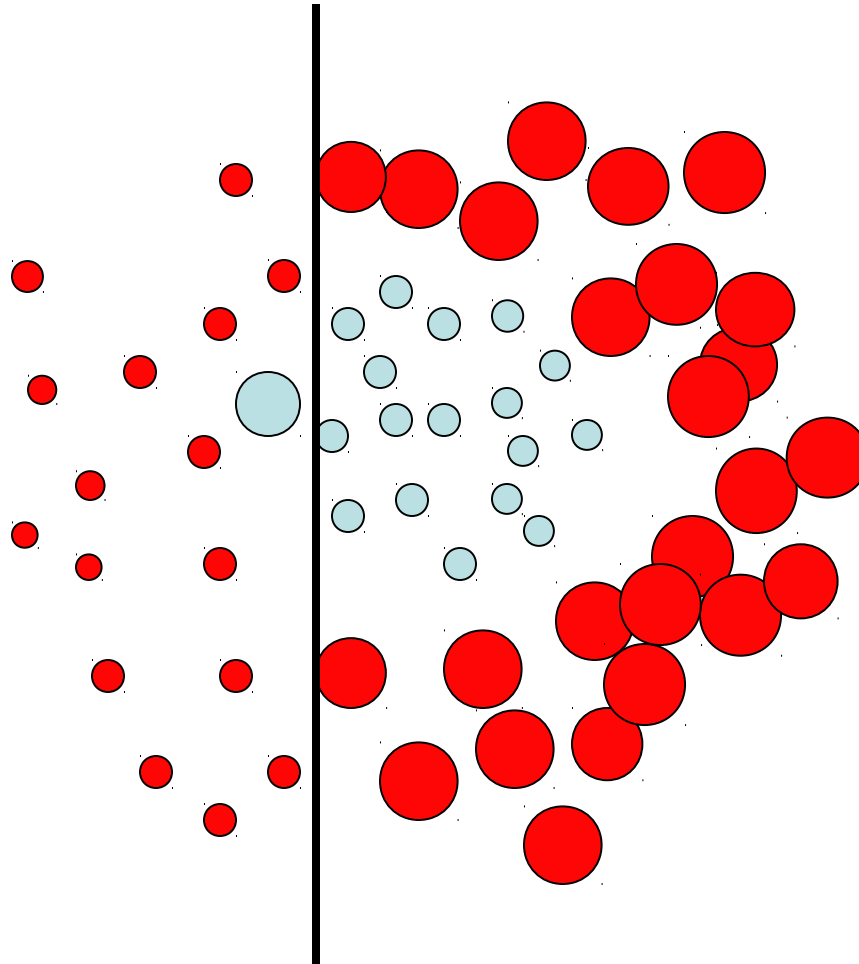
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

and a weight:
 $w_t = 1$

This one seems to be the best

This is a '**weak classifier**': It performs slightly better than chance.

Toy example



Each data point has
a class label:

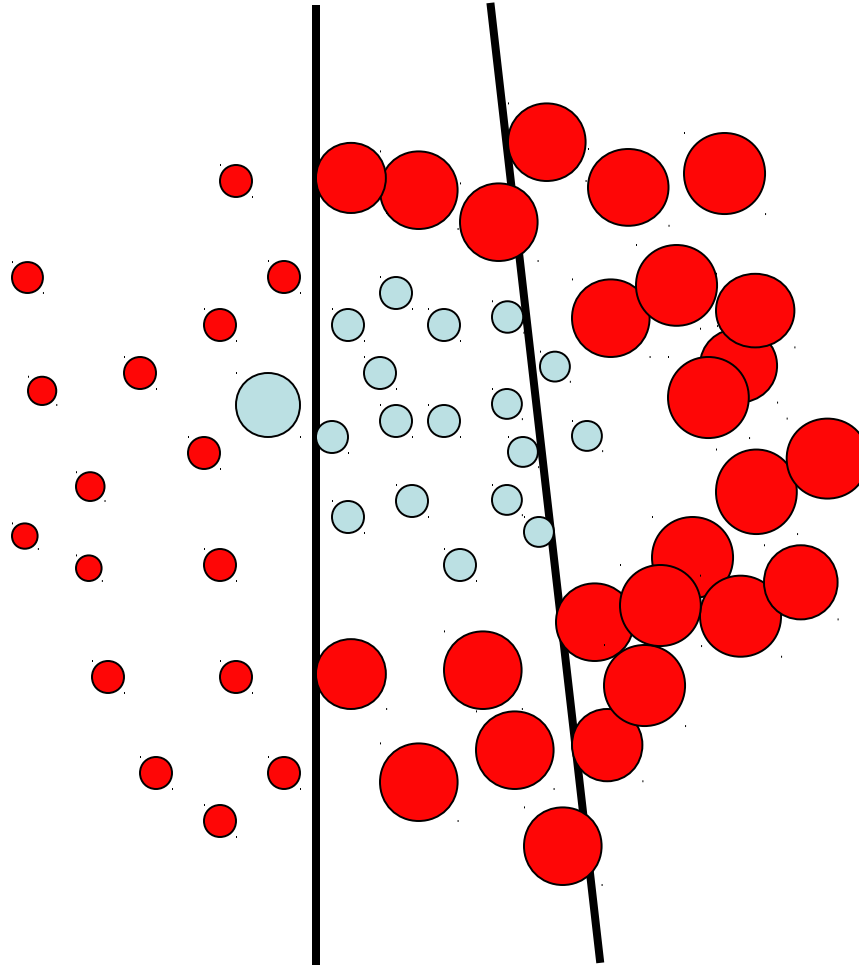
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has
a class label:

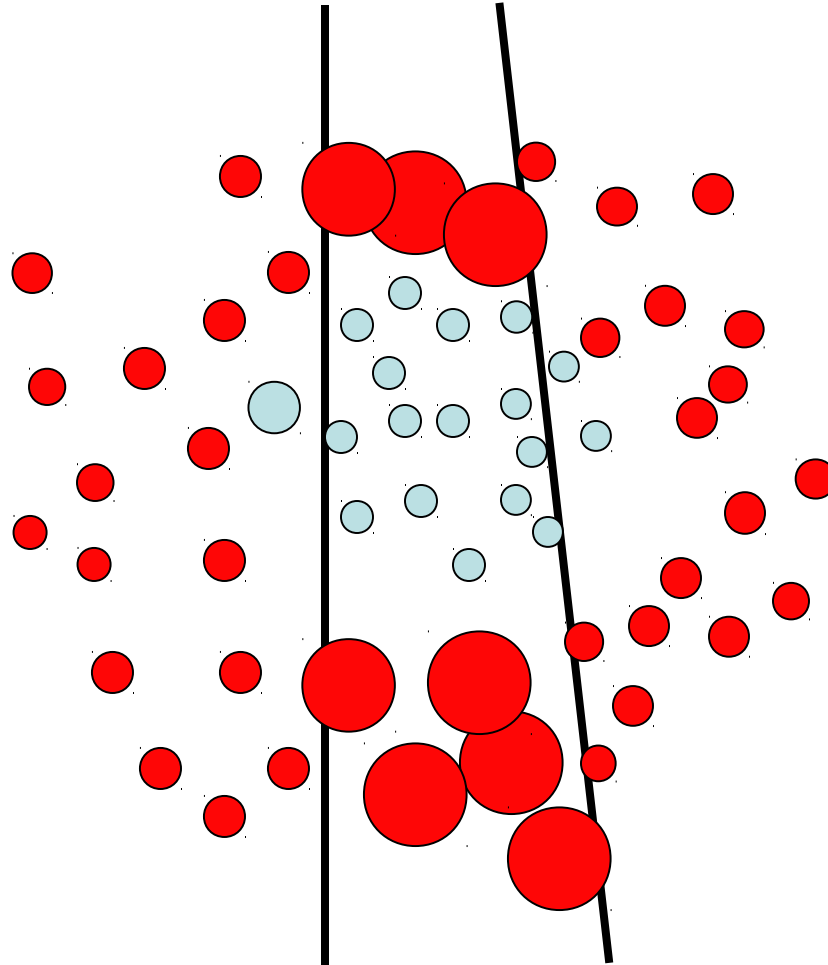
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has
a class label:

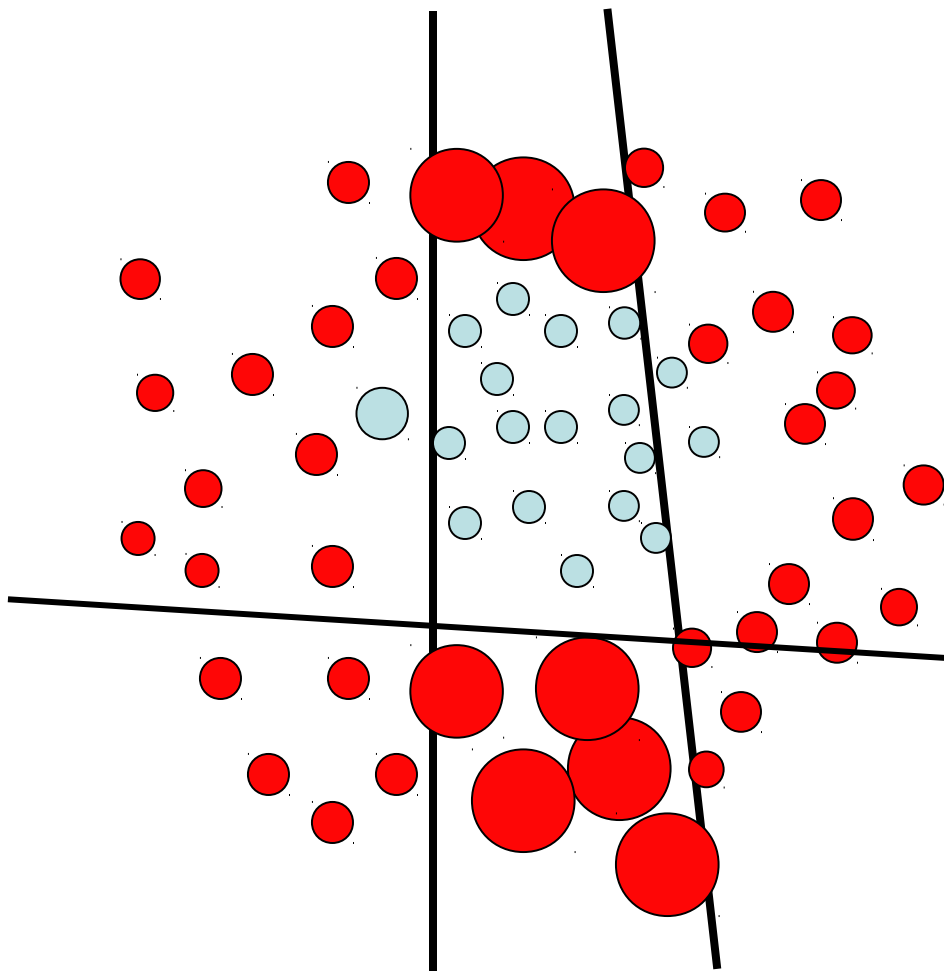
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



Each data point has
a class label:

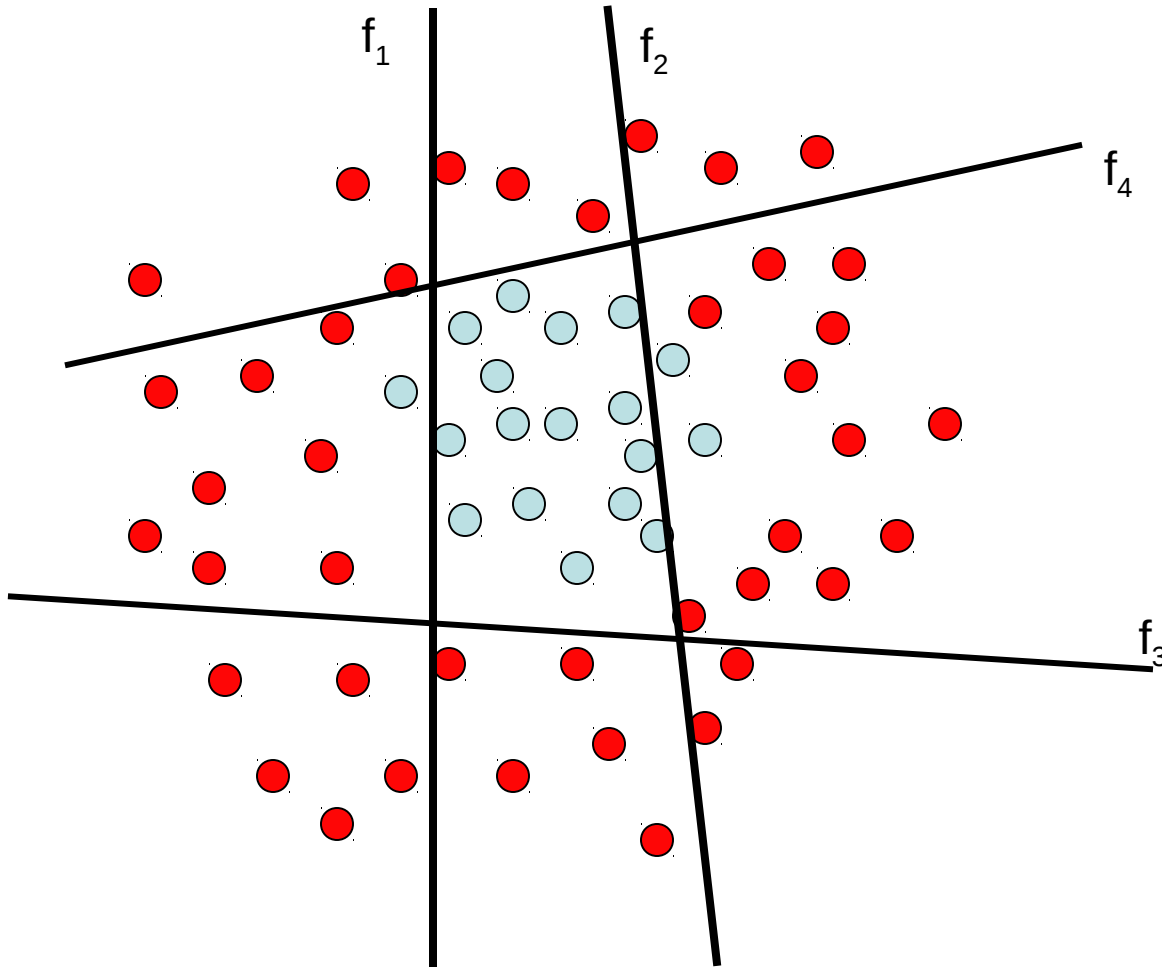
$$y_t = \begin{cases} +1 & (\bullet) \\ -1 & (\circ) \end{cases}$$

We update the weights:

$$w_t \leftarrow w_t \exp\{-y_t H_t\}$$

We set a new problem for which the previous weak classifier performs at chance again

Toy example



The strong (non-linear) classifier is built as the combination of all the weak (linear) classifiers.

Boosting

- Different cost functions and minimization algorithms result in various flavors of Boosting
- In this demo, I will use gentleBoosting: it is simple to implement and numerically stable.

Overview of section

- Object detection with classifiers
- Boosting
 - **Gentle boosting**
 - Weak detectors
 - Object model
 - Object detection

Boosting

Boosting fits the additive model

$$F(x) = f_1(x) + f_2(x) + f_3(x) + \dots$$

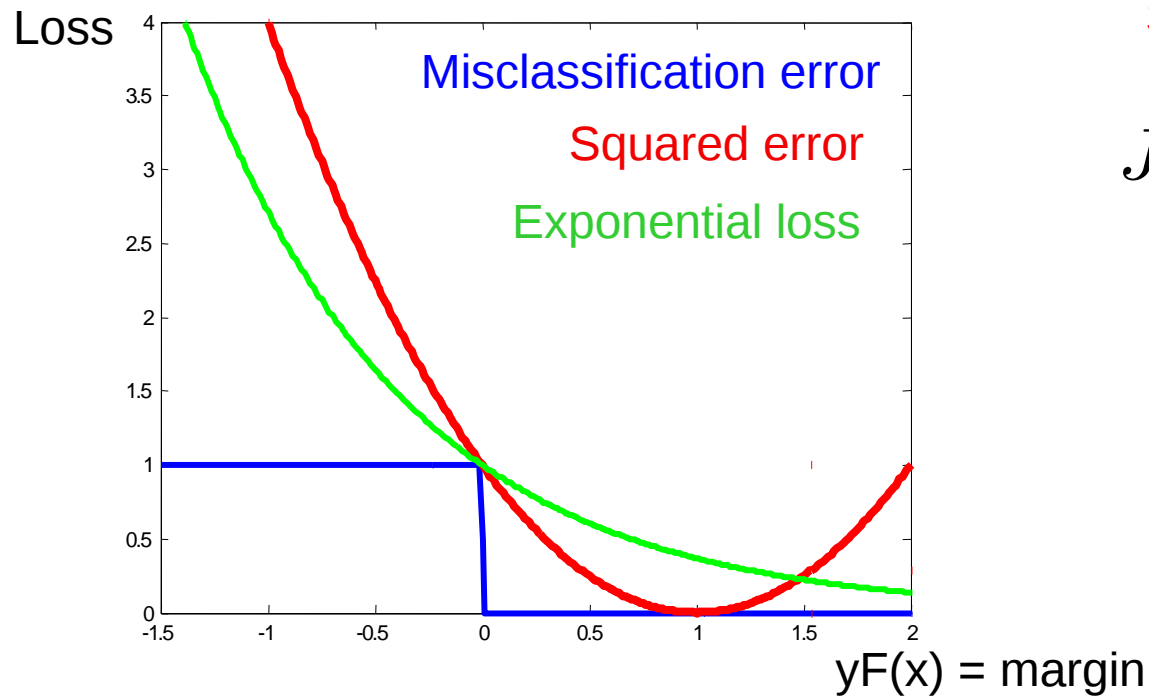
by minimizing the exponential loss

$$J(F) = \sum_{t=1}^N e^{-y_t F(x_t)}$$

↑ ↑
Training samples

The exponential loss is a differentiable upper bound to the misclassification error.

Exponential loss



Squared error

$$J = \sum_{t=1}^N [y_t - F(x_t)]^2$$

Exponential loss

$$J = \sum_{t=1}^N e^{-y_t F(x_t)}$$

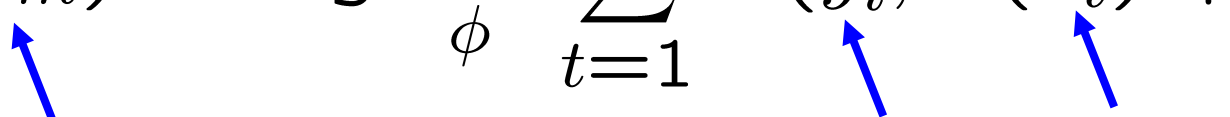
Boosting

Sequential procedure. At each step we add

$$F(x) \leftarrow F(x) + f_m(x)$$

to minimize the residual loss

$$(\phi_m) = \arg \min_{\phi} \sum_{t=1}^N J(y_i, F(x_t) + f(x_t; \phi))$$



Parameters
weak classifier

Desired output

input

gentleBoosting

- At each iteration:

We chose $f_m(x)$ that minimizes the cost:

$$J(F + f_m) = \sum_{t=1}^N e^{-y_t(F(x_t) + f_m(x_t))}$$

Instead of doing exact optimization, gentle Boosting minimizes a Taylor approximation of the error:

$$J(F) \propto \sum_{t=1}^N \boxed{e^{-y_t F(x_t)}} (y_t - f_m(x_t))^2 \rightarrow$$

Weights at this iteration

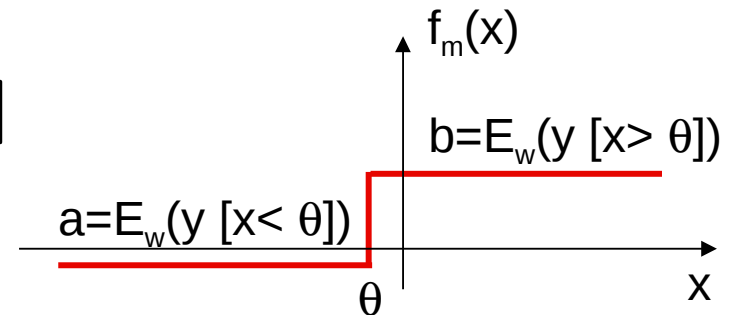
At each iterations we just need to solve a weighted least squares problem

Weak classifiers

- The input is a set of weighted training samples (x, y, w)
- Regression stumps: simple but commonly used in object detection.

$$f_m(x) = a[x_k < \theta] + b[x_k \geq \theta]$$

Four parameters: $[a, b, \theta, k]$



gentleBoosting.m

```
function classifier = gentleBoost(x, y, Nrounds)
```

```
...
```

```
for m = 1:Nrounds
```

```
    fm = selectBestWeakClassifier(x, y, w);
```

```
    w = w .* exp(- y .* fm);
```

```
    % store parameters of fm in classifier
```

```
    ...
```

```
end
```

Initialize weights $w = 1$

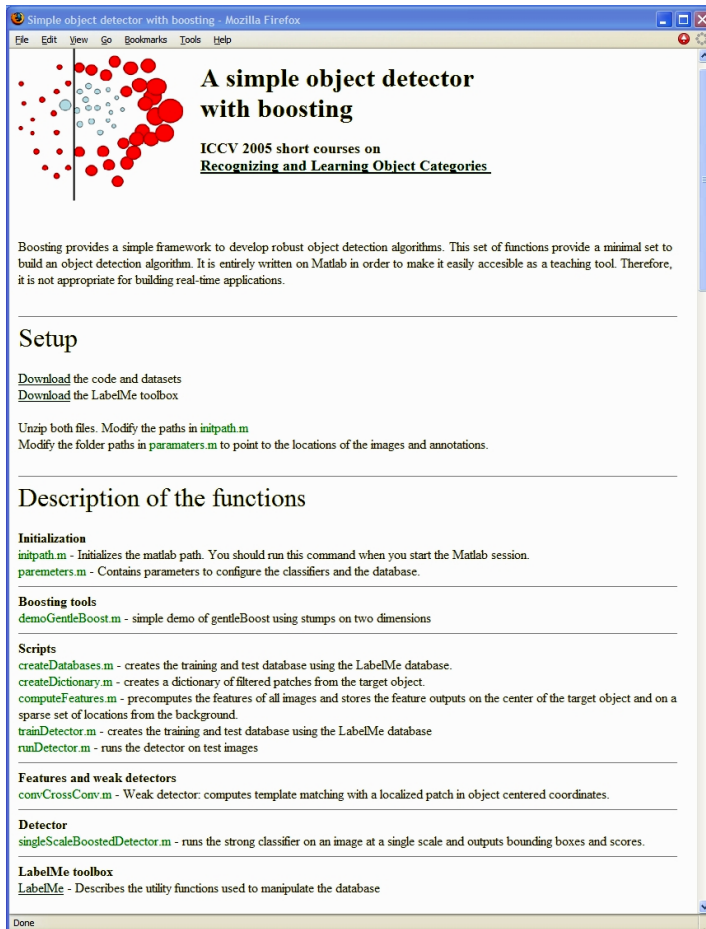
Solve weighted least-squares

Re-weight training samples

Demo gentleBoosting

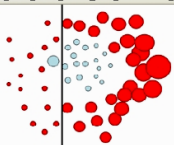
Demo using Gentle boost and stumps with hand selected 2D data:

> demoGentleBoost.m



Simple object detector with boosting - Mozilla Firefox

File Edit View Go Bookmarks Tools Help



A simple object detector with boosting

ICCV 2005 short courses on
Recognizing and Learning Object Categories

Boosting provides a simple framework to develop robust object detection algorithms. This set of functions provide a minimal set to build an object detection algorithm. It is entirely written on Matlab in order to make it easily accessible as a teaching tool. Therefore, it is not appropriate for building real-time applications.

Setup

[Download](#) the code and datasets
[Download](#) the LabelMe toolbox

Unzip both files. Modify the paths in [initpath.m](#)
Modify the folder paths in [parameters.m](#) to point to the locations of the images and annotations.

Description of the functions

Initialization

[initpath.m](#) - Initializes the matlab path. You should run this command when you start the Matlab session.
[parameters.m](#) - Contains parameters to configure the classifiers and the database.

Boosting tools

[demoGentleBoost.m](#) - simple demo of gentleBoost using stumps on two dimensions

Scripts

[createDatabases.m](#) - creates the training and test database using the LabelMe database.
[createDictionary.m](#) - creates a dictionary of filtered patches from the target object.
[computeFeatures.m](#) - precomputes the features of all images and stores the feature outputs on the center of the target object and on a sparse set of locations from the background.
[trainDetector.m](#) - creates the training and test database using the LabelMe database
[runDetector.m](#) - runs the detector on test images

Features and weak detectors

[convCrossConv.m](#) - Weak detector: computes template matching with a localized patch in object centered coordinates.

Detector

[singleScaleBoostedDetector.m](#) - runs the strong classifier on an image at a single scale and outputs bounding boxes and scores.

LabelMe toolbox

[LabelMe](#) - Describes the utility functions used to manipulate the database

Done

Flavors of boosting

- AdaBoost (Freund and Shapire, 1995)
- Real AdaBoost (Friedman et al, 1998)
- LogitBoost (Friedman et al, 1998)
- Gentle AdaBoost (Friedman et al, 1998)
- BrownBoosting (Freund, 2000)
- FloatBoost (Li et al, 2002)
- ...