HW-2

S. Suprabath Reddy
EE15BTECH11026

1) For K-class linear discriminant classifier,

$$\hat{y}_K(\vec{x}) = W_K^T \vec{x} + w_{K0}$$

If $\hat{y}_K(\vec{x}) > \hat{y}_j(\vec{x})$ for $j \neq K$,

then output is class K.

Consider two points $\vec{x}_A$ and $\vec{x}_B$ which are lying in the decision region R_K.

Any point $\hat{\vec{x}}$ that lies on line connecting $\vec{x}_A$ and $\vec{x}_B$

is $\hat{\vec{x}} = \lambda \vec{x}_A + (1-\lambda)\vec{x}_B$ where $0 \leq \lambda \leq 1$.

From the linearity of discriminant functions.

$$\cancel{\hat{y}_K(\hat{\vec{x}}) = \lambda y_K(\vec{x}_A)}$$

$$\hat{y}_K(\hat{\vec{x}}) = \lambda y_K(\vec{x}_A) + (1-\lambda) y_K(\vec{x}_B) \quad \text{--- (1)}$$

Since $\vec{x}_A$ and $\vec{x}_B$ lie inside R_K

$$\hat{y}_K(\vec{x}_A) > \hat{y}_j(\vec{x}_A) \quad \& \quad \hat{y}_K(\vec{x}_B) > \hat{y}_j(\vec{x}_B)$$

$$(1) \Rightarrow \hat{y}_K(\hat{\vec{x}}) > \lambda(\hat{y}_j(\vec{x}_A)) + (1-\lambda) y_j(\vec{x}_B)$$

$$\hat{y}_K(\hat{\vec{x}}) \geq \hat{y}_j(\hat{\vec{x}})$$

$\therefore$ $\hat{\vec{x}}$ lies inside R_K.

$\therefore$ R_K is convex

2) maximize of u subject to

$$y^{(i)} \cdot \frac{(\vec{w}^T \vec{x}^{(i)} + w_0)}{\|\vec{w}\|} \geq u$$

$$\Rightarrow y^{(i)} (\vec{w}^T \vec{x}^{(i)} + w_0) \geq u \|\vec{w}\|$$

If $\|\vec{w}\| = \frac{1}{u}$

$\Rightarrow$ Maximization of $u \Rightarrow$ minimization of $\|\vec{w}\|^2$

such that $y^{(i)} (\vec{w}^T \vec{x}^{(i)} + w_0) \geq 1$

$$L_p = \min_{w_0, \vec{w}} \frac{1}{2} \|\vec{w}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y^{(i)} (\vec{w}^T \vec{x}^{(i)} + w_0) - 1 \right]$$

where $\alpha_i \geq 0$ — (1)

We want to min L_p w.r.t. $\vec{w}$ and $\vec{w}_0$.

set $\nabla_{\vec{w}} L_p = 0 \Rightarrow \nabla_{\vec{w}} \left(\frac{1}{2} \vec{w}^T \vec{w} - \sum_{i=1}^{N} \alpha_i \left(y^{(i)} (\vec{w}^T \vec{x}^{(i)} + w_0) - 1 \right) \right) = 0$

$$\Rightarrow \vec{w} = \sum_{i=1}^{N} \alpha_i y^{(i)} \cdot \vec{x}^{(i)} \quad \text{— (2)}$$

Set $\frac{\partial L_p}{\partial w_0} = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i \cdot y^{(i)} = 0$ — (3)

From (2) & (3), solve for α_i by substituting in (1).

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} \cdot y^{(j)} \cdot \vec{x}^{(i)T} \cdot \vec{x}^{(j)}$$

such that $\alpha_i \geq 0$ and $\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$

In addition to above condition, our optimal solutions α_i's must satisfy

$$\alpha_i \left(y^{(i)} (\vec{w}^T \cdot \vec{x}^{(i)} + w_0) - 1 \right) = 0$$

3) Output at the m^{th} layer

$$Z_m = \sigma(\alpha_{m0} + \vec{\alpha}_m^T \cdot \vec{x})$$

where $\sigma(x) = \dfrac{1}{1+e^{-x}}$; $\vec{\alpha}_m = [\alpha_{m1} \cdots \alpha_{md}]^T$

α_{m0} = bias associated with m^{th} hidden node.

$$\hat{y}_k(\vec{x}) = g_k(\beta_{k0} + \vec{\beta}_k^T \cdot \vec{z})$$

where $g_k(\vec{x}) = \dfrac{e^{x_k}}{\sum_{j=1}^{K} e^{x_j}}$

Parameters - $[\theta : \alpha_{m0}, \vec{\alpha}_m, \beta_{k0}, \vec{\beta}_k]$; $1 \le m \le M$, $1 \le k \le K$

Cost function: $R(\theta) = \sum_{i=1}^{N} \| \vec{y}^{(i)} - \hat{y}(\vec{x}^{(i)}) \|^2$

$$= \sum_{i=1}^{N} \sum_{k=1}^{K} (y_k^{(i)} - \hat{y}_k(\vec{x}^{(i)}))^2$$

$$= \sum_{i=1}^{N} R^{(i)}(\theta) \quad \text{where } R^{(i)}(\theta) = \sum_{k=1}^{K} (y_k^{(i)} - \hat{y}_k(\vec{x}^{(i)}))^2$$

To find locally optimal parameter θ, find

$$\frac{\partial R^{(i)}(\theta)}{\partial \beta_{km}} = \frac{\partial}{\partial \beta_{km}} \sum_{k'=1}^{K} (y_{k'}^{(i)} - \hat{y}_{k'}^{(i)})^2$$

$$= \frac{\partial}{\partial \beta_{km}} \sum_{k'=1}^{K} (y_{k'}^{(i)} - g_{k'}(\beta_{k'0} + \beta_{k'}^T \cdot z^{(i)}))^2$$

$$\cancel{\Rightarrow \frac{\partial R^{(i)}(\theta)}{\partial \beta_{km}} = 2(y_{k'}^{(i)} - g_k(\beta}$$

$$\Rightarrow \frac{\partial R^{(i)}(\theta)}{\partial \beta_{km}} = 2(y_k^{(i)} - g_k(\beta_{k0} + \vec{\beta}_k^T \cdot \vec{z}^{(i)})). (-g_k' \cdot (\beta_{k0} + \beta_k^T \cdot z^{(i)})). z_m^{(i)}$$

$$= \delta_k^{(i)} \cdot z_m^{(i)} \quad \text{where } \delta_k^{(i)} = -2(y_k^{(i)} - g_k(\beta_{k0} + \beta_k^T \cdot \vec{z}^{(i)})). g_k'(\beta_{k0} + \beta_k^T \cdot z^{(i)})$$

$\textcircled{1}$

$$\frac{\partial R^{(i)}(\theta)}{\partial \alpha_{ml}} = s_m^{(i)} \cdot x_l^{(i)} \quad — ②$$

$$\hookrightarrow s_m^{(i)} = \left(\sum_{k=1}^{K} \delta_k^{(i)} \cdot \beta_{km} \right) \cdot \sigma'\left(\alpha_{m0} + \vec{\alpha}_m^T \cdot \vec{x}^{(i)}\right)$$

$\hookrightarrow$ ③

Final updated weights

$$\left.\begin{aligned} \beta_{km}^{(r+1)} &= \beta_{km}^{(r)} - \gamma_r \sum_{i=1}^{N} \frac{\partial R^{(i)}(\theta)}{\partial \beta_{km}^{(r)}} \\ \alpha_{ml}^{(r+1)} &= \alpha_{ml}^{(r)} - \gamma_r \cdot \sum_{i=1}^{N} \frac{\partial R^{(i)}(\theta)}{\partial \alpha_{ml}^{(r)}} \end{aligned}\right\} \quad — ④$$

From ① & ②, $\delta_k^{(i)}$; $s_m^{(i)}$ are errors from current model at output & hidden layer units.

③ is a back propogation eqn.

With the updated ~~weights~~ values in ④, we can implement back-propagation algorithm.

In forward pass, the current weights are fixed and predicted values are computed

In backward pass, errors $\delta_{ki}^{(i)}$ are computed, then using ③, to give errors $s_m^{(i)}$.

Both sets of errors are used to computed gradients.

4) Assume the below given neural network

x_1 $\quad\vdots\quad$ y_1

$\vdots$ $\quad\vdots\quad$ $\vdots$

x_n $\quad\quad\quad$ y_m

n inputs $\quad$ k nodes $\quad$ m-outputs

Weights, w_{10}, w_1 $\quad$ Weights w_{20}, w_2

$$h = w_1 x + w_{10} \quad -①$$

$$h_a = \frac{1}{1+e^{-x}}$$

$$z = w_2 h_a + w_{20} \quad -②$$

Loss function = loss entropy function

$$L = -\sum_{i=1}^{M} y^{(i)} \cdot \log(\hat{y}^{(i)})$$

$$\Rightarrow \frac{\partial L}{\partial w_2^{(i)}} = -\left(\left(\sum_{j \neq i} \frac{y^{(j)}}{\cancel{\hat{y}^{(j)}}}\left(-\hat{y}^{(i)} \cdot \cancel{\hat{y}^{(j)}}\right) + \frac{y^{(i)}}{\cancel{\hat{y}^{(i)}}} \cdot \cancel{\hat{y}^{(i)}}\left(1-\hat{y}^{(i)}\right)\right) \cdot h_a\right)$$

$$= -h_a\left(-\sum_{j=1}^{M} y^{(j)} \cdot \hat{y}^{(i)} + y^{(i)}\right)$$

$$\sum_{j=1}^{M} y^{(j)} = 1$$

$$\therefore \frac{\partial L}{\partial w_2^{(i)}} = h_a\left(\hat{y}^{(i)} - y^{(i)}\right)$$

$$w_2^{(i)} \text{ updated} = w_2^{(i)} - h_a\left(\hat{y}^{(i)} - y^{(i)}\right)$$

$$\frac{\partial L}{\partial w_1^{(i)}} = \frac{\partial L}{\partial h_a^{(i)}} \cdot \frac{\partial h_a^{(i)}}{\partial h^{(i)}} \cdot \frac{\partial h^{(i)}}{\partial w_1^{(i)}}$$

$$= \frac{\partial L}{\partial h_a^{(i)}} \cdot h^{(i)} \cdot (1-h^{(i)}) \cdot x \quad \text{(From ① \& ②)}.$$

$$= \frac{\partial L}{\partial z}\left(\cancel{w}_2^{(i)} * h^{(i)} \cdot (1 - h^{(i)})\right) \cdot x$$

↳ all weights from i^{th} layer to output layer.

$$= \underbrace{(\hat{y} - y) * w_2^{(i)} \cdot h^{(i)} \cdot (1 - h^{(i)})}_{\text{be } p} \cdot x$$

$\therefore\ w_1^{(i)}$ updated $= w_1^{(i)} + p \times x$