

Computational Data Analysis Machine Learning

Yao Xie, Ph.D.

Associate Professor

Harold R. and Mary Anne Nash Early Career Professor
H. Milton Stewart School of Industrial and Systems
Engineering

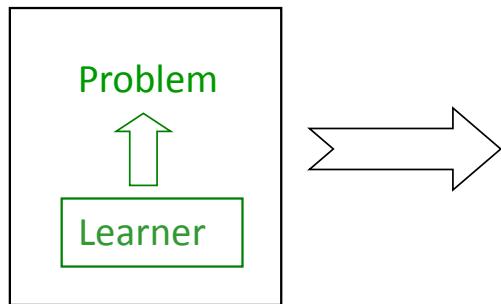
Boosting



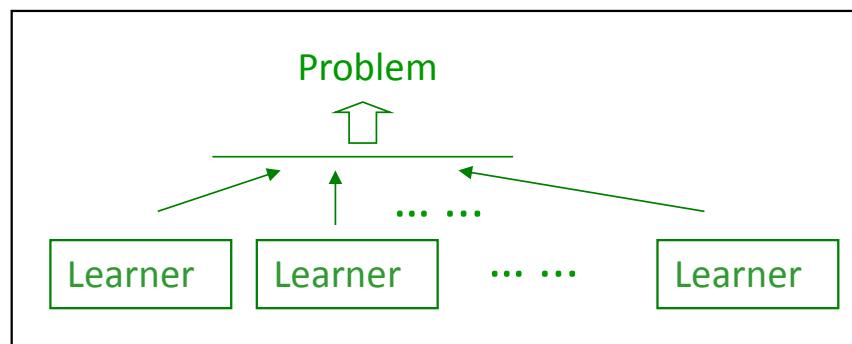
Rationale: Combination of methods

- There is no algorithm that is always the most accurate
- We can select simple “weak” classification or regression methods and combine them into a single “strong” method
- Different learners use different
 - Algorithms, Hyperparameters, Representations (Modalities), Training sets, Subproblems

Previously:



Ensemble:



Email spam classification

- 4601 email messages
- Label: regular, spam
- Objective: design automatic spam detector that filter out spam before clogging users' mailboxes
- Bag-of-words model: 57 most commonly occurring words of spam email (example)

	george	you	your	hp	free	hpl	!	our	re	edu	remove
spam	0.00	2.26	1.38	0.02	0.52	0.01	0.51	0.51	0.13	0.01	0.28
email	1.27	1.27	0.44	0.90	0.07	0.43	0.11	0.18	0.42	0.29	0.01

- Boosting algorithm decision rule

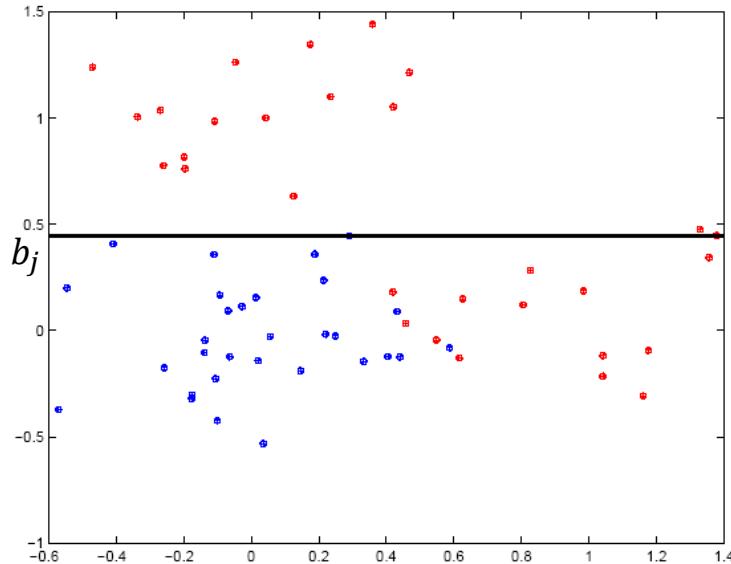
```
if (%george < 0.6) & (%you > 1.5)    then spam  
else email
```

Decision stump

- Let $y \in \{\pm 1\}$
- Decision stump:

$$h(x; \theta_j) = \text{sign}(w_j x_j + b_j)$$

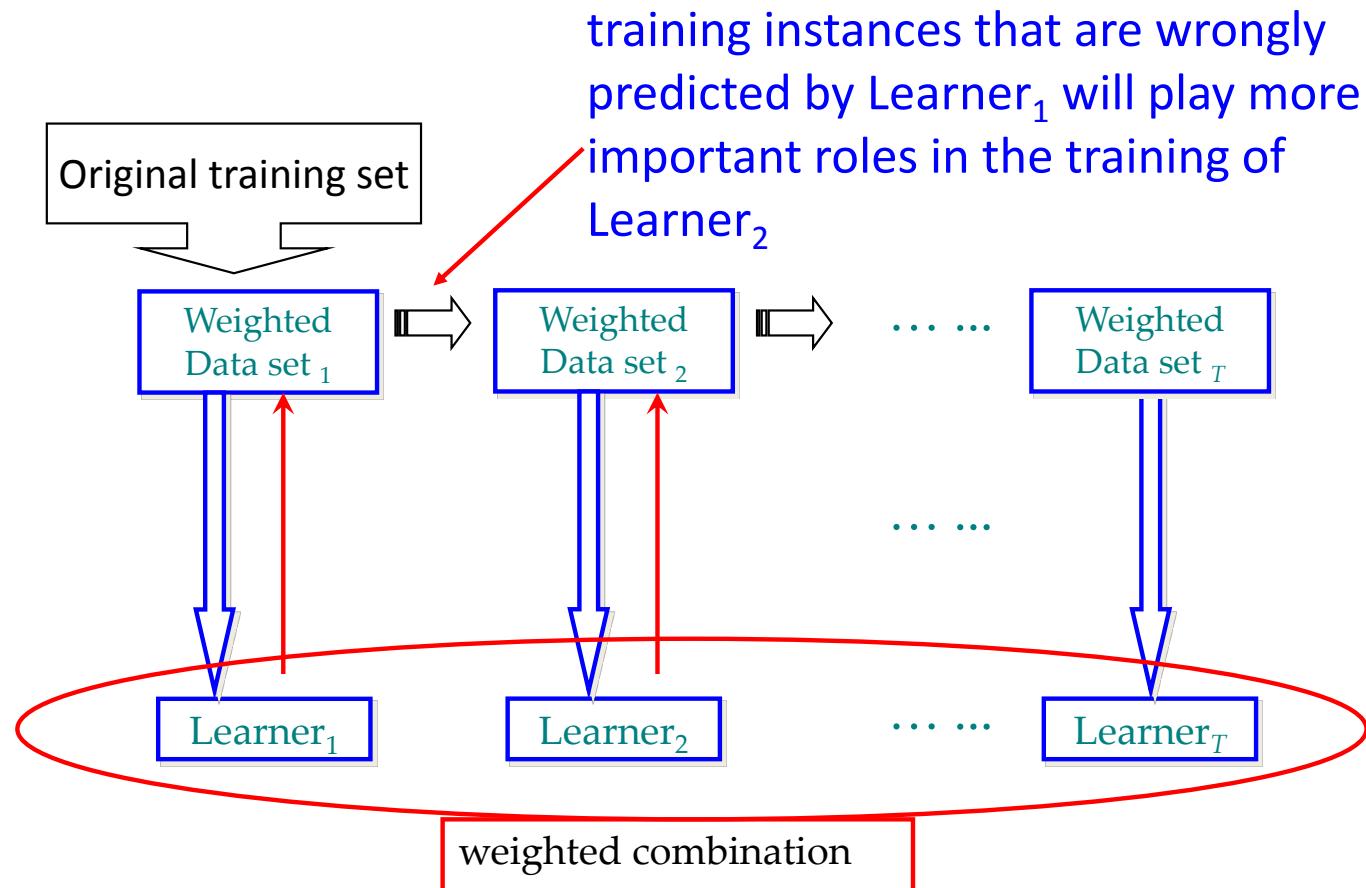
- Each decision stump pays attention to only a single dimension of the input vector



Boosting

- Boosting: general methods of converting rough rules of thumb (or weak classifier) into highly accurate prediction rule
- A family of methods which produce a sequence of classifiers
 - Each classifier is dependent on the previous one and focuses on the previous one's errors
 - Examples that are incorrectly predicted in the previous classifiers are chosen more often or weighted more heavily when estimating a new classifier.
- Questions:
 - How to choose “hardest” examples?
 - How to combine these classifiers?

Adaboost flow chart



AdaBoost

- constructing D_t :

- $D_1(i) = 1/m$
- given D_t and h_t : $\epsilon_t = \sum_{i=1}^m D_t(i) \mathbb{I}\{y_i \neq h_t(x_i)\}$

$$\begin{aligned} D_{t+1}(i) &= \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t} & \text{if } y_i = h_t(x_i) \\ e^{\alpha_t} & \text{if } y_i \neq h_t(x_i) \end{cases} \\ &= \frac{D_t(i)}{Z_t} \exp(-\alpha_t y_i h_t(x_i)) \end{aligned}$$

where Z_t = normalization constant

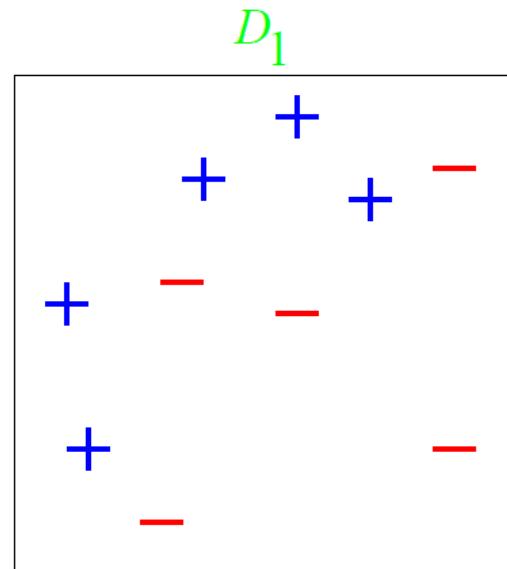
$$\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right) > 0$$

- final classifier:

- $H_{\text{final}}(x) = \text{sign} \left(\sum_t \alpha_t h_t(x) \right)$

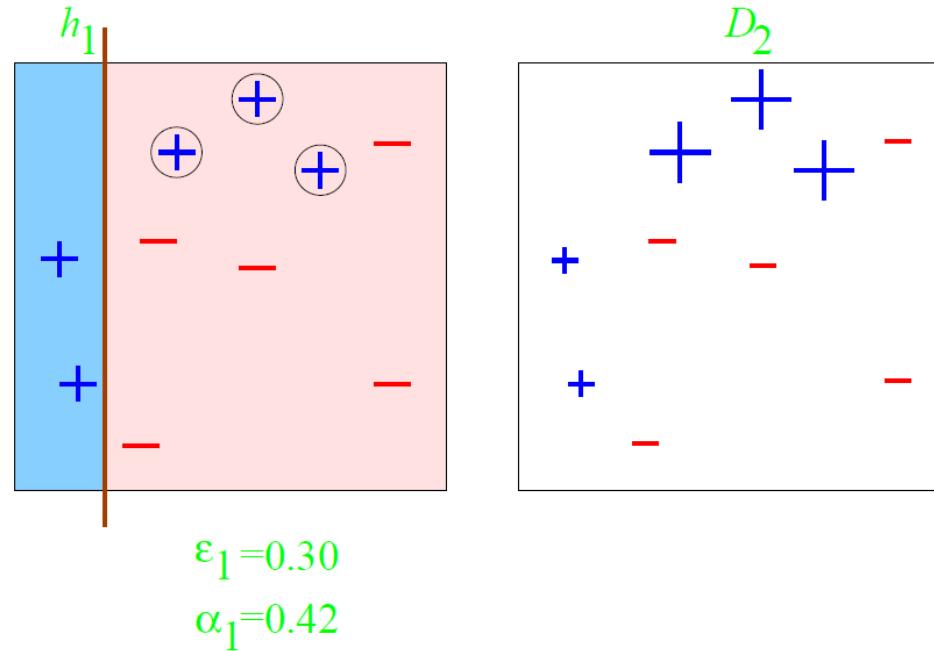
Toy Example

- Weak classifier (rule of thumb): vertical or horizontal half-planes (or decision stump)
- Let $y \in \{\pm 1\}$
- Uniform weights on all examples



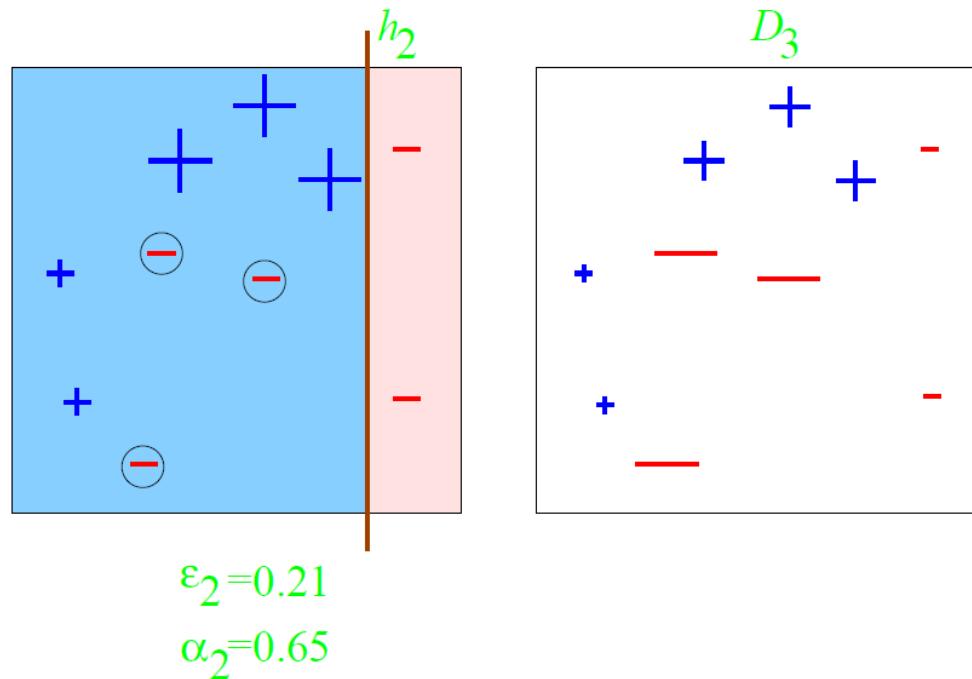
Boosting round 1

- Choose a decision stump (weak classifier)
- Some data points obtain higher weights because they are classified incorrectly



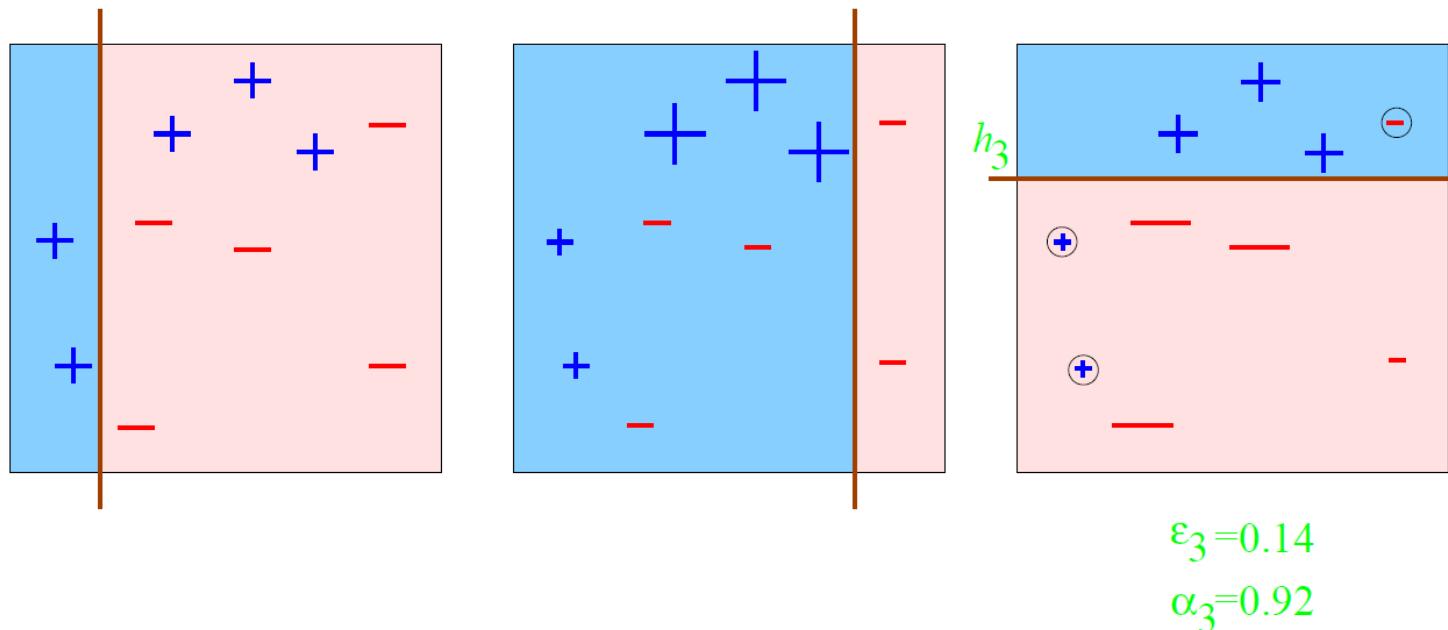
Boosting round 2

- Choose a new decision stump
- Reweight again. For incorrectly classified examples, weight increased



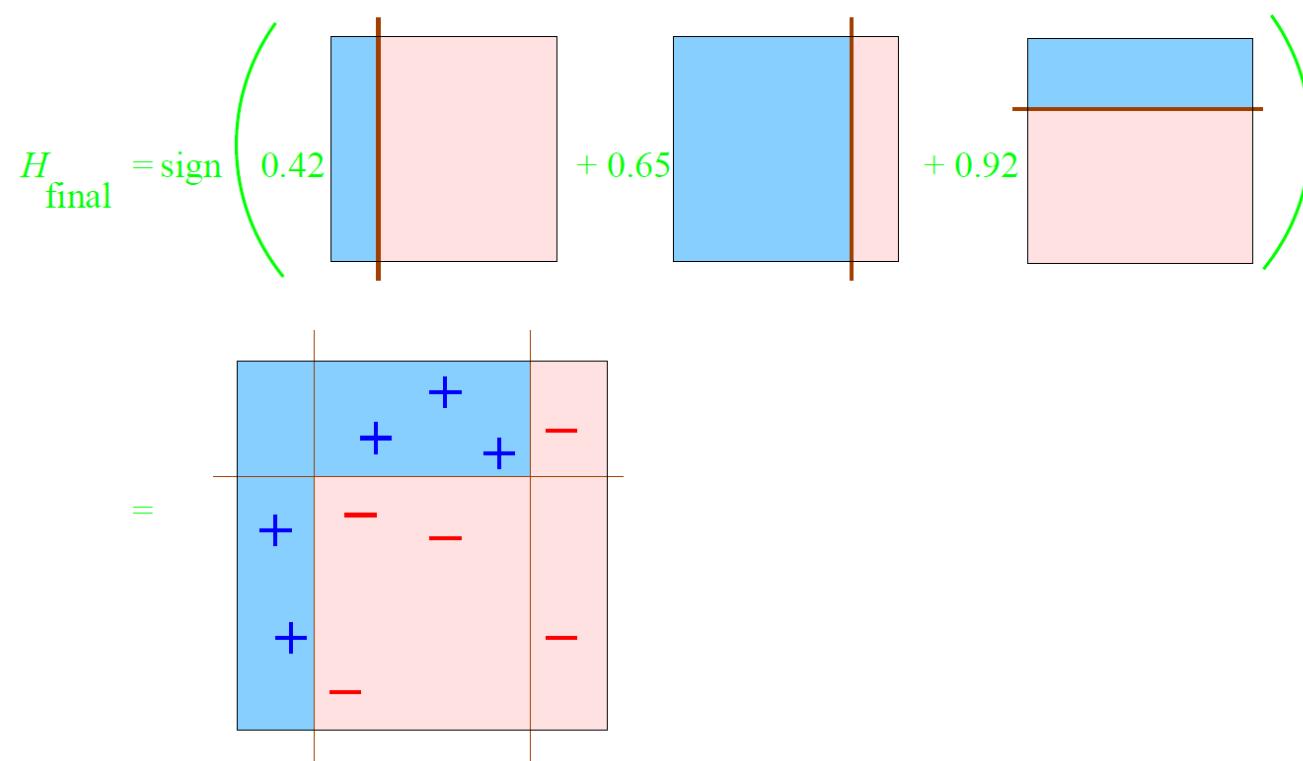
Boosting round 3

- Repeat the same process
- Now we have 3 classifiers



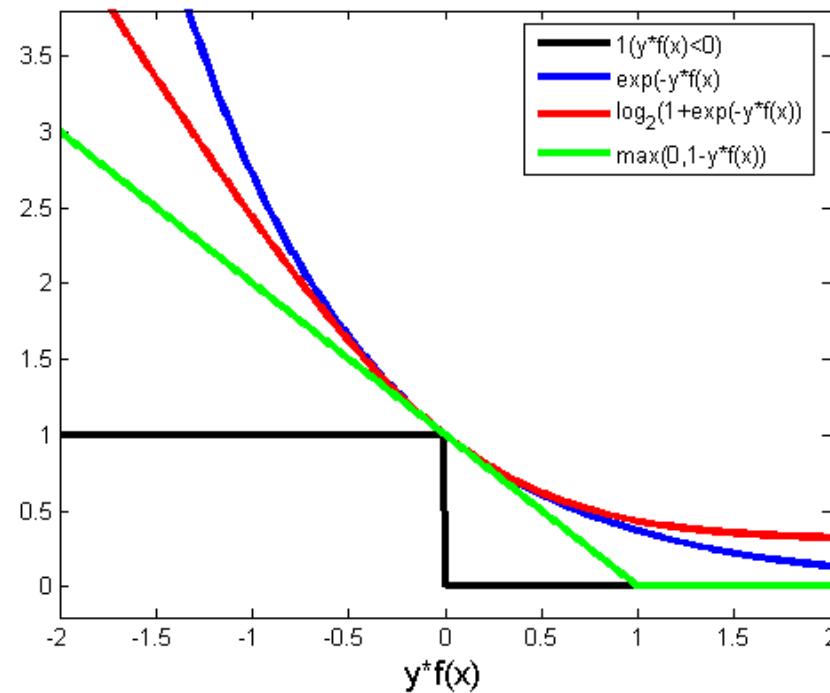
Boosting aggregate classifier

- Final classifier is weighted combination of weak classifiers



Loss functions

- Loss function: $l(y, f(x))$
- Eg., $\mathbb{I}\{y * f(x) < 0\}$



Understanding boosting: exponential loss

- Combined classifier

$$f_t(x) = \alpha_1 h_1(x; \theta_1) + \cdots + \alpha_t h_t(x; \theta_t)$$

- Exponential loss

$$\begin{aligned}\hat{L}(f) &= \frac{1}{m} \sum_{i=1}^m \exp(-y^i f_t(x^i)) \\ &= \frac{1}{m} \sum_{i=1}^m \exp\left(-y^i (f_{t-1}(x^i) + \alpha_t h_t(x; \theta_t))\right) \\ &= \frac{1}{m} \sum_{i=1}^m \exp(-y^i f_{t-1}(x^i)) \exp(-y^i \alpha_t h_t(x^i; \theta_t))\end{aligned}$$

Understanding boosting: exponential loss

- Define weighting on data point

$$D_t(i) = \exp(-y^i f_{t-1}(x^i))$$

- Then

$$\hat{L}(f) = \frac{1}{m} \sum_i^m D_t(i) \exp(-y^i \alpha_t h_t(x^i; \theta_t))$$

- The combined classifier f_{t-1} based on $t - 1$ iterations defines a weighted loss criterion for the next simple classifier to add
- each training sample is weighted by its "classifiability" (or difficulty) seen by the classifier we have built so far

Adding $h(x; \theta_n)$ sequentially

$$\bullet \hat{L}(f) = \frac{1}{m} \sum_{i=1}^m \exp\left(-y^i f_{t-1}(x^i)\right) \exp(-y^i \alpha_t h_t(x^i; \theta_t))$$

$D_t(i)$

- At iteration t , Suppose $f_{t-1}(x)$ is already given to you
- Define weighting on data point $D_t(i)$, then

$$\hat{L}(f) = \frac{1}{m} \sum_i^m D_t(i) \exp(-y^i \alpha_t h_t(x^i; \theta_t))$$

- Try to find α_t and h_t

Find h_t Linearization of loss function

- Assuming α_t is small, the empirical loss criterion reduces to

$$\exp(-y^i \alpha_t h_t(x^i; \theta_t)) \approx 1 - y_i \alpha_t h_t(x^i; \theta_t)$$

- We could choose a new component classifier to optimize this weighted agreement

$$\begin{aligned} & \frac{1}{m} \sum_i^m D_t(i) \exp(-y^i \alpha_t h_t(x^i; \theta_t)) \\ & \approx \frac{1}{m} \sum_i^m D_t(i) (1 - y^i \alpha_t h_t(x^i; \theta_t)) \\ & = \frac{1}{m} \sum_i^m D_t(i) - \frac{1}{m} \alpha_t \sum_i^m D_t(i) y^i h_t(x^i; \theta_t) \end{aligned}$$

Find h_t : a possible algorithm

- At stage t we find θ_t that maximize (or at least give a sufficiently high) weighted agreement:

$$\theta_t \leftarrow \operatorname{argmax}_{\theta} \sum_{i=1}^m D_t(i) y^i h_t(x^i; \theta)$$
$$\epsilon_t := \sum_{i=1}^m D_t(i) I\{y^i \neq h_t(x^i; \theta)\}$$

- Note that $y \in \{\pm 1\}$, and $h_t(x) \in \{\pm 1\}$
- $y^i h_t(x^i; \theta) \in \{\pm 1\}$

Find α_t

- Note that $y^i \in \{\pm 1\}$, and $h_t(x^i) \in \{\pm 1\}$, $y^i h_t(x^i; \theta) \in \{\pm 1\}$

$$\frac{\partial L}{\partial \alpha_t} = - \sum_{i=1}^m D_t(i) \exp(-y^i \alpha_t h_t(x^i; \theta_t)) y^i h_t(x^i; \theta_t)$$

$$= - \sum_{i:h_t(x^i)=y^i} D_t(i) \exp(-\alpha_t) + \sum_{i:h_t(x^i)\neq y^i} D_t(i) \exp(\alpha_t)$$
$$= (\epsilon_t - 1) \exp(-\alpha_t) + \epsilon_t \exp(\alpha_t) = 0$$

$$\Rightarrow \frac{1 - \epsilon_t}{\epsilon_t} = \exp(2\alpha_t) \Rightarrow \frac{1}{2} \ln\left(\frac{1 - \epsilon_t}{\epsilon_t}\right) = \alpha_t$$

Find α_t

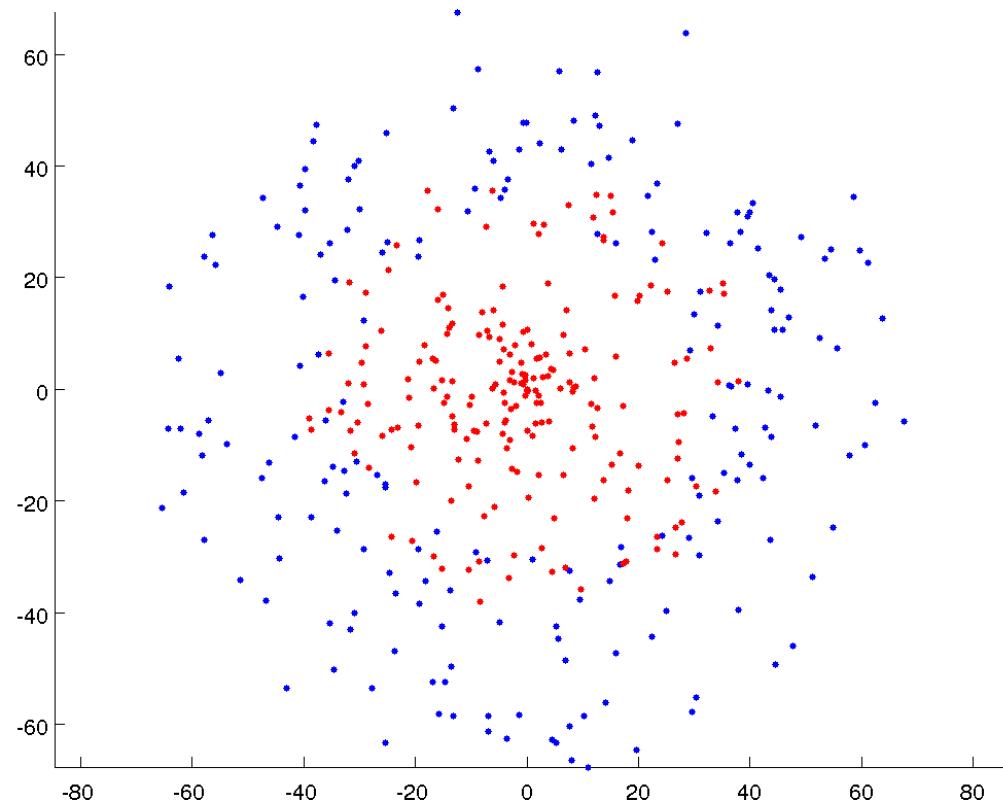
- Then we go back and find the “votes” α_t associated with the new classifier by minimizing the **original** weighted (exponential) loss

$$L(\alpha_t) = \sum_i^m D_t(i) \exp(-y^i \alpha_t h_t(x^i; \theta_t))$$

- Set derivative to 0 and solve for α_t

$$\frac{\partial L}{\partial \alpha_t} = - \sum_i^m D_t(i) \exp(-y^i \alpha_t h_t(x^i; \theta_t)) y^i h_t(x^i; \theta_t) = 0$$

Two rings dataset (demo)



Overall comparison

TABLE 10.1. Some characteristics of different learning methods. Key: $\blacktriangle = \text{good}$, $\blacklozenge = \text{fair}$, and $\blacktriangledown = \text{poor}$.

Characteristic	Neural Nets	SVM	Trees	MARS	k-NN, Kernels
Natural handling of data of “mixed” type	\blacktriangledown	\blacktriangledown	\blacktriangle	\blacktriangle	\blacktriangledown
Handling of missing values	\blacktriangledown	\blacktriangledown	\blacktriangle	\blacktriangle	\blacktriangle
Robustness to outliers in input space	\blacktriangledown	\blacktriangledown	\blacktriangle	\blacktriangledown	\blacktriangle
Insensitive to monotone transformations of inputs	\blacktriangledown	\blacktriangledown	\blacktriangle	\blacktriangledown	\blacktriangledown
Computational scalability (large N)	\blacktriangledown	\blacktriangledown	\blacktriangle	\blacktriangle	\blacktriangledown
Ability to deal with irrelevant inputs	\blacktriangledown	\blacktriangledown	\blacktriangle	\blacktriangle	\blacktriangledown
Ability to extract linear combinations of features	\blacktriangle	\blacktriangle	\blacktriangledown	\blacktriangledown	\blacklozenge
Interpretability	\blacktriangledown	\blacktriangledown	\blacklozenge	\blacktriangle	\blacktriangledown
Predictive power	\blacktriangle	\blacktriangle	\blacktriangledown	\blacklozenge	\blacktriangle

