Artificial Intelligence Fundamentals

Learning: Boosting

- Binary classification – classify the elements of a given set into two groups on a given rule

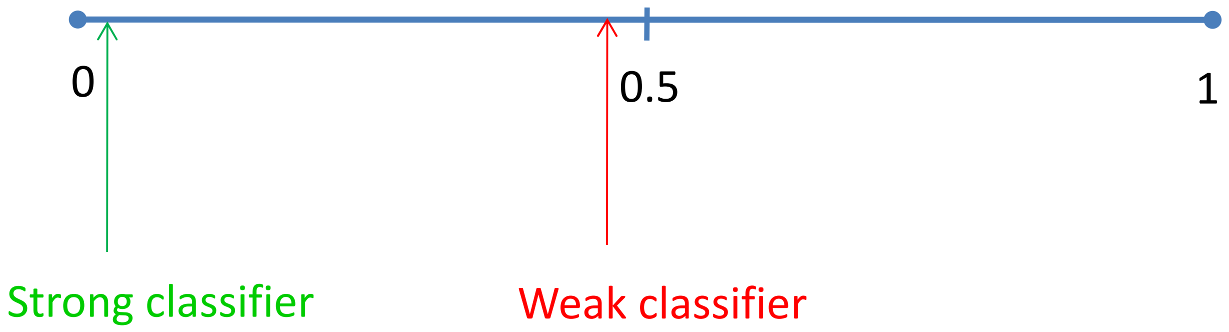
- Finding the classification rule can be a difficult task

- A crowd can be smarter than the participant in the crowd?

Classifiers strong/weak

- Suppose we have a set of classifiers
  h which give as the output {-1 , +1}

- Error rate

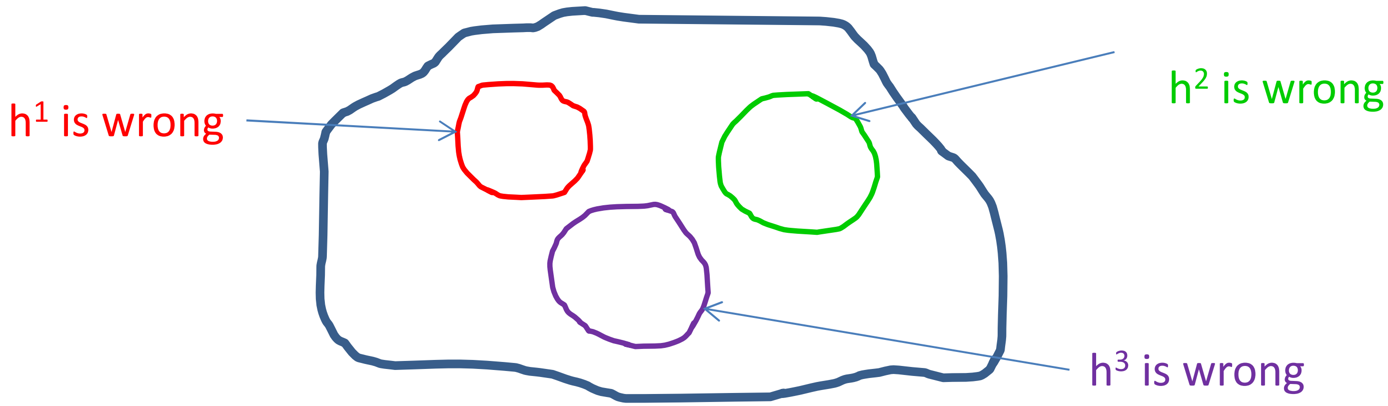


- Can we make a strong classifier by combining several of these weak classifiers and let them vote?

$$H(x) = sign\left(h^1(x) + h^2(x) + \ldots + h^n(x)\right) \quad x \text{ - is a sample}$$

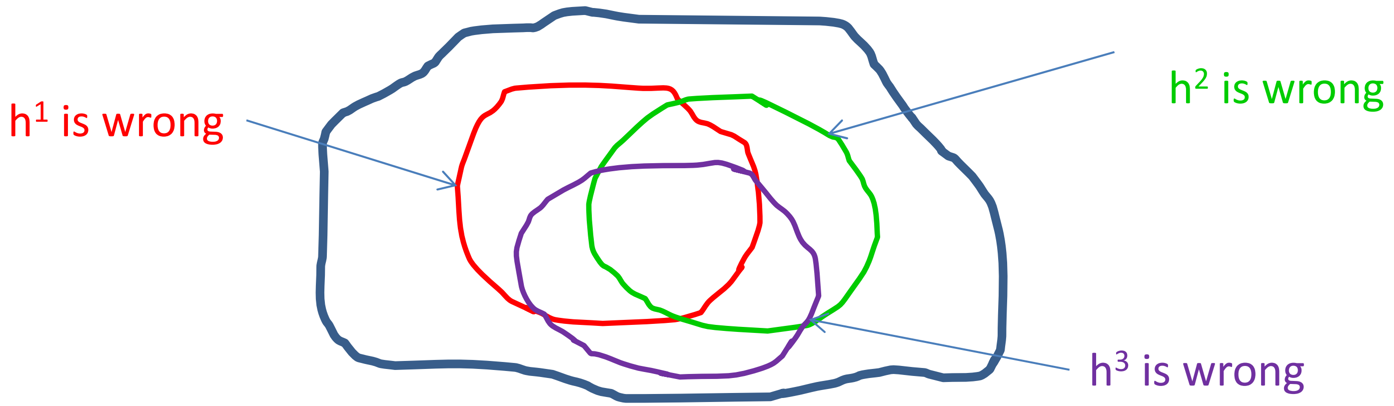
The perfect classifiers

$$H(x) = sign\left(h^1(x) + h^2(x) + h^3(x)\right)$$



- If it’s look like this, always we will have 0 error

A real situation

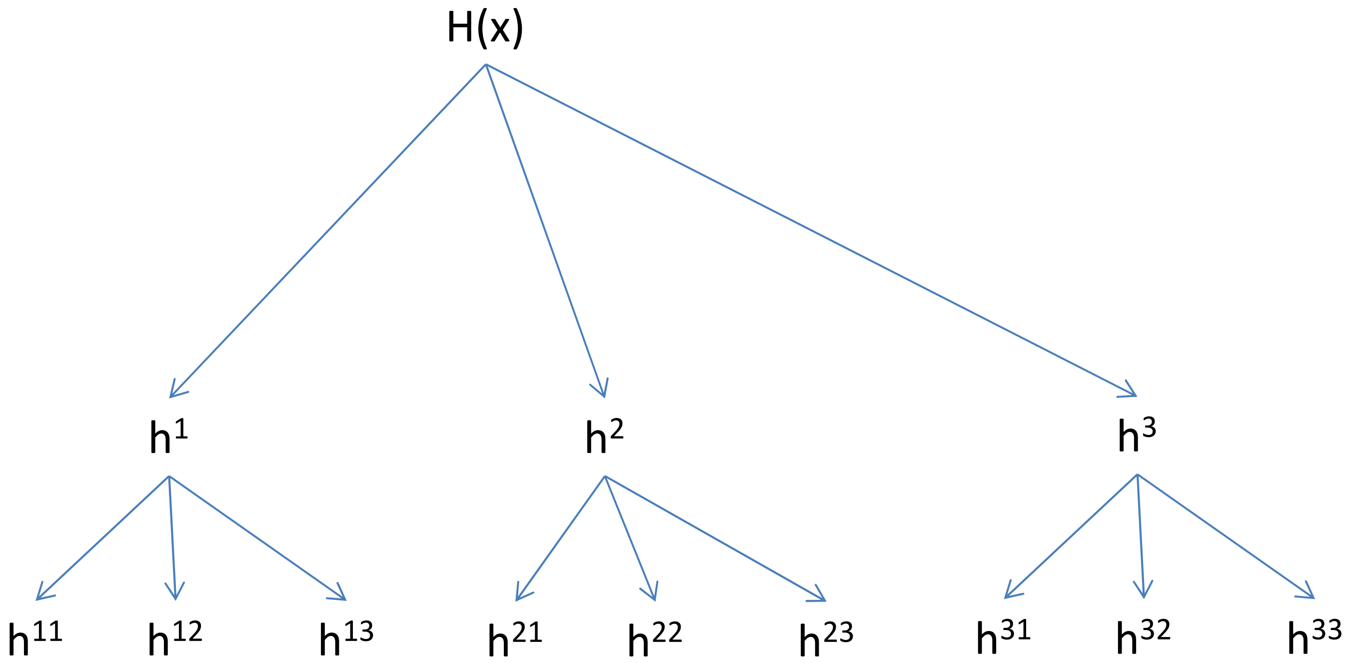


- Is the area overlapping by at least 2 classifiers sufficiently smaller than the area covers by each individual tests for wrong cases?

Idea #1

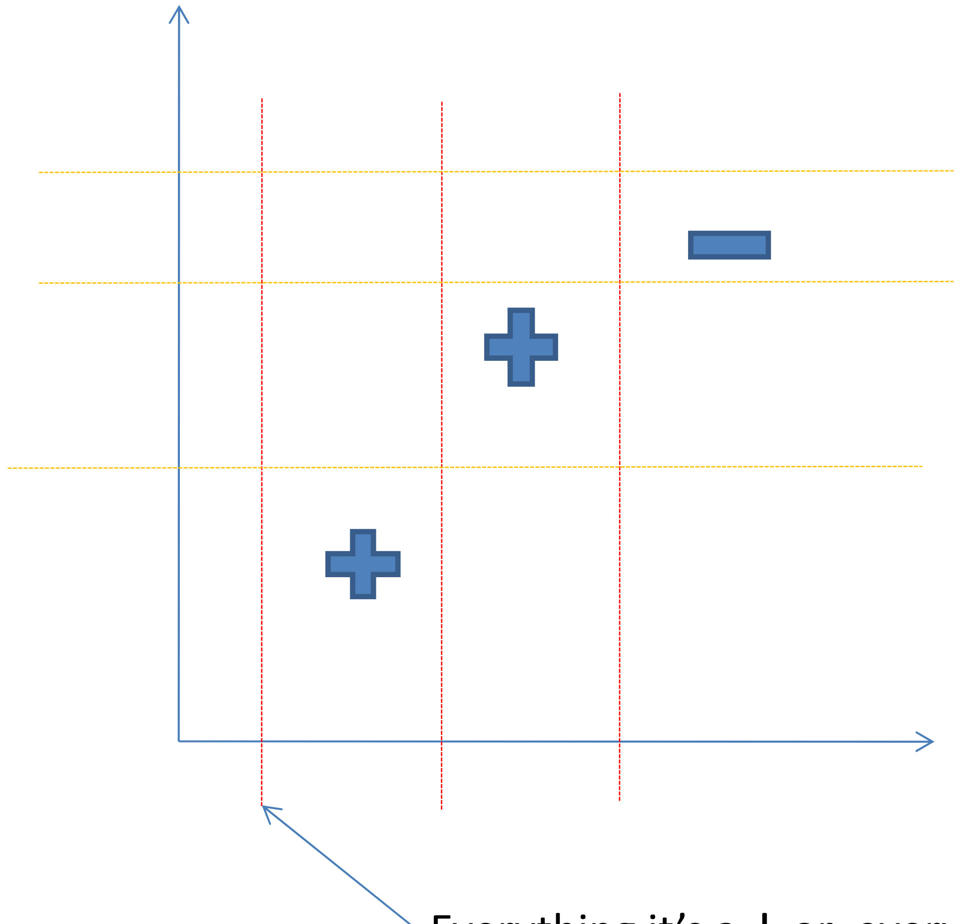
- We use undisturbed DATA to produce h^1
- We use DATA with an exaggeration of h^1 errors (disturbed set of data) to produce h^2
- We use DATA with an exaggeration of data where h^1 give a different answer than h^2 to produce h^3

Idea #2



- Get out the vote

Idea #3 – Example of classifiers



For each horizontal stump we could have 2
cases:

Up + , Down –

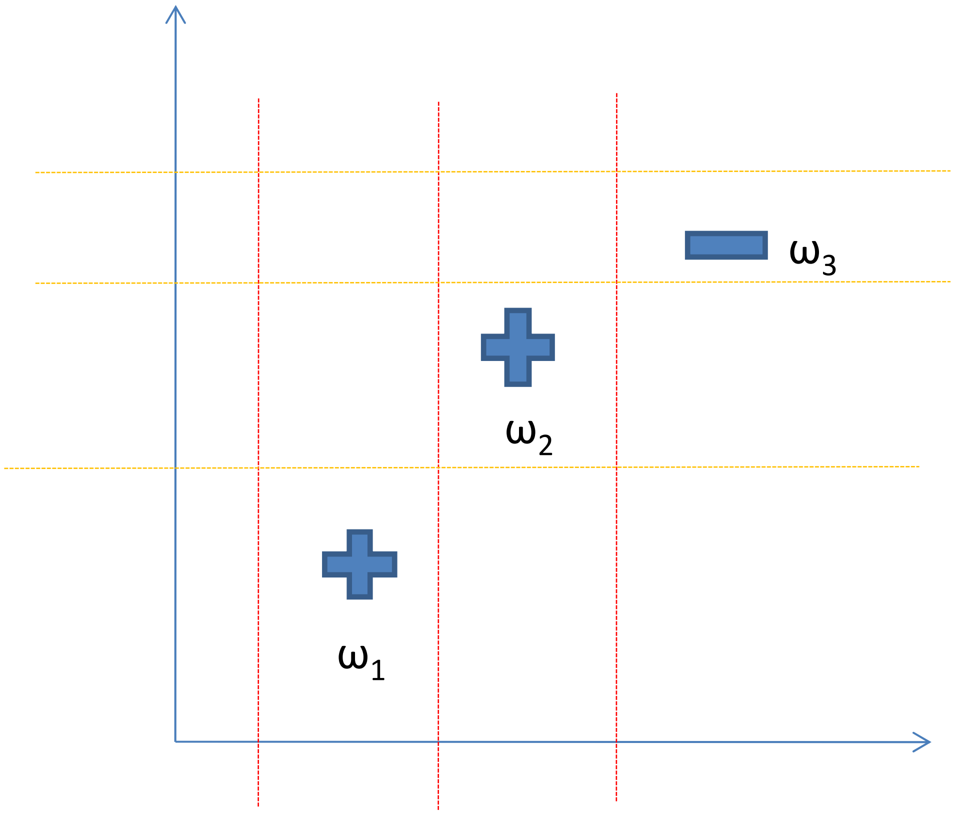
or

Up - , Down +

Similar for vertical stumps:
Left – Right.

Everything it's a + or everything it's a - . It's an extra test.

- Decision tree stumps – a single test
- Could be 12 decision tree stumps: for each dimension we have # of lines * 2 (we have 2 dimensions: 2*3*2=12)



$$Error = \sum_{\substack{WRONG \\ CASES}} \frac{1}{N}$$

$N - \#$ of cases

In the beginning:

$$\omega_i^1 = \frac{1}{N}$$

- Weighted the samples

$$Error^t = \sum_{\substack{i-WRONG \\ CASES}} \omega_i^t$$

- Enforced a distribution

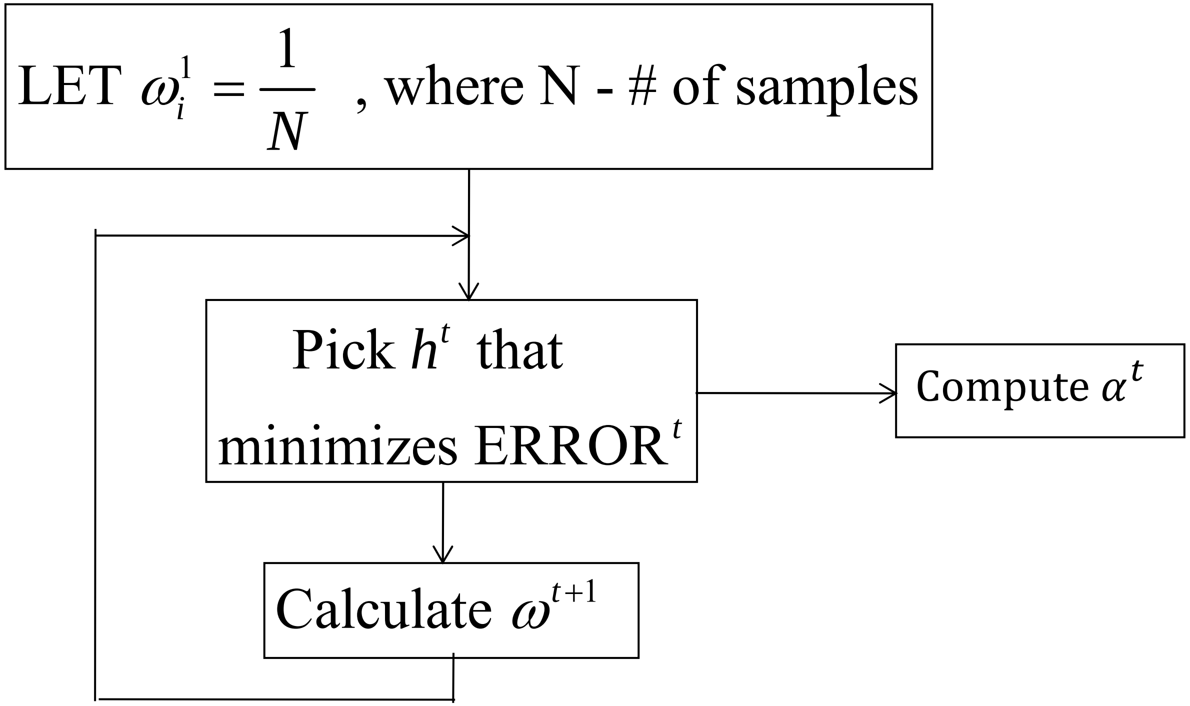
$$\sum_{\substack{ALL \\ CASES}} \omega_i^t = 1$$

Idea #4

$$H(x) = sign\left(\alpha^1 h^1(x) + \alpha^2 h^2(x) + \alpha^3 h^3(x) + \ldots.\right)$$

- Build a classifier in multiple steps
- We don’t treat equally each one on the crowd -> wisdom of weighted crowd of experts

Idea #5


LET $\omega_i^1 = \frac{1}{N}$, where N - # of samples
Pick h^t that minimizes ERROR^t
Compute α^t
Calculate ω^{t+1}

Idea #6

- Suppose that: $$\omega_i^{t+1} = \frac{\omega_i^t}{Z} e^{-\alpha^t h^t(x) y(x)}$$

$$h(x) = \begin{cases} +1 & \text{for samples the classifier thinks belongs to the class} \\ -1 & \text{for samples that the classifier thinks do not belong to the class} \end{cases}$$

$y(x) \in \{+1, -1\}$ - the desired output

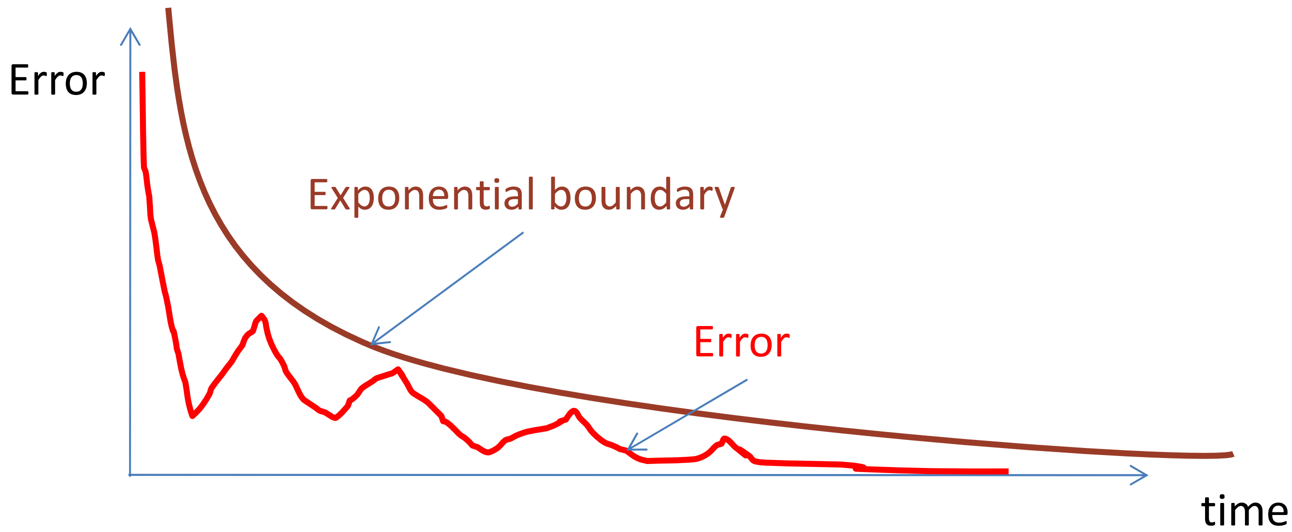
Z - the normalizer, in order to have a distribution

Minimize the error

- The error BOUND is minimized for the whole #4 if:

$$\alpha^t = \frac{1}{2}\ln\frac{1-E^t}{E^t}, \text{ where } E \text{ is the ERROR at time } t$$

- The error will be bounded by an exponential decay function
- It's guaranteed to converge on 0



Ada Boost

- You use uniform weights to start.
- For each step, you find the classifier that yields the lowest error rate for the current weights, w_i^t
- You use that best classifier, $h^t(x_i)$, to compute the error rate associated with the step, E^t
- You determine the alpha for the step, α^t from the error for the step, E^t .
- With the alpha in hand, you compute the weights for the next step, w_i^{t+1}, from the weights for the current step, w_i^t , taking care to include a normalizing factor, Z^t , so that the new weights add up to 1.
- You stop successfully when $H(x_i)$ correctly classifies all the samples, x_i; you stop unsuccessfully if you reach a point where there is no weak classifier, one with an error rate $< 1/2$.

Change the weights

$$\omega_i^{t+1} = \frac{\omega_i^t}{Z}\begin{cases}\sqrt{\dfrac{E^t}{1-E^T}} & \text{if it's correct}\\[2ex] \sqrt{\dfrac{1-E^t}{E^T}} & \text{if it's wrong}\end{cases}$$

$$Z = \sqrt{\frac{E^t}{1-E^t}}\sum_{CORRECT}\omega_i^t + \sqrt{\frac{1-E^t}{E^t}}\sum_{WRONG}\omega_i^t$$

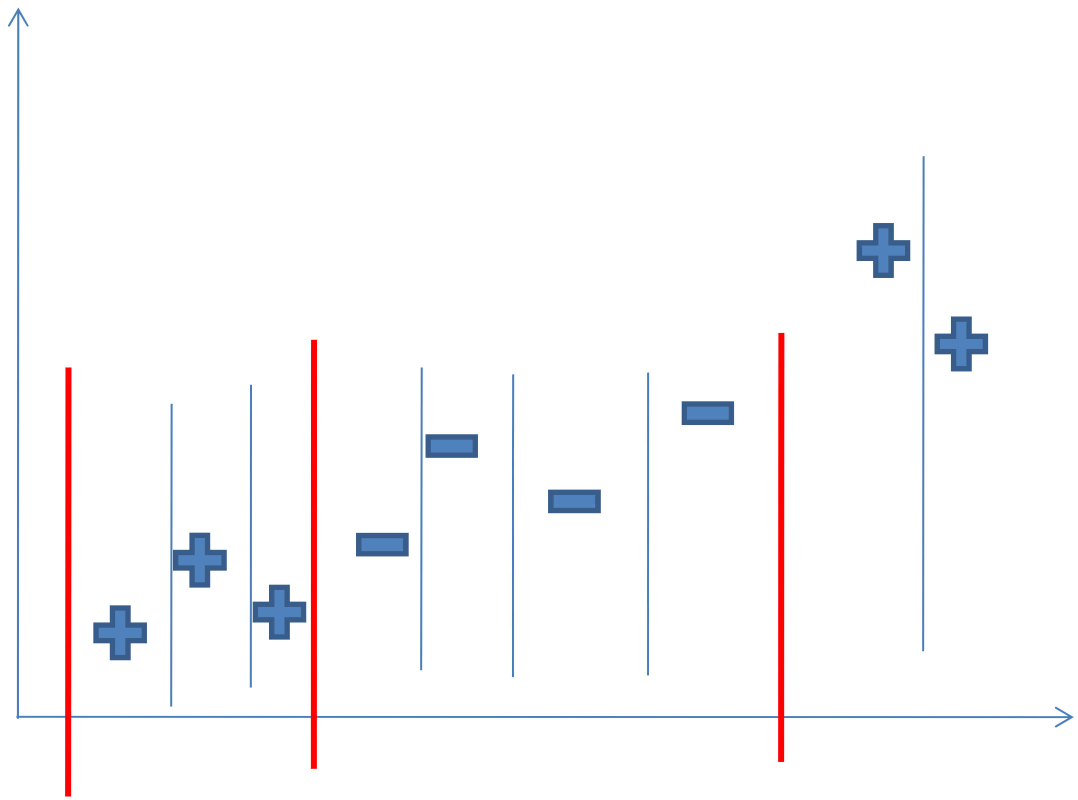
$$= \sqrt{\frac{E^t}{1-E}}(1-E^t) + \sqrt{\frac{1-E^t}{E^t}}E^t = 2\sqrt{E^t(1-E^t)}$$

$$\omega_i^{t+1} = \frac{\omega_i^t}{2}\begin{cases}\dfrac{1}{1-E^t} & \text{if it's correct}\\[2ex] \dfrac{1}{E^t} & \text{if it's wrong}\end{cases}$$

$$\sum_{CORRECT}\omega_i^{t+1} = \frac{1}{2}\frac{1}{1-E^t}\sum_{CORRECT}\omega^t = \frac{1}{2}$$

$$\sum_{WRONG}\omega_i^{t+1} = \frac{1}{2}$$

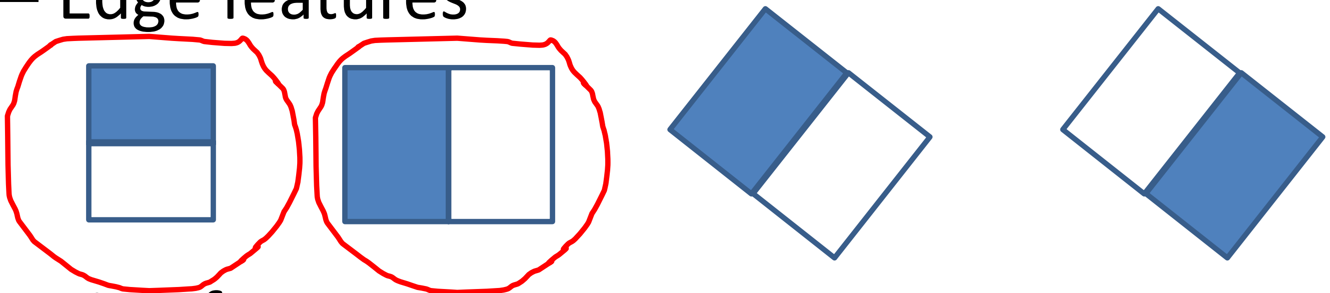
Improvements



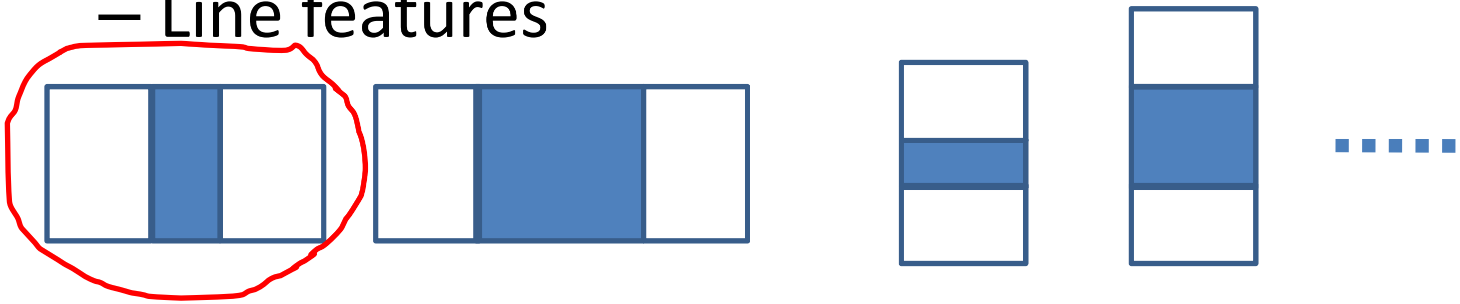
- Tests that really matter
- Immune to overfitting

Face detection

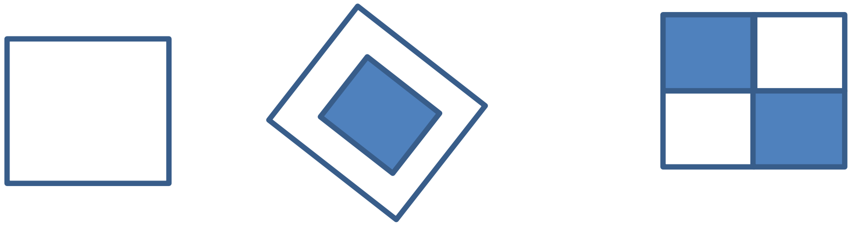
- Haar-like features
  - Edge features



  - Line features

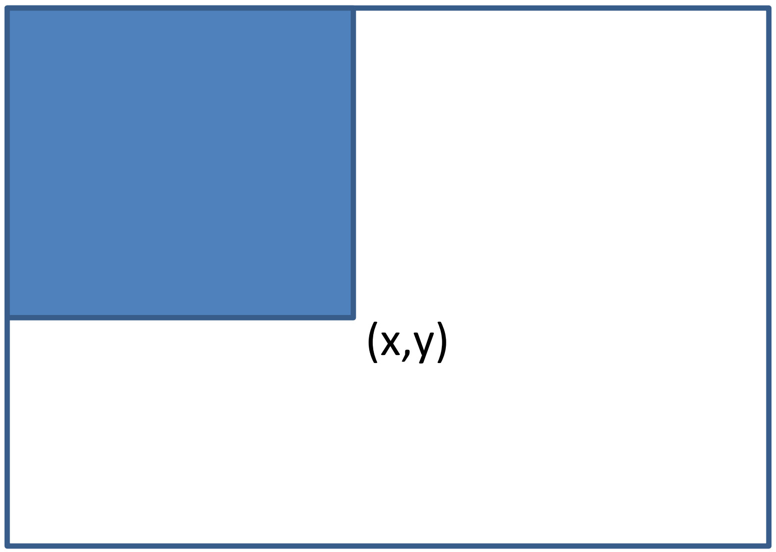


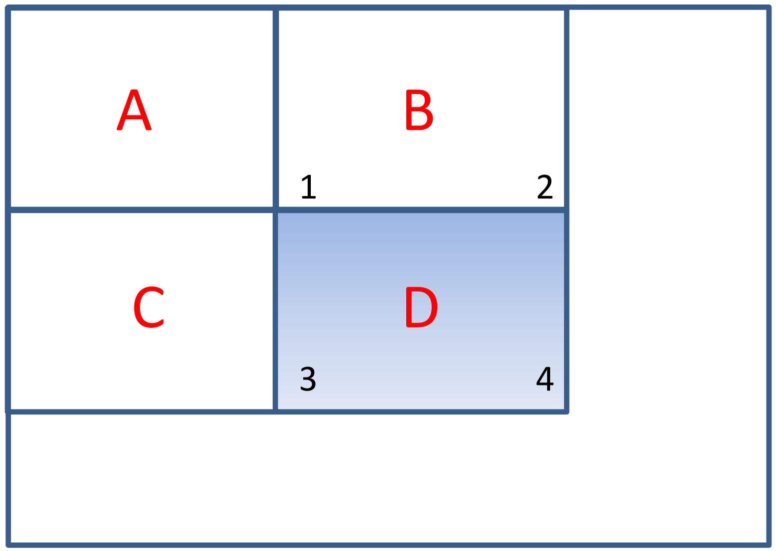
  - Other features



Face detection

- Integral image – each pixel (x,y) is the sum of all pixels above and left of *x,y* applied to original image





$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y'),$$

where $ii(x,y)$ – integral image

$i(x,y)$ – original image

$$v1 = ii(location\_1) = \sum_A i$$

$$v2 = ii(location\_2) = \sum_A i + \sum_B i$$

$$v3 = ii(location\_3) = \sum_A i + \sum_C i$$

$$v4 = ii(location\_4) = \sum_A i + \sum_B i + \sum_C i + \sum_D i$$

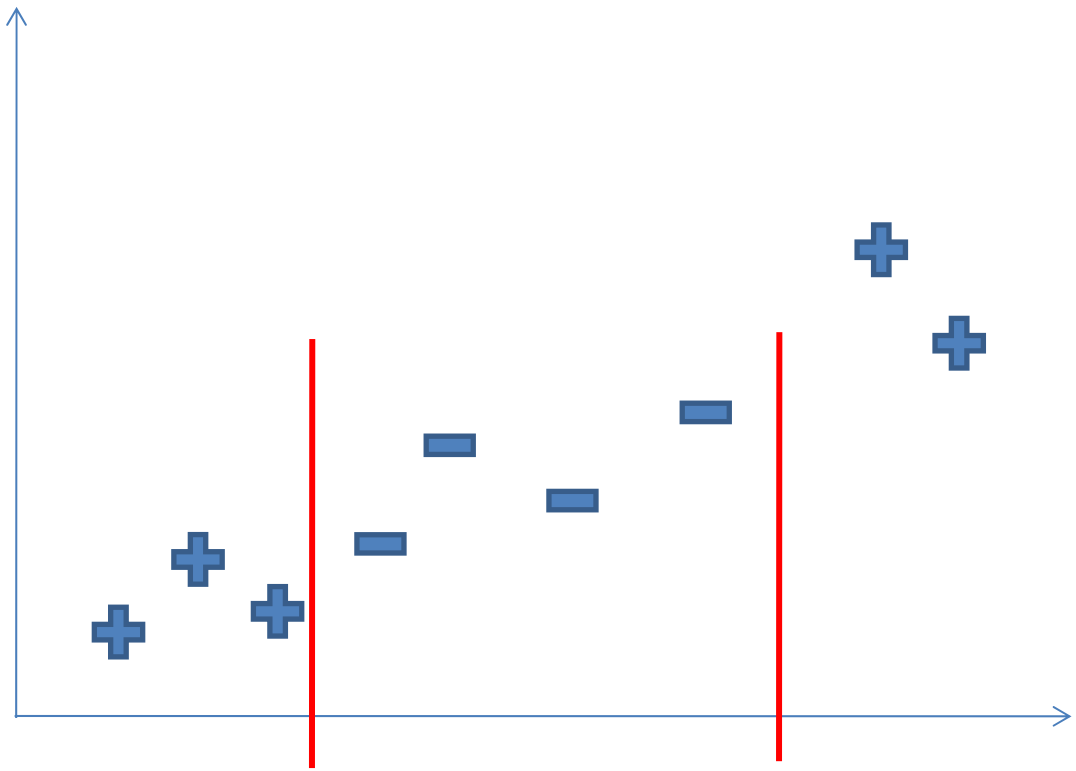
$$rect(D) = v1 + v4 - v2 - v3$$

Face detection

- For 24x24 pixels image – 162.336 features

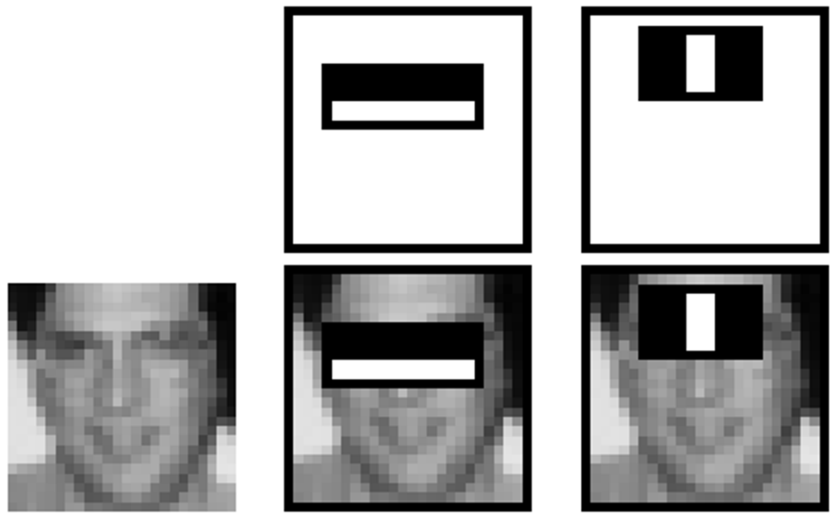
Face detection

- Choosing the threshold for each classifier



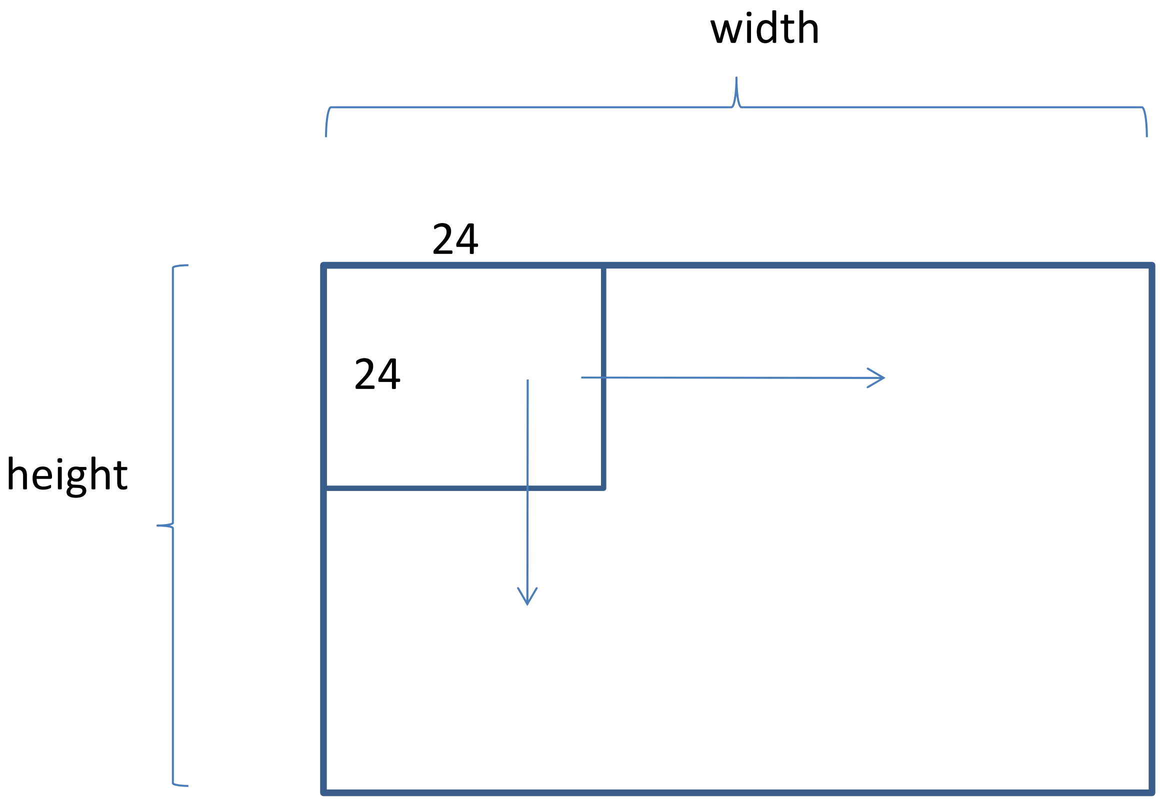
Face detection

- The first and second features selected by AdaBoost



Face detection

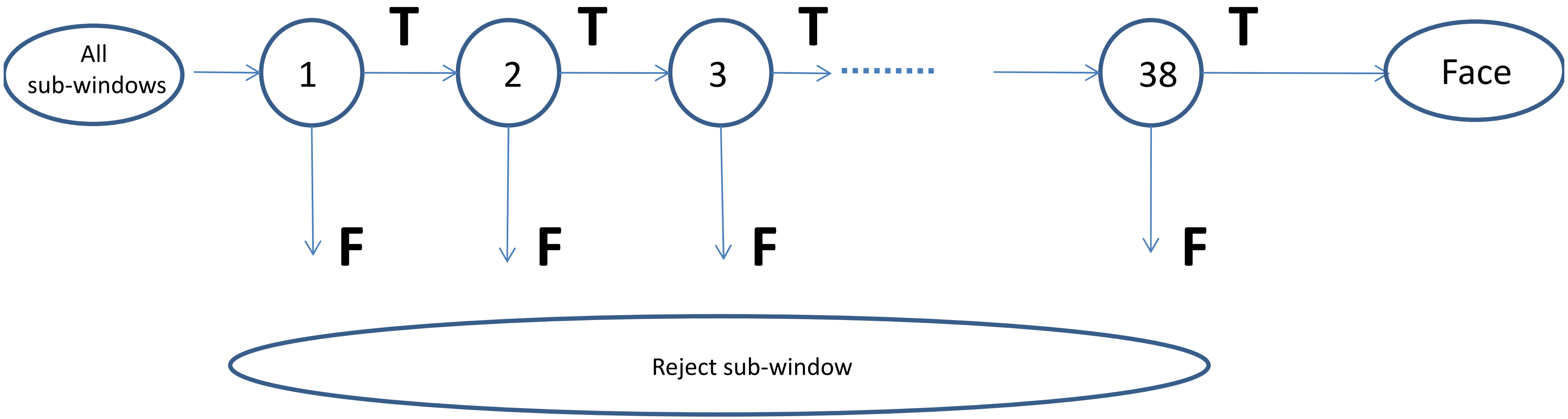
- Sub-window – 24x24 pixels



Starts from top-left corner and go to left 1 pixel at a time. When reach the end of the row, go down 1 pixel and start again from the left.

Face detection

- Cascade of classifier
  - #of features in the first 5 layers: 1, 10, 25, 25 and 50
  - total # of features in all layers - 6061



Related resources

- P. Viola, M. Jones, “Robust Real-Time Face Detection”, http://www.vision.caltech.edu/html-files/EE148-2005-Spring/pprs/viola04ijcv.pdf