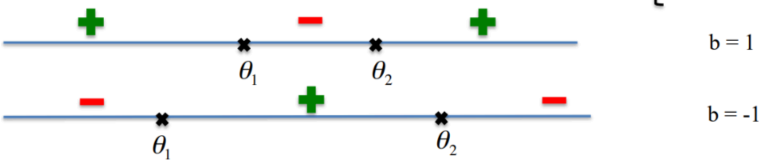
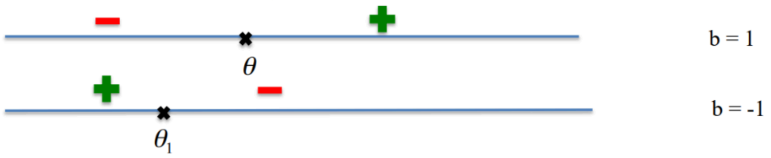
Weak learnability - example

Let $\mathcal{X}$ = R, $\mathcal{H}$ is the class of 3-piece classifiers (signed intervals):

$$H = \{h_{\theta_1,\theta_2,b} \mid \theta_1,\theta_2 \in R, \theta_1 < \theta_2, b \in \{-1,+1\}\} \quad h_{\theta_1,\theta_2,b}(x) = \begin{cases} +b, \text{ if } x < \theta_1 \text{ or } x > \theta_2 \\ -b, \text{ if } \theta_1 \le x \le \theta_2 \end{cases}$$



Consider $\mathcal{B}$ the class of Decision Stumps = class of 1-node decision trees
$\mathcal{B}$ = {$h_{\theta,b}$:**R**→{-1,1}, $h_{\theta,b}(x)$ = sign(x – θ) × b, θ ∈ **R**, b ∈ {-1,+1}}.



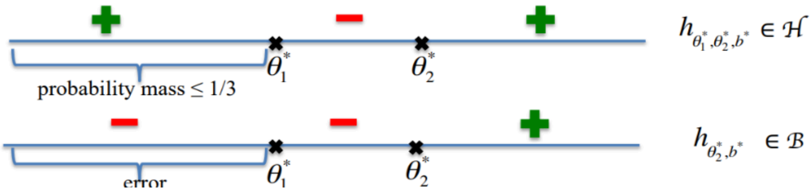
ERM$_{\mathcal{B}}$ is a γ-weak learner for $\mathcal{H}$, for γ = 1/12.

ERM$_{\mathcal{B}}$ is a γ-weak learner for $\mathcal{H}$, for γ = 1/12.

Proof: Consider a $h^* = h_{\theta_1^*,\theta_2^*,b^*} \in \mathcal{H}$ that labels a training set S.

Consider a distribution $\mathcal{D}$ over $\mathcal{X}$ = **R**. Then, we are sure that at least one of the regions $(-\infty,\theta_1^*)$, $[\theta_1^*,\theta_2^*]$, $(\theta_2^*,+\infty)$ has a probability mass wrt $\mathcal{D} \le 1/3$.

Consider, without loss of generality that $\mathcal{D}((-\infty,\theta_1^*)) = P_{x\sim\mathcal{D}}(x\in(-\infty,\ \theta_1^*)) \le 1/3$
Then the hypothesis $h_{\theta,b} \in$ B, where θ = θ_2^*, b = b* errors on $(-\infty,\ \theta_1^*)$.



$\mathcal{B}$ = {$h_{\theta,b}$:**R**→{-1,1}, $h_{\theta,b}(x)$ = sign(x – θ) × b, θ ∈ **R**, b ∈ {-1,+1}}.

It is easy to show that VCdim($\mathcal{B}$) = VCdim(class of signed thresholds) = 2.

The fundamental theorem of learning states that if the hypothesis class $\mathcal{B}$ has VCdim($\mathcal{B}$) = d, then the sample complexity of agnostic PAC learning $\mathcal{B}$ satisfies:

$$C_1 \frac{d + \log(1/\delta)}{\epsilon^2} \le m_{\mathcal{H}}(\epsilon,\delta) \le C_2 \frac{d + \log(1/\delta)}{\epsilon^2}$$

So, if the sample size is greater than the sample complexity, then with probability of at least 1 – δ, the ERM$_{\mathcal{B}}$ rule learns in the agnostic case a hypothesis such that:

$$L_{\mathcal{D}}(\mathrm{ERM_B(S)}) \le \min_{h\in\mathcal{H}} L_{\mathcal{D}}(h) + \epsilon \ = 1/3 + \varepsilon$$

Take ε = 1/12 than we obtain 1/3 + 1/12 = 5/12 ≤ 1 – 1/12 . Than ERM$_{\mathcal{B}}$ is a γ-weak learner for $\mathcal{H}$, where γ = 1/12.

Efficient implementation of ERM for Decision Stumps

In practice, we use the following base hypothesis class of decision stumps over $\mathbf{R}^d$ for weak learners:

$\mathcal{H}_{DS}{}^{d} = \{h_{i,\theta,b}: \mathbf{R}^d \to \{-1,1\},\ h_{i,\theta,b}(\mathbf{x}) = \mathrm{sign}(\theta - x_i) \times b,\ 1 \le i \le d,\ \theta \in \mathbf{R},\ b \in \{-1,+1\}\}$
-pick a coordinate i (from 1 to d), project the input $x = (x_1, x_2, \ldots, x_d)$ on the i-th coordinate and obtain x_i, if $x_i \le$ threshold θ label the example with b, else with –b

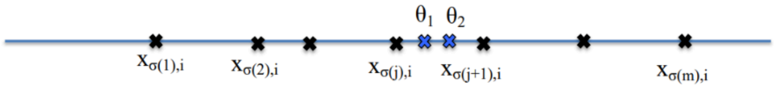
How to implement efficient ERM rule for the class $\mathcal{H}_{DS}{}^{d}$?

Let $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$ be a training set of size *m*. We want to find the best h_{i^*,θ^*,b^*} which minimizes the training error on S:

$$h_{i^*,\theta^*,b^*} = \underset{h_{i,\theta,b} \in H_{DS}^{d}}{\mathrm{argmin}}\, L_S(h_{i,\theta,b}) = \underset{\substack{1 \le i \le d \\ \theta \in R \\ b \in \{-1,+1\}}}{\mathrm{argmin}}\, L_S(h_{i,\theta,b})$$

We have $1 \le i \le d$, $\theta \in \mathbf{R}$, $b \in \{-1,+1\}$. We fix $i \in \{1, 2, \ldots, d\}$ and $b \in \{-1,+1\}$. Then we are interested in minimizing the error on $S_i = ((x_{1,i}, y_1), \ldots, (x_{m,i}, y_m))$. By sorting $x_{1,i}, x_{2,i}, \ldots, x_{m,i}$ we obtain $x_{\sigma(1),i} \le x_{\sigma(2),i} \le \ldots \le x_{\sigma(m),i}$

We have that $\theta \in \mathbf{R}$. Pick θ_1 and θ_2 in $[x_{\sigma(j),i}, x_{\sigma(j+1),i})$



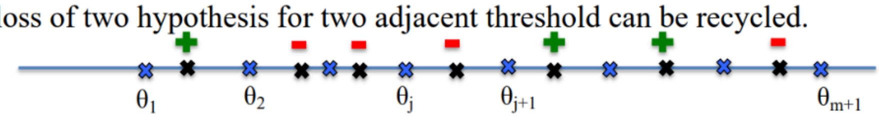
We see that $h_{i,\theta 1,b}$ and $h_{i,\theta 2,b}$ have the same error. For all $\theta \in [x_{\sigma(j),i}\ x_{\sigma(j+1),i})$ we obtain the same error, all the hypothesis $h_{i,\theta,b}$ are very similar.

In fact, we can restrict the search over all $\theta \in \mathbf{R}$ just to m + 1 values of θ, chosen in the intervals $(-\infty, x_{\sigma(1),i})$, $[x_{\sigma(1),i}, x_{\sigma(2),i})$, …, $[x_{\sigma(m),i}, +\infty)$ by taking a representative threshold in each interval. For example we can take the set of representative thresholds as containing extreme points + middle of the segments:
$\Theta_i = \{x_{\sigma(1),i}-1,\ 1/2 \times (x_{\sigma(1),i} + x_{\sigma(2),i}),\ \ldots,\ 1/2 \times (x_{\sigma(m-1),i} + x_{\sigma(m),i}),\ x_{\sigma(m),i}+1\}$

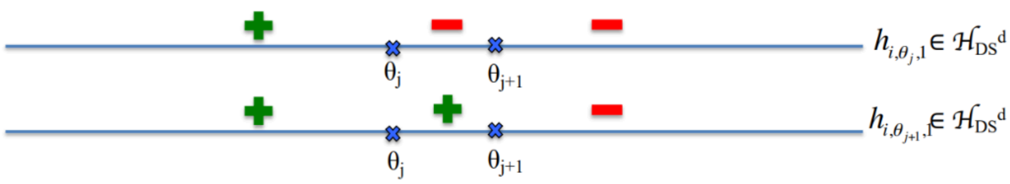
$\mathcal{H}_{DS}{}^{d} = \{h_{i,\theta,b}: \mathbf{R}^d \to \{-1,1\},\ h_{i,\theta,b}(\mathbf{x}) = \mathrm{sign}(\theta - x_i) \times b,\ 1 \le i \le d,\ \theta \in \mathbf{R},\ b \in \{-1,+1\}\}$

We have $1 \le i \le d$, $\theta \in \Theta_i$, $|\Theta_i| = m+1$, $b \in \{-1,+1\}$. So we have $d \times (m+1) \times 2$ possible hypothesis. Each hypothesis takes O(m) runtime. So the runtime is polynomial.

You can decrease the entire runtime using dynamic programming as computing the loss of two hypothesis for two adjacent threshold can be recycled.



Suppose b = 1. Then $L_S(h_{i,\theta_{j+1},1}) = L_S(h_{i,\theta_j,1}) - \dfrac{1}{m} y_j$



A formal description of boosting

Given:

- training set S = (($\mathbf{x_1}$, y_1), …, ($\mathbf{x_m}$, y_m)) of size m, $y_i \in \{-1, +1\}$
- weak learner
- T – number of rounds

For t = 1,...,T (number of rounds):

- construct distribution $\mathbf{D}^{(t)}$ on {1,..., m} ($\mathbf{x_1}$, $\mathbf{x_2}$, …, $\mathbf{x_m}$)

- find weak classifier ("rule of thumb") $h_t : \mathcal{X} \rightarrow \{-1,+1\}$ with error $\varepsilon_t < 0.5$ on $\mathbf{D}^{(t)}$:

$$\varepsilon_t = \Pr_{i \sim D^{(t)}}[h_t(x_i) \neq y_i] = \sum_{i=1}^{m} D^{(t)}(i) \times 1_{[h_t(x_i) \neq y_i]}$$

- output final/combined classifier h_{final} using all the learned weak classifiers h_t

Each round involves building the distribution $\mathbf{D}^{(t)}$ as well as a single call to the weak learner. Therefore, if the weak learner can be implemented efficiently (as happens in the case of ERM with respect to decision stumps – improper learning) then the total training process will be efficient. Different variants of boosting comes from constructing distribution $\mathbf{D}^{(t)}$ + obtaining the final classifier h_{final}

- construct distribution $\mathbf{D}^{(t)}$ on {1,..., m}:
  - $\mathbf{D}^{(1)}(i) = 1/m$
  - given $\mathbf{D}^{(t)}$ and h_t: $D^{(t+1)}(i) = \dfrac{D^{(t)}(i) \times e^{-w_t h_t(x_i) y_i}}{Z_{t+1}}$

where Z_{t+1} normalization factor ($\mathbf{D}^{(t+1)}$ is a distribution): $Z_{t+1} = \sum_{i=1}^{m} D^{(t)}(i) \times e^{-w_t h_t(x_i) y_i}$

w_t is a weight: $w_t = \frac{1}{2}\ln(\frac{1}{\varepsilon_t} - 1) > 0$ as the error $\varepsilon_t < 0.5$

ε_t is the error of h_t on $\mathbf{D}^{(t)}$: $\varepsilon_t = \Pr_{i \sim D^{(t)}}[h_t(x_i) \neq y_i] = \sum_{i=1}^{m} D^{(t)}(i) \times 1_{[h_t(x_i) \neq y_i]}$

If example $\mathbf{x}_i$ is correctly classified then $h_t(\mathbf{x}_i) = y_i$ so at the next iteration t+1 its importance (probability distribution) will be decreased to:

$$D^{(t+1)}(i) = \frac{D^{(t)}(i) \times e^{-w_t}}{Z_{t+1}} = \frac{D^{(t)}(i) \times e^{-\frac{1}{2}\ln(\frac{1}{\varepsilon_t} - 1)}}{Z_{t+1}} = \frac{D^{(t)}(i) \times (\frac{1}{\varepsilon_t} - 1)^{-\frac{1}{2}}}{Z_{t+1}} = \frac{D^{(t)}(i) \times \sqrt{\frac{\varepsilon_t}{1 - \varepsilon_t}}}{Z_{t+1}}$$

Z_{t+1}

where Z_{t+1} normalization factor ($\mathbf{D}^{(t+1)}$ is a distribution): $Z_{t+1} = \sum_{i=1}^{n} D^{(t)}(i) \times e^{-w_t h_t(x_i) y}$

w_t is a weight: $w_t = \frac{1}{2}\ln(\frac{1}{\varepsilon_t} - 1) > 0$ as the error $\varepsilon_t < 0.5$

ε_t is the error of h_t on $\mathbf{D}^{(t)}$: $\varepsilon_t = \Pr_{i \sim D^{(t)}}[h_t(x_i) \neq y_i] = \sum_{i=1}^{m} D^{(t)}(i) \times 1_{[h_t(x_i) \neq y_i]}$

If example $\mathbf{x}_i$ is correctly classified then $h_t(\mathbf{x}_i) = y_i$ so at the next iteration t+1 its importance (probability distribution) will be decreased to $D^{(t+1)}(i) = \dfrac{D^{(t)}(i) \times e^{-w_t}}{Z_{t+1}}$

If example x_i is misclassified then $h_t(x_i) \neq y_i$ so at the next iteration t+1 its importance (probability distribution) will be increased to $D^{(t+1)}(i) = \dfrac{D^{(t)}(i) \times e^{w_t}}{Z_{t+1}}$

output final/combined classifier h_{final}: $h_{final}(x) = sign(\sum_{t=1}^{T} w_t h_t(x))$