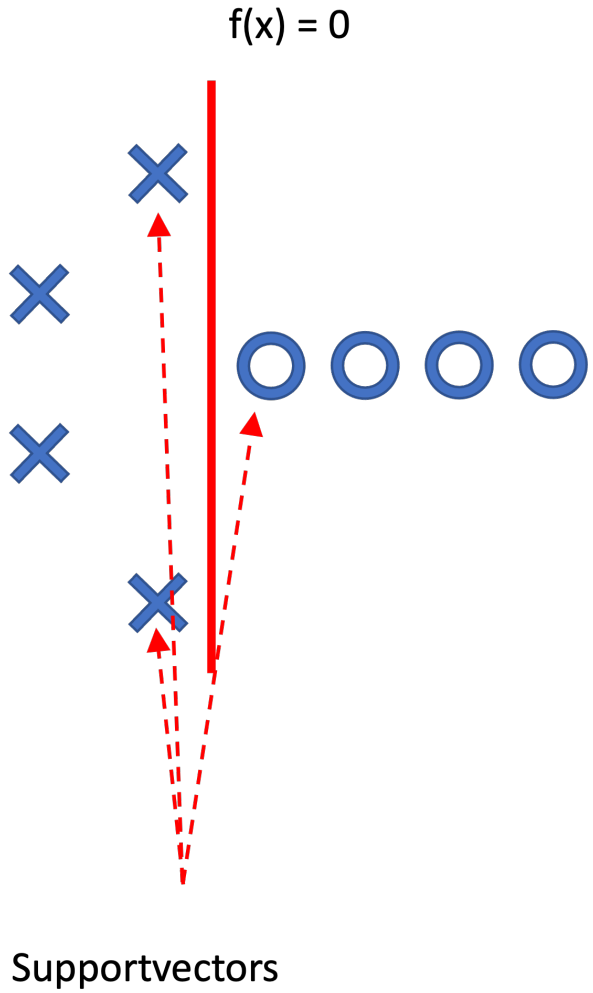
Solutions to the exam in
Neural Networks and Learning Systems - TBMI26 / 732A55
Exam 2021-08-28

Part 1

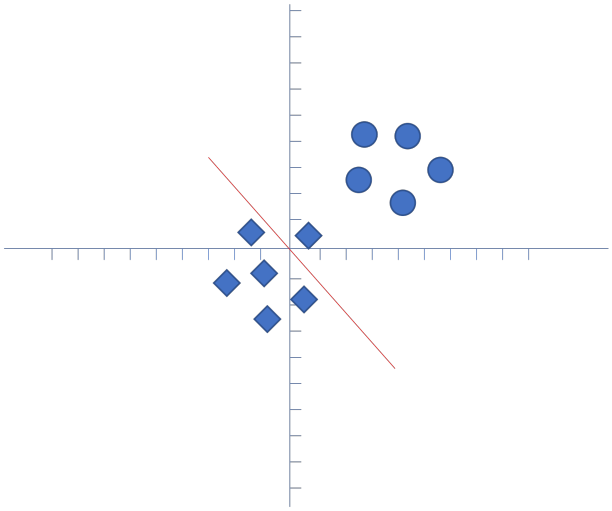
1. The following methods are unsupervised learning methods:

   - Mixture of Gaussians
   - k-means
   - PCA

2. See figure:



3. See figure:



4. The principal components are uncorrelated.
5. Image segmentation
6. The k-means algorithm is used for clustering.
7. By data augmentation, i.e. rotating, scaling or shifting the images.
8. By uisng the "kernel trick".
9. By monitoring the error on test data and stop when this error starts to increase.
10. It describes the expected accumulated (discounted) future reward for each state.

11. The "vanishing gradient" problem refers to problem of propagating the error back in deep neural networks when using sigmoid activation functions, which has a small derivative almost everywhere. To avoid this, we can us alternative activation functions such as the ReLU function.

12. Auto-encoders use unsupervised learning. They can be used e.g. for outlier detection.

13. A kernel function defines the inner product $\kappa(\mathbf{x}_1, \mathbf{x}_2) = \Phi(\mathbf{x}_1)^T \Phi(\mathbf{x}_2)$ in the new feature space. Thus, $\kappa(\mathbf{x}_1, \mathbf{x}_2)$ specifies the feature space by defining how distances and angles are measured, instead of explicitly stating the mapping function $\Phi(\mathbf{x})$.

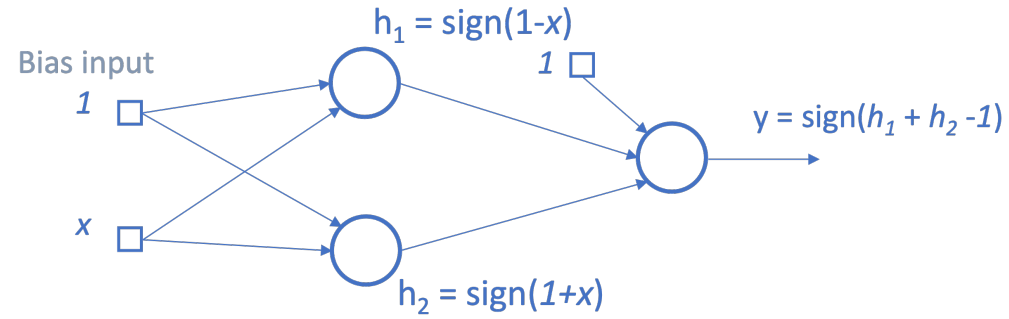
    The distance between $\mathbf{x}_1$ and $\mathbf{x}_2$ in the new feature space is

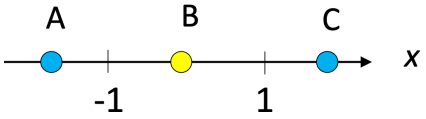
$$\begin{aligned}
\|\Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_2)\| &= \sqrt{(\Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_2))^T (\Phi(\mathbf{x}_1) - \Phi(\mathbf{x}_2))} \\
&= \sqrt{\Phi(\mathbf{x}_1)^T \Phi(\mathbf{x}_1) - 2\Phi(\mathbf{x}_1)^T \Phi(\mathbf{x}_2) + \Phi(\mathbf{x}_2)^T \Phi(\mathbf{x}_2)} \\
&= \sqrt{\kappa(\mathbf{x}_1, \mathbf{x}_1) - 2\,\kappa(\mathbf{x}_1, \mathbf{x}_2) + \kappa(\mathbf{x}_2, \mathbf{x}_2)} \\
&= \sqrt{1 - 2\,e^{-\frac{1}{4}} + 1} = 0.6651
\end{aligned}$$

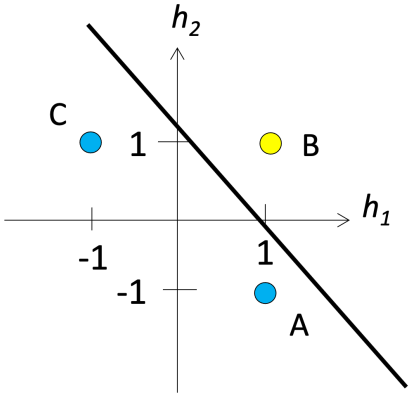
14. For example this:

-1	2	-1
-1	2	-1
-1	2	-1

15. See figure:







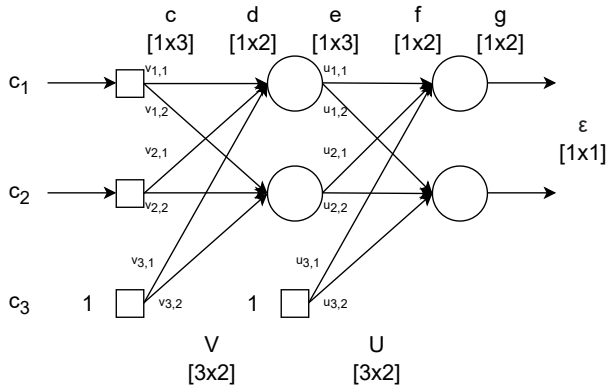


Figure 1: Neural network diagram a).

Part 2

1. a) Figure 1 shows a labeled diagram of the first network. We choose this naming for the variables to facilitate their reuse for b). We define the variables at each stage of the network:

$$c_j = i\text{-th input}, \quad d_k = \sum_{j=1}^{3} c_j v_{j,k},$$

$$e_k = \begin{cases} \max(0, d_k) & k = \{1, 2\}, \\ 1, & k = 3 \end{cases} \quad f_l = \sum_{k=1}^{3} e_k u_{k,l},$$

$$\epsilon = \sum_{l=1}^{2} (g_l - y_l)^2,$$

where y_l represents the target value for the l-th output.

From the above definitions, we calculate all partial derivatives between contiguous stages of the network:

$$\frac{\partial \epsilon}{\partial g_l} = \frac{\partial}{\partial g_l} \sum_{l=1}^{2} (g_l - y_l)^2 = 2(g_l - y_l),$$

$$\frac{\partial g_l}{\partial f_l} = \frac{\partial}{\partial f_l} f_l = 1,$$

$$\frac{\partial f_l}{\partial e_k} = \frac{\partial}{\partial e_k} \sum_{k=1}^{3} e_k u_{k,l} = u_{k,l}, \quad \frac{\partial f_l}{\partial u_{k,l}} = \frac{\partial}{\partial u_{k,l}} \sum_{k=1}^{3} e_k u_{k,l} = e_k,$$

$$\frac{\partial e_k}{\partial d_k} = \frac{\partial}{\partial d_k} \max(0, d_k) = \text{step}(d_k), \quad k = 1, 2$$

$$\frac{\partial d_k}{\partial c_j} = \frac{\partial}{\partial c_j} \sum_{j=1}^{3} c_j v_{j,k} = v_{j,k}, \quad \frac{\partial d_k}{\partial v_{j,k}} = \frac{\partial}{\partial v_{j,k}} \sum_{j=1}^{3} c_j v_{j,k} = c_j.$$

Using these, we can find the gradient of the loss with respect to the weight matrices:

$$\frac{\partial \epsilon}{\partial u_{k,l}} = \frac{\partial \epsilon}{\partial g_l} \frac{\partial g_l}{\partial f_l} \frac{\partial f_l}{\partial u_{k,l}} = 2(g_l - y_l) e_k,$$

$$\frac{\partial \epsilon}{\partial v_{j,k}} = \left(\sum_{l=1}^{2} \frac{\partial \epsilon}{\partial g_l} \frac{\partial g_l}{\partial f_l} \frac{\partial f_l}{\partial e_k} \right) \frac{\partial e_k}{\partial d_k} \frac{\partial d_k}{\partial v_{j,k}} = \left(\sum_{l=1}^{2} 2(g_l - y_l) u_{k,l} \right) \text{step}(d_k) c_j,$$

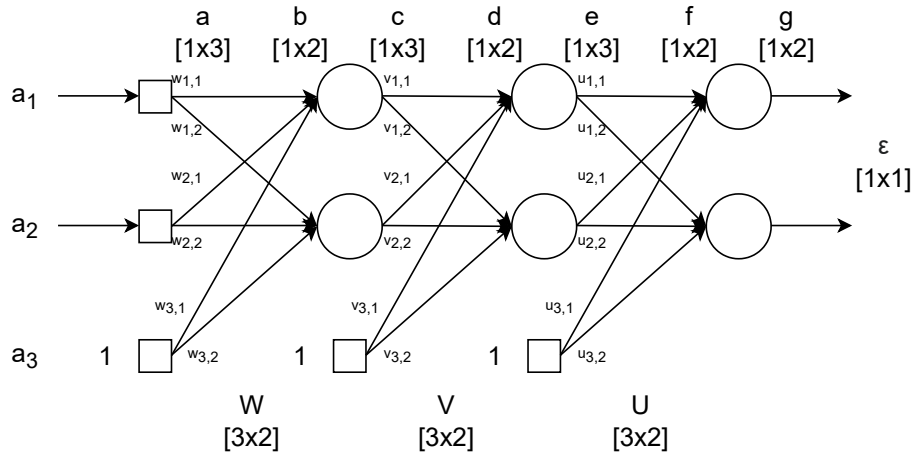


Figure 2: Neural network diagram b).

and the weight update rules are

$$u_{k,l} \leftarrow u_{k,l} - \eta \frac{\partial \epsilon}{\partial u_{k,l}},$$
$$v_{j,k} \leftarrow v_{j,k} - \eta \frac{\partial \epsilon}{\partial v_{j,k}}.$$

b) Figure 2 shows the new network, which has one more hidden layer than the previous. Due to our naming convention, we can reuse most of the definitions and results from a). We define the new variables in the network:

$$a_i = i\text{-th input}, \quad b_j = \sum_{i=1}^{3} a_i w_{i,j},$$
$$c_j = \begin{cases} \max(0, b_j) & j = \{1, 2\}, \\ 1, & j = 3 \end{cases}$$

We calculate the new partial derivatives between contiguous stages of the network:

$$\frac{\partial c_j}{\partial b_j} = \frac{\partial}{\partial b_j} \max(0, b_j) = \text{step}(b_j), \quad j = 1, 2$$
$$\frac{\partial b_j}{\partial a_i} = \frac{\partial}{\partial a_i} \sum_{i=1}^{3} a_i w_{i,j} = w_{i,j}, \quad \frac{\partial b_j}{\partial w_{i,j}} = \frac{\partial}{\partial w_{i,j}} \sum_{i=1}^{3} a_i w_{i,j} = a_i.$$

The gradients of the loss with respect ton the weight matrices are

$$\frac{\partial \epsilon}{\partial u_{k,l}} = \frac{\partial \epsilon}{\partial g_l} \frac{\partial g_l}{\partial f_l} \frac{\partial f_l}{\partial u_{k,l}} = 2(g_l - y_l) e_k,$$
$$\frac{\partial \epsilon}{\partial v_{j,k}} = \left(\sum_{l=1}^{2} \frac{\partial \epsilon}{\partial g_l} \frac{\partial g_l}{\partial f_l} \frac{\partial f_l}{\partial e_k} \right) \frac{\partial e_k}{\partial d_k} \frac{\partial d_k}{\partial v_{j,k}} = \left(\sum_{l=1}^{2} 2(g_l - y_l) u_{k,l} \right) \text{step}(d_k) c_j,$$
$$\frac{\partial \epsilon}{\partial w_{i,j}} = \left(\sum_{k=1}^{2} \left(\sum_{l=1}^{2} \frac{\partial \epsilon}{\partial g_l} \frac{\partial g_l}{\partial f_l} \frac{\partial f_l}{\partial e_k} \right) \frac{\partial e_k}{\partial d_k} \frac{\partial d_k}{\partial c_j} \right) \frac{\partial c_j}{\partial b_j} \frac{\partial b_j}{\partial w_{i,j}}$$
$$= \left(\sum_{k=1}^{2} \left(\sum_{l=1}^{2} 2(g_l - y_l) u_{k,l} \right) \text{step}(d_k) v_{j,k} \right) \text{step}(b_j) a_i$$

and the weight update rules are

$$u_{k,l} \leftarrow u_{k,l} - \eta \frac{\partial \epsilon}{\partial u_{k,l}},$$
$$v_{j,k} \leftarrow v_{j,k} - \eta \frac{\partial \epsilon}{\partial v_{j,k}},$$
$$w_{i,j} \leftarrow w_{i,j} - \eta \frac{\partial \epsilon}{\partial w_{i,j}},$$

2. a) Figure 3(a) shows the state of the problem after the second AdaBoost iteration and the results of the third AdaBoost iteration. After selecting the threshold yielding minimum error we get $\epsilon_3 = \frac{7}{22}$ and $\alpha_3 = \frac{1}{2}\ln\frac{1-\epsilon_3}{\epsilon_3} = \frac{1}{2}\ln\frac{15/22}{7/22} = \frac{\ln 15/7}{2}$. The weights of the correctly classified samples are multiplied by $e^{-\alpha_3} = \frac{1}{\sqrt{15/7}}$, while the weight of those of the misclassified samples are multiplied by $e^{\alpha_3} = \sqrt{15/7}$. The sum of the weights is $\frac{15}{22}\frac{1}{\sqrt{15/7}} + \frac{7}{22}\sqrt{15/7} = \frac{7}{11}\sqrt{15/7}$, and after normalizing they take the values shown in Figure 3(a).

   b) We have the following data:

$$\mathbf{C} = \begin{bmatrix}\mathbf{c}_1\\ \mathbf{c}_2\\ \mathbf{c}_3\end{bmatrix} = \begin{bmatrix}-1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\\ 1 & 1 & 1 & 1 & -1 & 1 & 1 & 1\\ -1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\end{bmatrix}$$

$$\boldsymbol{\alpha} = \begin{bmatrix}\alpha_1\\ \alpha_2\\ \alpha_3\end{bmatrix} = \begin{bmatrix}0.97\\ 0.65\\ 0.38\end{bmatrix}$$

The strong classifier results are shown in Figure 3(c). The strong classifier $\mathbf{H}_i$ is defined as $sign\left(\sum_i \alpha_i \mathbf{c}_i\right)$ where $\mathbf{c}_i$ are the classifications by the weak classifiers.

$$\mathbf{H}_1 = sign(\alpha_1\mathbf{c}_1) = \begin{bmatrix}-1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\end{bmatrix}$$

$$\mathbf{H}_2 = sign(\sum_{i=1}^{2}\alpha_i\mathbf{c}_i) = \begin{bmatrix}-1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\end{bmatrix}$$

$$\mathbf{H}_3 = sign(\sum_{i=1}^{3}\alpha_i\mathbf{c}_i) = \begin{bmatrix}-1 & 1 & 1 & 1 & -1 & -1 & -1 & -1\end{bmatrix}$$

All three strong classifiers produce the same classification, giving an accuracy of $\frac{7}{8}$.

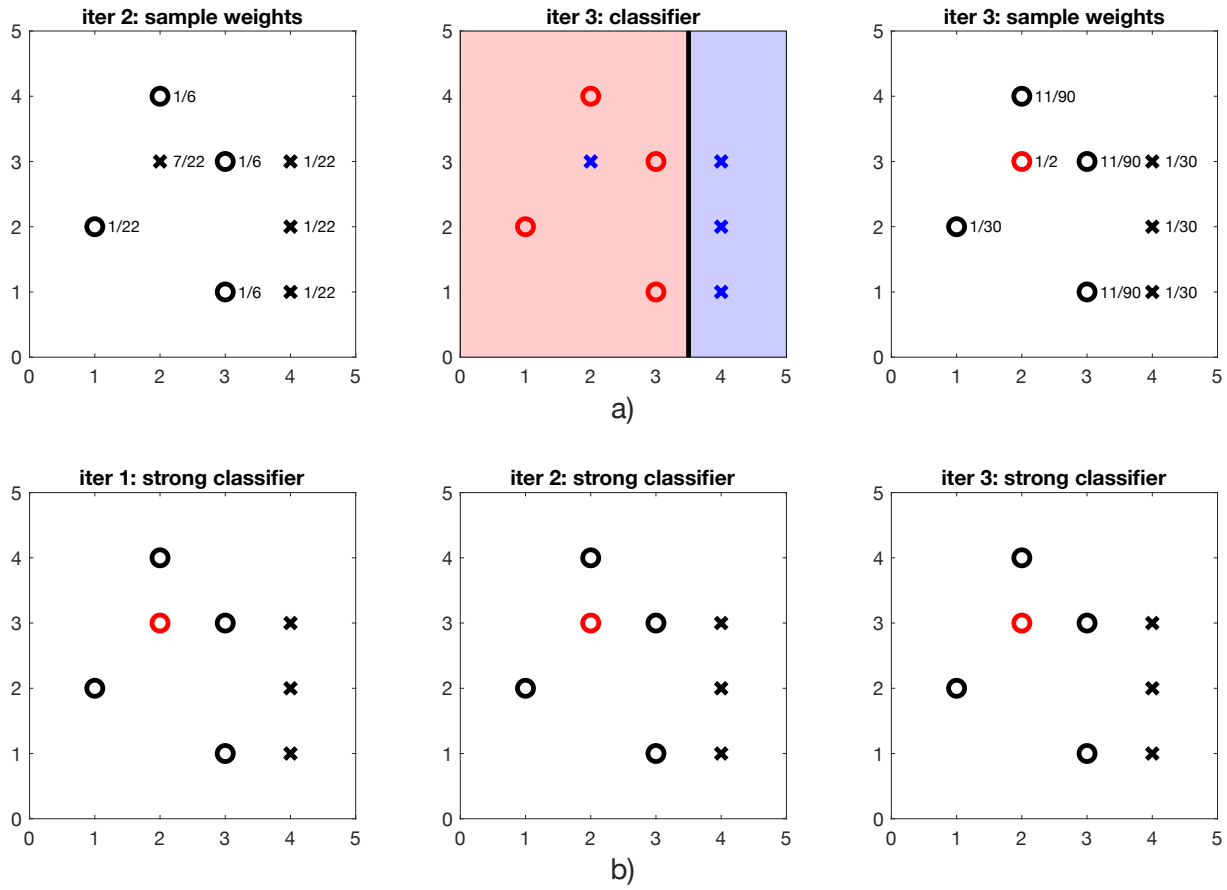


Figure 3: a) Left: initial weights at the start of iteration 3. Middle: optimal decision stump for iteration 3. Blue crosses and red circles represent the positive and negative classes, respectively. Right: updated weights at the end of iteration 3. Black and red indicate correct and incorrect classifications, respectively. b) Strong classifier results based on 1, 2, and 3 weak classifiers. Black and red indicate correct and incorrect classifications, respectively.

3. The direction that optimally separates the two classes is given by

$$\mathbf{w} = \mathbf{C}_{\text{tot}}^{-1} (\mathbf{m}_{\text{x}} - \mathbf{m}_{\text{o}})$$

where $\mathbf{C}_{\text{tot}}$ is the sum of the individual covariance matrices for the two respective classes, and $\mathbf{m}_{\text{x}}$ and $\mathbf{m}_{\text{o}}$ are the centers of the two classes. We begin with the "crosses" class:

$$\mathbf{X}_{\text{x}} = \begin{pmatrix} -2 & -1 & 0 & -1 \\ -2 & -1 & -1 & 0 \end{pmatrix}$$

$$\mathbf{C}_{\text{x}} = \frac{1}{N_{\text{x}} - 1} (\mathbf{X}_{\text{x}} - \mathbf{m}_{\text{x}}) (\mathbf{X}_{\text{x}} - \mathbf{m}_{\text{x}})^T = \left[\mathbf{m}_{\text{x}} = \begin{pmatrix} -1 \\ -1 \end{pmatrix} \right] = \frac{1}{3} \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Now for the "circles" class:

$$\mathbf{X}_{\text{o}} = \begin{pmatrix} -1 & 1 & 2 & 2 \\ 1 & 0 & 1 & -2 \end{pmatrix}$$

$$\mathbf{C}_{\text{o}} = \frac{1}{N_{\text{o}} - 1} (\mathbf{X}_{\text{o}} - \mathbf{m}_{\text{o}}) (\mathbf{X}_{\text{o}} - \mathbf{m}_{\text{o}})^T = \left[\mathbf{m}_{\text{o}} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right] = \frac{1}{3} \begin{pmatrix} 6 & 1 \\ 1 & 2 \end{pmatrix}$$

We now get:

$$\mathbf{C}_{\text{tot}} = \mathbf{C}_{\text{x}} + \mathbf{C}_{\text{o}} = \frac{2}{3} \begin{pmatrix} 4 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\mathbf{C}_{\text{tot}}^{-1} = \frac{1}{\left(\frac{2}{3}\right)^2 (4 * 2 - 1 * 1)} * \frac{2}{3} \begin{pmatrix} 2 & -1 \\ -1 & 4 \end{pmatrix} = \frac{3}{14} \begin{pmatrix} 2 & -1 \\ -1 & 4 \end{pmatrix}$$

$$\mathbf{w} = \mathbf{C}_{\text{tot}}^{-1} (\mathbf{m}_{\text{x}} - \mathbf{m}_{\text{o}}) = \frac{-3}{7} \begin{pmatrix} 1 \\ 3 \end{pmatrix}$$

Normalize $\mathbf{w}$:

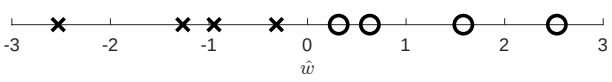
$$\hat{\mathbf{w}} = \frac{\mathbf{w}}{||\mathbf{w}||_2} = \frac{1}{\sqrt{10}} \begin{pmatrix} 1 \\ 3 \end{pmatrix}$$

Project the data on $\hat{\mathbf{w}}$ which gives the following:

$$\mathbf{Y}_{\text{x}} = \hat{\mathbf{w}}^T \mathbf{X}_{\text{x}} = \frac{1}{\sqrt{10}} \begin{pmatrix} -8 & -4 & -3 & -1 \end{pmatrix}$$

$$\mathbf{Y}_{\text{o}} = \hat{\mathbf{w}}^T \mathbf{X}_{\text{o}} = \frac{1}{\sqrt{10}} \begin{pmatrix} 2 & 1 & 5 & 8 \end{pmatrix}$$

The projected data is shown in the figure below.



4. The update of Q-value in state S_k and action a_j is given by:

$$\hat{Q}(S_k, a_j) \leftarrow (1-\alpha)\hat{Q}(S_k, a_j) + \alpha(r + \gamma \max_a \hat{Q}(S_k, a))$$

We can update the values of the Q-function by following the sequence of actions. Starting with *Sequence 1*:

$$\begin{aligned}
\hat{Q}(1, up) &= (1-\alpha)\hat{Q}(1, up) + \alpha(r + \gamma \max_a \hat{Q}(1)) \\
&= (1-\alpha)0 + \alpha(0 + \gamma 0) = 0
\end{aligned}$$

$$\begin{aligned}
\hat{Q}(3, left) &= (1-\alpha)\hat{Q}(3, left) + \alpha(r + \gamma \max_a \hat{Q}(4)) \\
&= (1-\alpha)0 + \alpha(0 + \gamma 0) = 0
\end{aligned}$$

$$\begin{aligned}
\hat{Q}(4, up) &= (1-\alpha)\hat{Q}(4, up) + \alpha(r + \gamma \max_a \hat{Q}(6)) \\
&= (1-\alpha)0 + \alpha(0 + \gamma 0) = 0
\end{aligned}$$

$$\begin{aligned}
\hat{Q}(6, up) &= (1-\alpha)\hat{Q}(6, up) + \alpha(r + \gamma \max_a \hat{Q}(7)) \\
&= (1-\alpha)0 + \alpha(5 + \gamma 0) = 5\alpha
\end{aligned}$$

Following now *Sequence 2* of actions:

$$\begin{aligned}
\hat{Q}(1, up) &= (1-\alpha)\hat{Q}(1, up) + \alpha(r + \gamma \max_a \hat{Q}(3)) \\
&= (1-\alpha)0 + \alpha(0 + \gamma 0) = 0
\end{aligned}$$

$$\begin{aligned}
\hat{Q}(3, up) &= (1-\alpha)\hat{Q}(3, up) + \alpha(r + \gamma \max_a \hat{Q}(5)) \\
&= (1-\alpha)0 + \alpha(-10 + \gamma 0) = -10\alpha
\end{aligned}$$

$$\begin{aligned}
\hat{Q}(5, left) &= (1-\alpha)\hat{Q}(5, left) + \alpha(r + \gamma \max_a \hat{Q}(6)) \\
&= (1-\alpha)0 + \alpha(0 + 5\alpha\gamma) \\
&= 5\alpha^2\gamma
\end{aligned}$$

$$\begin{aligned}
\hat{Q}(6, up) &= (1-\alpha)\hat{Q}(6, up) + \alpha(r + \gamma \max_a \hat{Q}(7)) \\
&= (1-\alpha)5\alpha + \alpha(5 + \gamma 0) \\
&= 10\alpha - 5\alpha^2
\end{aligned}$$

Following *Sequence 2* of actions again:

$$\begin{aligned}
\hat{Q}(1, up) &= (1-\alpha)\hat{Q}(1, up) + \alpha(r + \gamma \max_a \hat{Q}(3)) \\
&= (1-\alpha)0 + \alpha(0 + \gamma \max(0, -10\alpha)) \\
&= 0
\end{aligned}$$

$$
\begin{aligned}
\hat{Q}(3, up) &= (1-\alpha)\hat{Q}(3, up) + \alpha(r + \gamma \max_a \hat{Q}(5)) \\
&= (1-\alpha)(-10\alpha) + \alpha(-10 + \gamma(5\alpha^2\gamma)) \\
&= -20\alpha + 10\alpha^2 + 5\alpha^3\gamma^2
\end{aligned}
$$

$$
\begin{aligned}
\hat{Q}(5, left) &= (1-\alpha)\hat{Q}(5, left) + \alpha(r + \gamma \max_a \hat{Q}(6)) \\
&= (1-\alpha)(5\alpha^2\gamma) + \alpha(0 + \gamma(10\alpha - 5\alpha^2)) \\
&= 15\alpha^2\gamma - 10\alpha^3\gamma
\end{aligned}
$$

$$
\begin{aligned}
\hat{Q}(6, up) &= (1-\alpha)\hat{Q}(6, up) + \alpha(r + \gamma \max_a \hat{Q}(7)) \\
&= (1-\alpha)(10\alpha - 5\alpha^2) + \alpha(5 + \gamma 0) \\
&= 15\alpha - 15\alpha^2 + 5\alpha^3
\end{aligned}
$$