Keertana . V . Chidambaram

PS - 4.

(1) (a) $\sum_{i: y_i \neq h_{m+1}(x_i)} W_i^{m+1}$ $\left\{ \text{since } \sum_{i=1}^{n} W_i^{m+1} = 1 \right.$ i.e weights are normalized

$$= \frac{\sum_{i: y_i \neq h_{m+1}(x_i)} W_i^{m+1}}{\sum_{i: y_i \neq h_{m+1}(x_i)} W_i^{m+1} + \sum_{i: y_i = h_{m+1}(x_i)} W_i^{m+1}}$$

$$A = \frac{1}{1 + \left\{ \frac{\sum_{i: y_i = h_{m+1}(x_i)} W_i^{m+1}}{\sum_{i: y_i \neq h_{m+1}(x_i)} W_i^{m+1}} \right\}}$$

consider the term in the curly brackets,

we know $W_i^{m+1} = \frac{1}{Z} W_i^{(m)} e^{-\alpha_{m+1} y_i h_{m+1}(x_i)}$

so. $$B = \frac{\sum_{i: y_i = h_{m+1}(x_i)} W_i^{m+1}}{\sum_{i: y_i \neq h_{m+1}(x_i)} W_i^{m+1}} = \frac{\sum_{i: y_i = h_{m+1}(x_i)} \frac{1}{Z} W_i^{m} e^{-\alpha_{m+1} \times 1}}{\sum_{i: y_i \neq h_{m+1}(x_i)} \frac{1}{Z} W_i^{m} e^{-\alpha_{m+1} \times -1}}$$

$\left\{ \text{since } y_i h_{m+1}(x_i) \begin{cases} 1 & \text{if } y_i = h_{m+1}(x_i) \text{ and} \\ -1 & \text{if } y_i \neq h_{m+1}(x_i) \end{cases} \right.$

$$= \frac{e^{-2\alpha_{m+1}} \sum_{i: y_i = h_{m+1}(x_i)} w_i^m}{\sum_{i: y_i \neq h_{m+1}(x_i)} w_i^m}$$

but we know that $\alpha_{m+1} = \frac{1}{2} \log \left[\frac{1-\varepsilon_{m+1}}{\varepsilon_{m+1}} \right]$

so $e^{-2\alpha_{m+1}} = e^{-2 \times \frac{1}{2} \log \left(\frac{1-\varepsilon_{m+1}}{\varepsilon_{m+1}} \right)}$

$$\frac{\varepsilon_{m+1}}{1-\varepsilon_{m+1}}$$

But we know that

$$\varepsilon_{m+1} = \sum_{i: y_i \neq h_{m+1}(x_i)} w_i^m.$$

and thus $1-\varepsilon_{m+1} = \sum_{i: y_i = h_{m+1}(x_i)} w_i^m.$

$$B = \frac{\varepsilon_{m+1}}{1-\varepsilon_{m+1}} \times \frac{1-\varepsilon_{m+1}}{\varepsilon_{m+1}} = 1$$

$$\therefore A = \frac{1}{1+B} = \frac{1}{1+1} = \underline{\underline{\frac{1}{2}}}. \quad \square$$

1 b Proved in 1.a:

$$\sum_{i: h_m(x_i) \neq y_i} w_i^m = \frac{1}{2}$$

Suppose $h_1 \ldots h_m$ models comprise H_M.

Adding h_{m+1}, h_{m+1} is determined by

$$\min_{h_{m+1}} \sum_{i: h_{m+1}(x_i) \neq y_i} w_i^m = \varepsilon_{m+1}$$

If $h_{m+1} = h_m$, (i.e optimal $h_{m+1} = h_m$)

$$\varepsilon_{m+1} = \sum_{i: h_m(x_i) \neq y_i} w_i^m = \frac{1}{2}$$

We know

$$\alpha_{m+1} = \log\left(\frac{1-\varepsilon_{m+1}}{\varepsilon_{m+1}}\right)$$

$$= \log\left(\frac{1-1/2}{1/2}\right)$$

$$= \log(1) = 0.$$

Also a classifier h_m is weak only if $\varepsilon_m < 1/2$. If $\varepsilon_m \geq 1/2$ it is as good as a random classifier. So $h_{m+1} \neq h_m$.

$\therefore$ if $h_{m+1} = h_m$, it will not get added to the ensemble because $\alpha_{m+1} = 0$,

thus we can't have $h_{m+1} = h_m$ for any m.

① ⓒ A "weak" classifier h_m is useful only if error $\varepsilon_m = 1/2 - \varepsilon$ with $\varepsilon > 0$.

if $h_{m+1} = h_m$

then $\varepsilon_{m+1} = 1/2$ (as proved before)

but if $h_{m+k} = h_m$

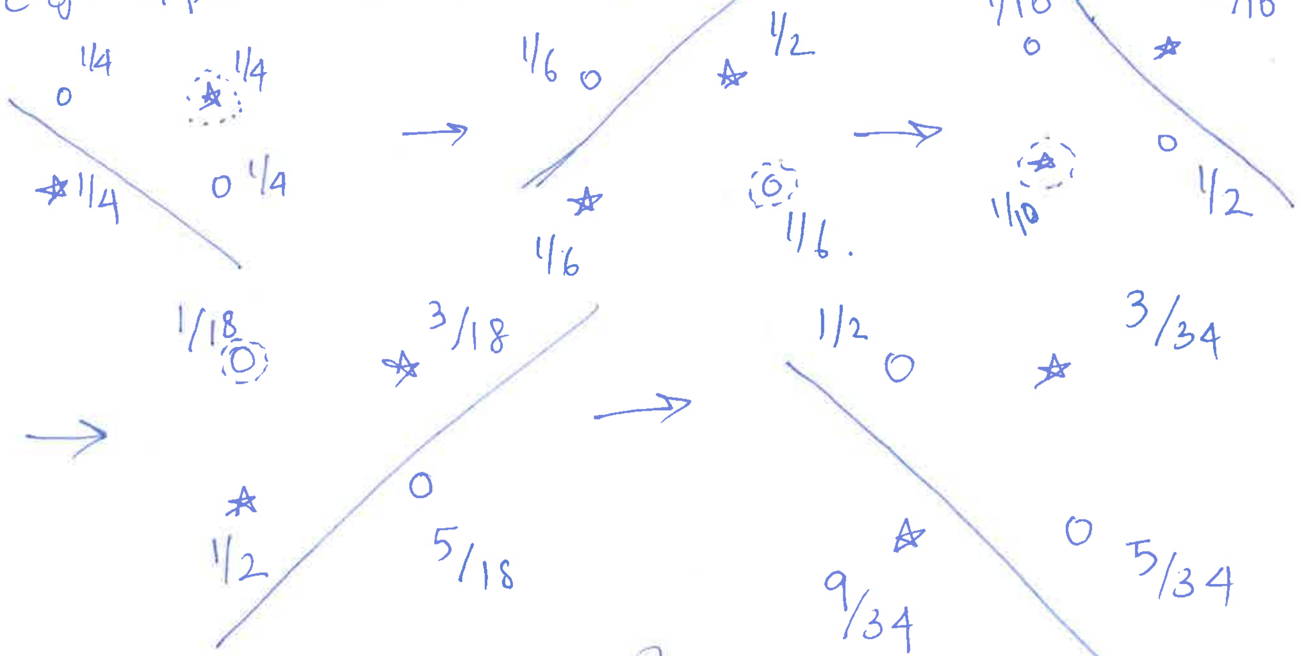
for some $k > 1$,

then $\varepsilon_{m+k} = \sum_{i : h_{m+k}(x_i) \neq y_i} W_{m+k-1}$

need not necessarily be $1/2$.

So a classifier model can repeat itself in the AdaBoost algo

e.g.: A point classification:



{Note: idea borrowed from Michael Hanna}

classifier is same as the first classifier.

③

②

Empirical exponential loss;

$$L = \sum_{i=1}^{n} w_i^{m-1} e^{-\alpha_m y_i h_m(x_i)}$$

Given $h_m(x_i)$, the α_m that minimizes L satisfies:

$$\frac{\partial L}{\partial \alpha_m} = 0$$

$$\Rightarrow \sum_{i=1}^{n} w_i^{m-1} e^{-\alpha_m y_i h_m(x_i)} \times -y_i h_m(x_i) = 0 .$$

we know $y_i h_m(x_i) = \begin{cases} 1 & \text{if } y_i = h_m(x_i) \\ -1 & \text{o.w.} \end{cases}$

so $\sum_{i:y_i = h_m(x_i)} w_i^{m-1} e^{-\alpha_m} \times -1 + \sum_{i:y_i \neq h_m(x_i)} w_i^{m-1} e^{\alpha_m} \times 1 = 0.$

α_m is independent of i so

$$-e^{-\alpha_m} \sum_{i:y_i = h_m(x_i)} w_i^{m-1} + e^{\alpha_m} \sum_{i:y_i \neq h_m(x_i)} w_i^{m-1} = 0.$$

we know $\varepsilon_m = \sum_{i:y_i \neq h_m(x_i)} w_i^{m-1}$

and $1 - \varepsilon_m = \sum_{i:y_i = h_m(x_i)} w_i^{m-1}$,

so $-e^{-\alpha_m}(1-\varepsilon_m) + e^{\alpha_m}\varepsilon_m = 0.$

$$\Rightarrow e^{2\alpha_m} = \frac{1-\varepsilon_m}{\varepsilon_m} \Rightarrow \alpha_m = \frac{1}{2}\log\left[\frac{1-\varepsilon_m}{\varepsilon_m}\right].$$

③ $p(y=1 \mid x; w) = \dfrac{1}{1+\exp\left(\sum_{j=1}^{d} w_j \, \phi_j'(x)\right)} = \dfrac{1}{1+\exp(w.\phi'(x))}$

Define the new feature space

$\phi'(x_0)$ as:

$$\phi'(x_0) = \begin{bmatrix} K(x_0, x_1) \\ K(x_0, x_2) \\ \vdots \\ K(x_0, x_n) \end{bmatrix}_{n \times 1} \equiv K(x_0, x)$$

and

w is an $n \times 1$ vector.

where $x_1, x_2, \ldots x_n$ are the training data

and $K(x_0, x_i)$ is the kernel defined as

$$K(x_0, x_i) = \phi(x_0)^T \phi(x_i)$$

where ϕ is the original feature space.

prediction for x_0 is given as.

$$\hat{y} = \text{sign}\left(W . \phi'(x_0)\right) = \text{sign}\left(W . K(x_0, x)\right)$$

$\therefore$ prediction only depends on the kernel values.

Gradient (as given in notes + adding regularizer term):

$$\text{grad} = \nabla_W \left[\log p(y_i \mid x_i; W) + \lambda \|W\|^2\right]$$

$$= -\left[y_i - \sigma(W \cdot K(x_i, x))\right] K(x_i, x) + 2\lambda W$$

So gradient on a single example also depends on the training data only through the kernel.