Neural Networks for Classification

EE698V - Machine Learning for Signal Processing

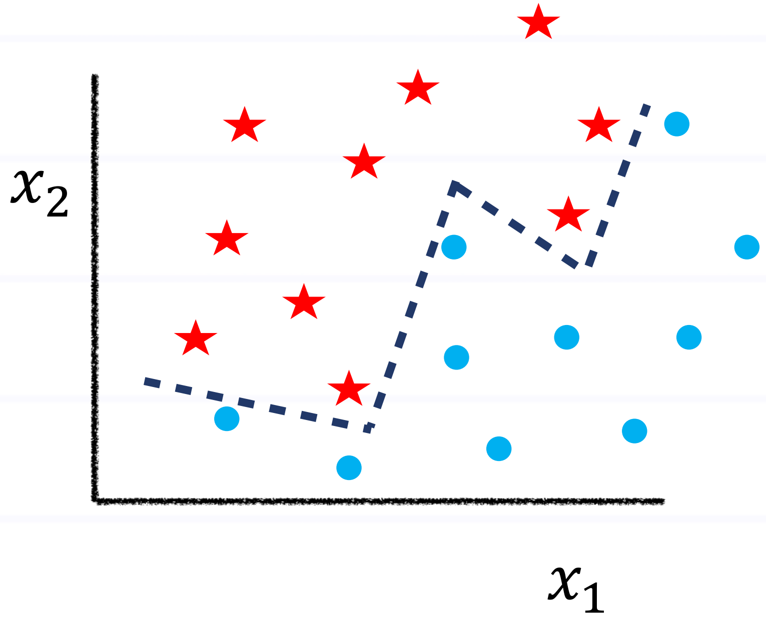
Vipul Arora

Non-linear Classification

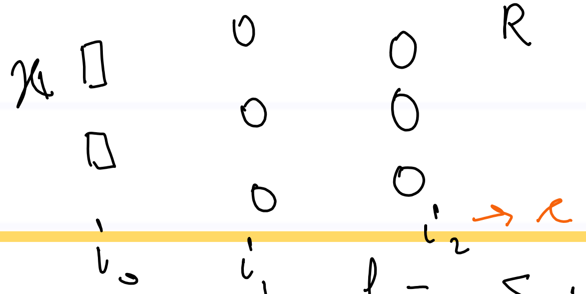
Reference: PRML Sec 5.1, 5.2

Non-linear Classifiers

- Remember, non-linear models can be more powerful than linear ones



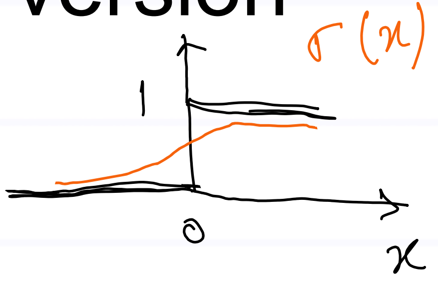
Multi-class Classification



$y = s(h)$

$h_{i_2} = \sum_{i_1} w_{i_2 i_1} v_{i_1}$

- The output of model is $\boldsymbol{h}$ or $h_c \in \mathbb{R}, c = 0,1,2,\ldots, C-1$
- $y_c = \operatorname{argmax}_c h_c$, $\boldsymbol{y}$ is a one-hot vector
- argmax is not differentiable, so we need a soft version

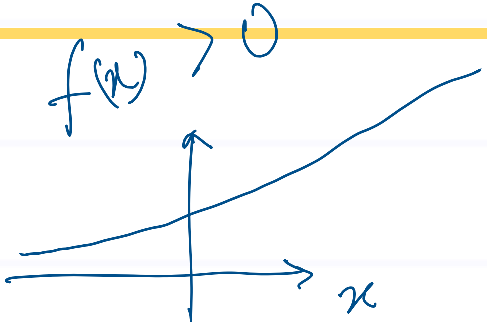


- $\boldsymbol{y} = \text{softmax}(\boldsymbol{h})$; i.e., $y_c = \dfrac{e^{h_c}}{\sum_{c'} e^{h_{c'}}}$

$h = \begin{bmatrix} 2.1 \\ 0.3 \\ 0.8 \end{bmatrix}$ $\quad y = \begin{bmatrix} e^{2.1}/\text{den} \\ e^{0.3}/\text{den} \\ e^{0.8}/\text{den} \end{bmatrix} = \begin{bmatrix} .7 \\ .11 \\ .18 \end{bmatrix}$ $\quad \text{den} = e^{2.1} + e^{0.3} + e^{0.8}$ $\quad y = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Why exp in softmax?

$$y_c = \frac{h_c}{\sum_{c'} h_{c'}} \leftarrow \text{could be -ve or 0}$$



exp.

$$= \frac{2.1}{2.1 + 0.8 + 0.3}$$

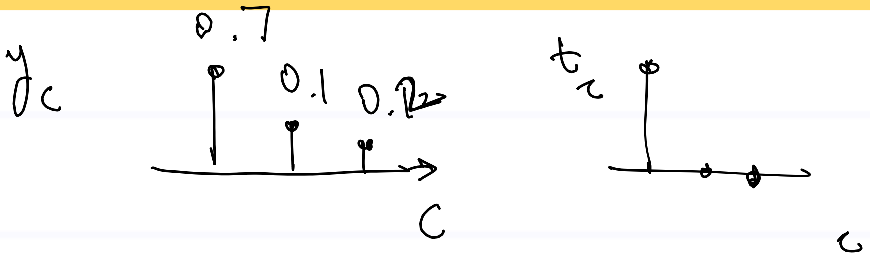
$$= \frac{\sigma(2.1)}{\sigma(2.1) + \sigma(0.8) + \sigma(0.3)}$$

$\sigma(2.1) \rightarrow 1$

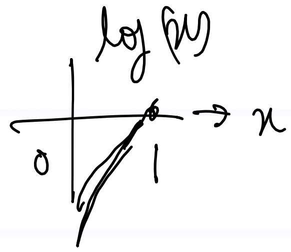
$\rightarrow 1 \qquad \rightarrow 0.9 \qquad \rightarrow 0.5$

$$y = \begin{bmatrix} 1/2.4 \\ 0.9/2.4 \\ 0.5/2.4 \end{bmatrix}$$

Multi-class Classification: Loss function



- Mean Squared Error?
- We need a stronger pull to 0 or 1



- y_c can be seen as $P(c)$ as $\sum_c y_c = 1$, so also t_c
- Categorical cross-entropy
  - $E_{Xent} = -\sum_c t_c \log(y_c)$

$$\overset{}{=} -(\underset{}{\overset{c=0}{1 \log 0.7}} + \overset{c=1}{0} + \overset{c=2}{0})$$

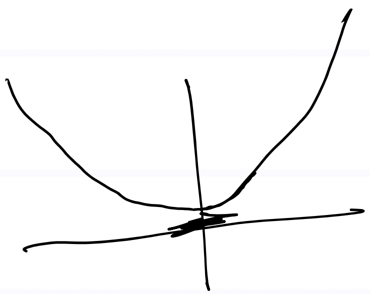
Why not MSE?

$$E_{MSE} = \frac{1}{2}\sum_c (t_c - y_c)^2$$

$$E_{xent} = \sum_c -t_c \log y_c$$

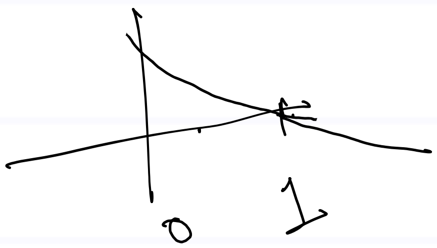
$$\frac{\partial E_{MSE}}{\partial y_{c'}} = (t_c - y_{c'})\left(-\frac{\partial y_{c'}}{\partial y_{c'}}\right) = -0.3$$

$$y_c \approx 0.7$$



$$\frac{\partial E_{xent}}{\partial y_{c'}} = -\frac{\overset{\delta_{cc'}}{\cancel{t_c}}}{y_{c'}} \frac{\partial y_{c'}}{\partial y_{c'}} = -\frac{1}{0.7} = -1.4$$

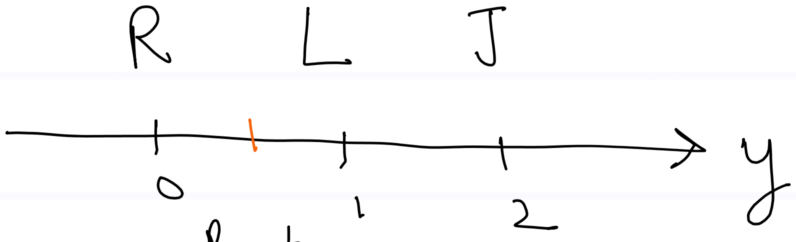
$$y_c \approx 0.5$$



$$\frac{\partial E_{xent}}{\partial y_{c'}} = -2 \qquad \frac{\partial E_{MSE}}{\partial y_{c'}} = -0.5$$

Why not represent target as {0,1,2}?

$$t = \left\{\frac{0}{2}, \frac{1}{2}, \frac{2}{2}\right\}$$



$$t = \left\{ \begin{matrix} 1 \\ 0 \\ 0 \end{matrix}, \begin{matrix} 0 \\ 1 \\ 0 \end{matrix}, \begin{matrix} 0 \\ 0 \\ 1 \end{matrix} \right\} \leftarrow \text{one hot}$$

flower ~ R, L $\quad$ ~ R, J

$$\begin{matrix} 0.5 \\ 0.5 \\ 0 \end{matrix} \qquad \begin{matrix} 0.5 \\ 0 \\ 0.5 \end{matrix}$$

Multi-label Classification

attributes

beard	0	1
glasses	0	1
hair	0	1

- The output of model is $\boldsymbol{h}$ or $h_l \in \mathbb{R}, l = 0,1,2,\dots, L-1$

$h_{i_2} = \sum_{i_1} w_{i_2 i_1} v_{i_1}$

- $y_l = \mathrm{sign}(h_l)$ or $y_l = (\mathrm{sign}(h_l)+1)/2$
- sign is not differentiable, so we need a soft version

tc

- $y_l = \mathrm{sigmoid}(h_l)$, $\boldsymbol{y}$ is a multi-hot vector

$$\begin{matrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & \dots \\ 1 & 1 & 1 & 1 & 0 & 0 \end{matrix}$$

$$\boldsymbol{h} = \begin{bmatrix} 2.1 \\ 0.8 \\ -0.3 \end{bmatrix} \qquad \boldsymbol{y} = \begin{bmatrix} 0.8 \\ 0.65 \\ 0.4 \end{bmatrix}$$

Multi-label Classification: Loss function

- Probabilistic interpretation:
  - $P(l = 1) = y_l$
  - $P(l = 0) = 1 - y_l$

$E = 0$

when $y = 1$ for $t = 1$

and $y = 0$ for $t = 0$

- Binary Cross-entropy
  - $E_{binXent} = -\sum_l (\, t_l \log y_l + (1 - t_l) \log (1 - y_l)\,)$

beard, $t = 1$

$y = 0.7$

$E_{binXent} = -1 \log 0.7$

$t = 0$

$y = 0.7$

$E = -(1 - 0) \log 0.3$

Optimization

- You can find optimal model parameters (or weights) using gradient descent
- All you need is $\partial E / \partial \boldsymbol{w}$
- Can you find $\partial E_{Xent} / \partial h_c$ or $\partial E_{binXent} / \partial h_l$?

Perceptron Algorithm

- DIY
- Reference: PRML Section 4.1.7