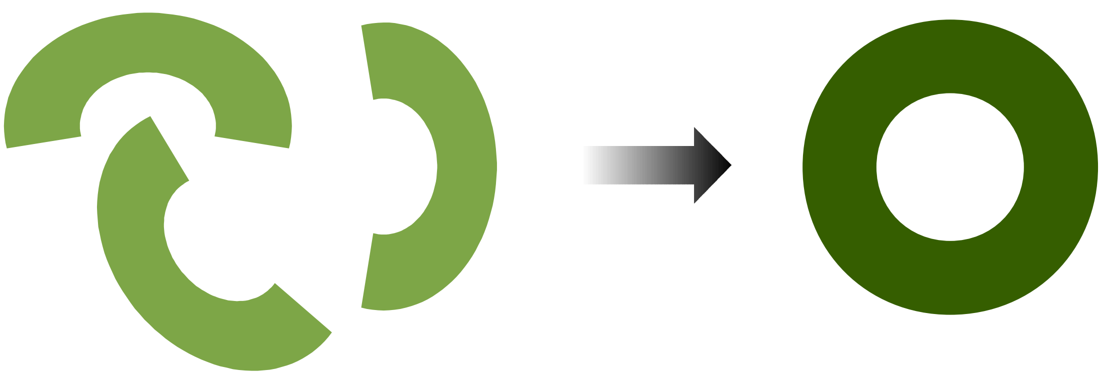
Boosting & AdaBoost

Luis De Alba
ldealbar@cc.hut.fi
5.3.2008

Boosting

- Definition:
  - Method of producing a very accurate prediction rule by combining inaccurate rules of thumb.
- Objective:
  - Improve the accuracy of any given learning algorithm (LMS, SVM, Perceptron, etc.)

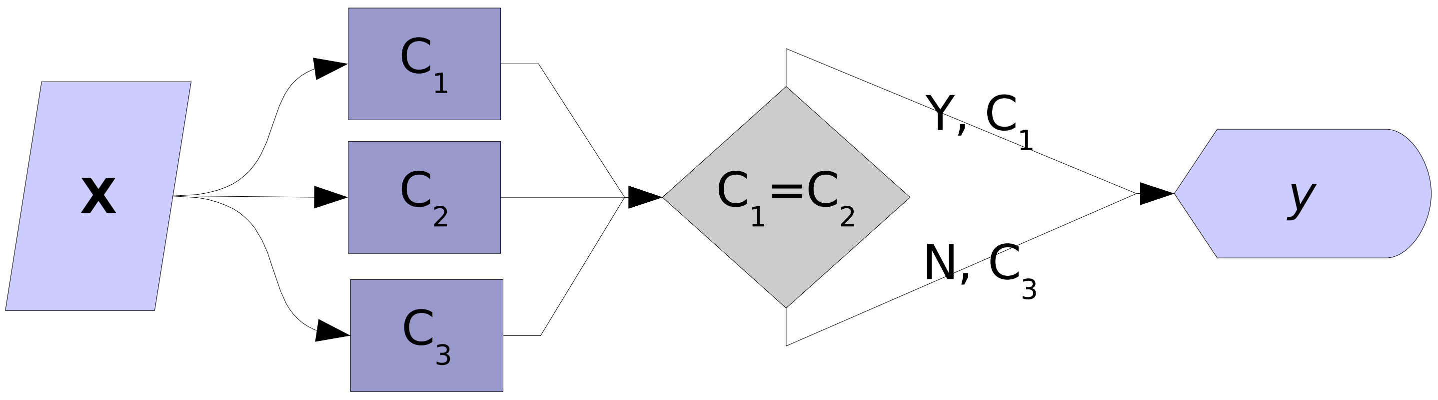
- How to boost in 2 steps:
  - Select a Classifier with an accuracy greater than chance (random)
  - Form an ensemble of the selected Classifiers whose joint decision rule has high accuracy.

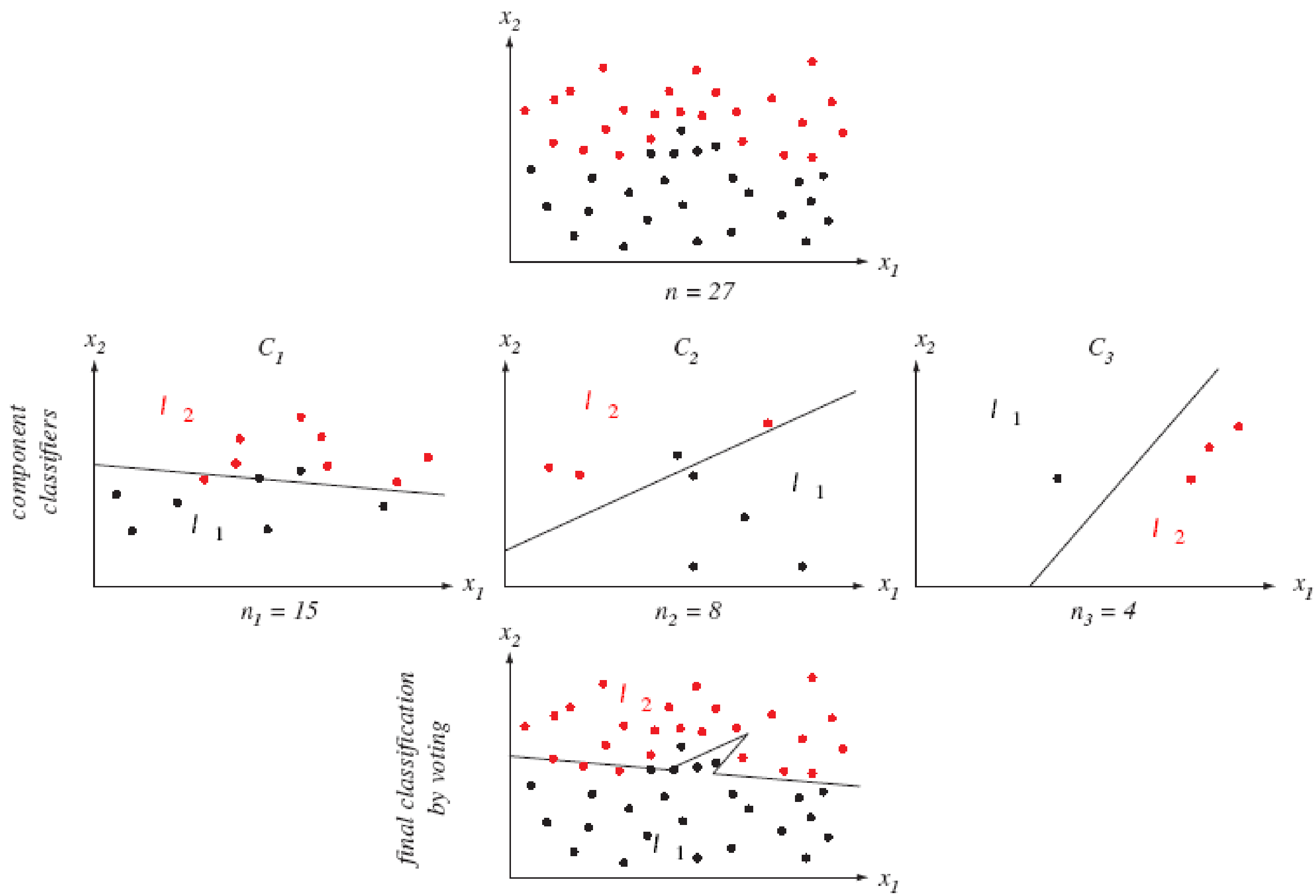


- Example 1: Golf Tournament
  - Tiger Woods VS
  - 18 average+ players

- Example 2: Classification Problem
  - 2 dimensions, 2 classes, 3 Classifiers
  - D training sets $\{\mathbf{x}_1, y_1 \ldots \mathbf{x}_m, y_m\}$
- a) Train C_1 with D_1 where D_1 is randomly selected.
- b) Train C_2 with D_2 where only 50% of the selected set is correctly classified by C_1

- c) Train C_3 with D_3 where the classification, of selected D_3, disagrees between C_1 and C_2
- Classification task:





x_2
x_1
$n = 27$
component classifiers
C_1
C_2
C_3
$n_1 = 15$
$n_2 = 8$
$n_3 = 4$
final classification by voting

- Duda, et al.

AdaBoost

- Adaptive Boosting aka“AdaBoost”
- Introduced by Yoav Freud and Robert Schapire in 1995. Both from AT&T Labs.
- Advantages:
  - As many weak learners as needed (wished)
  - “Focuses in” on the informative or “difficult” patterns

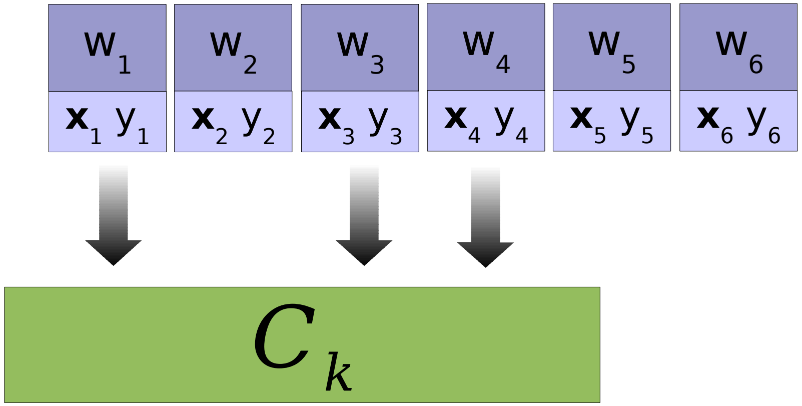
- Algorithm Inputs:
  - Training set: $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)$
    - $Y = \{-1, +1\}$ (two classes)
  - Weak or Base algorithm C
    - LMS, Perceptron, etc.
  - Number or rounds (calls) to the weak algorithm $k = 1, \ldots, K$

- Each training tuple receives a weight that determines its probability of being selected.
- At initialization all weights across training set are uniform:

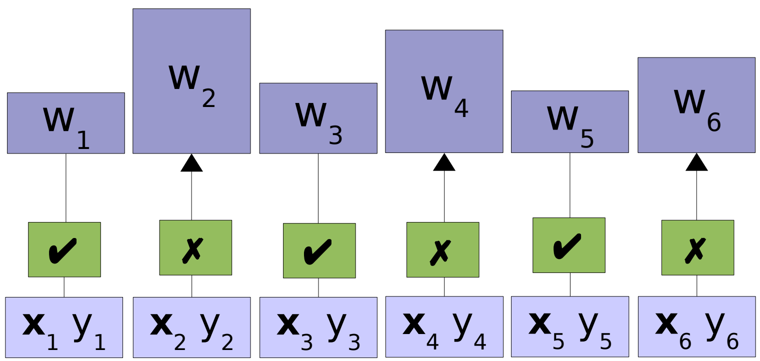
w_1	w_2	w_3	w_m
$\mathbf{x}_1$ y_1	$\mathbf{x}_2$ y_2	$\mathbf{x}_3$ y_3	$\mathbf{x}_m$ y_m

$$W_1(i) = 1/m$$

- For each iteration k a random set is selected from the training set according to weights.
- Classifier C_k is trained with the selection



- **Increase** weights of training patterns **misclassified** by C_k
- **Decrease** weights of training patterns **correctly** classified by C_k



- Repeat for all K

- Algorithm:

initialize

$D = \{ \mathbf{x}_1, y_1, \ldots \mathbf{x}_m, y_m \}, K, W_1(i) = 1/m, i = 1, \ldots, m$

loop

For $k = 1$ to K

Train weak learner C_k using

D_k sampled according to $W_k(i)$

$E_k \leftarrow$ training error of C_k measured using D_k

$$\alpha_k \leftarrow \frac{1}{2} \ln[(1 - E_k)/E_k]$$

$$W_{k+1}(i) \leftarrow \frac{W_k(i) e^{(-\alpha_k y_i C_k(\mathbf{x}_i))}}{Z_k}$$

return

C_k and α_k for all K

- Output of the final classifier for any given input.

$$H(\boldsymbol{x}) = \text{sign}\left(\sum_{k=1}^{K} \alpha_k C_k(\boldsymbol{x}) \right)$$

Analysis

- Error calculation:

$$E_k = \frac{\sum_{i=1}^{m} if\,(C_k(\boldsymbol{x}_i) \neq y_i) : 1 : 0}{m}$$

  - Error is the summation and normalization of all wrongly classified sets by the weak learner.
  - Weak learner shall do better than if only random guesses i.e., $E_k < 0.5$

- α measurement:

$$\alpha_k \leftarrow \frac{1}{2}\ln\left[\frac{(1-E_k)}{E_k}\right]$$

  - It measures the importance assigned to C_k
  - There are two things to note:
    - α >=0 if E <= ½
    - α gets larger as E gets smaller

- Normalization factor:
  - Z_k is a normalization factor so that W_{k+1} will be a distribution.
  - Possible calculation:

$$Z_k = \sum_{i=1}^{m} W_{k+1}(i)$$

- Hypothesis:

$$H(\boldsymbol{x}) = \mathrm{sign}\left(\sum_{k=1}^{K} \alpha_k C_k(\boldsymbol{x})\right)$$

  - H is a **weighted** majority vote of the K weak hypothesis where α_t is the **weight** associated to C_k
  - Each instance of **x** outputs a prediction whose **sign** represents the class (-1, +1) and the magnitude $|C_k(\boldsymbol{x})|$ is the **confidence**.

Why better than random?

- Suppose three classifiers:
  - Worse than random (*E=0.75*)
    - Alpha=-0.549 CC=**1.731** IC=**0.577**
  - Random (*E=0.50*)
    - Alpha=0.0 CC=1.0 IC=1.0
  - Better than random (*E=0.25*)
    - Alpha=0.549 CC=**0.577** IC=**1.731**

CC=Correctly Classified Weight Factor

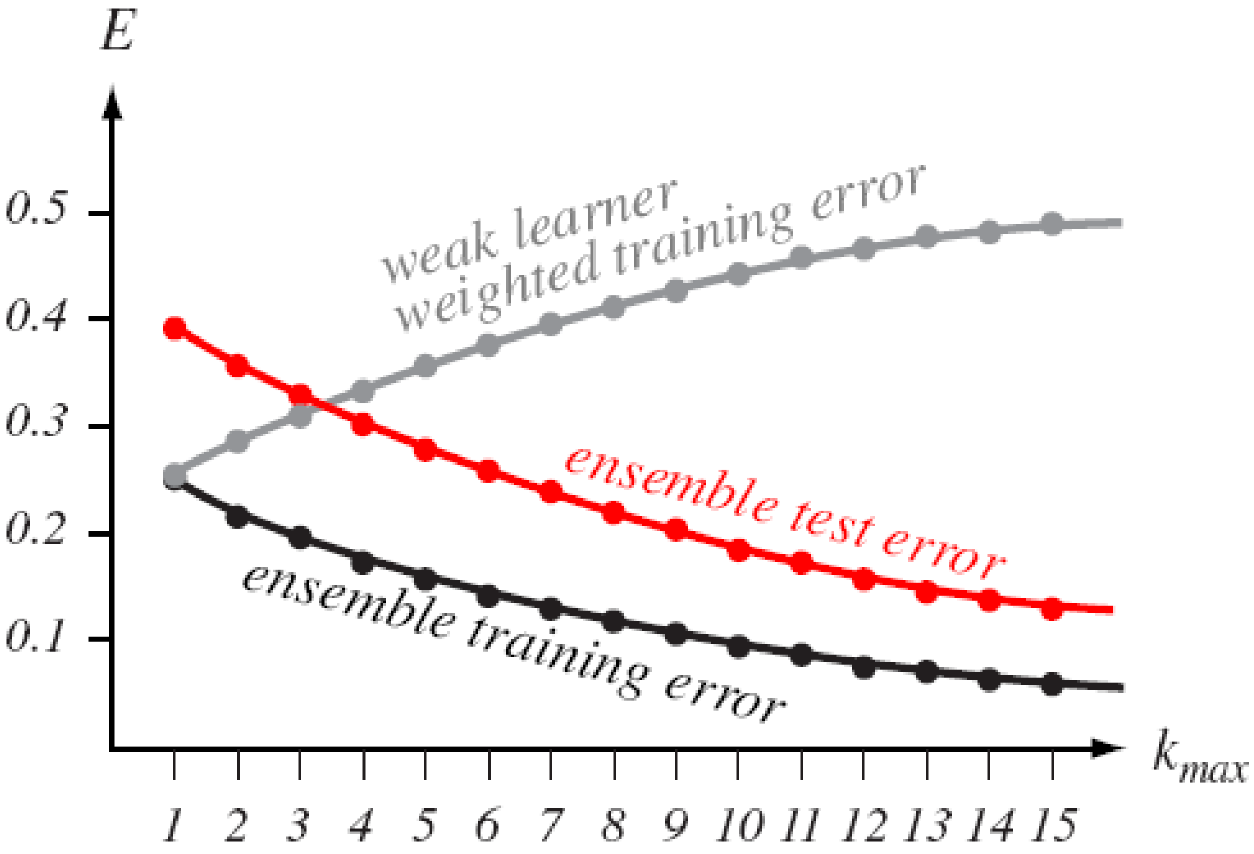
IC=Incorrectly Classified Weight Factor

Generalization Error

- Based on sample size *m*, the VC-dimension *d* and the boosting rounds *K*.

$$\tilde{O}\left(\sqrt{\frac{Kd}{m}}\right) \rightarrow \tilde{O}\left(\sqrt{\frac{d}{m\theta^2}}\right)$$

  - Overfit when *K* to large.
  - Experiments showed that AdaBoost do not overfit.
  - It continues improving after training error is zero.

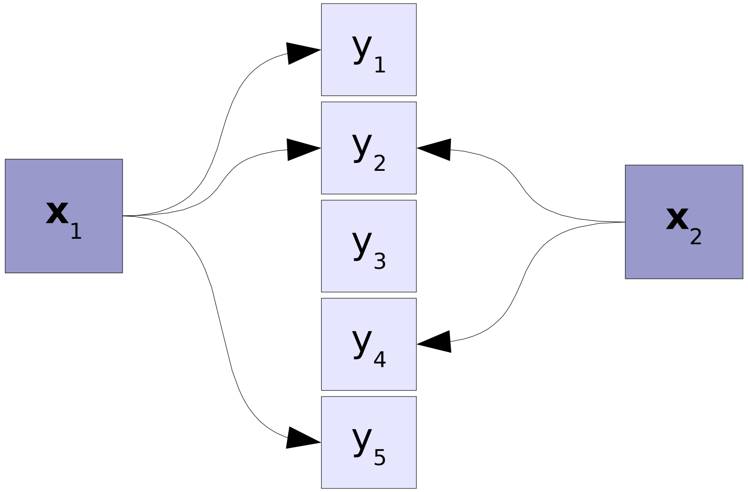

E
0.5
0.4
0.3
0.2
0.1
weak learner
weighted training error
ensemble test error
ensemble training error
k_{max}
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

- Duda, et al.

Multi-class, Multi-label

- In the multi-label case, each instance $\boldsymbol{x} \in X$ may belong to multiple labels in Y.

$$(\boldsymbol{x}, Y) \text{ where } Y \subseteq \mathcal{Y}$$



- Decompose the problem into *n* orthogonal **binary** classification problems.

For $Y \subseteq \mathcal{Y}$, define $Y[\lambda]$ for $\lambda \in \mathcal{Y}$ to be

$$Y[\lambda] = \begin{cases} +1 \text{ if } \lambda \in Y \\ -1 \text{ if } \lambda \notin Y \end{cases}$$

$H: X * Y \rightarrow \{-1, +1\}$ defined by $H(x, \lambda) = H(x)[\lambda]$

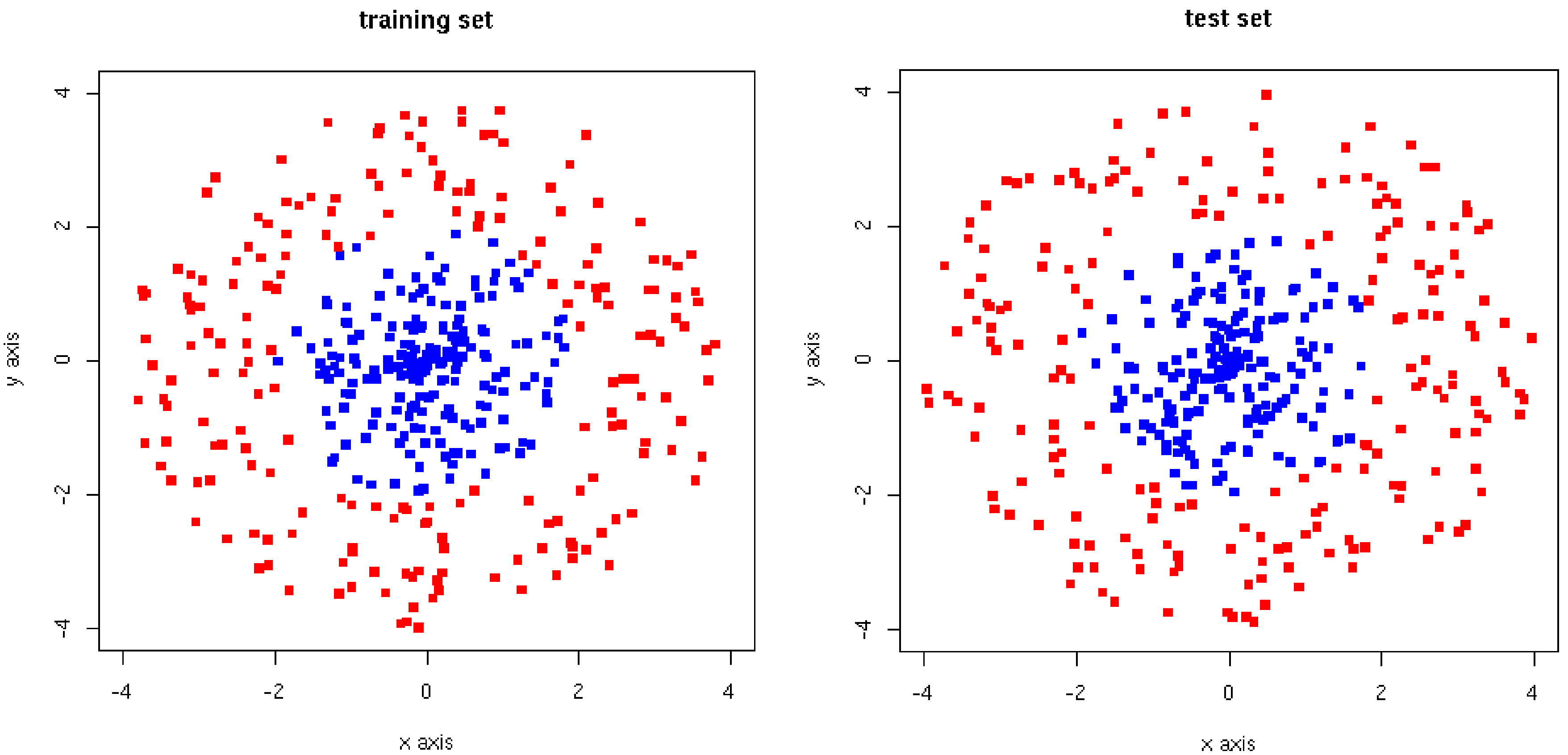
- Then, replace each training example by *k* examples.

$$\left(\boldsymbol{x}_i, Y_i\right) \rightarrow \left(\left(\boldsymbol{x}_i, \lambda\right), Y_i[\lambda]\right) \text{ for } \lambda \in \mathcal{Y}$$

- The algorithm is called AdaBoost.MH [3]

Homework hints

- Weak learning function is a simple Perceptron
- Do not forget how **Alpha** is computed.
- Do not forget what is the **Weight**'s vector and how it is computed.
- Do not be lazy !


training set
test set
y axis
x axis
4
2
0
-2
-4

Conclusion

- AdaBoost does not require prior knowledge of the weak learner.
- The performance is completely dependent on the learner and the training data.
- AdaBoost can identify outliers based on the weight's vector.
- It is susceptible to noise or to examples with very large number of outliers.

Bibliography

- [1] A Short Introduction to Boosting. Yoav Freud, Robert Schapire. 1999
- [2] Patter Classification. Richard Duda, et al. Wiley Interscience. 2nd Edition. 2000. [s. 9.5.2]
- [3] Improved boosting algorithms using confidence-rated predictions. Robert Schapire, Yoram Singer. 1998 [s. 6,7]