Ensemble: Boosting Models

Bagging vs Boosting Ideas

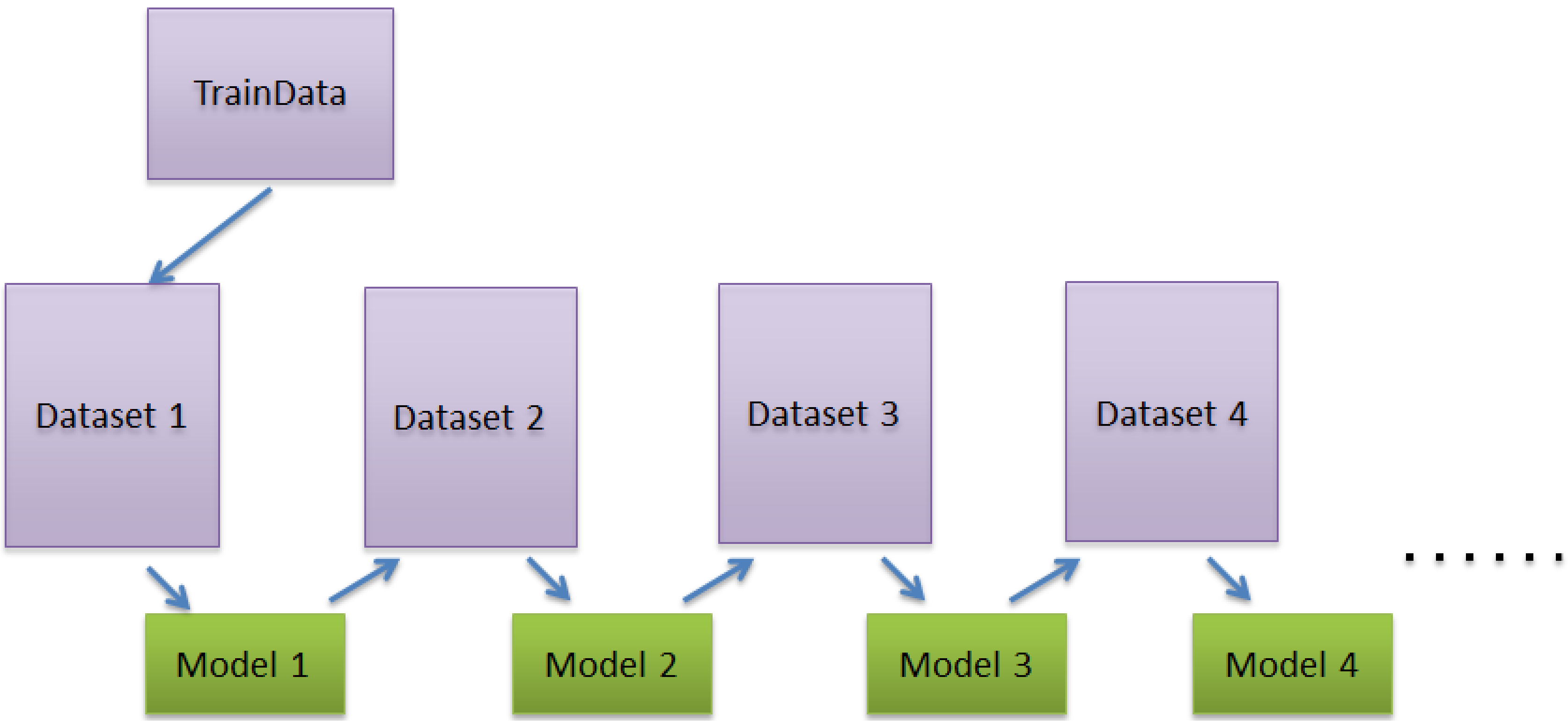
- Bagging: Each member learns on sample of train data. For prediction, each member has an equal weight for their vote and majority vote is taken as final prediction.
- Boosting: Each member has an expert learning on weighted sample of train data. For prediction, each member has weighted vote and weighted vote is taken as final prediction.

Boosting: Intuitive thought

- Bagging uses bootstrapped samples for diversification while boosting uses weights for data points for same purpose.

- At step (b+1), the idea is to set higher weight for the individuals which are misclassified in b^{th} step. The constructions of the successive models is sequential by nature.

Boosting: Intuitive thought



*Each model corrects the mistakes or shortcomings of its predecessor.

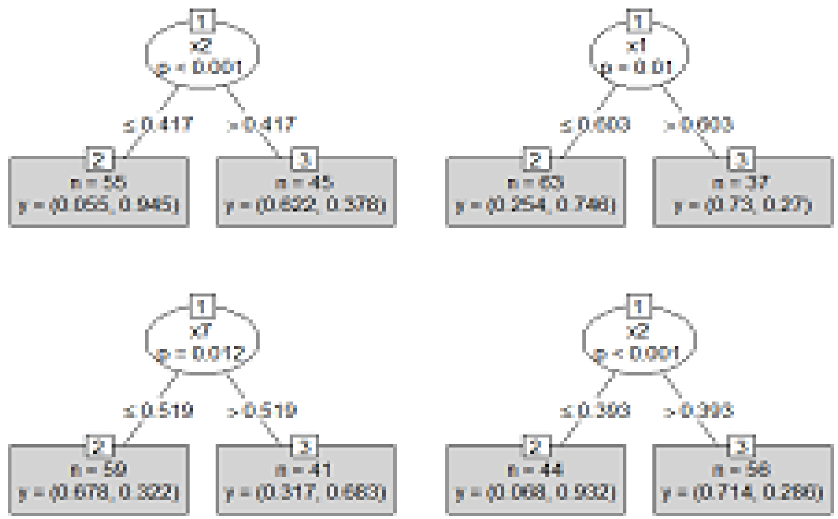
Properties of Boosting

- By guiding the learning at each step, boosting reduces the bias; by combining them, it also reduces the variance.

- Boosting can be applied to any kind of model. But trees are advantageous because we can modulate the properties of the model(more or less in depth).

Boosted Trees

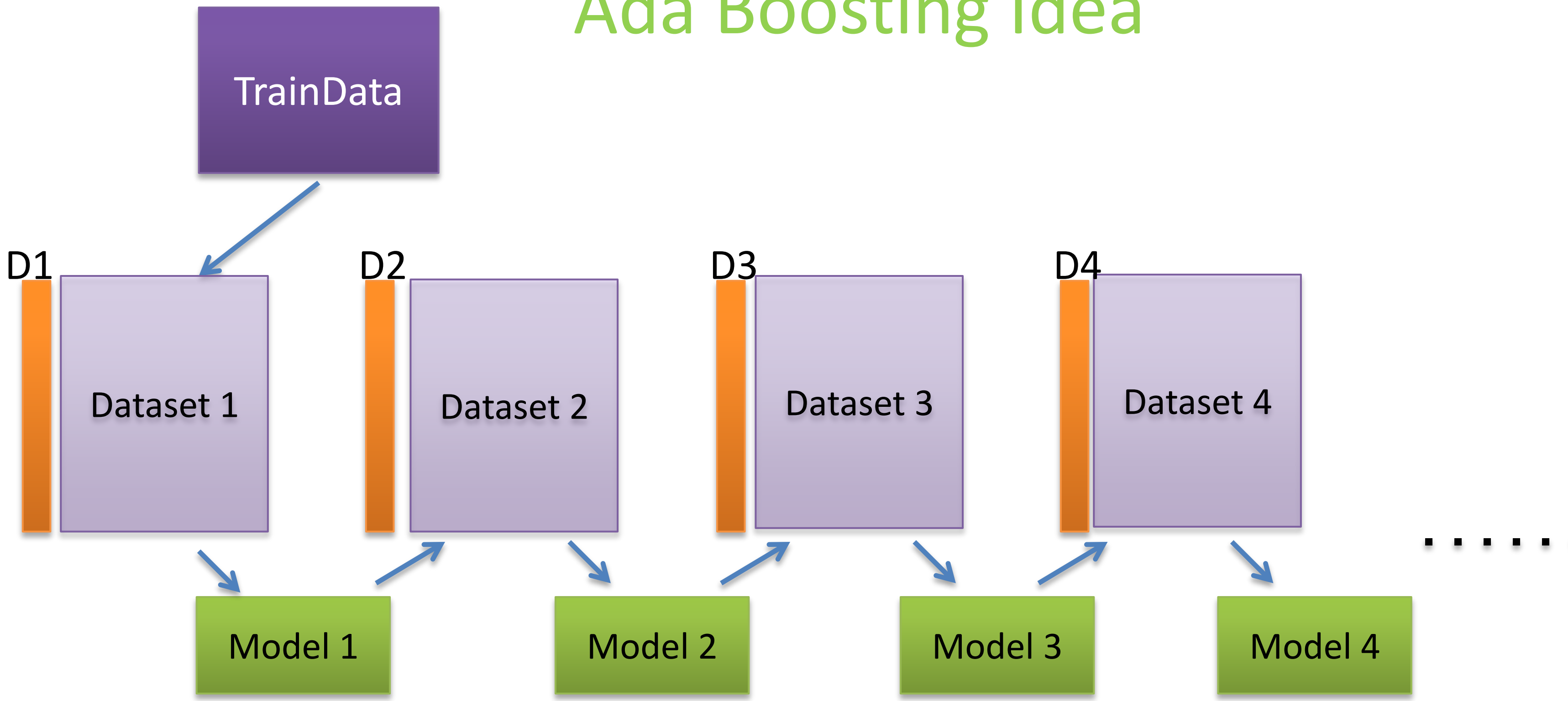
- Since boosting reduces the bias, we can use simpler one-level decision tree model which has high bias but a very low variance.



- Increase the tree depth little to take into account interactions among variables.

Adaptive Boosting (AdaBoost)

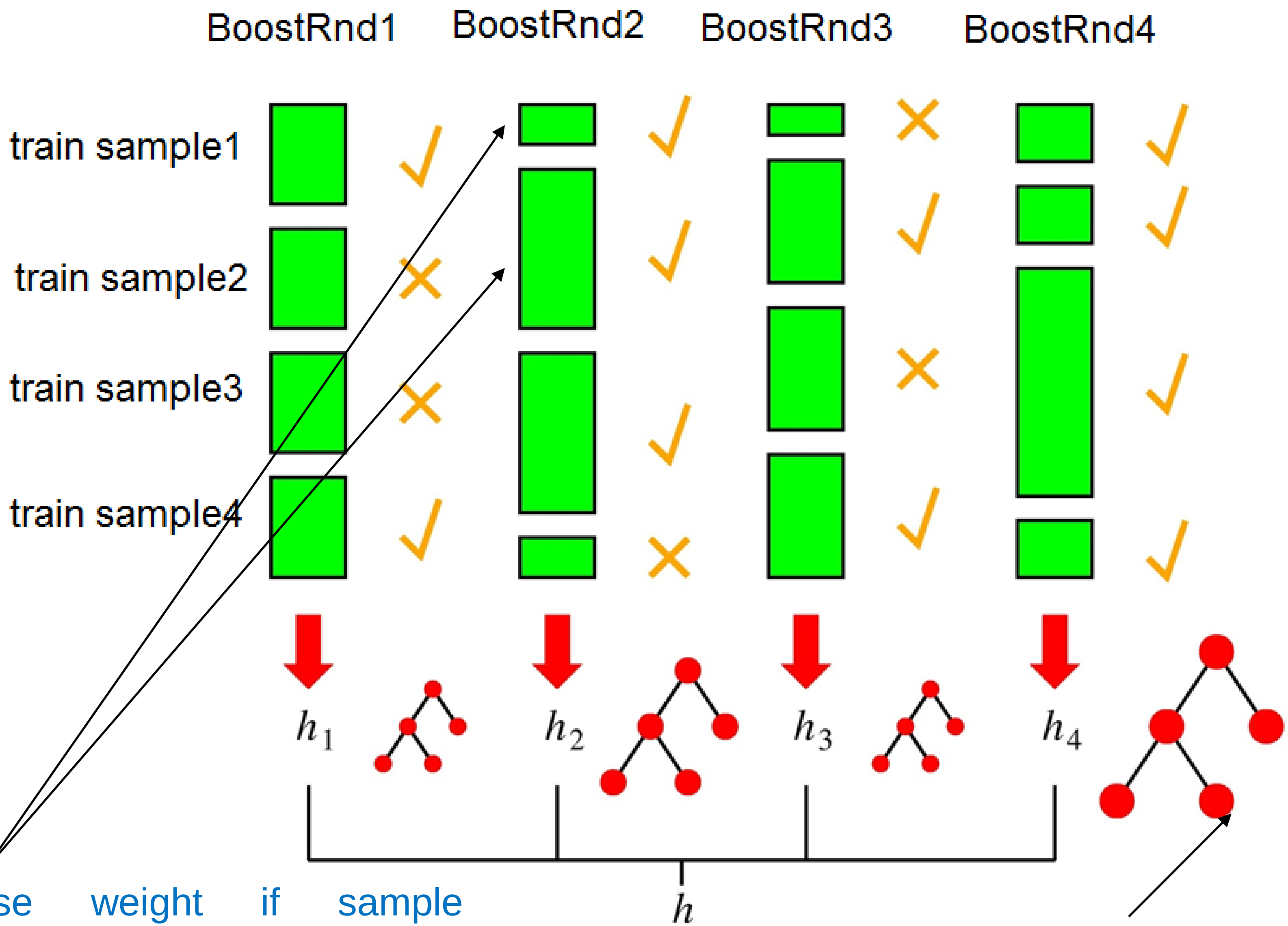
Ada Boosting Idea



Same dataset is used for each model but each dataset is associated with different distribution of weights(or probabilities) for its samples.

The model building algorithm must take sample weights into consideration. The higher weighted data samples must be classified correctly.

Ada Boosting Idea



Increase weight if sample misclassified otherwise decrease weight

The model with high accuracy gets strong weighted vote.

AdaBoost Algorithm

Adaboost Pseudo-code

$D_k(i)$: Example i weight after learner k

α_k: Learner k weight

Set uniform example weights.

for Each base learner **do**

Train base learner with weighted sample.

Test base learner on all data.

Set learner weight with weighted error.

Set example weights based on ensemble predictions.

end for

Adaboost is short for "Adaptive Boosting", because the algorithm adapts weights on the base learners and training examples.

Detailed AdaBoost Algorithm

Adaboost Pseudo-code

$D_k(i)$: Example i weight after learner k

α_k: Learner k weight

$\forall i : D_0(i) \leftarrow \frac{1}{N}$

for k=1 to K **do**

  $\mathcal{D} \leftarrow$ data sampled with D_{k-1}.

  $h_k \leftarrow$ base learner trained on $\mathcal{D}$

  $\epsilon_k \leftarrow \sum_{i=1}^{N} D_{k-1}(i)\delta[h_k(x_i) \neq y_i]$

  $\alpha_k \leftarrow \frac{1}{2} \log \frac{1-\epsilon_k}{\epsilon_k}$

  $D_k(i) \leftarrow \frac{D_{k-1}(i)e^{-\alpha_k y_i h_k(x_i)}}{Z_k}$

end for

Detailed AdaBoost Algorithm

Decision rule:

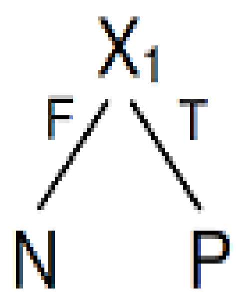
$$H(x) = sign(\sum_{k=1}^{K} \alpha_k h_k(x))$$

Example

	X_1	X_2	X_3	X_4	C			
	T	T	F	F	P			
	T	T	T	T	N			
	T	F	F	T	P			
	T	F	T	F	P			
	F	F	F	T	N			
	F	T	F	F	P			
	F	T	F	F	N			

AdaBoosting-Iteration1

P_1	X_1	X_2	X_3	X_4	C			
0.1429	T	T	F	F	P			
0.1429	T	T	T	T	N			
0.1429	T	F	F	T	P			
0.1429	T	F	T	F	P			
0.1429	F	F	F	T	N			
0.1429	F	T	F	F	P			
0.1429	F	T	F	F	N			

$h_1 =$ 

AdaBoosting-Iteration1

P_1	X_1	X_2	X_3	X_4	C			
0.1429	T	T	F	F	P	✔		
0.1429	T	T	T	T	N	✘		
0.1429	T	F	F	T	P	✔		
0.1429	T	F	T	F	P	✔		
0.1429	F	F	F	T	N	✔		
0.1429	F	T	F	F	P	✘		
0.1429	F	T	F	F	N	✔		

Compute ε_1

$$\varepsilon_1 = \sum_i P_1(i)[h_1(i) \neq c(i)]$$

$$= 0.1429 + 0.1429$$

$$= 0.2858$$

Compute α_1

$$\varepsilon_1 = 0.2858$$

$$\alpha_1 = \frac{1}{2}\ln\left(\frac{1-\varepsilon_1}{\varepsilon_1}\right)$$

$$= \frac{1}{2}\ln\left(\frac{1-0.2858}{0.2858}\right) = 0.4581$$

AdaBoosting-Iteration1

P_1	X_1	X_2	X_3	X_4	C			
0.1429	T	T	F	F	P	✔	0.0799	
0.1429	T	T	T	T	N	✘	0.2259	
0.1429	T	F	F	T	P	✔	0.0799	
0.1429	T	F	T	F	P	✔	0.0799	
0.1429	F	F	F	T	N	✔	0.0799	
0.1429	F	T	F	F	P	✘	0.2259	
0.1429	F	T	F	F	N	✔	0.0799	

Compute multiplier

$\varepsilon_1 = 0.2858$

$\alpha_1 = 0.4581$

$e^{\alpha_1} = 1.581$ misclassified instances

$e^{-\alpha_1} = 0.5593$ correct instances

correct instances

$$P_2 = P_1 e^{\alpha_1}$$
$$= .1429 * .5593$$
$$= .0799$$

misclassified instances

$$P_2 = P_1 e^{-\alpha_1}$$
$$= .1429 * 1.581$$
$$= .2259$$

AdaBoosting-Iteration1

P_1	X_1	X_2	X_3	X_4	C			P_2
0.1429	T	T	F	F	P	✔	0.0799	0.0939
0.1429	T	T	T	T	N	✘	0.2259	
0.1429	T	F	F	T	P	✔	0.0799	
0.1429	T	F	T	F	P	✔	0.0799	
0.1429	F	F	F	T	N	✔	0.0799	
0.1429	F	T	F	F	P	✘	0.2259	
0.1429	F	T	F	F	N	✔	0.0799	

Normalize P_2

$$Z = 5 * 0.0799 + 2 * 0.2259 = 0.8513$$

AdaBoosting-Iteration1

P_1	X_1	X_2	X_3	X_4	C			P_2
0.1429	T	T	F	F	P	✔	0.0799	0.0939
0.1429	T	T	T	T	N	✘	0.2259	0.2653
0.1429	T	F	F	T	P	✔	0.0799	0.0939
0.1429	T	F	T	F	P	✔	0.0799	0.0939
0.1429	F	F	F	T	N	✔	0.0799	0.0939
0.1429	F	T	F	F	P	✘	0.2259	0.2653
0.1429	F	T	F	F	N	✔	0.0799	0.0939

Normalize P_2 $Z = 5 * 0.0799 + 2 * 0.2259 = 0.8513$

AdaBoosting-Iteration2

P_2	X_1	X_2	X_3	X_4	C			
0.0939	T	T	F	F	P			
0.2653	T	T	T	T	N			
0.0939	T	F	F	T	P			
0.0939	T	F	T	F	P			
0.0939	F	F	F	T	N			
0.2653	F	T	F	F	P			
0.0939	F	T	F	F	N			

$h_2 =$

AdaBoosting-Iteration2

P_2	X_1	X_2	X_3	X_4	C			
0.0939	T	T	F	F	P	✔	0.0451	
0.2653	T	T	T	T	N	✔	0.1276	
0.0939	T	F	F	T	P	✘	0.1953	
0.0939	T	F	T	F	P	✔	0.0451	
0.0939	F	F	F	T	N	✔	0.0451	
0.2653	F	T	F	F	P	✔	0.1276	
0.0939	F	T	F	F	N	✘	0.1953	

Multipliers:

$$\varepsilon_2 = 0.1878$$

$$\alpha_2 = 0.7322$$

$$e^{\alpha_1} = 2.080$$

$$e^{-\alpha_1} = 0.4808$$

AdaBoosting-Iteration2

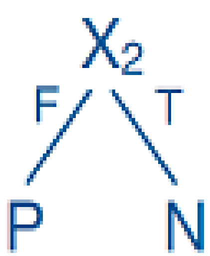
P_2	X_1	X_2	X_3	X_4	C			
0.0939	T	T	F	F	P	✔	0.0451	0.0577
0.2653	T	T	T	T	N	✔	0.1276	0.1634
0.0939	T	F	F	T	P	✘	0.1953	0.25
0.0939	T	F	T	F	P	✔	0.0451	0.0577
0.0939	F	F	F	T	N	✔	0.0451	0.0577
0.2653	F	T	F	F	P	✔	0.1276	0.1634
0.0939	F	T	F	F	N	✘	0.1953	0.25

Normalize:

$$\begin{aligned} Z = {} & 3*0.0451 \\ & +2*0.1276 \\ & +2*0.1953 \\ = {} & 0.7811 \end{aligned}$$

AdaBoosting-Iteration3

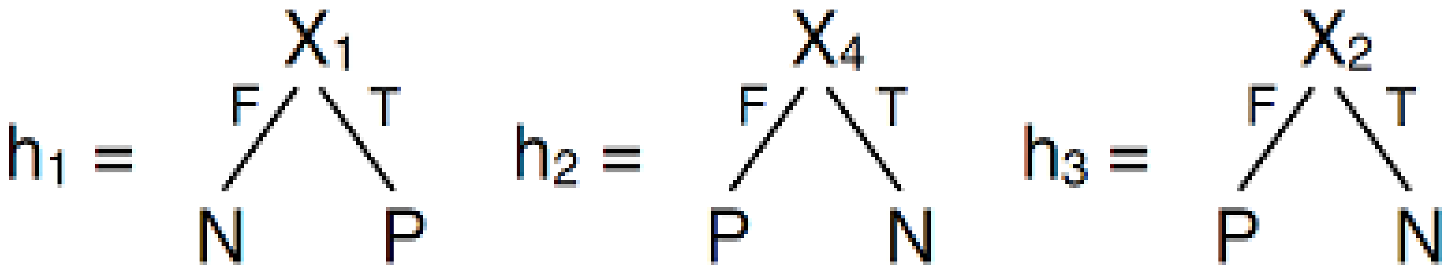
P_2	X_1	X_2	X_3	X_4	C			
0.0577	T	T	F	F	P			
0.1634	T	T	T	T	N			
0.25	T	F	F	T	P			
0.0577	T	F	T	F	P			
0.0577	F	F	F	T	N			
0.1634	F	T	F	F	P			
0.25	F	T	F	F	N			

$h_3 =$ 

$$\varepsilon_3 = 0.2788$$

$$\alpha_3 = 0.4752$$

AdaBoosting - Final Model



$\alpha_1 = 0.4581$ $\alpha_2 = 0.7322$ $\alpha_3 = 0.4752$

AdaBoosting - Pros

- Good performances in prediction
- Easy to configure (**B**)
- Variable importance measurement
- Operates on bias and the variance
- Do not need large tree (quickness)
- We can modulate the depth of the tree in order to take into account the interactions between the variables

AdaBoosting - Cons

- No (obvious) parallelisation of the computations
- If base model is too simple/weak: underfitting
- If base model is too complex/strong: overfitting
- Not robust against outliers or noise on the class attribute, excessive weights