Artificial Neural Networks (Gerstner). Solutions for week 1

Simple Perceptrons, Geometric interpretation, Discriminant function

Exercise 1. Gradient of quadratic error function

We define the mean square error in a data base with P patterns as

$$E^{\mathrm{MSE}}(\boldsymbol{w}) = \frac{1}{2}\frac{1}{P}\sum_{\mu}[t^{\mu} - \hat{y}^{\mu}]^2 \tag{1}$$

where the output is

$$\hat{y}^{\mu} = g(a^{\mu}) = g(\boldsymbol{w}^T\boldsymbol{x}^{\mu}) = g(\sum_k w_k x_k^{\mu}) \tag{2}$$

and the input is the pattern $\boldsymbol{x}^{\mu}$ with components $x_1^{\mu} \dots x_N^{\mu}$.

a. Calculate the update of weight w_j by gradient descent (batch rule)

$$\Delta w_j = -\eta \frac{dE}{dw_j} \tag{3}$$

Hint: Apply chain rule

b. Rewrite the formula by taking one pattern at a time (stochastic gradient descent). What is the difference to the batch rule?

c. Rewrite your result in b in vector notation (hint: use the weight vector $\boldsymbol{w}$ and the input vector $\boldsymbol{x}^{\mu}$). Show that the update after application of pattern μ can be written as

$$\Delta \boldsymbol{w} = \eta\delta(\mu)\boldsymbol{x}^{\mu}$$

where $\delta(\mu)$ is a scalar number that depends on μ. Express $\delta(\mu)$ in terms of $t^{\mu}, \hat{y}^{\mu}, g'$.

Solution:

a.

$$\begin{aligned}\frac{dE}{dw_j} &= \frac{d}{dw_j}\frac{1}{2}\frac{1}{P}\sum_{\mu}\left[t^{\mu} - g\left(\sum_k w_k x_k^{\mu}\right)\right]^2 \\ &= \frac{1}{2}\frac{1}{P}\sum_{\mu} 2\,[t^{\mu} - \hat{y}^{\mu}]\,\frac{d}{dw_j}\left[t^{\mu} - g\left(\sum_k w_k x_k^{\mu}\right)\right] \\ &= \frac{1}{P}\sum_{\mu}[t^{\mu} - g(a^{\mu})]\,(-x_j^{\mu})g'(a^{\mu})\end{aligned}$$

hence

$$\Delta w_j = \eta\,\frac{1}{P}\sum_{\mu}[t^{\mu} - g(a^{\mu})]\,x_j^{\mu} g'(a^{\mu})$$

b.

$$\Delta w_j(\mu) = \eta\,[t^{\mu} - g(a^{\mu})]\,x_j^{\mu} g'(a^{\mu})$$

Here, the weight update is performed on the error signal of a single pattern. While in (a), the error signal is averaged over all patterns. From a geometric perspective, the batch rule updates the separation hyperplane in the direction so that the hyperplane linearly separates all points. On the other hand, the stochastic *online* rule, will update the hyperplane so that a single randomly chosen point is pushed on the correct side of the hyperplane. Looking at the hyperplane updates, the stochastic rule leads to more noisy updates (the hyperplane moves in many different direction depending on the selected example) than the batch rule.

c. We know that $\boldsymbol{w} = [w_1, \ldots, w_d]$ where d is the dimensionality of the input space. $\Delta\boldsymbol{w} = [\Delta w_1, \ldots, \Delta w_d]$ and therefore the update rule in b in the vector form can be written as

$$\Delta\boldsymbol{w} = \eta \left[[t^\mu - g(a^\mu)]\, x_1^\mu g'(a^\mu), \ldots, [t^\mu - g(a^\mu)]\, x_d^\mu g'(a^\mu) \right].$$

Taking the common scalar terms outside and substituting $g(a^\mu)$ with $\hat{y}^\mu$, we have

$$\Delta\boldsymbol{w} = \eta\ [t^\mu - \hat{y}^\mu]\, g'(a^\mu) x^\mu$$

and thus $\delta(\mu) = [t^\mu - \hat{y}^\mu]\, g'(a^\mu)$.

Exercise 2. Perceptron Algorithm as stochastic gradient descent

We work in $n+1$ dimensions (in other words the threshold is integrated in the weight vector). We define the Heaviside step function $\theta(x) = 1$ for $x > 0$ and zero otherwise. We define $\tilde{t}^\mu = 2(t^\mu - 0.5)$. Hence $\tilde{t} = \pm 1$.

The classic perceptron has a gain function $g(a) = \theta(a)$.

Show that the perceptron algorithm performs stochastic gradient descent on the (piecewise) linear error function:

$$E^{\text{perc}}(\boldsymbol{w}) = -\sum_\mu \left[\tilde{t}^\mu a^\mu\right] \theta\left[-\tilde{t}^\mu a^\mu\right] \tag{4}$$

where $a^\mu = \sum_k w_k x_k^\mu$.

Hint: Show first that the error function vanishes for all patterns that are correctly classified. It is non–negative for misclassified patterns. Then calculate the gradient for one of the misclassified pattern.

Solution:

We first express the error function in terms of correctly classified patterns μ_c and misclassified patterns μ_m:

$$\begin{aligned} E^{\text{perc}}(\mathbf{w}) &= -\sum_{\mu_c} \left[\tilde{t}^{\mu_c} a^{\mu_c}\right] \theta\left[-\tilde{t}^{\mu_c} a^{\mu_c}\right] - \sum_{\mu_m} \left[\tilde{t}^{\mu_m} a^{\mu_m}\right] \theta\left[-\tilde{t}^{\mu_m} a^{\mu_m}\right] \\ &= -\sum_{\mu_m} \left[\tilde{t}^{\mu_m} a^{\mu_m}\right] \end{aligned}$$

where the first term vanishes because, for correctly classified patterns, $\tilde{t}^{\mu_c} a^{\mu_c} \geq 0$ and therefore $\theta\left[-\tilde{t}^{\mu_c} a^{\mu_c}\right] = 0$. For the remaining misclassified terms, $\tilde{t}^{\mu_m} a^{\mu_m} \leq 0$. Therefore, the overall error will always be greater than or equal to 0.

For stochastic gradient descent, we consider the component of the error due to a single pattern μ and take the gradient with respect to the weights, giving us

$$\Delta E^{\text{perc},\mu}(\mathbf{w}) = \begin{cases} 0 & \text{if } \mu \in \mu_c \quad \text{and} \\ -\tilde{t}^\mu \mathbf{x}^\mu & \text{if } \mu \in \mu_m \ . \end{cases}$$

Noting that $t^{\mu_c} - \theta[a^{\mu_c}] = 0$ for correctly classified patterns and, for misclassified patterns,

$$\begin{aligned} t^{\mu_m} &= 1 - \theta[a^{\mu_m}] \\ 2t^{\mu_m} - 1 &= t^{\mu_m} - \theta[a^{\mu_m}] \\ \tilde{t}^{\mu_m} &= t^{\mu_m} - \theta[a^{\mu_m}], \end{aligned}$$

we can rewrite the gradient as

$$\Delta E^{\text{perc},\mu}(\mathbf{w}) = -(t^\mu - \theta[a^\mu]) \cdot \mathbf{x}^\mu$$

corresponding to the update rule

$$\begin{aligned} \mathbf{w}^{t+1} &= \mathbf{w}^t - \eta \cdot \Delta E^{\text{perc},\mu}(\mathbf{w}) \\ &= \mathbf{w}^t + \eta \cdot (t^\mu - \theta[a^\mu]) \cdot \mathbf{x}^\mu \end{aligned}$$

which is the perceptron learning algorithm.

Exercise 3. Apply the Perceptron Algorithm

A data base $(\boldsymbol{x}^\mu, t^\mu)$ with $1 \leq \mu \leq 6$ contains three positive examples ($t^\mu = +1$) at the points
$x^1 = (2, -1); x^3 = (2, 0.5); x^5 = (0.5, -1)$
and three negative examples ($t^\mu = 0$) at the points
$x^2 = (-1, 1); x^4 = (0.2, -0.2); x^6 = (2, 1)$.

a. Construct one of the possible separating hyperplanes by hand. Express the parameters of the hyperplane in terms of the three weight parameters (where the third weight parameter is the threshold) of the perceptron.

b. A perceptron algorithm which takes patterns sequentially one after the other starting with pattern $\mu = 1$ is applied to the above problem using an initialization $w = (1, 0)$ and threshold $\theta = 0$.

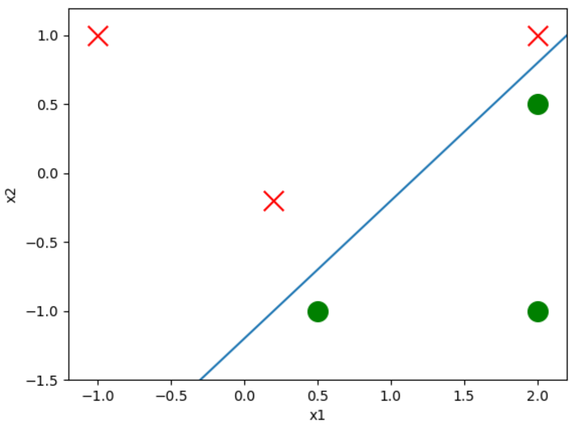
   Consider a learning rate $\eta = 2$ and give the resulting weight vector during the first 6 steps of the iteration.

c. Draw a figure on paper and plot the separating hyperplane before learning $\boldsymbol{w}(0)$, and immediately after the first change $\boldsymbol{w}(1)$, (not every iteration leads to a change)

d. repeat the same with learning rate $\eta = 0.5$ and plot the result in a separate figure.

Solution:

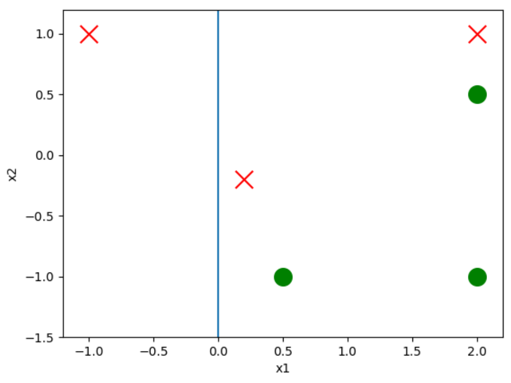
a. One possible hyperplane, given by $1x_1 - 1x_2 - 1.2 = 0$ (parameters $[1, -1, 1.2]$).



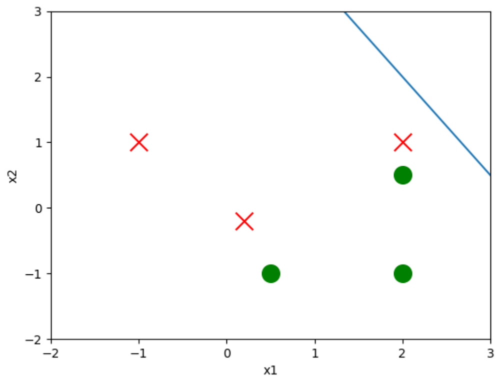
b. The perceptron output and weight updates for the first 6 steps are given by:

$$
\begin{aligned}
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([2., -1., -1.])] = 1 \\
w^1 &= ([1., 0., 0.]) + 2 \cdot (1-1) \cdot ([2., -1., -1.]) = ([1., 0., 0.]) \\
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([-1., 1., -1.])] = 0 \\
w^2 &= ([1., 0., 0.]) + 2 \cdot (0-0) \cdot ([-1., 1., -1.]) = ([1., 0., 0.]) \\
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([2., 0.5, -1.])] = 1 \\
w^3 &= ([1., 0., 0.]) + 2 \cdot (1-1) \cdot ([2., 0.5, -1.]) = ([1., 0., 0.]) \\
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([0.2, -0.2, -1.])] = 1 \\
w^4 &= ([1., 0., 0.]) + 2 \cdot (0-1) \cdot ([0.2, -0.2, -1.]) = ([0.6, 0.4, 2.]) \\
\theta[a^\mu] &= \theta[([0.6, 0.4, 2.]) \cdot ([0.5, -1., -1.])] = 0 \\
w^5 &= ([0.6, 0.4, 2.]) + 2 \cdot (1-0) \cdot ([0.5, -1., -1.]) = ([1.6, -1.6, 0.]) \\
\theta[a^\mu] &= \theta[([1.6, -1.6, 0.]) \cdot ([2., 1., -1.])] = 1 \\
w^6 &= ([1.6, -1.6, 0.]) + 2 \cdot (0-1) \cdot ([2., 1., -1.]) = ([-2.4, -3.6, 2.])
\end{aligned}
$$

c.



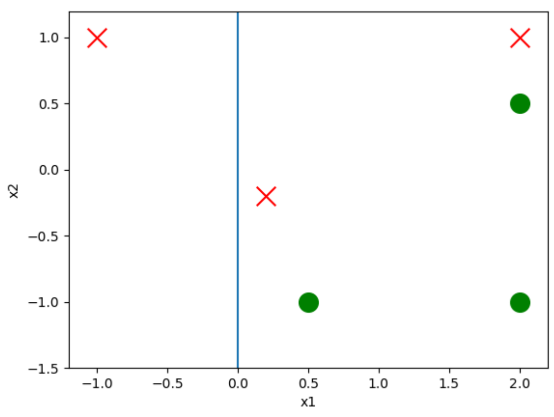
Initial hyperplane.



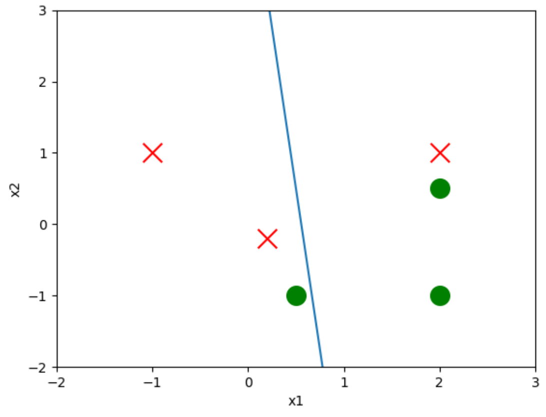
First weight update (from x^4).

d. Using $\eta = 0.5$,

$$
\begin{aligned}
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([2., -1., -1.])] = 1 \\
w^1 &= ([1., 0., 0.]) + 0.5 \cdot (1 - 1) \cdot ([2., -1., -1.]) = ([1., 0., 0.]) \\
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([-1., 1., -1.])] = 0 \\
w^2 &= ([1., 0., 0.]) + 0.5 \cdot (0 - 0) \cdot ([-1., 1., -1.]) = ([1., 0., 0.]) \\
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([2., 0.5, -1.])] = 1 \\
w^3 &= ([1., 0., 0.]) + 0.5 \cdot (1 - 1) \cdot ([2., 0.5, -1.]) = ([1., 0., 0.]) \\
\theta[a^\mu] &= \theta[([1., 0., 0.]) \cdot ([0.2, -0.2, -1.])] = 1 \\
w^4 &= ([1., 0., 0.]) + 0.5 \cdot (0 - 1) \cdot ([0.2, -0.2, -1.]) = ([0.9, 0.1, 0.5]) \\
\theta[a^\mu] &= \theta[([0.9, 0.1, 0.5]) \cdot ([0.5, -1., -1.])] = 0 \\
w^5 &= ([0.9, 0.1, 0.5]) + 0.5 \cdot (1 - 0) \cdot ([0.5, -1., -1.]) = ([1.15, -0.4, 0.]) \\
\theta[a^\mu] &= \theta[([1.15, -0.4, 0.]) \cdot ([2., 1., -1.])] = 1 \\
w^6 &= ([1.15, -0.4, 0.]) + 0.5 \cdot (0 - 1) \cdot ([2., 1., -1.]) = ([0.15, -0.9, 0.5])
\end{aligned}
$$



Initial hyperplane.



First weight update (from x^4).

Exercise 4. Apply the stochastic gradient descent algorithm from Exercise 1

Use a transfer function $g(a) = 1/[1 + exp(-\beta a)]$ and take patterns from Exercise 3 in the same order and with the learning rate $\eta = 0.5$ as you did for the perceptron algorithm. What are the differences to the perceptron algorithm? What are the problems?

Hint: Choose first $\beta = 0.5$ and then $\beta = 5$. Consider a pattern that is correctly classified and one that is misclassified, for example patterns 5 and 6. Evaluate the gradient g' and comment on its absolute value for cases where a pattern is misclassified.

Solution:

The gradient of the transfer function is $g'(a) = \beta g(a)(1 - g(a))$, we can now rewrite the update equation as

$$
\begin{aligned}
\Delta w_j &= \eta \left[t^\mu - g(a^\mu)\right] x_j^\mu g'(a^\mu) \\
&= \eta \left[t^\mu - g(a^\mu)\right] x_j^\mu \beta g(a^\mu)(1 - g(a^\mu))
\end{aligned}
$$

Using $\beta = 0.5$,

$$
\begin{aligned}
g[a^1] &= g[(1., 0., 0.) \cdot (2., -1., -1.)] = 0.731 \\
w^1 &= (1., 0., 0.) + 0.5 \cdot (1 - 0.73) \cdot (2., -1., -1.) \cdot 0.098 = (1.026, -0.013, -0.013) \\
g[a^2] &= g[(1.026, -0.013, -0.013) \cdot (-1., 1., -1.)] = 0.374 \\
w^2 &= (1.026, -0.013, -0.013) + 0.5 \cdot (0 - 0.374) \cdot (-1., 1., -1.) \cdot 0.117 = (1.048, -0.035, 0.009) \\
g[a^3] &= g[(1.048, -0.035, 0.009) \cdot (2., 0.5, -1.)] = 0.738 \\
w^3 &= (1.048, -0.035, 0.009) + 0.5 \cdot (1 - 0.738) \cdot (2., 0.5, -1.) \cdot 0.097 = (1.074, -0.029, -0.004) \\
g[a^4] &= g[(1.074, -0.029, -0.004) \cdot (0.2, -0.2, -1.)] = 0.528 \\
w^4 &= (1.074, -0.029, -0.004) + 0.5 \cdot (0 - 0.528) \cdot (0.2, -0.2, -1.) \cdot 0.125 = (1.067, -0.022, 0.029) \\
g[a^5] &= g[(1.067, -0.022, 0.029) \cdot (0.5, -1., -1.)] = 0.565 \\
w^5 &= (1.067, -0.022, 0.029) + 0.5 \cdot (1 - 0.565) \cdot (0.5, -1., -1.) \cdot 0.123 = (1.080, -0.049, 0.002) \\
g[a^6] &= g[(1.080, -0.049, 0.002) \cdot (2., 1., -1.)] = 0.742 \\
w^6 &= (1.080, -0.049, 0.002) + 0.5 \cdot (0 - 0.742) \cdot (2., 1., -1.) \cdot 0.096 = (1.009, -0.084, 0.038)
\end{aligned}
$$

Here, the weight update is performed even for correctly classified points and the amplitude of each update depends on the distance from the hyperplane. The gradient of the sigmoid function g' is always positive. For small βs, the slope of the central part of the sigmoid - around the decision threshold of 0.5 - is small, almost linear, and for a big range of values. Therefore, the gradient is similar for most points. For bigger βs, the gradient around the decision is bigger. Therefore, examples falling close to the decision threshold will result in a higher weight update amplitude. However, the range of the saturated regime, where the gradient is close to 0, is increased. Once a data point falls into this regime it can be trapped if no other points can change the parameters.

with $w = (1, 0)$ and $\theta = 0$:

β	μ	t^μ	$g(a^\mu)$	$g'(a^\mu)$
0.5	5	1	0.562	0.123
0.5	6	0	0.731	0.076
5	5	1	0.924	0.351
5	6	0	0.999	0.0002

Exercise 5. Adaline algorithm

A friend of yours argues as follows: in a perceptron with $\hat{y}^\mu = \theta(a^\mu)$, a sufficient condition that all patterns are correctly classified is that $a^\mu = +1$ for all positive examples $t^\mu = +1$, and $a^\mu = -1$ for all negative examples $t^\mu = 0$.

He therefore suggests to directly optimize the error function

$$E^{\mathrm{Ada}}(\boldsymbol{w}) = \frac{1}{2} \sum_\mu \left[\tilde{t}^\mu - a^\mu\right]^2 \tag{5}$$

where $\tilde{t}^\mu = 2(t^\mu - 0.5)$ and $a^\mu = \sum_k w_k x_k^\mu$.

To convince him that this error function is not a good idea, show him the following example.

A data base $(\boldsymbol{x}^\mu, t^\mu)$ with $1 \leq \mu \leq 12$ contains six positive examples $(t^\mu = +1)$ at the points
$x^1 = (1, 1); x^2 = (1, -1); x^3 = (\alpha, 2); x^4 = (\alpha, 1); x^5 = (\alpha, -2); x^6 = (\alpha, -1)$ where $\alpha \geq 1$ is a parameter; and six negative examples $(\tilde{t}^\mu = -1)$ at the points
$x^7 = (-1, 3); x^8 = (-1, 2); x^9 = (-1, 1); x^{10} = (-1, -1); x^{11} = (-1, -2); x^{12} = (-1, -3)$.

a. Plot the points.

b. Choose a weight vector $(w, 0, b)$ that gives rise to a linear discriminant function $d(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{x}$ which separates positive and negative examples. Does your solution depend on the choice of α?

c. Insert the data points and the weight vector $(w, 0, b)$ with arbitrary w and b into the error function E^{Ada} and find those w and b that minimize the error function. Express your result as a function of α.

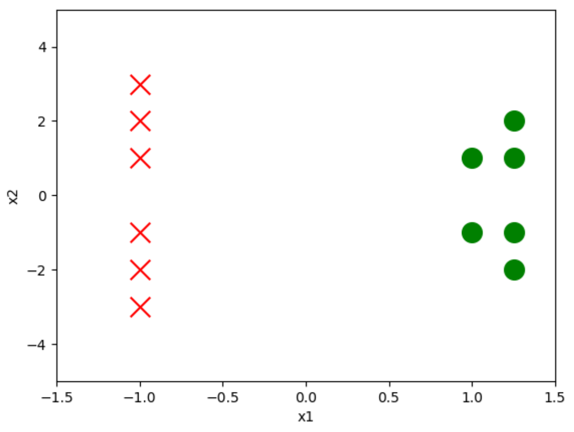
d. Determine the maximal value of the parameter α for which the optimization in (c) still gives a correct discriminant function.

e. Conclude the argument.

Remark: $E^{\text{Ada}}(\boldsymbol{w})$ was called the error function of the historical 'Adaline' algorithm.

Solution:

a.



Plot of points, where we have chosen $\alpha = 1.25$.

b. We note that the constraints give $x_1 = b/w$. One possible solution is $b = 0$, $w = 1$, and therefore $x_1 = 0$, which would separate the points (see figure). Since $\alpha \geq 1$, the positive examples parameterized by α will always be correctly separated by this function. In fact, the choice of α does not affect whether any vertical–line solution is correct.

c.

$$
\begin{aligned}
E^{\text{Ada}}(\mathbf{w}) &= \frac{1}{2}\sum_{\mu}\left[\tilde{t}^{\mu} - a^{\mu}\right]^2 \\
&= \frac{1}{2}\left[2\cdot(1-(w-b))^2 + 4\cdot(1-(\alpha w - b))^2 + 6\cdot(-1-(-w-b))^2\right]
\end{aligned}
$$

Taking the derivatives with respect to w and b and setting to 0 gives

$$
\begin{aligned}
w &= \frac{3(\alpha+2)}{2\alpha^2+2\alpha+5} \\
b &= \frac{\alpha^2+\alpha-2}{2\alpha^2+2\alpha+5}
\end{aligned}
$$

d. The points will only be separated by a vertical discriminant defined by $x_1 = b/w < 1$. Rearranging the above equations we get

$$
\begin{aligned}
\frac{b}{w} = \frac{\alpha^2+\alpha-2}{3(\alpha+2)} &< 1 \\
\frac{\alpha-1}{3} &< 1 \\
\alpha &< 4
\end{aligned}
$$

Therefore, for any $\alpha \geq 4$, the discriminant that minimizes E^{Ada} will not separate the two classes.

e. Based on what we found in (d), there exist linearly separable classes of points for which E^{Ada} is minimized by an incorrect solution (i.e. a discriminant that does not separate the two classes). We therefore conclude that it is a poor error function.

   Intuitively, we note that a^{μ} is proportional to the signed distance to the discriminant. Even correctly classified points can contribute to the error function if they are far from the discriminant (i.e. $|a^{\mu}| > 1$). This has the effect of pulling the discriminant towards outlier points.