Boosting

Lecturer: Sang Hwa Lee

What is AdaBoost ?

❖ Concept of AdaBoosting

AdaBoost is an algorithm for constructing a "strong" classifier as linear combination of "simple" "weak" classifiers $h_t(x)$: $X \rightarrow \{-1, +1\}$.

$$f(x) = \sum_{t=1}^{T} \alpha_t h_t(x)$$

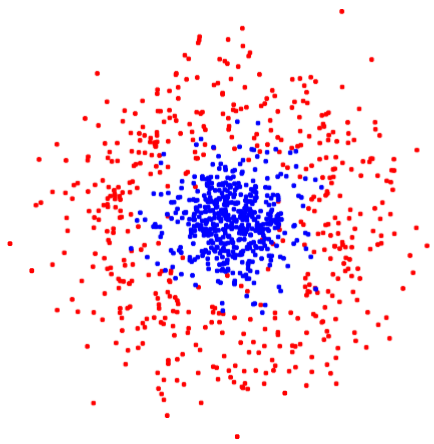
❖ Terminology

$h_t(x)$ … "weak" or basis classifier, hypothesis, "feature"

$H(x) = sign(f(x))$ … "strong" or final classifier/hypothesis

AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in X$, $y_i \in \{-1, +1\}$

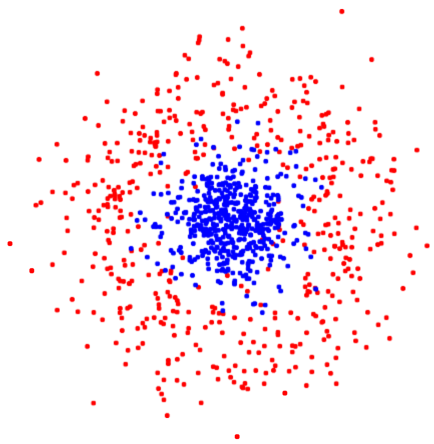


AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in X$, $y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:



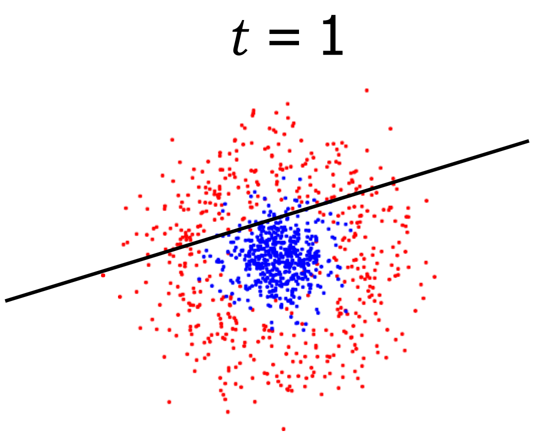
AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m); x_i \in X, y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$



AdaBoost Algorithm

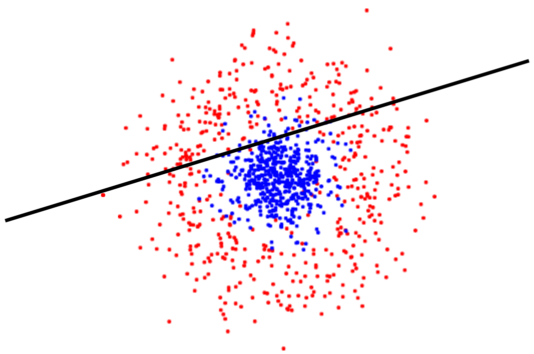
Given: $(x_1, y_1), \ldots, (x_m, y_m); x_i \in X, y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

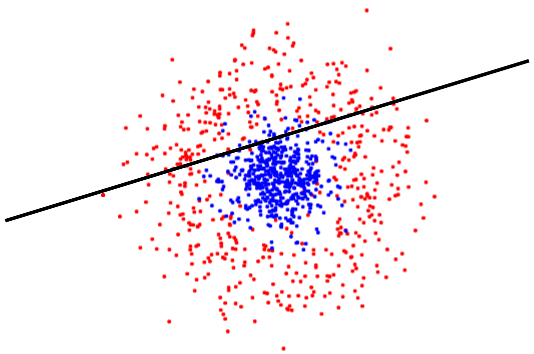
- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop

$t = 1$



AdaBoost Algorithm

$t = 1$



Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in X$, $y_i \in \{-1, +1\}$

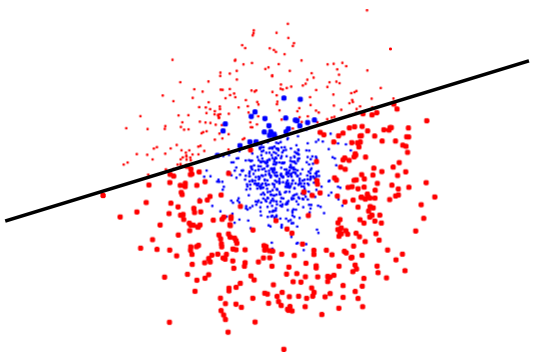
Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$

AdaBoost Algorithm

$t = 1$



Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in X$, $y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i) [\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2} \log(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

$$D_{t+1}(i) = \frac{D_t(i) exp(-a_t y_i h_t(x_i))}{Z_t}$$

where Z_t is normalisation factor

AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m);\ x_i \in X,\ y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

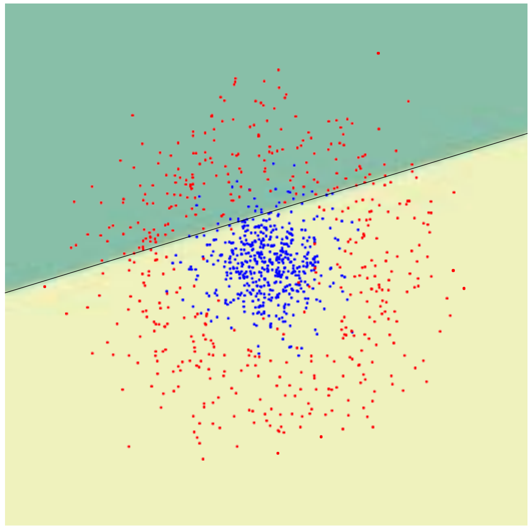
$$D_{t+1}(i) = \frac{D_t(i)exp(-a_t y_i h_t(x_i))}{Z_t}$$

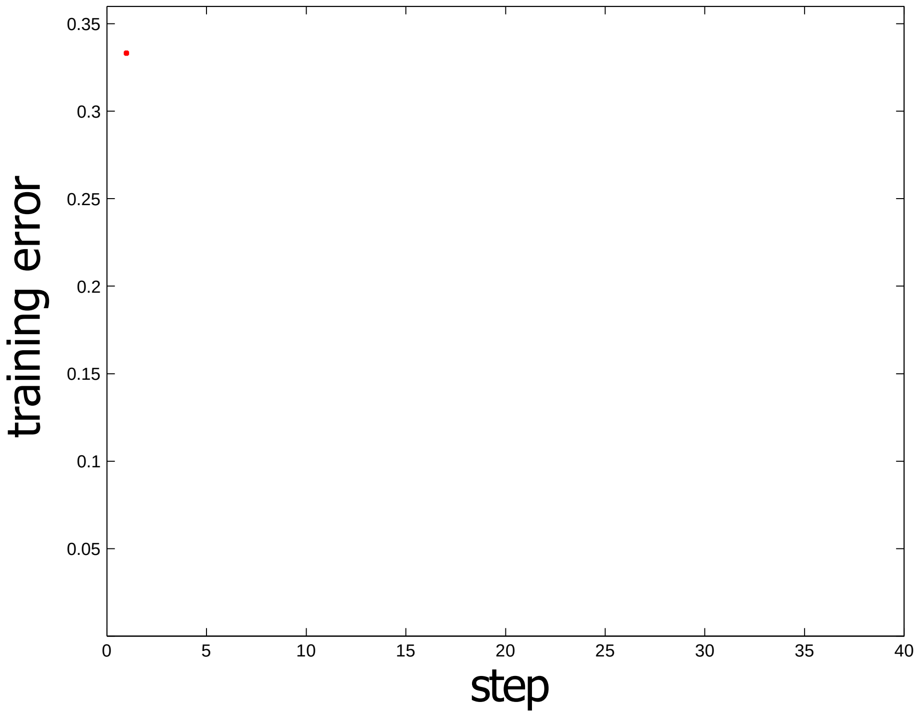
where Z_t is normalisation factor

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 1$





AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m);\ x_i \in X,\ y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
- Update

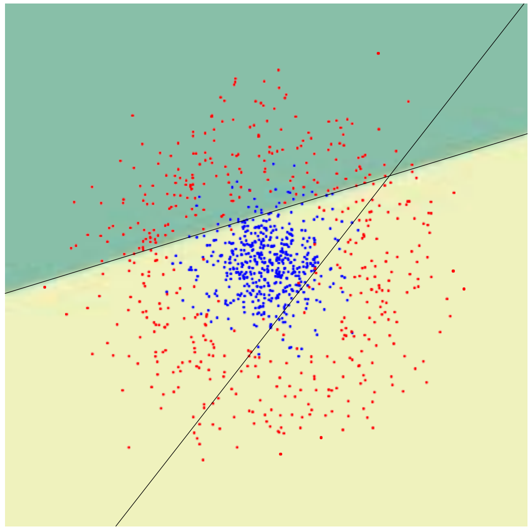
$$D_{t+1}(i) = \frac{D_t(i)exp(-a_t y_i h_t(x_i))}{Z_t}$$

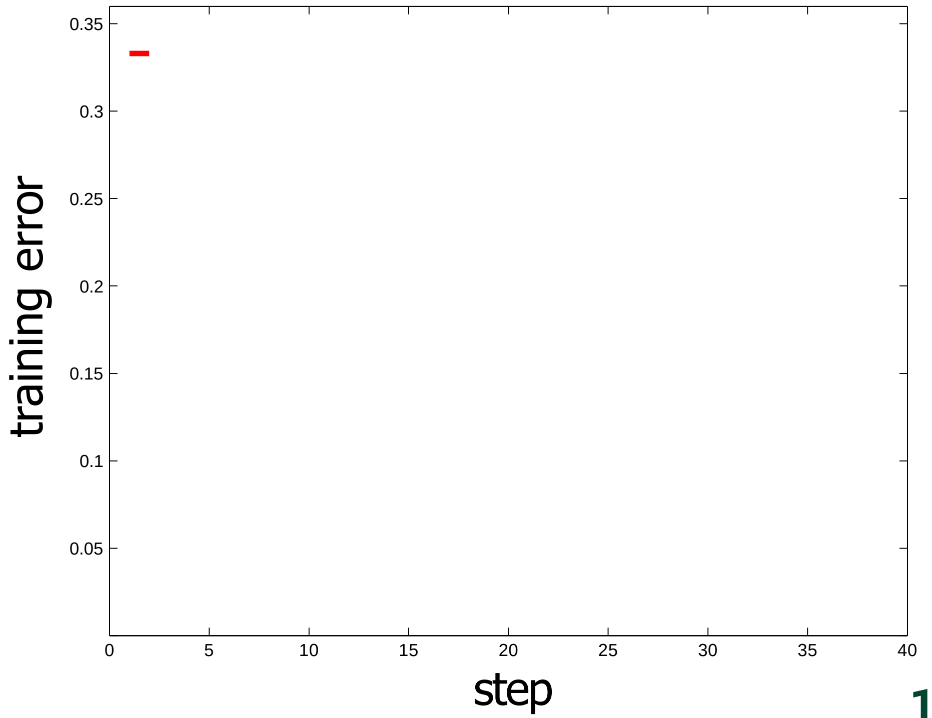
where Z_t is normalisation factor

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 2$





AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m); x_i \in X, y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

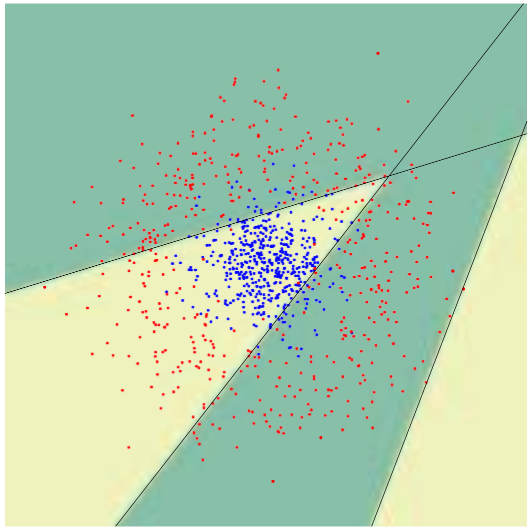
$$D_{t+1}(i) = \frac{D_t(i) exp(-a_t y_i h_t(x_i))}{Z_t}$$

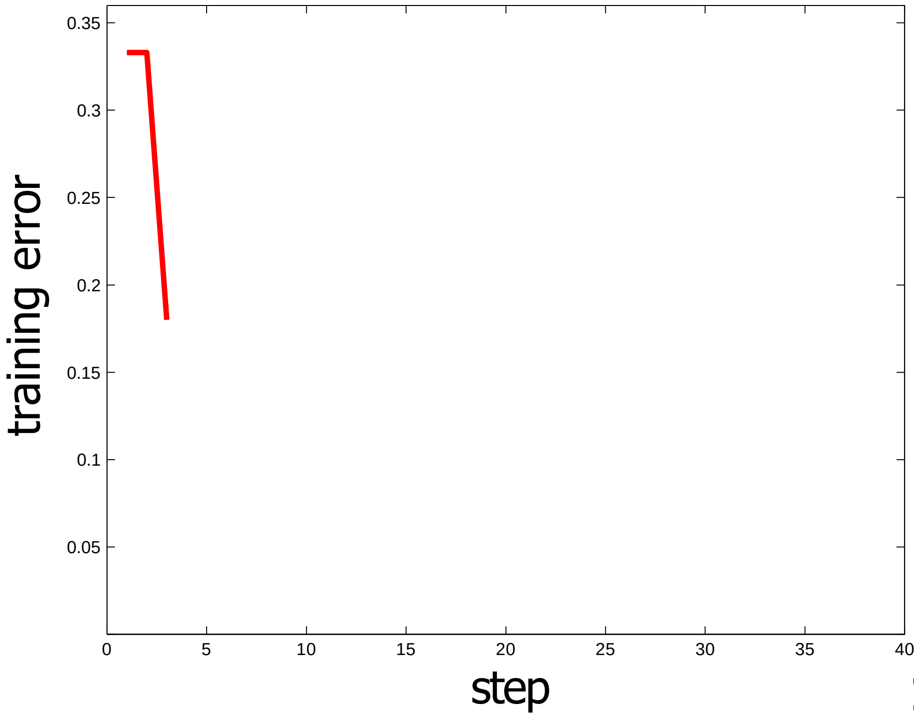
where Z_t is normalisation factor

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 3$





AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in X$, $y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

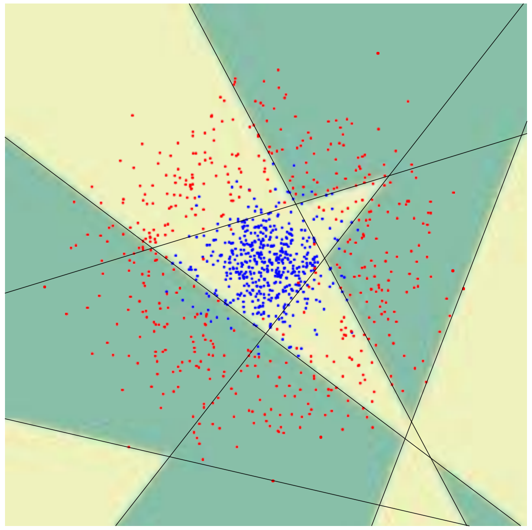
$$D_{t+1}(i) = \frac{D_t(i) exp(-a_t y_i h_t(x_i))}{Z_t}$$

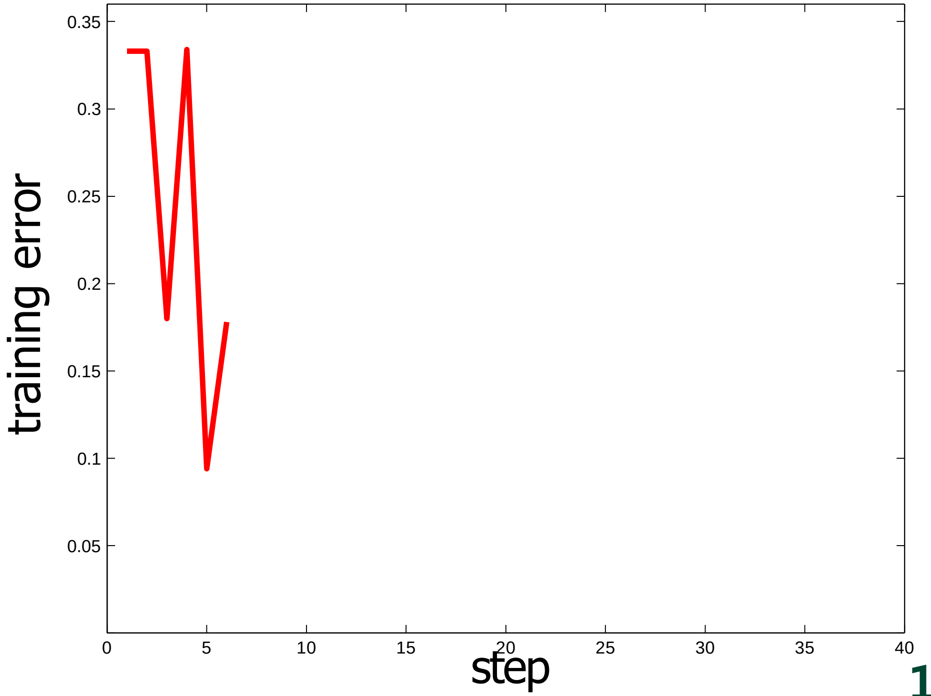
where Z_t is normalisation factor

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 6$





AdaBoost Algorithm

Given: $(x_1, y_1), \ldots, (x_m, y_m)$; $x_i \in X$, $y_i \in \{-1, +1\}$

Initialise weights $D_1(i) = 1/m$

For $t = 1, \ldots, T$:

- Find $h_t = \arg\min_{h_j \in \mathcal{H}} \epsilon_j = \sum_{i=1}^{m} D_t(i)[\![y_i \neq h_j(x_i)]\!]$
- If $\epsilon_t \geq 1/2$ then stop
- Set $\alpha_t = \frac{1}{2}\log(\frac{1-\epsilon_t}{\epsilon_t})$
- Update

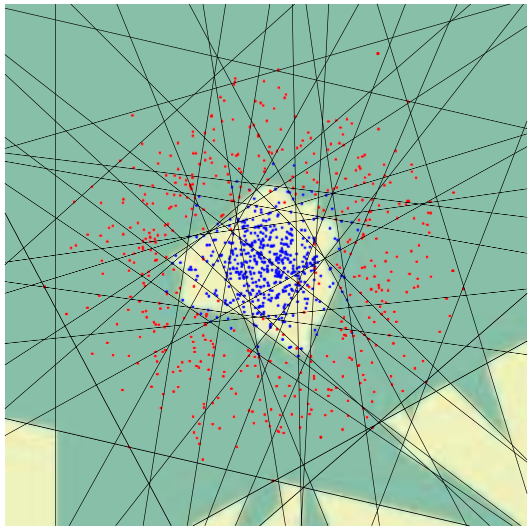
$$D_{t+1}(i) = \frac{D_t(i)exp(-a_t y_i h_t(x_i))}{Z_t}$$

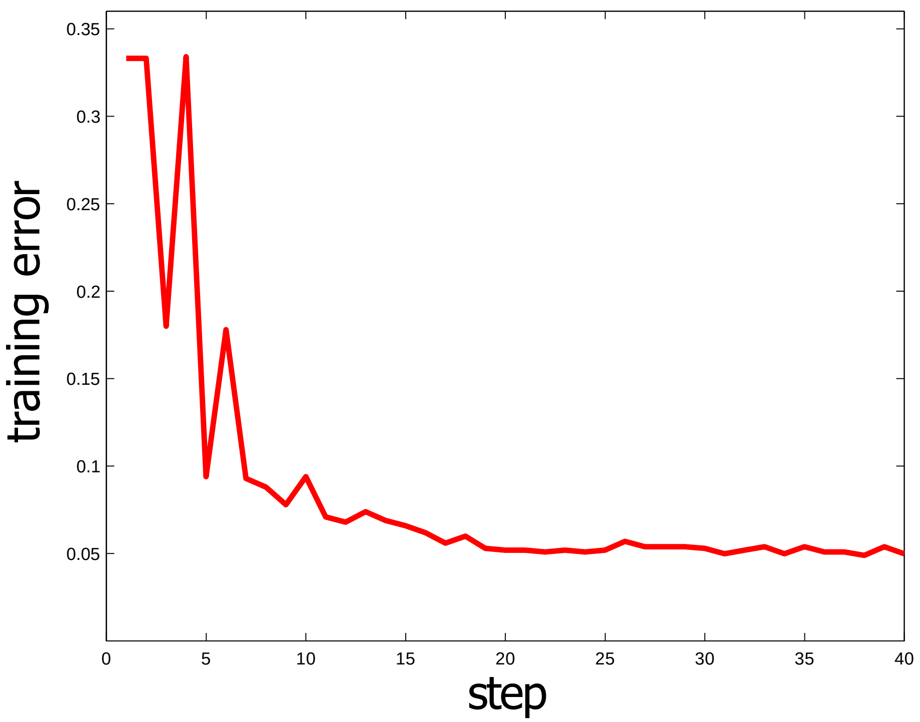
where Z_t is normalisation factor

Output the final classifier:

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right)$$

$t = 40$





Reweighting

Effect on the training set

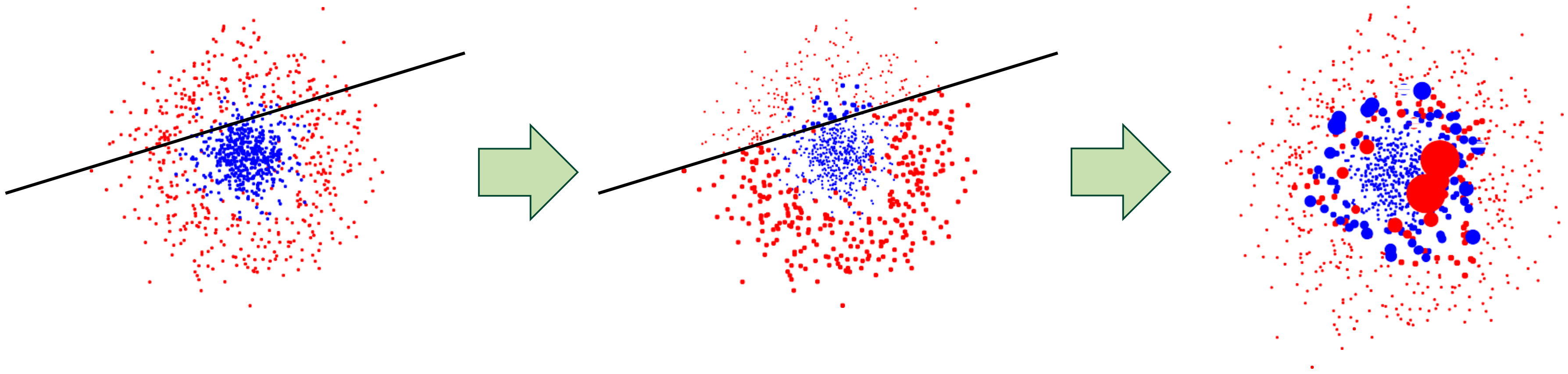
$$D_{t+1}(i) = \frac{D_t(i)exp(-a_t y_i h_t(x_i))}{Z_t}$$

$$exp(-\alpha_t y_i h_t(x_i)) \begin{cases} < 1, & y_i = h_t(x_i) \\ > 1, & y_i \neq h_t(x_i) \end{cases}$$

$\Rightarrow$ Increase (decrease) weight of wrongly (correctly) classified examples

$\Rightarrow$ The weight is the upper bound on the error of a given example

$\Rightarrow$ All information about previously selected "features" is captured in D_t



Upper Bound Theorem (1)

Theorem: The following upper bound holds on the training error of H

$$\frac{1}{m}|\{i : H(x_i) \neq y_i\}| \leq \prod_{t=1}^{T} Z_t$$

Proof: By unravelling the update rule

$$D_{T+1}(i) = \frac{D_t(i)exp(-a_t y_i h_t(x_i))}{Z_t}$$

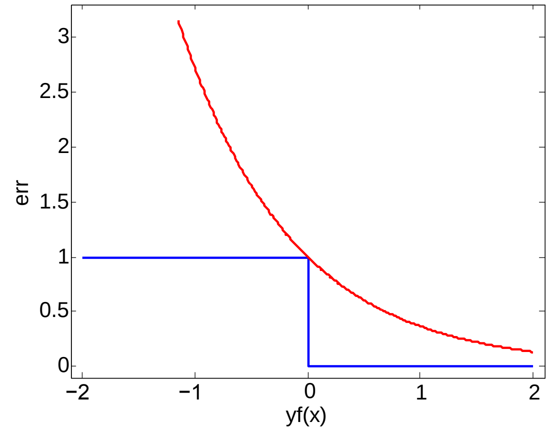
$$= \frac{exp(-\sum_t \alpha_t y_i h_t(x_i))}{m \prod_t Z_t} = \frac{exp(-y_i f(x_i))}{m \prod_t Z_t}$$

If $H(x_i) \neq y_i$ then $y_i f(x_i) \leq 0$ implying that $exp(-y_i f(x_i)) > 1$, thus

$$[\![H(x_i) \neq y_i]\!] \leq exp(-y_i f(x_i))$$

$$\frac{1}{m}\sum_i [\![H(x_i) \neq y_i]\!] \leq \frac{1}{m}\sum_i exp(-y_i f(x_i))$$

$$= \sum_i (\prod_t Z_t) D_{T+1}(i) = \prod_t Z_t$$



Upper Bound Theorem (2)

- Instead of minimising the training error, its upper bound can be minimized. This can be done by minimising Z_t in each training round by:
  - Choosing optimal h_t, and
  - Finding optimal a_t
- AdaBoost can be proved to maximise margin
- AdaBoost iteratively fits an additive logistic regression model

Choosing optimal α_t

We attempt to minimise $Z_t = \sum_i D_t(i) exp(-\alpha_t y_i h_t(x_i))$:

$$\begin{aligned}
\frac{dZ}{d\alpha} = -\sum_{i=1}^{m} D(i) y_i h(x_i) e^{-y_i \alpha_i h(x_i)} &= 0 \\
-\sum_{i:y_i = h(x_i)} D(i) e^{-\alpha} + \sum_{i:y_i \neq h(x_i)} D(i) e^{\alpha} &= 0 \\
-e^{-\alpha}(1-\epsilon) + e^{\alpha}\epsilon &= 0 \\
\alpha &= \frac{1}{2} \log \frac{1-\epsilon}{\epsilon}
\end{aligned}$$

$\Rightarrow$ The minimisator of the upper bound is $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$

Boosting Variations and Applications

Variations of Boosting

- **Logitboost, Gentleboost, Floatboost, Klboost,...**
  - How to update the weights
  - How to combine the weak classifiers

- **Example: Gentleboost**
  - Update: $f_m(x) = P(y=1 \mid x) - P(y=0 \mid x)$

    Instead of likelihood ratio

$$f_m(x) = \frac{1}{2} \log \frac{P_w(y=1|x)}{P_w(y=-1|x)}$$

KLBoost

- Boost using Kullbeck-Leibler distance
  - Distnace between two pdf

$$D(P \| Q) = \sum_x P(x) \log\left(\frac{P(x)}{Q(x)}\right)$$

- Weak classifier
  - Vector(line) projection (inner product)
- KL Analysis
  - Weak classifier selection
  - Feature vector that maximizes KL distance between two classes

KLBoost Procedure (I)

- Given learning set, and initial weights

$$\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N), \text{ where } x_i \in \mathbf{R}^d,\ y_i = \{+1, -1\}\} \qquad \omega_i^{(0)} = \frac{1}{N}$$

- Find an optimal feature (projection vector)

$$\phi_k^* = \arg\max_{\phi} KL(\phi) = \int \left[h_k^+(\phi^T x) - h_k^-(\phi^T x) \right] \log \frac{h_k^+(\phi^T x)}{h_k^-(\phi^T x)} \, d\phi^T x$$

- Combine weak classifiers

$$\lambda_i(\phi_i^T x) = \alpha_i \log \left(\frac{h_i^+(\phi_i^T x)}{h_i^-(\phi_i^T x)} \right)$$

- Update the weights

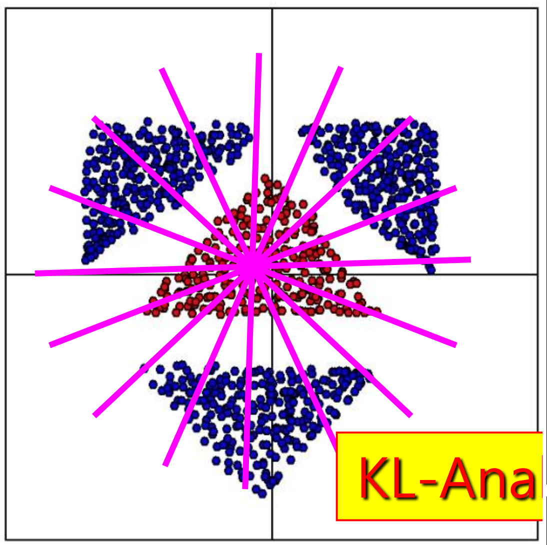
$$\omega_{k+1}(x_i) = \frac{1}{Z_k} \omega_k(x_i) \exp\{-\beta_k y_i F_k(x_i)\} \qquad \beta \leftarrow \ln\left(\frac{1-\varepsilon_M}{\varepsilon_M} \right)$$

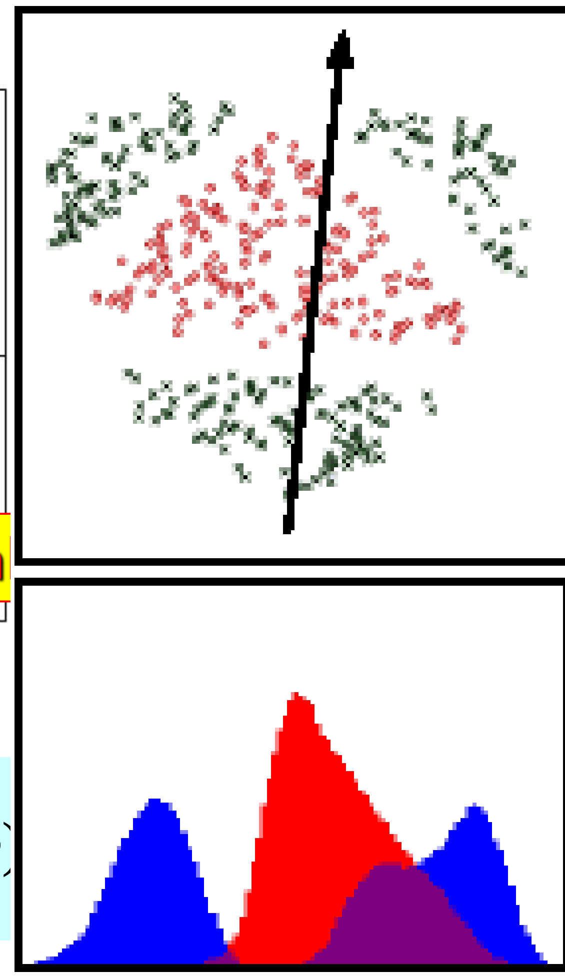
- Strong classifier

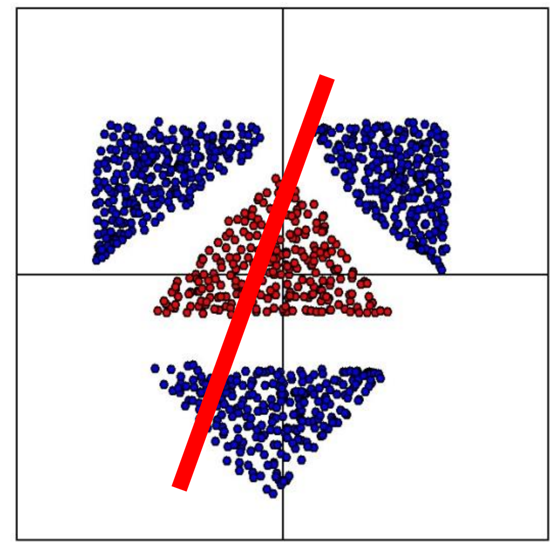
$$\boxed{F(x) = sign\left[\sum_{i=1}^{N} \lambda i(\phi_i^T x) \right]}$$

Example of KLBoost (I)

- **2D vector classification**
  - Line projection and histogram







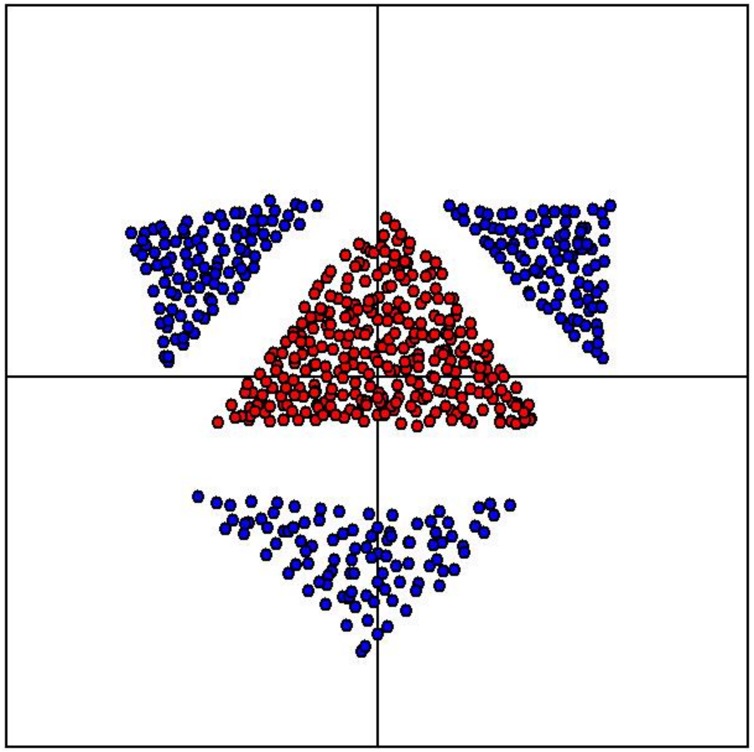
Feature Bank

KL Feature

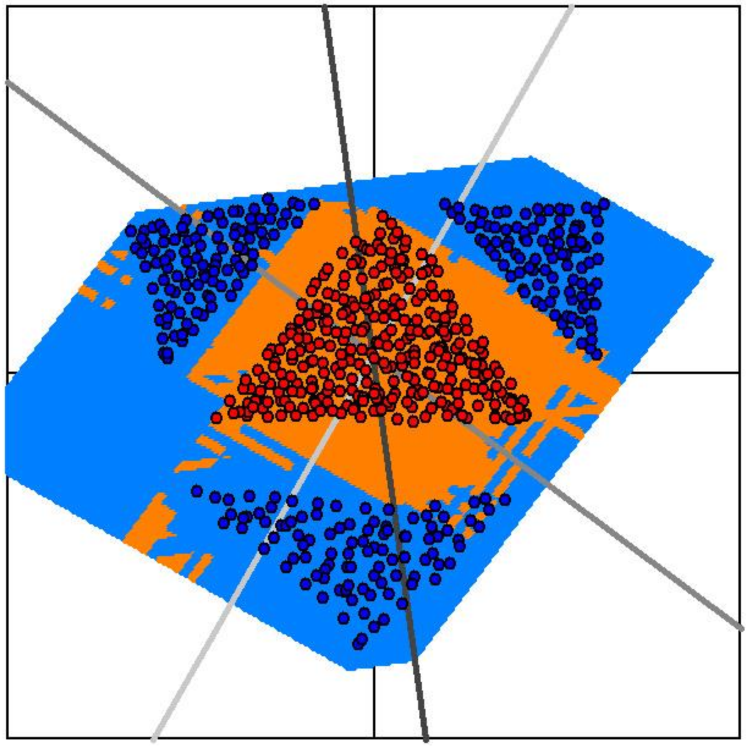
$$\phi_k^* = \arg\max_{\phi} KL(\phi \; \ldots \;)\Big]\log\frac{h_k^+(\phi^T x)}{h_k^-(\phi^T x)}d\phi^T x$$

Example of KLBoost (II)

- **KL Boosting example 1**
  - 2D data classification



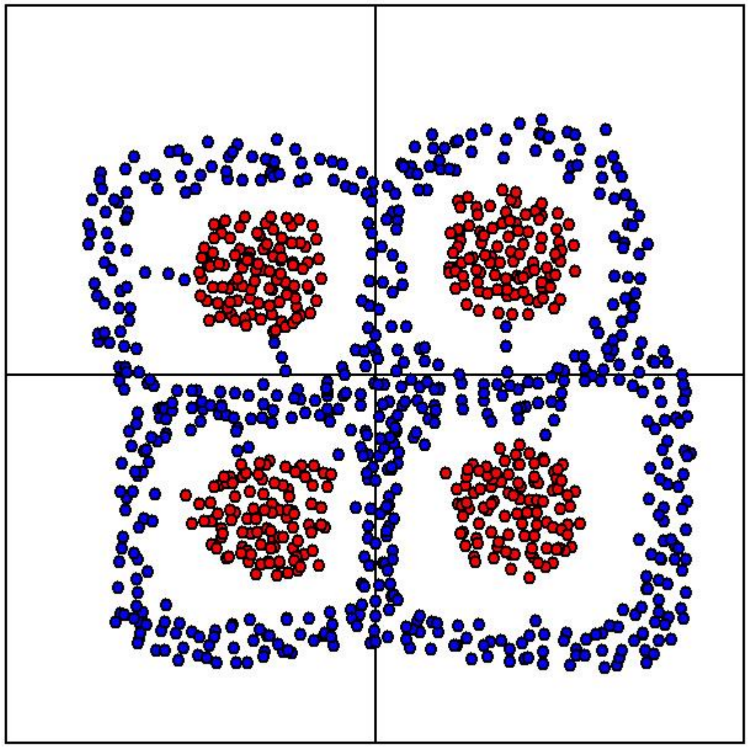
2-class 2D data



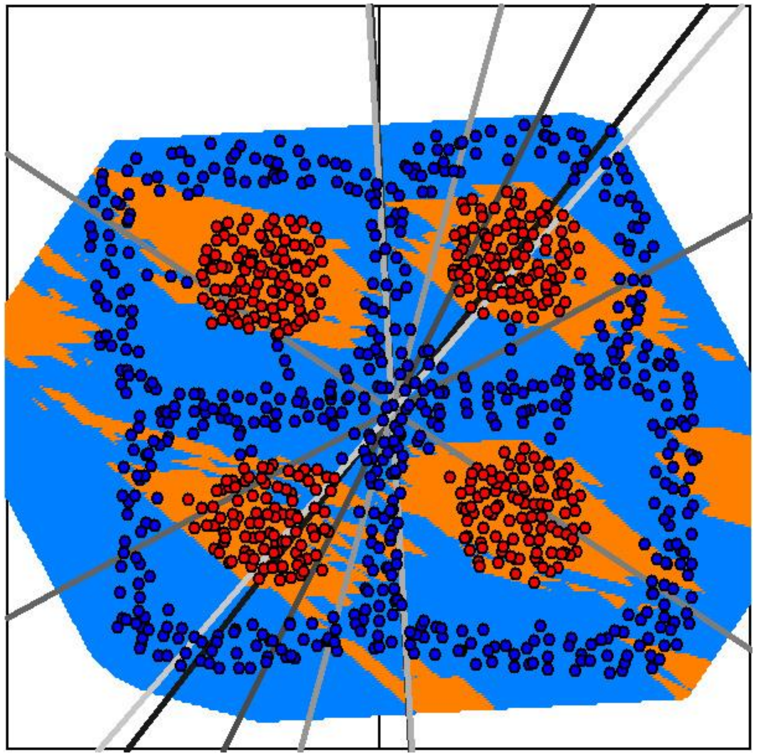
Classified data

Example of KLBoost (III)

- **KL Boosting example 2**
  - 2D data classification



2-class 2D data



Classified data

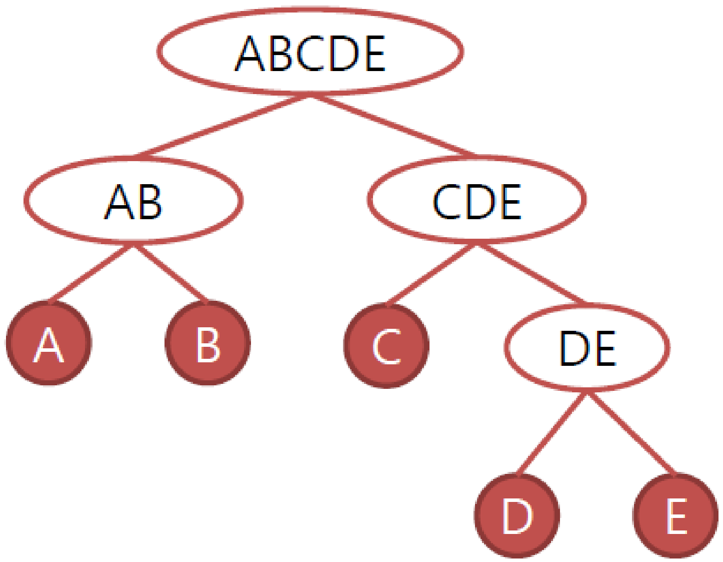
Exemplary Application of Boosting

- For application of boosting process
  - Design the weak classifiers
  - Design matching schemes
- Weak classifiers
  - Easy to generate (weak classifiers bank)
  - Simple calculation
  - Intuitive idea (operation)
- Example: Car model recognition
  - What features are used?
  - How are the weak classifiers designed?

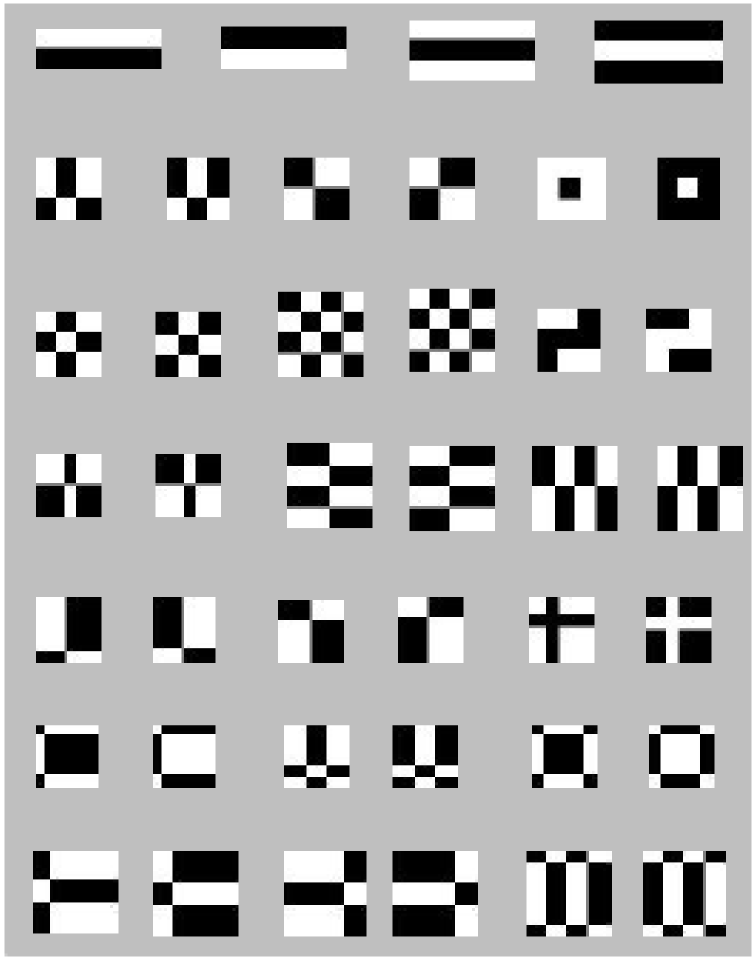
2D Haar-Like Patterns

❖Design some 2D Haar-Like Patterns

Car models



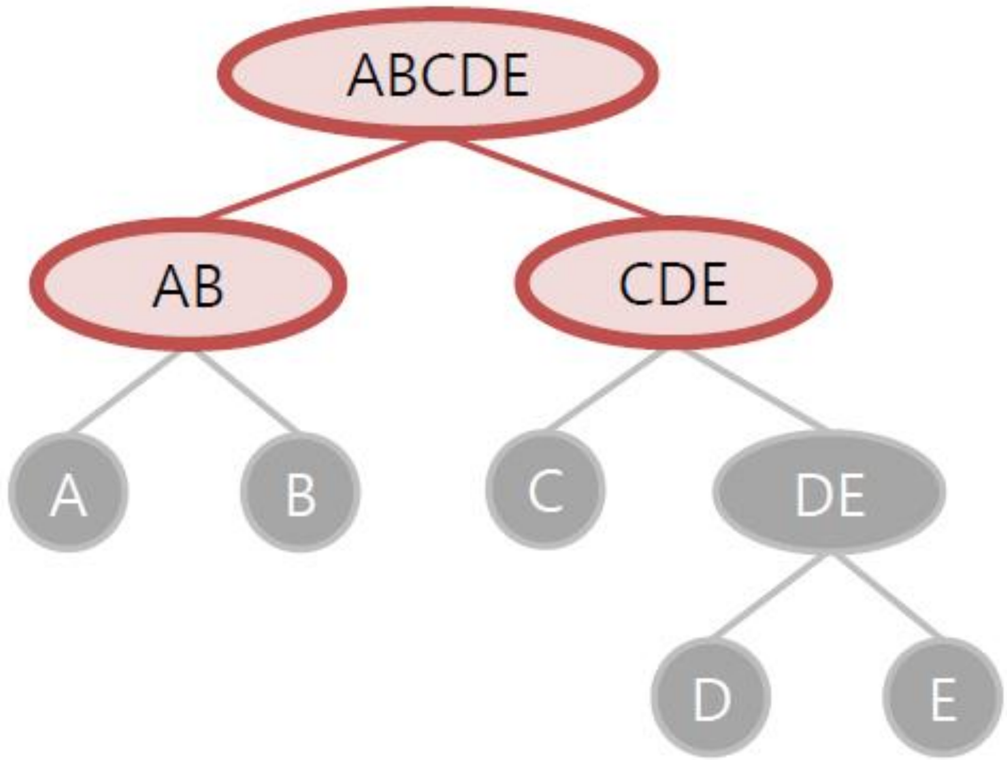
Cascaded tree structure



2D Haar-like Patterns
(features)

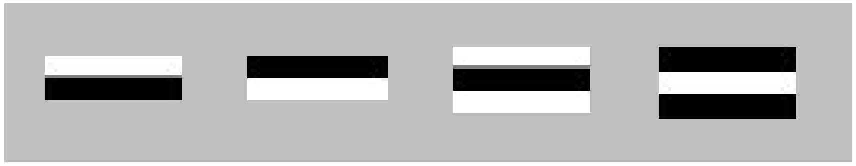
2D Haar-Like Patterns

Car models



Car models

Mean outputs
- 120
- 80
- 70
- 65

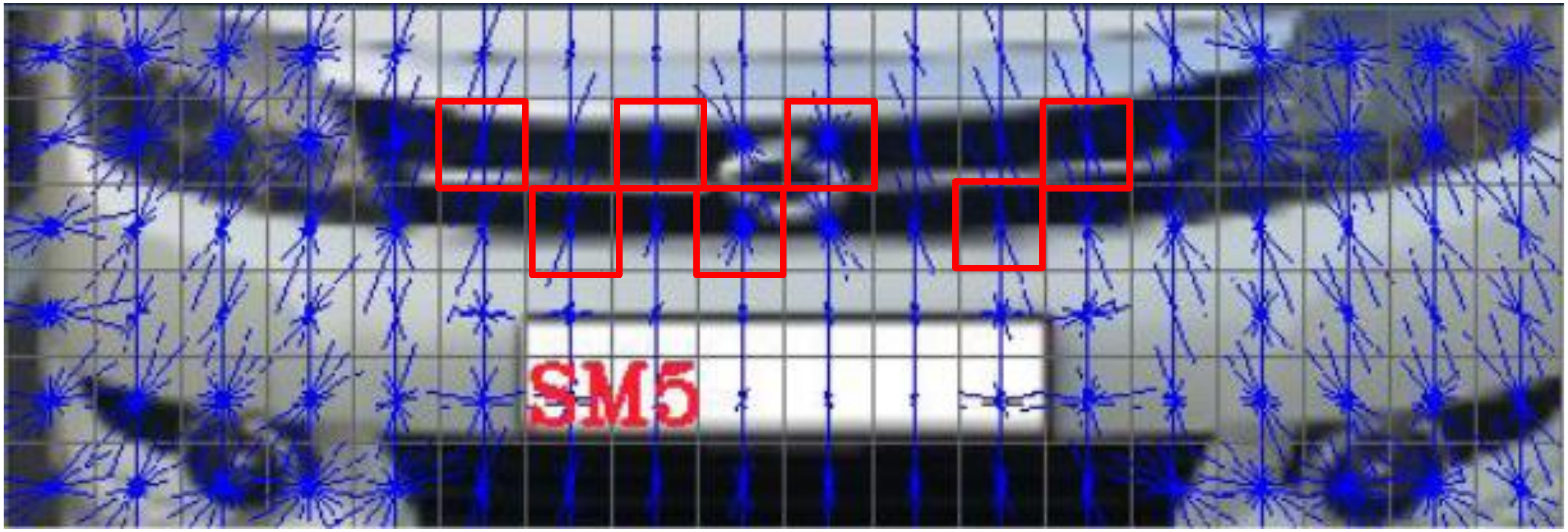


2D Harr-like Patterns
(features)

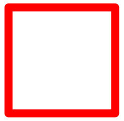
Mean outputs
- 30
- 40
- 45
- 50

HOG

- **Histogram of Gradient orientations**
  - **Partial distribution**
  - **Select maximally discriminative parts**



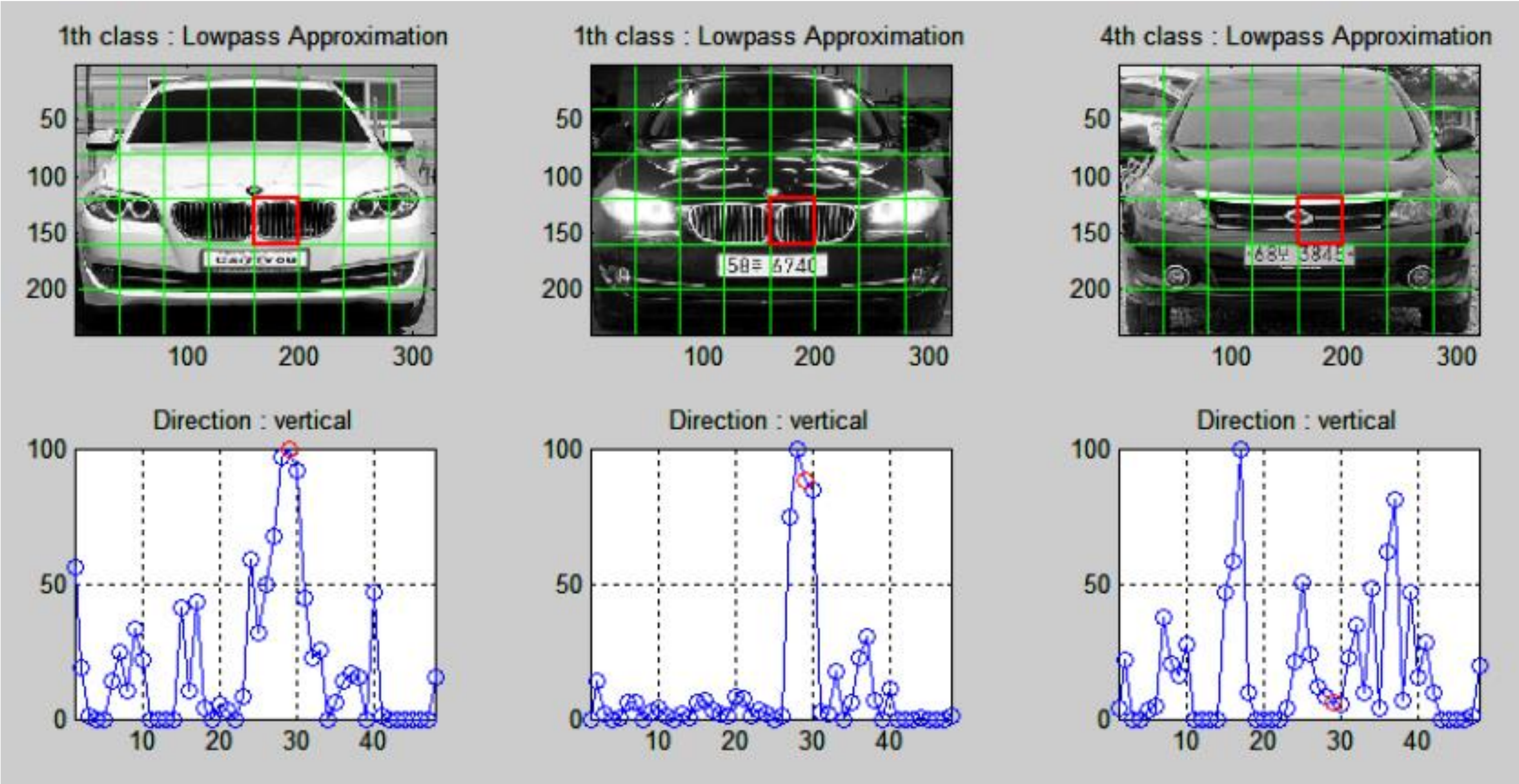
Features: HOG for subblocks (parts)



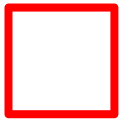
Most discriminative parts

DFT Coefficients

❖Specific distribution of DFT Coefficients



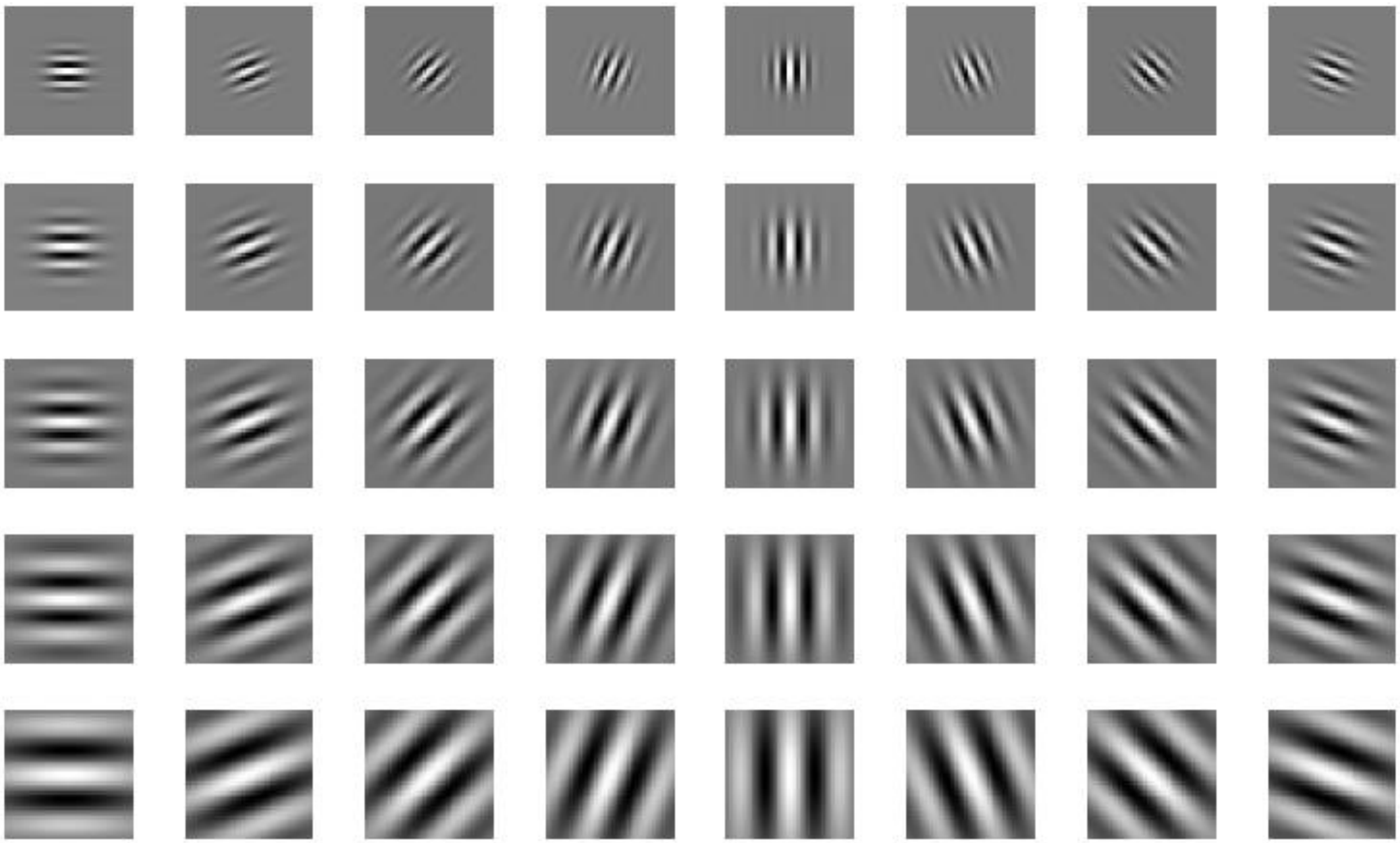
Features: Distribution of DFT Coefficients



Most discriminative parts

2D Gabor Patterns

- **Gabor wavelet filter**
  - Texture analysis
  - 5 scales, 8 orientations



2D Gabor Patterns

- **Gabor filter response**