# CS373 Data Mining and Machine Learning Lecture 8



Jean Honorio
Purdue University



*(originally prepared by Tommi Jaakkola, MIT CSAIL)*

# Today's topics

- Ensembles and Boosting
  - ensembles, relation to feature selection
  - myopic forward-fitting and boosting
  - understanding boosting

# Ensembles

- An ensemble classifier combines a set of m “weak” base learners into a “strong” ensemble

$$\underbrace{h_m(\underline{x})}_{\text{ensemble discriminant function}} = \alpha_1 \underbrace{h(\underline{x};\underline{\theta}_1)}_{\text{base learner}} + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$

non-negative “votes”

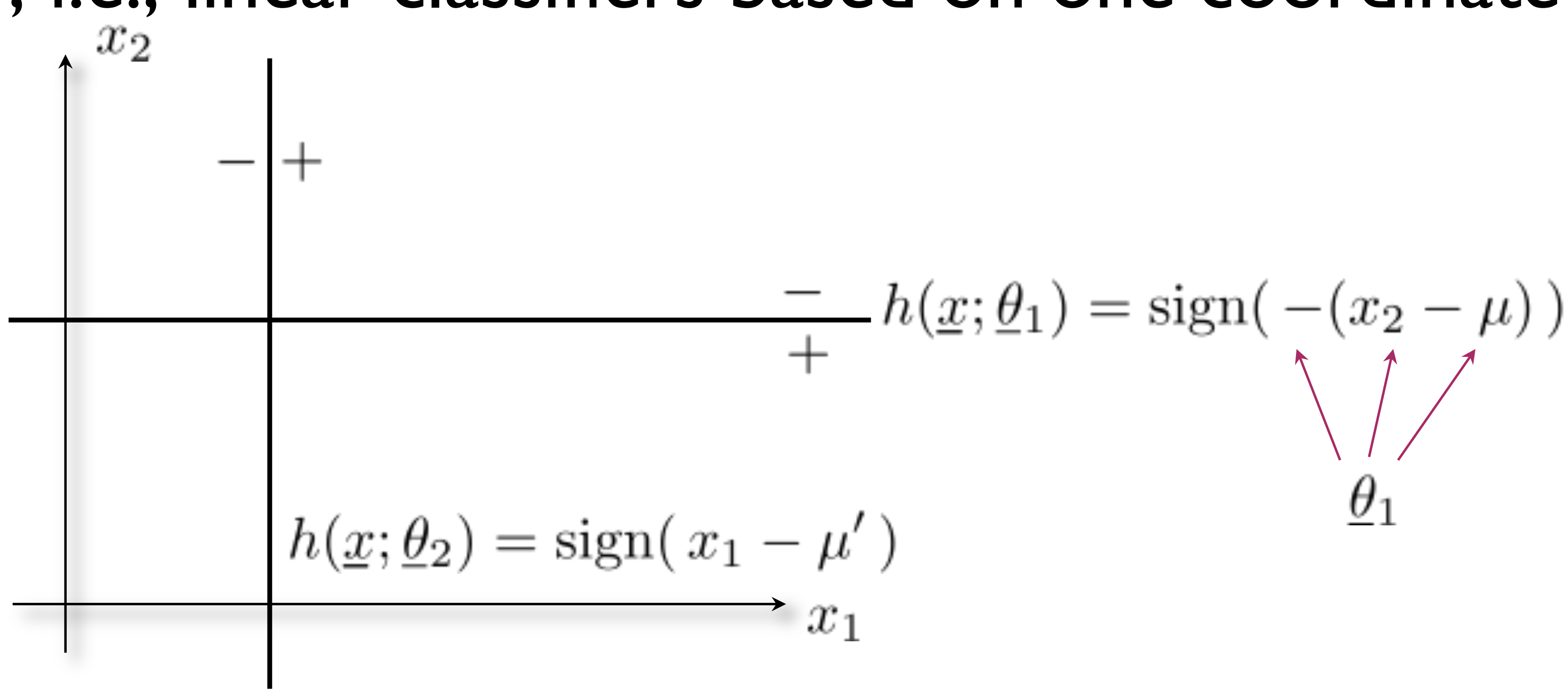
# Ensembles

- An ensemble classifier combines a set of m "weak" base learners into a "strong" ensemble

$$\underbrace{h_m(\underline{x})}_{\text{ensemble discriminant function}} = \underbrace{\alpha_1}_{\text{non-negative "votes"}} \underbrace{h(\underline{x};\underline{\theta}_1)}_{\text{base learner}} + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$



- The base learners are typically simple "decision stumps", i.e., linear classifiers based on one coordinate



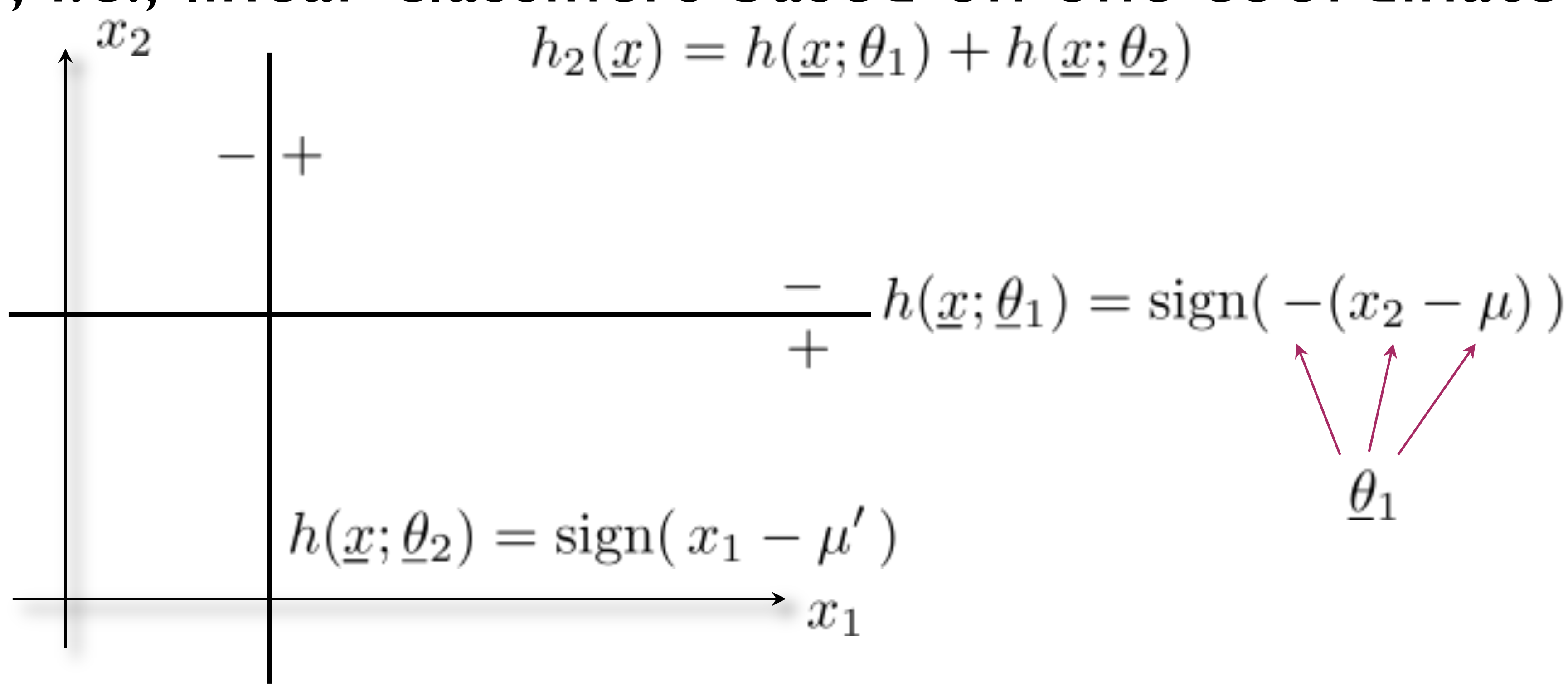
# Ensembles

- An ensemble classifier combines a set of m "weak" base learners into a "strong" ensemble

$$\underbrace{h_m(\underline{x})}_{\text{ensemble discriminant function}} = \underbrace{\alpha_1}_{\text{non-negative ``votes''}} \underbrace{h(\underline{x};\underline{\theta}_1)}_{\text{base learner}} + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$



- The base learners are typically simple "decision stumps", i.e., linear classifiers based on one coordinate



# Ensembles

- An ensemble classifier combines a set of m “weak” base learners into a “strong” ensemble

$$h_m(\underline{x}) = \alpha_1 h(\underline{x};\underline{\theta}_1) + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$

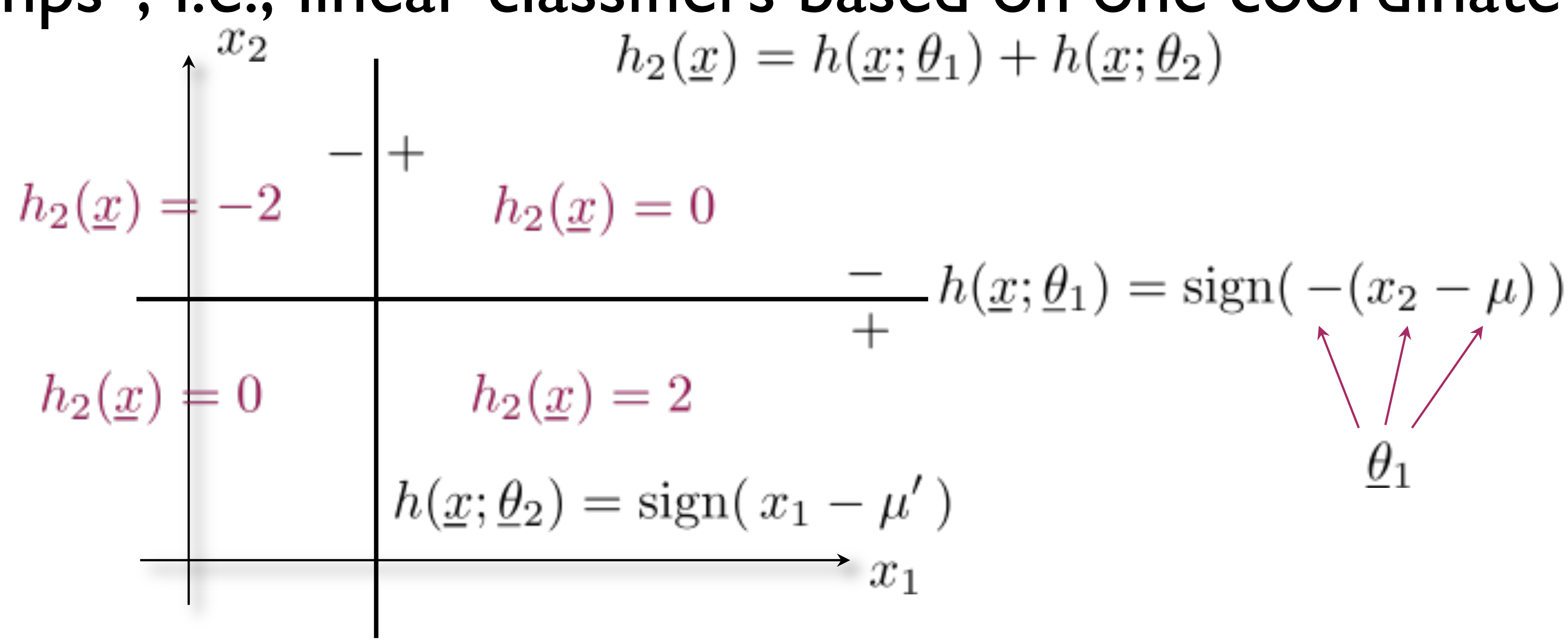
ensemble discriminant function

non-negative “votes”

base learner



- The base learners are typically simple “decision stumps”, i.e., linear classifiers based on one coordinate



# Ensemble learning

- We can view the ensemble learning problem as a coordinate selection problem

$$h_m(\underline{x}) = \begin{bmatrix} 0 \\ \cdots \\ \alpha_1 \\ \cdots \\ \alpha_m \\ 0 \\ \cdots \end{bmatrix} \cdot \begin{bmatrix} 0 \\ \cdots \\ h(\underline{x};\underline{\theta}_1) \\ \cdots \\ h(\underline{x};\underline{\theta}_m) \\ 0 \\ \cdots \end{bmatrix}$$

parameters or "votes" | coordinates indexed by $\underline{\theta}$



- The problem of finding the best "coordinates" to include corresponds to finding the parameters of the base learners (out of an uncountable set)

# Estimation criterion

- In principle, we can estimate the ensemble

$$h_m(\underline{x}) = \alpha_1 h(\underline{x}; \underline{\theta}_1) + \ldots + \alpha_m h(\underline{x}; \underline{\theta}_m)$$

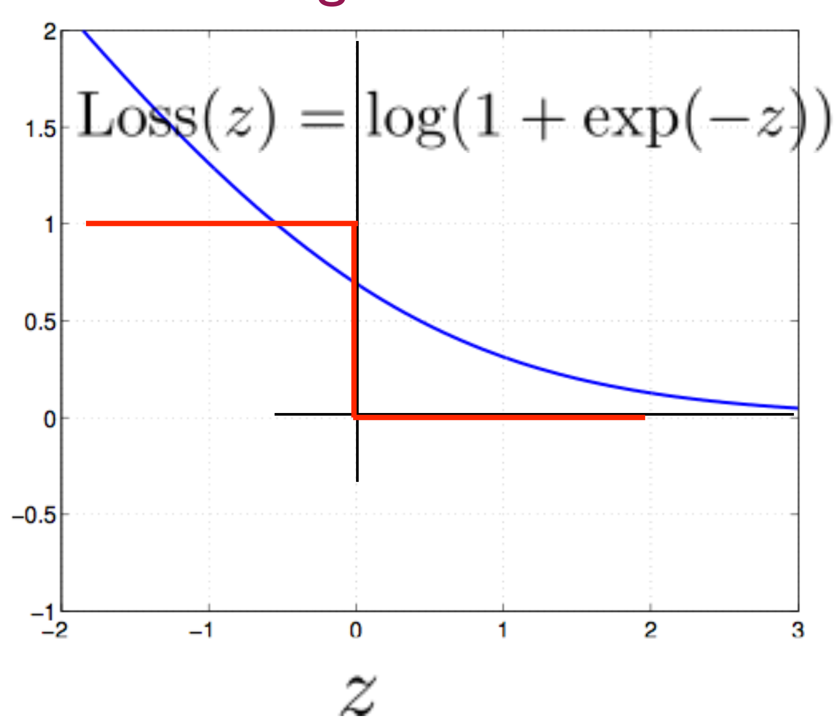
by minimizing the training loss

$$\sum_{t=1}^{n} \text{Loss}(y_t h_m(\underline{x}_t))$$

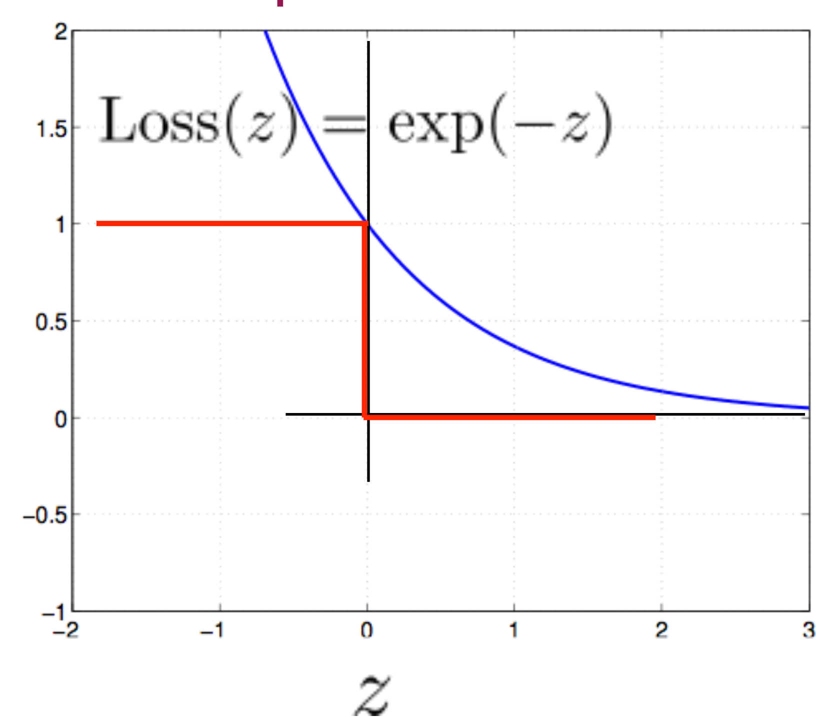


with respect to the parameters in the ensemble

logistic loss




exponential loss



# Estimation criterion

- In principle, we can estimate the ensemble

$$h_m(\underline{x}) = \alpha_1 h(\underline{x};\underline{\theta}_1) + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$

by minimizing the training loss

$$\sum_{t=1}^{n} \mathrm{Loss}(y_t h_m(\underline{x}_t))$$



with respect to the parameters in the ensemble



- This is a hard problem to solve jointly but we can add base learners sequentially (cf. forward fitting)

# Estimation criterion

- In principle, we can estimate the ensemble

$$h_m(\underline{x}) = \alpha_1 h(\underline{x}; \underline{\theta}_1) + \ldots + \alpha_m h(\underline{x}; \underline{\theta}_m)$$

by minimizing the training loss

$$\sum_{t=1}^{n} \text{Loss}(y_t h_m(\underline{x}_t))$$



with respect to the parameters in the ensemble



- This is a hard problem to solve jointly but we can add base learners sequentially (cf. forward fitting)

Fix $h_{m-1}(\underline{x})$

Find $\alpha_m$ and $\underline{\theta}_m$ that minimize

$$J(\alpha_m, \underline{\theta}_m) = \sum_{t=1}^{n} \text{Loss}\Big( \overbrace{\underbrace{y_t h_{m-1}(\underline{x}_t)}_{\text{fixed}} + \alpha_m y_t h(\underline{x}_t; \underline{\theta}_m)}^{y_t h_m(\underline{x}_t)} \Big)$$

# Myopic forward-fitting

$$J(\alpha_m, \underline{\theta}_m) = \sum_{t=1}^{n} \text{Loss}\Big( \overbrace{\underbrace{y_t h_{m-1}(\underline{x}_t)}_{\text{fixed}} + \alpha_m y_t h(\underline{x}_t; \underline{\theta}_m)}^{y_t h_m(\underline{x}_t)} \Big)$$

- Out of all $\underline{\theta}_m$ we wish to select one that minimizes the derivative of the loss at $\alpha_m = 0$ (has the most negative derivative at zero)



$$\left.\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\right|_{\alpha_m=0} = \sum_{t=1}^{n} \left[ \frac{\partial}{\partial z} \text{Loss}(z) \Big|_{z=y_t h_{m-1}(\underline{x}_t)} \right] y_t h(\underline{x}_t; \underline{\theta}_m)$$

$$= \sum_{t=1}^{n} \underbrace{\text{DLoss}\big( y_t h_{m-1}(\underline{x}_t) \big)}_{\text{fixed weights on training examples}} y_t h(\underline{x}_t; \underline{\theta}_m)$$

# Weights on training examples

$$\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\bigg|_{\alpha_m=0} = \sum_{t=1}^{n} \left[ \frac{\partial}{\partial z} \text{Loss}(z) \big|_{z=y_t h_{m-1}(\underline{x}_t)} \right] y_t h(\underline{x}_t; \underline{\theta}_m)$$

$$= \sum_{t=1}^{n} \underbrace{\text{DLoss}\big( y_t h_{m-1}(\underline{x}_t) \big)}_{\text{these derivatives are negative}} y_t h(\underline{x}_t; \underline{\theta}_m)$$

# Weights on training examples

$$\left.\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\right|_{\alpha_m=0} = \sum_{t=1}^{n} \left[ \frac{\partial}{\partial z} \text{Loss}(z)\Big|_{z=y_t h_{m-1}(\underline{x}_t)} \right] y_t h(\underline{x}_t; \underline{\theta}_m)$$

$$= \sum_{t=1}^{n} \underbrace{\text{DLoss}\big(\, y_t h_{m-1}(\underline{x}_t) \,\big)}_{\text{these derivatives are negative}} y_t h(\underline{x}_t; \underline{\theta}_m)$$



$$= \sum_{t=1}^{n} W_t \, \underbrace{(-y_t) h(\underline{x}_t; \underline{\theta}_m)}_{\text{error (agreement with the opposite label)}}$$





positive weight

error (agreement with the opposite label)

# Weights on training examples

$$\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\bigg|_{\alpha_m=0} = \sum_{t=1}^{n} \left[ \frac{\partial}{\partial z} \mathrm{Loss}(z) \big|_{z=y_t h_{m-1}(\underline{x}_t)} \right] y_t h(\underline{x}_t; \underline{\theta}_m)$$

$$= \sum_{t=1}^{n} \mathrm{DLoss}\big( y_t h_{m-1}(\underline{x}_t) \big)\, y_t h(\underline{x}_t; \underline{\theta}_m)$$

these derivatives are negative



$$= \sum_{t=1}^{n} W_t \, (-y_t) h(\underline{x}_t; \underline{\theta}_m)$$





positive weight

error (agreement with the opposite label)

Logistic loss: $W_t = g\big( -y_t h_{m-1}(\underline{x}_t) \big), \;\; g(z) = (1+\exp(-z))^{-1}$

Exponential loss: $W_t = \exp\big( -y_t h_{m-1}(\underline{x}_t) \big)$

# Weights on training examples

$$
\begin{aligned}
\left.\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\right|_{\alpha_m=0} &= \sum_{t=1}^{n} \left[ \frac{\partial}{\partial z} \mathrm{Loss}(z) \big|_{z=y_t h_{m-1}(\underline{x}_t)} \right] y_t h(\underline{x}_t; \underline{\theta}_m) \\
&= \sum_{t=1}^{n} \underbrace{\mathrm{DLoss}\big(\, y_t h_{m-1}(\underline{x}_t)\,\big)}_{\text{these derivatives are negative}}\, y_t h(\underline{x}_t; \underline{\theta}_m) \\
&= \sum_{t=1}^{n} \underbrace{W_t}_{\text{positive weight}} \underbrace{(-y_t) h(\underline{x}_t; \underline{\theta}_m)}_{\text{error (agreement with the opposite label)}}
\end{aligned}
$$



$$\text{Logistic loss:} \quad W_t = g\big(\, - y_t h_{m-1}(\underline{x}_t)\,\big), \;\; g(z) = (1 + \exp(-z))^{-1}$$

$$\text{Exponential loss:} \quad W_t = \exp\big(\, - y_t h_{m-1}(\underline{x}_t)\,\big)$$

- We can always normalize the weights without affecting the choice of $\underline{\theta}_m$

$$W_t \leftarrow \frac{W_t}{\sum_{i=1}^{n} W_i}$$

# General boosting algorithm

- We use a myopic forward-fitting method to estimate

$$h_m(\underline{x}) = \alpha_1 h(\underline{x};\underline{\theta}_1) + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$

Step 0: $h_0(\underline{x}) = 0,\ W_t = 1/n, t = 1, \ldots, n$

# General boosting algorithm

- We use a myopic forward-fitting method to estimate

$$h_m(\underline{x}) = \alpha_1 h(\underline{x};\underline{\theta}_1) + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$

Step 0: $h_0(\underline{x}) = 0,\ W_t = 1/n, t = 1, \ldots, n$

Step 1: Find $\hat{\underline{\theta}}_m$ that minimizes the weighted error



$$\left.\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\right|_{\alpha_m=0} = \sum_{t=1}^{n} W_t\,(-y_t) h(\underline{x}_t;\underline{\theta}_m)$$

# General boosting algorithm

- We use a myopic forward-fitting method to estimate

$$h_m(\underline{x}) = \alpha_1 h(\underline{x};\underline{\theta}_1) + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$

Step 0: $h_0(\underline{x}) = 0, \; W_t = 1/n, t = 1, \ldots, n$

Step 1: Find $\underline{\hat{\theta}}_m$ that minimizes the weighted error





$$\left.\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\right|_{\alpha_m = 0} = \sum_{t=1}^{n} W_t \, (-y_t) h(\underline{x}_t; \underline{\theta}_m)$$



- weights correspond to derivatives of the loss function
- the weight is large if the example is not classified correctly by the ensemble we have so far
- finding the parameters that minimize the weighted error is easily solved, e.g., for decision stumps

# General boosting algorithm

- We use a myopic forward-fitting method to estimate

$$h_m(\underline{x}) = \alpha_1 h(\underline{x};\underline{\theta}_1) + \ldots + \alpha_m h(\underline{x};\underline{\theta}_m)$$

Step 0: $h_0(\underline{x}) = 0,\ W_t = 1/n, t = 1,\ldots,n$

Step 1: Find $\hat{\underline{\theta}}_m$ that minimizes the weighted error



$$\left.\frac{\partial J(\alpha_m,\underline{\theta}_m)}{\partial \alpha_m}\right|_{\alpha_m=0} = \sum_{t=1}^{n} W_t\,(-y_t)h(\underline{x}_t;\underline{\theta}_m)$$



Step 2: Find $\hat{\alpha}_m$ that minimizes

$$J(\alpha_m,\hat{\underline{\theta}}_m) = \sum_{t=1}^{n} \text{Loss}\big(\underbrace{y_t h_{m-1}(\underline{x}_t)}_{\text{fixed}} + \alpha_m \underbrace{y_t h(\underline{x}_t;\hat{\underline{\theta}}_m)}_{\text{fixed}}\big)$$

- this is a 1-dimensional convex problem that can be solved easily.

# General boosting algorithm

- We use a myopic forward-fitting method to estimate

$$h_m(\underline{x}) = \alpha_1 h(\underline{x}; \underline{\theta}_1) + \ldots + \alpha_m h(\underline{x}; \underline{\theta}_m)$$

Step 0: $h_0(\underline{x}) = 0,\ W_t = 1/n, t = 1, \ldots, n$

Step 1: Find $\hat{\underline{\theta}}_m$ that minimizes the weighted error



$$\left.\frac{\partial J(\alpha_m, \underline{\theta}_m)}{\partial \alpha_m}\right|_{\alpha_m=0} = \sum_{t=1}^{n} W_t\,(-y_t) h(\underline{x}_t; \underline{\theta}_m)$$



Step 2: Find $\hat{\alpha}_m$ that minimizes

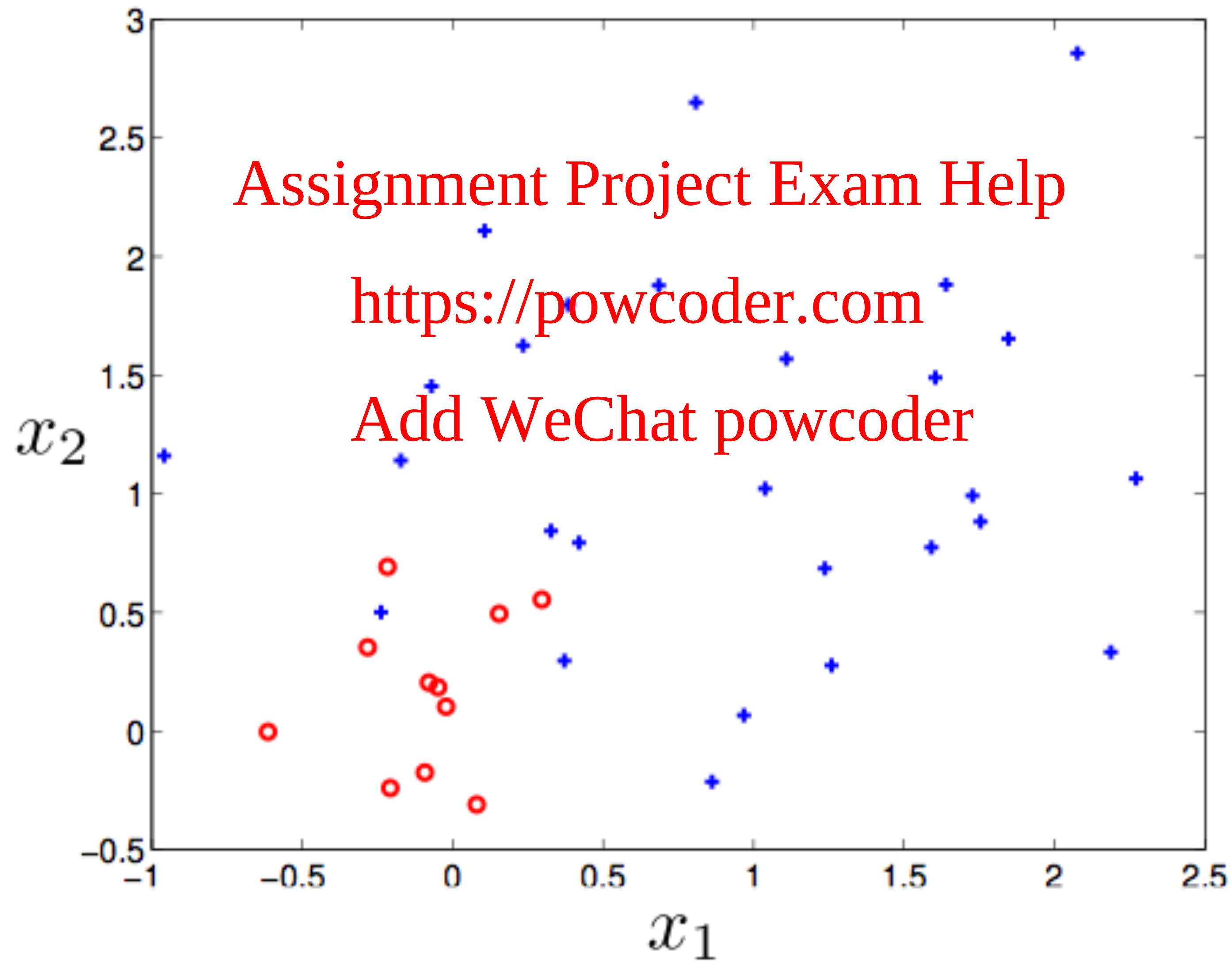
$$J(\alpha_m, \hat{\underline{\theta}}_m) = \sum_{t=1}^{n} \text{Loss}\Big(\underbrace{y_t h_{m-1}(\underline{x}_t)}_{\text{fixed}} + \alpha_m \underbrace{y_t h(\underline{x}_t; \hat{\underline{\theta}}_m)}_{\text{fixed}}\Big)$$

Step 3: Update example weights

$$W_t = -\text{DLoss}\Big(\underbrace{y_t h_{m-1}(\underline{x}_t) + \hat{\alpha}_m y_t h(\underline{x}_t; \hat{\underline{\theta}}_m)}_{y_t h_m(\underline{x}_t)}\Big)$$

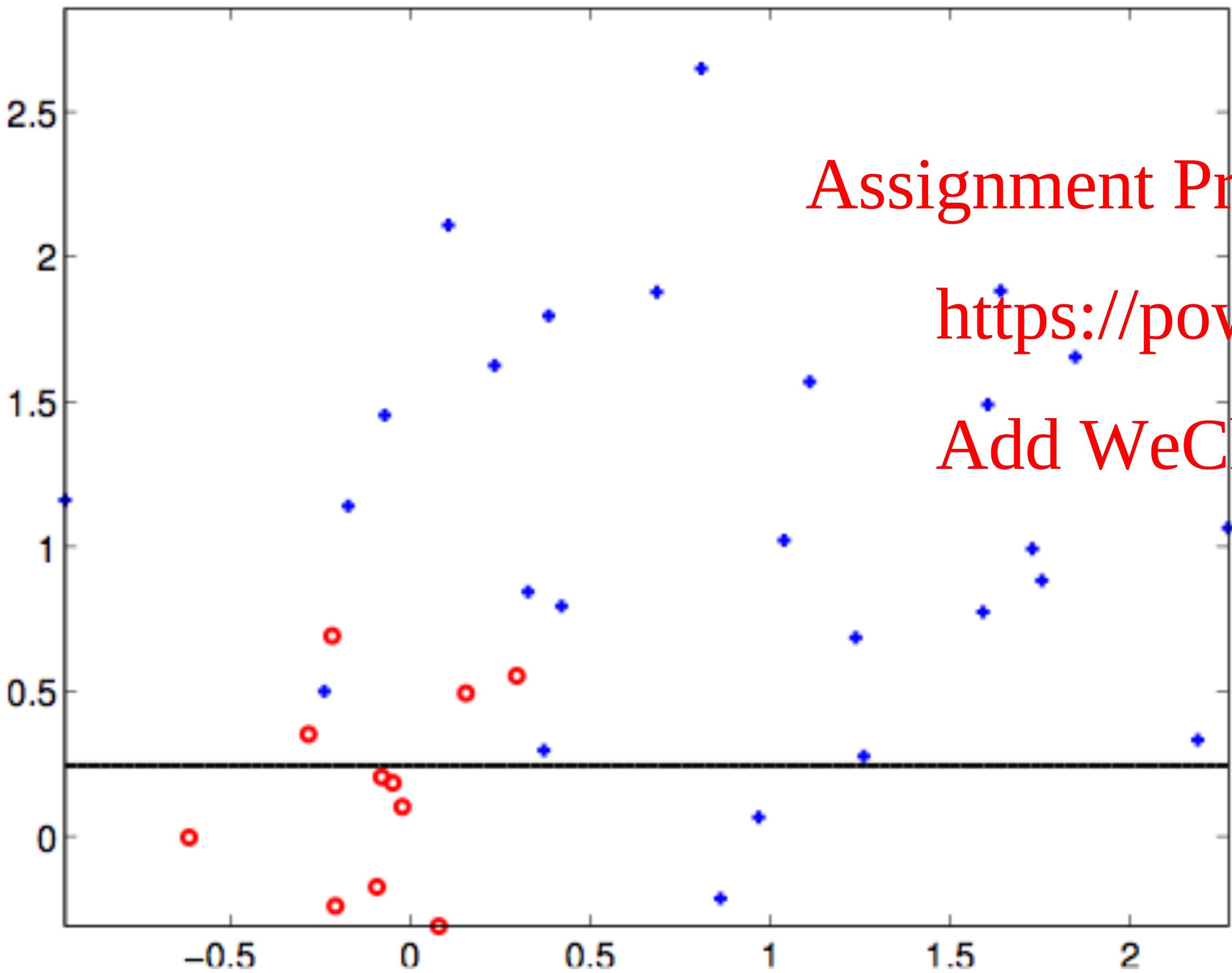
# Boosting example

Logistic loss $\text{Loss}(z) = \log(1 + \exp(-z))$

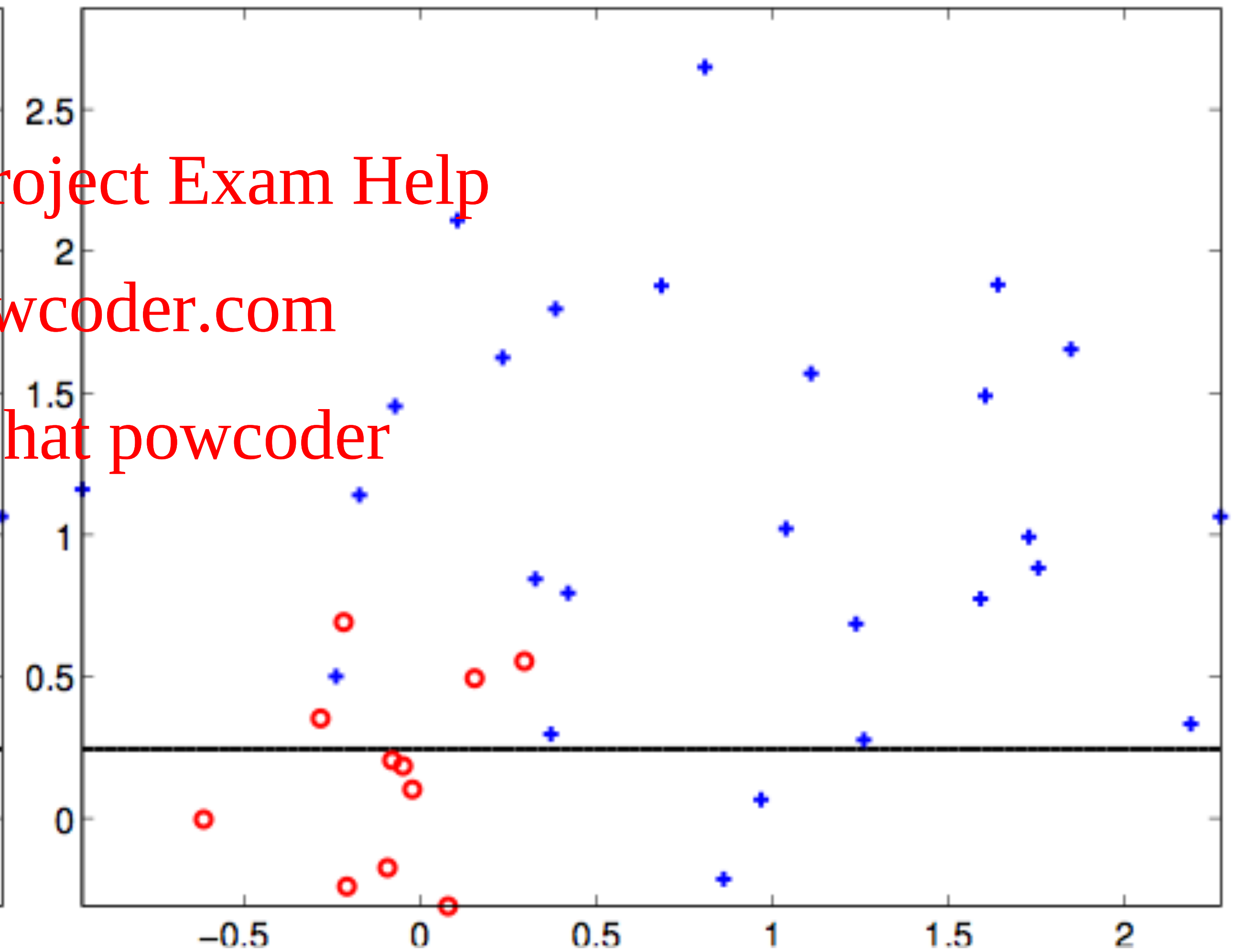


# Boosting example

$$h(\underline{x};\hat{\underline{\theta}}_1)$$

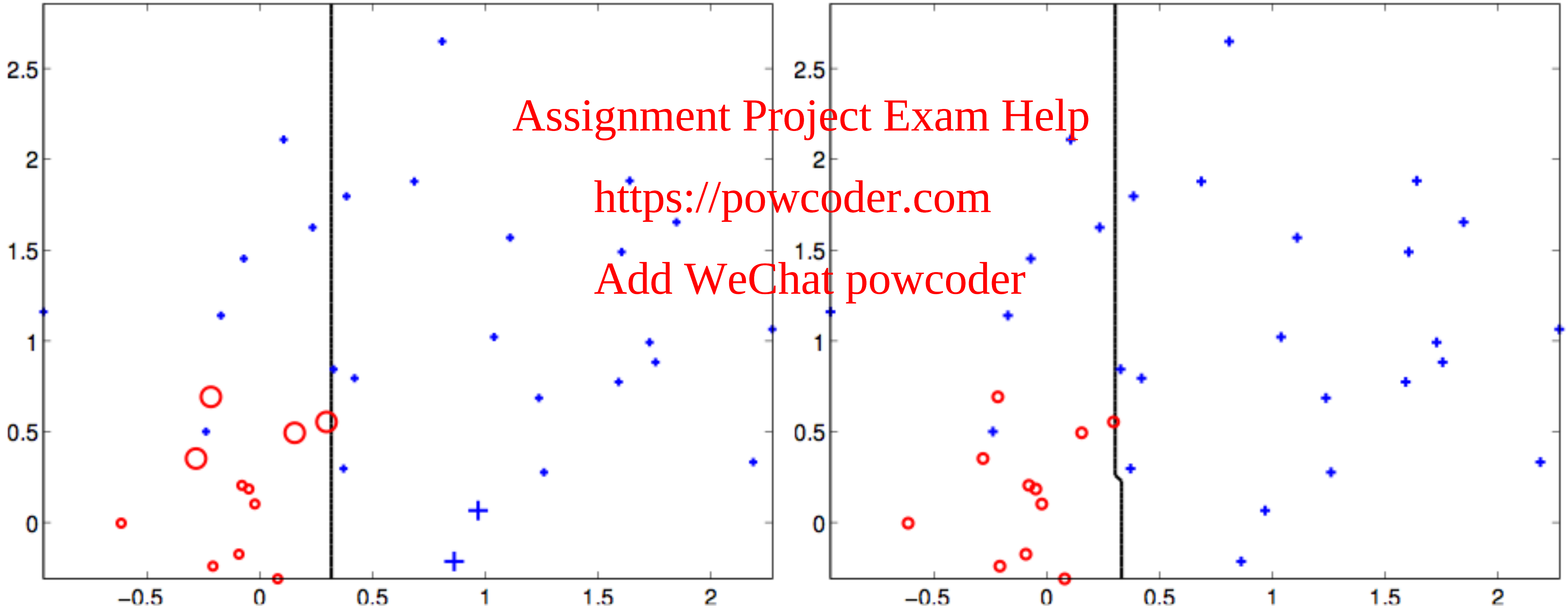



$$h_1(\underline{x}) = \hat{\alpha}_1 h(\underline{x};\hat{\underline{\theta}}_1)$$



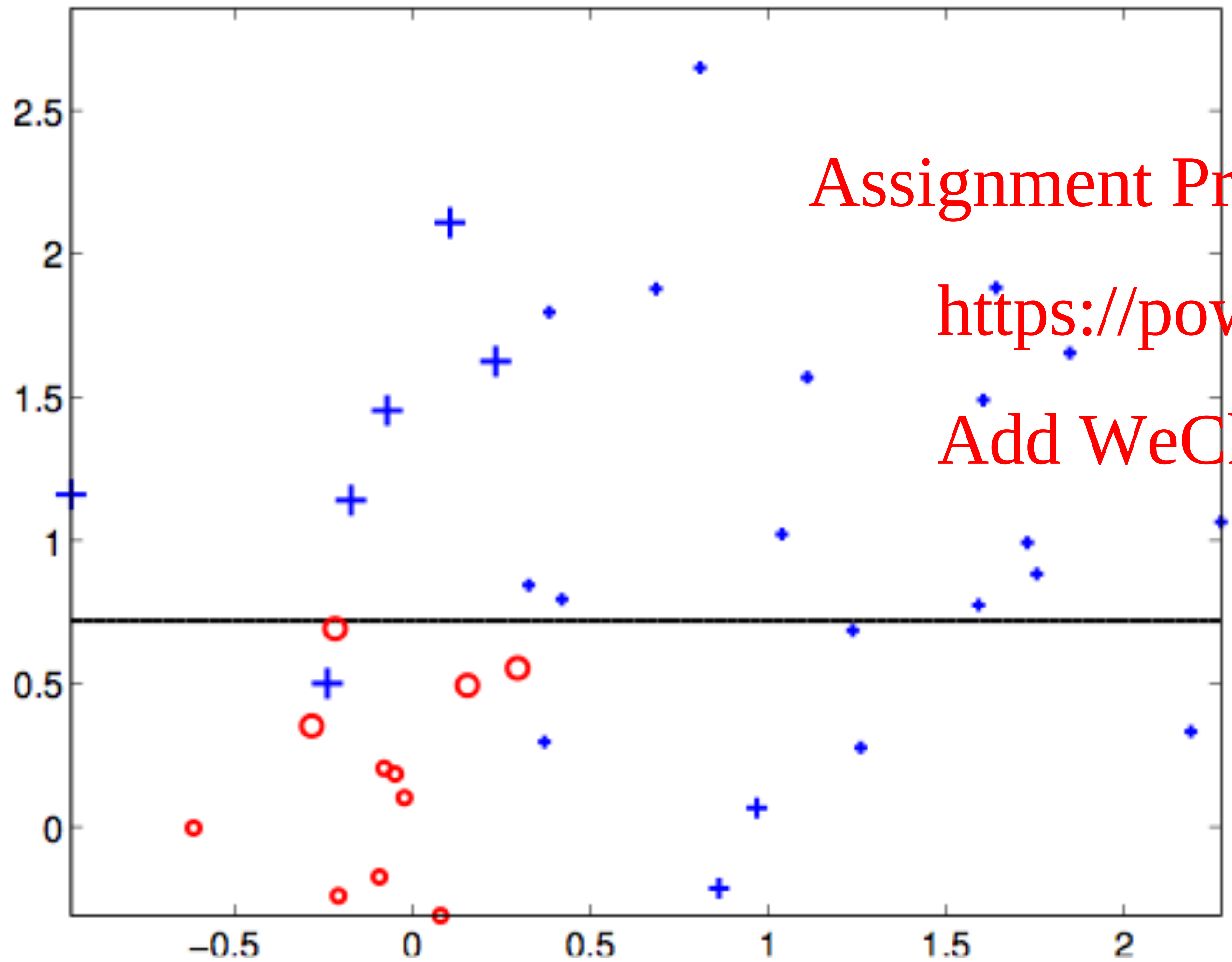
# Boosting example

$$h(\underline{x};\hat{\underline{\theta}}_2) \qquad\qquad h_2(\underline{x}) = \hat{\alpha}_1 h(\underline{x};\hat{\underline{\theta}}_1) + \hat{\alpha}_2 h(\underline{x};\hat{\underline{\theta}}_2)$$

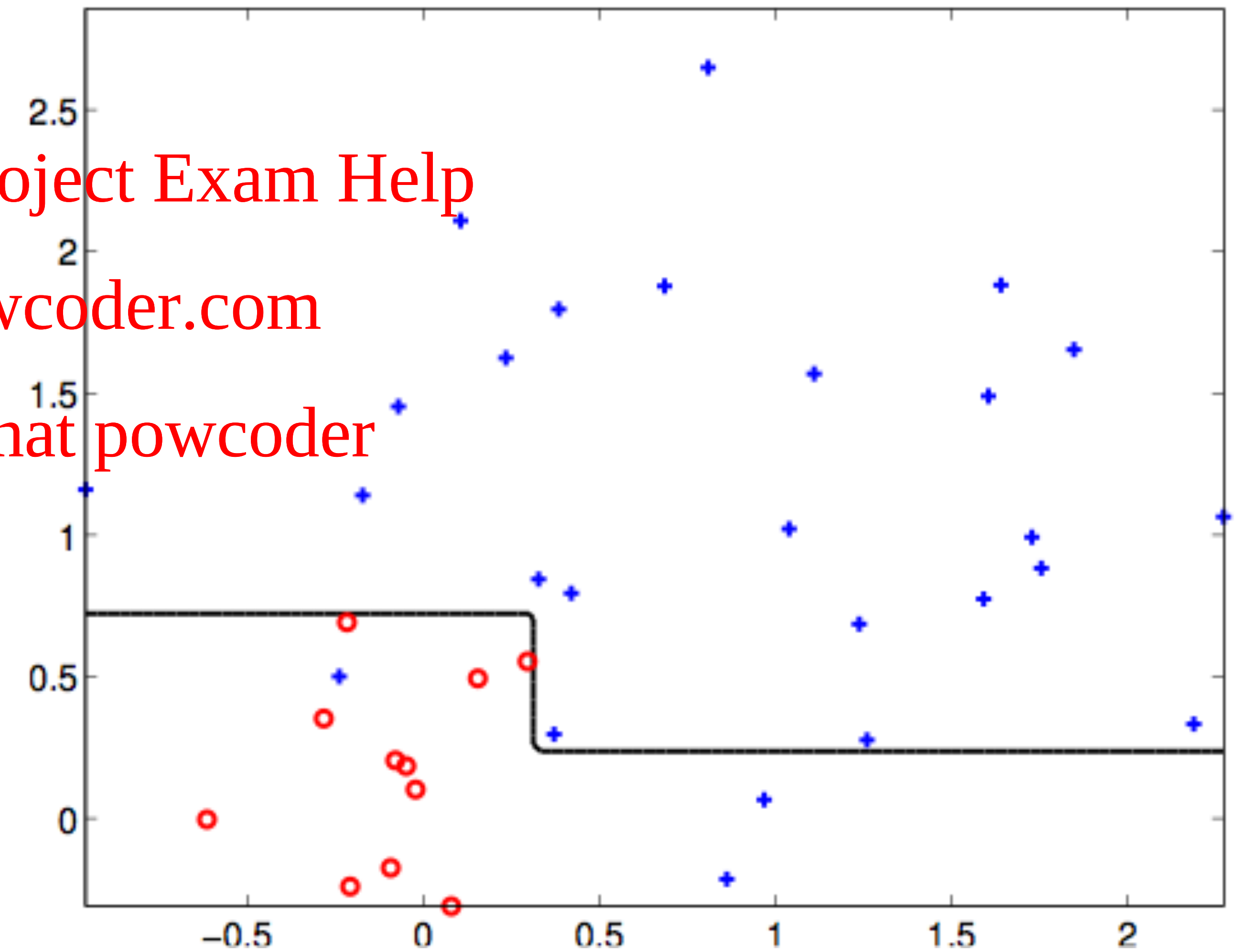


# Boosting example

$h(\underline{x}; \underline{\hat{\theta}}_3)$

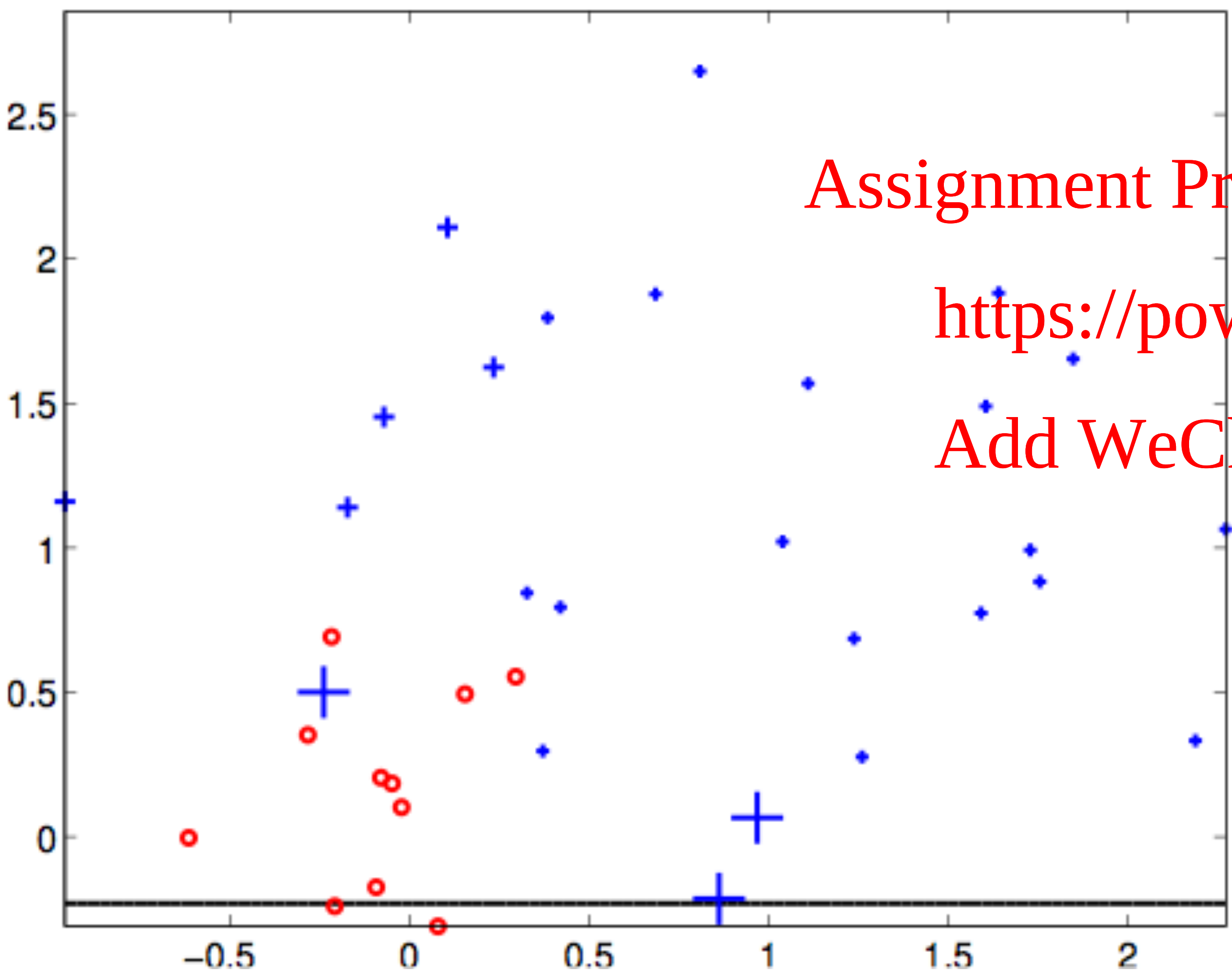



$h_3(\underline{x}) = \hat{\alpha}_1 h(\underline{x}; \underline{\hat{\theta}}_1) + \cdots + \hat{\alpha}_3 h(\underline{x}; \underline{\hat{\theta}}_3)$

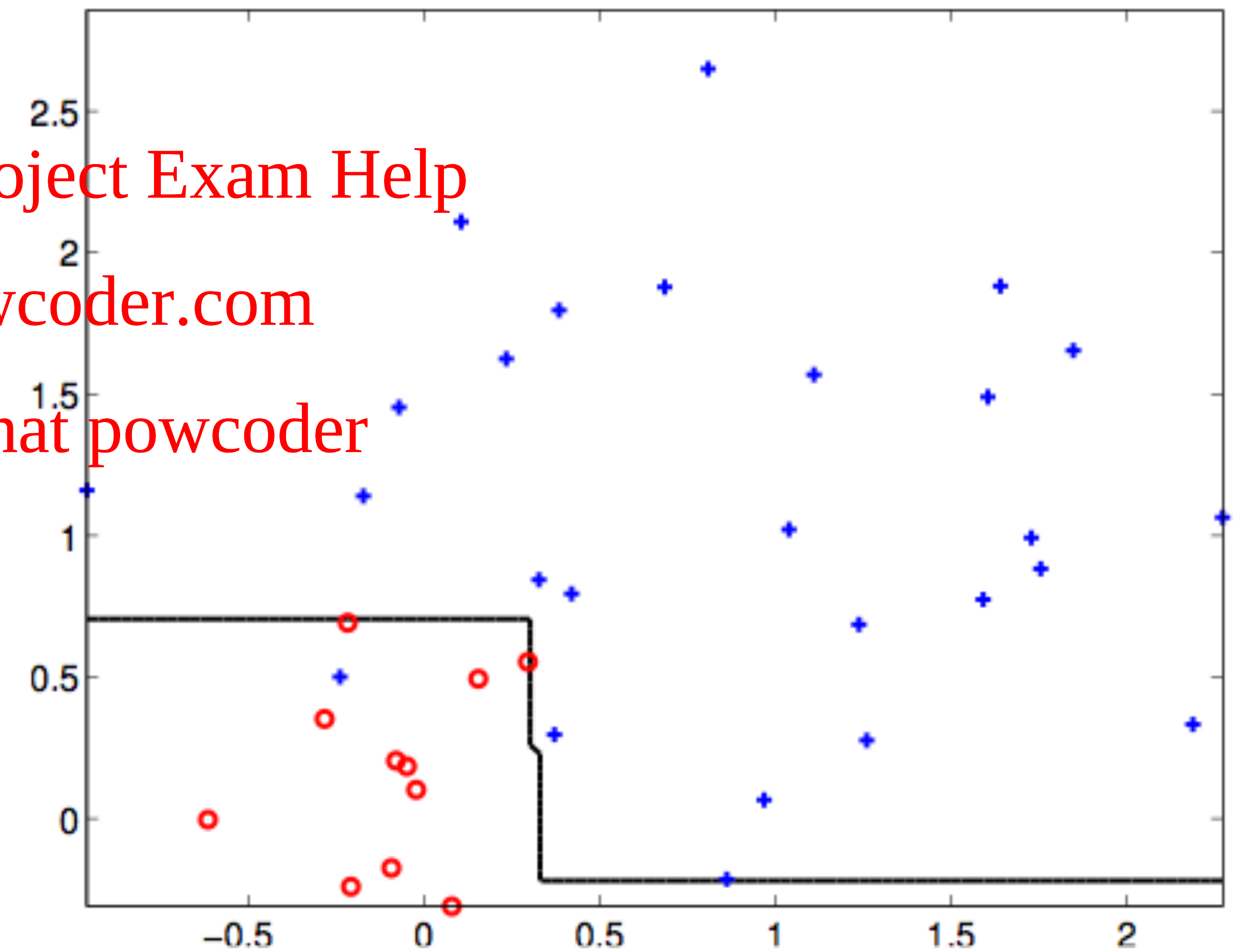


# Boosting example

$h(\underline{x}; \underline{\hat{\theta}}_4)$




$h_4(\underline{x}) = \hat{\alpha}_1 h(\underline{x}; \underline{\hat{\theta}}_1) + \cdots + \hat{\alpha}_4 h(\underline{x}; \underline{\hat{\theta}}_4)$



# Trying the same base learner again

- $\hat{\alpha}_m$ is set to minimize the loss given the base learner

$$J(\alpha_m, \underline{\hat{\theta}}_m) = \sum_{t=1}^{n} \text{Loss}\big(\, y_t h_{m-1}(\underline{x}_t) + \alpha_m y_t h(\underline{x}_t; \underline{\hat{\theta}}_m)\,\big)$$

- At the optimum value,



$$\frac{\partial J(\alpha_m, \underline{\hat{\theta}}_m)}{\partial \alpha_m}\Big|_{\alpha_m = \hat{\alpha}_m} =$$

$$\sum_{t=1}^{n} \underbrace{\text{DLoss}\big(\, y_t h_{m-1}(\underline{x}_t) + \hat{\alpha}_m y_t h(\underline{x}_t; \underline{\hat{\theta}}_m)\,\big)}_{\text{updated weights (up to normalization and overall sign)}} y_t h(\underline{x}_t; \underline{\hat{\theta}}_m) \quad = \quad 0$$

- Thus the current base learner is useless at the next iteration (but may be chosen again later on)

# Ensemble training error

- The boosting algorithm decreases the training loss

$$\sum_{t=1}^{n} \text{Loss}(\, y_t h_m(\underline{x}_t)\,)$$

monotonically while the base learners remain effective against the weighted error (derivative is not zero)



- For any non-increasing loss function,

$$\sum_{t=1}^{n} I_{[y_t h_m(\underline{x}_t) \le 0]} \le \frac{1}{\text{Loss}(0)} \sum_{t=1}^{n} \text{Loss}(\, y_t h_m(\underline{x}_t)\,)$$

Thus we have a monotonically decreasing upper bound on the 0-1 training error (classification error)

# Ensemble training error