Tim O'Connor

6038881588

oconnort@usc.edu

EE559 – Rajati – Summer 2024

Homework 7

1. For SVR, prove that $w_0 = y_j - \epsilon - \sum_{i\in S}(\lambda_i^+ - \lambda_i^-)\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, where $S$ is the set of indices of support vectors and $\boldsymbol{x}_j$ is a support vector at the *upper edge* of the $\epsilon$-tube. (10 pts)

   For $\boldsymbol{x}_j$ at the *upper edge*:

$$y_i - \hat{y}_i = \xi_i^+ + \epsilon = \epsilon \; (\xi_i^+ = 0 \; at \; edge)$$
$$y_j - \boldsymbol{w}^T\phi(\boldsymbol{x}_j) - w_0 = \epsilon$$
$$w_0 = y_j - \epsilon - \boldsymbol{w}^T\varphi(\boldsymbol{x}_j)$$
$$L(\boldsymbol{w}, w_0, \boldsymbol{\xi}^+, \boldsymbol{\xi}^-, \boldsymbol{\mu}^+, \boldsymbol{\mu}^-, \boldsymbol{\lambda}^+, \boldsymbol{\lambda}^-) = C\sum_{i=1}^{N}(\xi_i^+ + \xi_i^-) + \frac{1}{2}||\boldsymbol{w}||_2^2 - \sum_{i=1}^{N}(\mu_i^+\xi_i^+ - \mu_i^-\xi_i^-) - \sum_{i\in S}\lambda_i^+(\epsilon + \hat{y}_i - y_i) - \sum_{i\in S}\lambda_i^-(\epsilon - \hat{y}_i + y_i)$$
$$\hat{y} = \boldsymbol{w}^T\varphi(\boldsymbol{x}) + w_0$$
$$\left.\frac{\partial L}{\partial \boldsymbol{w}}\right|_{w=w^*} = \boldsymbol{w} - \sum_{i=1}^{N}\lambda_i^+\frac{\partial}{\partial \boldsymbol{w}}\hat{y}_i + \sum_{i=1}^{N}\lambda_i^-\frac{\partial}{\partial \boldsymbol{w}}\hat{y}_i = \boldsymbol{0}$$
$$\boldsymbol{w} = \sum_{i=1}^{N}(\lambda_i^+ - \lambda_i^-)\varphi(\boldsymbol{x}_i) = \sum_{i\in S}(\lambda_i^+ - \lambda_i^-)\varphi(\boldsymbol{x}_i)$$
$$w_0 = y_j - \epsilon - \sum_{i\in S}(\lambda_i^+ - \lambda_i^-)\varphi(\boldsymbol{x}_i)\,\varphi(\boldsymbol{x}_j)$$

2. For the following classification problem, design a single-layer perceptron, by using the multiclass Perceptron update rule. (20 pts)

$$\mathcal{D}_{\omega 1} = \left\{\boldsymbol{x_1} = \begin{bmatrix}1\\1\end{bmatrix}\right\}$$
$$\mathcal{D}_{\omega 2} = \left\{\boldsymbol{x_2} = \begin{bmatrix}1\\-1\end{bmatrix}\right\}$$
$$\mathcal{D}_{\omega 3} = \left\{\boldsymbol{x_3} = \begin{bmatrix}-1\\-1\end{bmatrix}\right\}$$

   Use one-hot encoding for classes, for example, $\omega_3$ should be represented using the following vector

$$\boldsymbol{y}_3 = \begin{bmatrix}-1\\-1\\1\end{bmatrix}$$

(a) Start with $\boldsymbol{W}(0) = \boldsymbol{0}_{2\times 3}$ and choose $\eta(i) = 0.5$ in $\boldsymbol{W}(i+1) = \boldsymbol{W}(i) + \eta(i)\boldsymbol{x}(i)\boldsymbol{e}^T$. Do not use the augmented space and assume that the biases are always zero (no update for biases).

Show multiple steps of your algorithm. Does it converge? Why? It is alright if you use a computer or calculator to perform the matrix calculation, but you should write down all the steps, and should not write a computer program to yield the final results.

Epoch 1:

$$\boldsymbol{W}(0) = \underset{2\times 3}{\boldsymbol{0}}$$

Step 1: $\boldsymbol{x}(1) = \boldsymbol{x}_1 = [1 \quad 1]^T, \boldsymbol{y}(1) = \boldsymbol{y}_1 = [1 \quad -1 \quad -1]^T, \boldsymbol{W}(1) = \boldsymbol{W}(0) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

$$\boldsymbol{e}(1) = \boldsymbol{y}(1) - f\big(\boldsymbol{W}^T(1)\boldsymbol{x}(1)\big)$$

$$\rightarrow \boldsymbol{W}^T(1)\boldsymbol{x}(1) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}^T \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} < \boldsymbol{w}_1^T(1), \boldsymbol{x}(1) > \\ < \boldsymbol{w}_2^T(1), \boldsymbol{x}(1) > \\ < \boldsymbol{w}_3^T(1), \boldsymbol{x}(1) > \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$f(x) = 2H(x) - 1, f(\boldsymbol{W}^T\boldsymbol{x}) = \boldsymbol{y}_1 \; if \; \boldsymbol{w}_1^T\boldsymbol{x} = \cdots = \boldsymbol{w}_S^T\boldsymbol{x}$$

$$\rightarrow f\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right) = \boldsymbol{y}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\boldsymbol{e}(1) = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ -2 \end{bmatrix}$$

$$\boldsymbol{W}(2) = \underset{2\times 3}{\boldsymbol{0}} + 0.5\begin{bmatrix} 1 \\ 1 \end{bmatrix} \cdot [0 \quad -2 \quad -2] = \begin{bmatrix} 0 & -1 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

Step 2: $\boldsymbol{x}(2) = \boldsymbol{x}_2 = [1 \quad -1]^T, \boldsymbol{y}(2) = \boldsymbol{y}_2 = [-1 \quad 1 \quad -1]^T$

$$\boldsymbol{e}(2) = \boldsymbol{y}(2) - f\big(\boldsymbol{W}^T(2)\boldsymbol{x}(2)\big)$$

$$\rightarrow \boldsymbol{W}^T(2)\boldsymbol{x}(2) = \begin{bmatrix} 0 & -1 & -1 \\ 0 & -1 & -1 \end{bmatrix}^T \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} < \boldsymbol{w}_1^T(2), \boldsymbol{x}(2) > \\ < \boldsymbol{w}_2^T(2), \boldsymbol{x}(2) > \\ < \boldsymbol{w}_3^T(2), \boldsymbol{x}(2) > \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$f(x) = 2H(x) - 1, f(\boldsymbol{W}^T\boldsymbol{x}) = \boldsymbol{y}_1 \; if \; \boldsymbol{w}_1^T\boldsymbol{x} = \cdots = \boldsymbol{w}_S^T\boldsymbol{x}$$

$$\rightarrow f\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right) = \boldsymbol{y}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\boldsymbol{e}(2) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \\ -2 \end{bmatrix}$$

$$\boldsymbol{W}(2) = \begin{bmatrix} 0 & -1 & -1 \\ 0 & -1 & 01 \end{bmatrix} + 0.5\begin{bmatrix} 1 \\ -1 \end{bmatrix} \cdot [-2 \quad 0 \quad -2]$$
$$= \begin{bmatrix} 0 & -1 & -1 \\ 0 & -1 & -1 \end{bmatrix} + \begin{bmatrix} -1 & 0 & -1 \\ 1 & 0 & 1 \end{bmatrix}$$
$$= \begin{bmatrix} -1 & -1 & -2 \\ 1 & -1 & 0 \end{bmatrix}$$

Step 3: $\boldsymbol{x}(3) = \boldsymbol{x}_2 = [-1 \quad -1]^T, \boldsymbol{y}(2) = \boldsymbol{y}_2 = [-1 \quad -1 \quad 1]^T$

$$\boldsymbol{e}(3) = \boldsymbol{y}(3) - f\big(\boldsymbol{W}^T(3)\boldsymbol{x}(3)\big)$$

$$\rightarrow \boldsymbol{W}^T(3)\boldsymbol{x}(3) = \begin{bmatrix} -1 & -1 & -2 \\ 1 & -1 & 0 \end{bmatrix}^T \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} < \boldsymbol{w}_1^T(3), \boldsymbol{x}(3) > \\ < \boldsymbol{w}_2^T(3), \boldsymbol{x}(3) > \\ < \boldsymbol{w}_3^T(3), \boldsymbol{x}(3) > \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 2 \end{bmatrix}$$

$$f(x) = 2H(x) - 1, f(\boldsymbol{W}^T\boldsymbol{x}) = \boldsymbol{y}_1 \; if \; \boldsymbol{w}_1^T\boldsymbol{x} = \cdots = \boldsymbol{w}_S^T\boldsymbol{x}$$
$$\rightarrow f\left(\begin{bmatrix}0\\0\\0\end{bmatrix}\right) = \boldsymbol{y}_1 = \begin{bmatrix}1\\1\\11\end{bmatrix}$$
$$\boldsymbol{e}(3) = \begin{bmatrix}-1\\-1\\1\end{bmatrix} - \begin{bmatrix}1\\1\\0\end{bmatrix} = \begin{bmatrix}-2\\-2\\0\end{bmatrix}$$
$$\boldsymbol{W}(3) = \begin{bmatrix}-1 & -1 & -2\\1 & -1 & 0\end{bmatrix} + 0.5\begin{bmatrix}-1\\-1\end{bmatrix} \cdot \begin{bmatrix}-2 & -2 & 0\end{bmatrix}$$
$$= \begin{bmatrix}-1 & -1 & -2\\1 & -1 & 0\end{bmatrix} + \begin{bmatrix}1 & 1 & 0\\1 & 1 & 0\end{bmatrix}$$
$$= \begin{bmatrix}0 & 0 & -2\\2 & 0 & 0\end{bmatrix}$$

Epoch 2:

$$\boldsymbol{W}(0) = \begin{bmatrix}0 & 0 & -2\\2 & 0 & 0\end{bmatrix}$$

Step 1: $\boldsymbol{x}(1) = \boldsymbol{x}_1 = [1 \quad 1]^T, \boldsymbol{y}(1) = \boldsymbol{y}_1 = [1 \quad -1 \quad -1]^T, \boldsymbol{W}(1) = \boldsymbol{W}(0) = \begin{bmatrix}0 & 0 & -2\\2 & 0 & 0\end{bmatrix}$

$$\boldsymbol{e}(1) = \boldsymbol{y}(1) - f\left(\boldsymbol{W}^T(1)\boldsymbol{x}(1)\right)$$
$$\rightarrow \boldsymbol{W}^T(1)\boldsymbol{x}(1) = \begin{bmatrix}0 & 0 & -2\\2 & 0 & 0\end{bmatrix}^T \begin{bmatrix}1\\1\end{bmatrix} = \begin{bmatrix}< \boldsymbol{w}_1^T(1), \boldsymbol{x}(1) >\\< \boldsymbol{w}_2^T(1), \boldsymbol{x}(1) >\\< \boldsymbol{w}_3^T(1), \boldsymbol{x}(1) >\end{bmatrix} = \begin{bmatrix}2\\0\\-2\end{bmatrix}$$
$$f(x) = 2H(x) - 1, f(\boldsymbol{W}^T\boldsymbol{x}) = \boldsymbol{y}_1 \; if \; \boldsymbol{w}_1^T\boldsymbol{x} = \cdots = \boldsymbol{w}_S^T\boldsymbol{x}$$
$$\rightarrow f\left(\begin{bmatrix}0\\0\\0\end{bmatrix}\right) = \begin{bmatrix}1\\1\\-1\end{bmatrix}$$
$$\boldsymbol{e}(1) = \begin{bmatrix}1\\-1\\-1\end{bmatrix} - \begin{bmatrix}1\\1\\-1\end{bmatrix} = \begin{bmatrix}0\\-2\\0\end{bmatrix}$$
$$\boldsymbol{W}(2) = \begin{bmatrix}0 & 0 & -2\\2 & 0 & 0\end{bmatrix} + 0.5\begin{bmatrix}1\\1\end{bmatrix} \cdot \begin{bmatrix}0 & -2 & 0\end{bmatrix}$$
$$= \begin{bmatrix}0 & 0 & -2\\2 & 0 & 0\end{bmatrix} + \begin{bmatrix}0 & -1 & 0\\0 & -1 & 0\end{bmatrix}$$
$$= \begin{bmatrix}0 & -1 & -2\\2 & -1 & 0\end{bmatrix}$$

Step 2: $\boldsymbol{x}(2) = \boldsymbol{x}_2 = [1 \quad -1]^T, \boldsymbol{y}(2) = \boldsymbol{y}_2 = [-1 \quad 1 \quad -1]^T$

$$\boldsymbol{e}(2) = \boldsymbol{y}(2) - f\left(\boldsymbol{W}^T(2)\boldsymbol{x}(2)\right)$$
$$\rightarrow \boldsymbol{W}^T(2)\boldsymbol{x}(2) = \begin{bmatrix}0 & -1 & -2\\2 & -1 & 0\end{bmatrix}^T \begin{bmatrix}1\\-1\end{bmatrix} = \begin{bmatrix}< \boldsymbol{w}_1^T(2), \boldsymbol{x}(2) >\\< \boldsymbol{w}_2^T(2), \boldsymbol{x}(2) >\\< \boldsymbol{w}_3^T(2), \boldsymbol{x}(2) >\end{bmatrix} = \begin{bmatrix}-2\\0\\-2\end{bmatrix}$$
$$f(x) = 2H(x) - 1, f(\boldsymbol{W}^T\boldsymbol{x}) = \boldsymbol{y}_1 \; if \; \boldsymbol{w}_1^T\boldsymbol{x} = \cdots = \boldsymbol{w}_S^T\boldsymbol{x}$$
$$\rightarrow f\left(\begin{bmatrix}0\\0\\0\end{bmatrix}\right) = \boldsymbol{y}_1 = \begin{bmatrix}-1\\1\\-1\end{bmatrix}$$

$$\boldsymbol{e}(2) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} - \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{W}(3) = \begin{bmatrix} 0 & -1 & -2 \\ 2 & -1 & 0 \end{bmatrix} + 0.5 \begin{bmatrix} 1 \\ -1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -1 & -2 \\ 2 & -1 & 0 \end{bmatrix}$$

Step 3: $\boldsymbol{x}(3) = \boldsymbol{x}_2 = [-1 \quad -1]^T, \boldsymbol{y}(2) = \boldsymbol{y}_2 = [-1 \quad -1 \quad 1]^T$

$$\boldsymbol{e}(3) = \boldsymbol{y}(3) - f\left(\boldsymbol{W}^T(3)\boldsymbol{x}(3)\right)$$

$$\rightarrow \boldsymbol{W}^T(3)\boldsymbol{x}(3) = \begin{bmatrix} 0 & -1 & -2 \\ 2 & -1 & 0 \end{bmatrix}^T \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} < \boldsymbol{w}_1^T(3), \boldsymbol{x}(3) > \\ < \boldsymbol{w}_2^T(3), \boldsymbol{x}(3) > \\ < \boldsymbol{w}_3^T(3), \boldsymbol{x}(3) > \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \\ 2 \end{bmatrix}$$

$$f(x) = 2H(x) - 1, f(\boldsymbol{W}^T\boldsymbol{x}) = \boldsymbol{y}_1 \; if \; \boldsymbol{w}_1^T\boldsymbol{x} = \cdots = \boldsymbol{w}_S^T\boldsymbol{x}$$

$$\rightarrow f\left(\begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}\right) = \boldsymbol{y}_1 = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

$$\boldsymbol{e}(3) = \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} - \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ 0 \end{bmatrix}$$

$$\boldsymbol{W}(4) = \begin{bmatrix} 0 & -1 & -2 \\ 2 & -1 & 0 \end{bmatrix} + 0.5 \begin{bmatrix} -1 \\ -1 \end{bmatrix} \cdot \begin{bmatrix} 0 & -2 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & -1 & -2 \\ 2 & -1 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & -2 \\ 2 & 0 & 0 \end{bmatrix}$$

Final weight at end of epoch 2 equals final weight at end of epoch 1.

$$(1): \eta(i) = \frac{1}{2} > 0$$

$$(2): \sum_{i=1}^{\infty} \eta(i) = \frac{\infty}{2} = \infty$$

$$(3): \lim_{m \to \infty} \frac{\sum_{i=1}^{m} \eta^2(i)}{\left(\sum_{i=1}^{m} \eta(i)\right)^2} = \lim_{m \to \infty} \frac{\frac{m}{4}}{\frac{m^2}{4}} = \lim_{m \to \infty} \frac{1}{m} = 0$$

$$Algorithm\ converges$$

(b) Now redo the previous procedure, but this time deal with each of the columns of $\boldsymbol{W}$ as one perceptron, i.e. update each column (the weight associated with a linear discriminant) separately, for example the first iteration becomes:

$$\boldsymbol{W}(0) = [\boldsymbol{w}_1(0) \quad \boldsymbol{w}_2(0) \quad \boldsymbol{w}_3(0)]$$
$$\boldsymbol{w}_1(1) = \boldsymbol{w}_1(0) + \eta e_1 \boldsymbol{x}(1)$$
$$\boldsymbol{w}_2(1) = \boldsymbol{w}_2(0) + \eta e_2 \boldsymbol{x}(1)$$
$$\boldsymbol{w}_3(1) = \boldsymbol{w}_3(0) + \eta e_3 \boldsymbol{x}(1)$$

Where $e_i = y_{i1} - sign(\boldsymbol{w}_i(0)^T \boldsymbol{x}(1))$ is the difference between the $i^{th}$ element of $\boldsymbol{y}_1$ (the targer vector for $\boldsymbol{x}_1$) and the output of the $i^{th}$ neuron/linear discriminant $\boldsymbol{w}_i(1)$

Perform two epochs only. This is essentially to make you observe that a multicategory Perceptron algorithm is based on multiple binary problems.

Epoch 1:

Step 1: $\boldsymbol{x} = \boldsymbol{x}_1, \boldsymbol{y} = \boldsymbol{y}_1$

$$e_1 = 1 - sign(0) = 0$$
$$e_2 = -1 - sign(0) = -2$$
$$e_3 = -1 - sign(0) = -2$$
$$\boldsymbol{w}_1(1) = \boldsymbol{w}_1(0) + \eta e_1 \boldsymbol{x}(1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.5(0)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
$$\boldsymbol{w}_2(1) = \boldsymbol{w}_2(0) + \eta e_2 \boldsymbol{x}(1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.5(-2)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$
$$\boldsymbol{w}_3(1) = \boldsymbol{w}_3(0) + \eta e_3 \boldsymbol{x}(1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.5(-2)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

Step 2: $\boldsymbol{x} = \boldsymbol{x}_2, \boldsymbol{y} = \boldsymbol{y}_2$

$$e_1 = -1 - sign\left([0 \quad 0] \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = -2$$
$$e_2 = 1 - sign\left([-1 \quad -1] \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = 0$$
$$e_3 = -1 - sign\left([-1 \quad -1] \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = -2$$
$$\boldsymbol{w}_1(2) = \boldsymbol{w}_1(1) + \eta e_1 \boldsymbol{x}(1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.5(-2)\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$
$$\boldsymbol{w}_2(2) = \boldsymbol{w}_2(1) + \eta e_2 \boldsymbol{x}(1) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} + 0.5(0)\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$
$$\boldsymbol{w}_3(2) = \boldsymbol{w}_3(1) + \eta e_3 \boldsymbol{x}(1) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} + 0.5(-2)\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

Step 3: $\boldsymbol{x} = \boldsymbol{x}_3, \boldsymbol{y} = \boldsymbol{y}_3$

$$e_1 = -1 - sign\left([-1 \quad 1] \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right) = -1 - sign(0) = -2$$
$$e_2 = -1 - sign\left([-1 \quad -1] \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right) = -1 - sign(2) = -2$$
$$e_3 = 1 - sign\left([-2 \quad 0] \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right) = 1 - sign(2) = 0$$
$$\boldsymbol{w}_1(3) = \boldsymbol{w}_1(2) + \eta e_1 \boldsymbol{x}(2) = \begin{bmatrix} -1 \\ 1 \end{bmatrix} + 0.5(-2)\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

$$\boldsymbol{w}_2(3) = \boldsymbol{w}_2(2) + \eta e_2 \boldsymbol{x}(2) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} + 0.5(-2)\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{w}_3(3) = \boldsymbol{w}_3(2) + \eta e_3 \boldsymbol{x}(2) = \begin{bmatrix} -2 \\ 0 \end{bmatrix} + 0.5(0)\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

Epoch 1: $\boldsymbol{w}_1(0) = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \boldsymbol{w}_2(0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \boldsymbol{w}_3(0) = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$

Step 1: $\boldsymbol{x} = \boldsymbol{x}_1, \boldsymbol{y} = \boldsymbol{y}_1$

$$e_1 = 1 - sign\left([0 \quad 2] \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) = 1 - sign(2) = 0$$

$$e_2 = -1 - sign\left([0 \quad 0] \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) = -1 - sign(0) = -2$$

$$e_3 = -1 - sign\left([-2 \quad 0] \cdot \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) = -1 - sign(-2) = 0$$

$$\boldsymbol{w}_1(1) = \boldsymbol{w}_1(0) + \eta e_1 \boldsymbol{x}(1) = \begin{bmatrix} 0 \\ 2 \end{bmatrix} + 0.5(0)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

$$\boldsymbol{w}_2(1) = \boldsymbol{w}_2(0) + \eta e_2 \boldsymbol{x}(1) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 0.5(-2)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

$$\boldsymbol{w}_3(1) = \boldsymbol{w}_3(0) + \eta e_3 \boldsymbol{x}(1) = \begin{bmatrix} -2 \\ 0 \end{bmatrix} + 0.5(0)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

Step 2: $\boldsymbol{x} = \boldsymbol{x}_2, \boldsymbol{y} = \boldsymbol{y}_2$

$$e_1 = -1 - sign\left([0 \quad 2] \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = -1 - sign(-2) = 0$$

$$e_2 = 1 - sign\left([-1 \quad -1] \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = 1 - sign(0) = 0$$

$$e_3 = -1 - sign\left([-2 \quad 0] \cdot \begin{bmatrix} 1 \\ -1 \end{bmatrix}\right) = -1 - sign(-2) = 0$$

$$\boldsymbol{w}_1(2) = \boldsymbol{w}_1(1) + \eta e_1 \boldsymbol{x}(1) = \begin{bmatrix} 0 \\ 2 \end{bmatrix} + 0.5(0)\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

$$\boldsymbol{w}_2(2) = \boldsymbol{w}_2(1) + \eta e_2 \boldsymbol{x}(1) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} + 0.5(0)\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

$$\boldsymbol{w}_3(2) = \boldsymbol{w}_3(1) + \eta e_3 \boldsymbol{x}(1) = \begin{bmatrix} -2 \\ 0 \end{bmatrix} + 0.5(-2)\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

Step 3: $\boldsymbol{x} = \boldsymbol{x}_3, \boldsymbol{y} = \boldsymbol{y}_3$

$$e_1 = -1 - sign\left([0 \quad 2] \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right) = -1 - sign(-2) = 0$$

$$e_2 = -1 - sign\left([-1 \quad -1] \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right) = -1 - sign(2) = -2$$

$$e_3 = 1 - sign\left([-2 \quad 0] \cdot \begin{bmatrix} -1 \\ -1 \end{bmatrix}\right) = 1 - sign(2) = 0$$

$$\boldsymbol{w}_1(3) = \boldsymbol{w}_1(2) + \eta e_1 \boldsymbol{x}(2) = \begin{bmatrix} 0 \\ 2 \end{bmatrix} + 0.5(0)\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$$

$$\boldsymbol{w}_2(3) = \boldsymbol{w}_2(2) + \eta e_2 \boldsymbol{x}(2) = \begin{bmatrix} -1 \\ -1 \end{bmatrix} + 0.5(-2)\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{w}_3(3) = \boldsymbol{w}_3(2) + \eta e_3 \boldsymbol{x}(2) = \begin{bmatrix} -2 \\ 0 \end{bmatrix} + 0.5(0)\begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

$$Final\ \boldsymbol{w}_1 = \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \boldsymbol{w}_2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \boldsymbol{w}_3 = \begin{bmatrix} -2 \\ 0 \end{bmatrix}$$

3. Consider the two classes of patterns that are shown in the figure below. Design a multilayer neural network with the following architecture to distinguish these categories (30 pts)

Assuming the patterns are:

$$\begin{matrix} x_1 & x_2 \\ x_3 & x_4 \end{matrix} \to \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} : x_i \in \{0,1\}$$

$$\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \; in\ \omega_1, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \; in\ \omega_2$$

For the first hidden layer

$$\begin{aligned} \boldsymbol{w}_1^{(1)} &= [1 \quad 1 \quad 0 \quad 0]^T \\ \boldsymbol{w}_2^{(1)} &= [0 \quad 0 \quad 1 \quad 1]^T \end{aligned}$$

For each neuron in the hidden layer:

$$\begin{aligned} \boldsymbol{w}^T{}_1^{(1)}\boldsymbol{x} + w_{10}^{(1)} &= x_1 + x_2 + w_{10}^{(1)} = x_1 + x_2 - 1 \to x_1 \wedge x_2 \\ \boldsymbol{w}^T{}_2^{(1)}\boldsymbol{x} + w_{20}^{(1)} &= x_1 + x_2 + w_{20}^{(1)} = x_3 + x_4 - 1 \to x_3 \wedge x_4 \end{aligned}$$

Because inputs are binary and we desire an output to be binary, use step function as activation function, producing these inputs to the final hidden layer.

$$\begin{bmatrix} H(x_1 + x_2 - 1) \\ H(x_3 + x_4 - 1) \end{bmatrix}$$

With weight:

$$\boldsymbol{w}_1^{(2)} = [1 \quad 1]^T$$

Producing the output:

$$H(x_1 + x_2 - 1) + H(x_3 + x_4 - 1) =$$

Again, using the Heaviside step function as the activation function, we have the final output.

$$\begin{aligned} &y = H\big(H(x_1 + x_2 - 1) + H(x_3 + x_4 - 1)\big) \\ &y = 0 \to \boldsymbol{x} \in \omega_1 \\ &y = 1 \to \boldsymbol{x} \in \omega_2 \end{aligned}$$

4. Programming Assignment: Parkinsons Telemonitoring

(a) Download the Parkinsons Telemonitoring Data Set from: http://archive.ics.scu.edu/ml/datasets/Parkinsons+Telemonitoring .Choose 70% of the data randomly as the training set.

(b) Use metric learning with Gaussian kernels to estimate each of the outputs motor UPDRS and total UPDRS from the features. As metric leaning uses a low dimensional transformation of the features except the non-predictive feature subject#, use 5-fold cross-validation to decide the number of components form $M = 5{,}10{,}15, p$, where $p$ is the number of all predictive features you can use. Initialize the linear transformation with PCA features for $M = 5, 10, 15$ and with original features for $M = p$. This corresponds to setting intit as (default='auta'). Remember to standardize your features. Report the $R^2$ on training and test sets for each of the outputs. (30 pts)

(c) Use sklearn's neural network implementation to train a neural network with two outputs that predicts motor UPDRS and total UPDRS. Use a single layer. You are responsible to determine other architectural parameters of the network, including the number of neurons in the hidden and output layers, method of optimization, type of activation functions, and the L2 "regularization" parameter etc. You should determine the design parameters via trial and error, by testing your trained network on the test set and choosing the architecture that yields the smallest test error. For this part, set early-stopping=False. Remember to standardize your features. Report your $R^2$ on both training and test sets. (20 pts)

(d) Use the design parameters that you chose in the first part and train a neural network, but this time set early-stopping=True. Research what early stopping is, and compare the performance of you network on the test set with the previous network. You can leave the validation-fraction as the default (0.1) or change it to see whether you can obtain a better model. Remember to standardize your features. Report your $R^2$ on both training and test sets. (10 pts)

Note: there are a lot of design parameters in a neural network. If you are not sure how they work, just set them as the default of sklearn, but if you use them masterfully, you can have better models.