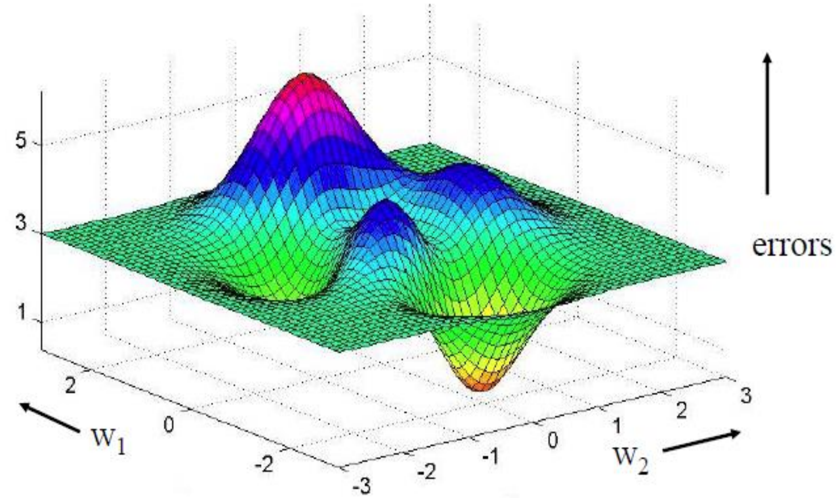
CISC452/CMPE452/COGS400
Adaline

Farhana Zulkernine

Error Measures

- Error is associated with the combination of weight values.
- (D-y) is a very straight forward error calculation which applies the same change $\Delta w_i = \pm c.|E|.x_i$ to all weights without considering how each weight affects the change in the output.
- $w_i \in \mathbb{R}$, a real number and can vary widely → impossible to try all combinations to get to minimum error.
- Solution → introduce the idea of an error surface which is a function of weights. We can compute *derivative of error with respect to the weight values to determine how each weight affects the change in error and apply a proportional amount of correction to that weight*.



Mean Squared Error

- Error $e = d – y$ d (desired), y (actual) output
- For *p* data points the **Mean Squared Error** will be:

  MSE $E = 1/p * \sum^{p} (d – y)^2$

  Using a **linear output** function, $y = a = \sum_{i=1}^{n} x_i w_i$

  MSE $E = 1/p * \sum^{p} (d – a)^2$
- Since MSE is a *quadratic function, its derivative exists everywhere*.
- It is also known as the Loss Function (LF).
- So we can calculate $\frac{dE}{dW}$ and adjust *w* towards the –ve slope of the error surface to reach minimum error.

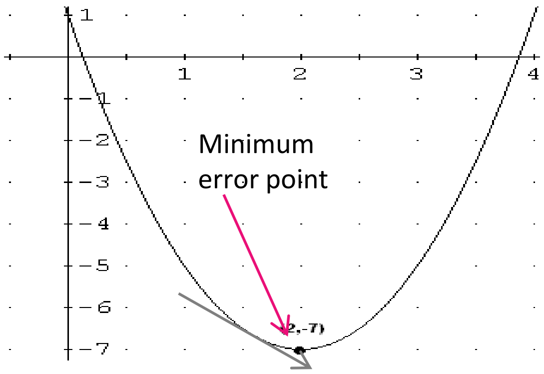
Adaline
(Widrow, 1959)

- An **Ada**ptive **Lin**ear **E**lement tries to minimize the total **amount of error** instead of aiming to reduce the *number of misclassifications*.
- The Adaline, proposed by Widrow (1959,1960), modifies weight in such a way to diminish the **Mean Squared Error (MSE)** at every data point which is the LF.
- The training examples are of form (x, d) where d ϵ $\mathbb{R}$.
- Output y ϵ $\mathbb{R}$, the network outputs the activation *a*.
  - Uses *linear output or activation function*. y=a

Gradient Descent Optimization

- To reduce total error or optimize the loss, Adaline uses gradient descent of loss function.
  - if $f(w_1, \ldots, w_n)$ is a *differentiable, scalar-valued output function of several variables*,
  - its **gradient** is the *vector* V, whose components are the n partial derivatives of $f = \frac{\partial f}{\partial W} = (\frac{\partial f}{\partial w_1}, \frac{\partial f}{\partial w_2}, \ldots, \frac{\partial f}{\partial w_n})$

Considering MSE and the error surface, we want to move towards the –ve **slope of the tangent of the error surface,** which points in the direction of the greatest rate of decrease of the error value towards **the minimum error point.**

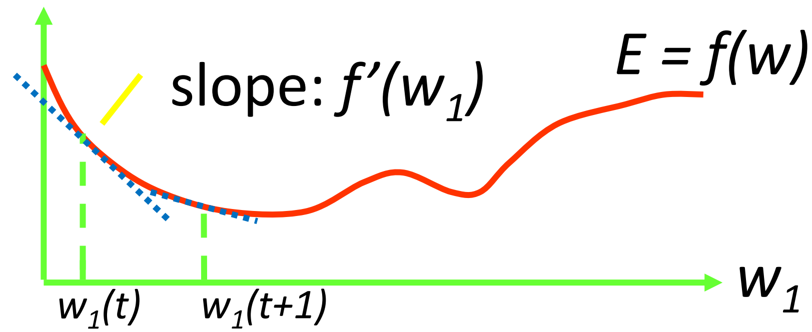


$f(x) = 2x^2 - 8x + 1$

Weight Adjustment using Gradient Descent

- Let's say that the partial derivative of the error $E=f(w)$, with respect to one of the weights w_1 is $f'(w_1)$, we can use Gradient Descent as the optimization function to reach minimum error by adjusting the weight using $w_1(t+1) = w_1(t) - \eta f'(w_1(t))$
- Where η is the *learning rate* (i.e., adjustment factor). Weight adjustments are done repeatedly in multiple cycles for all the data points until error falls below the acceptable threshold.

$$-\frac{\partial E}{\partial w_1} = -\frac{\partial f(w)}{\partial w_1} = f'(w_1)$$



- Other optimization functions such as *Adam optimizer* (study on your own) or *simulated annealing*.

Chain Rule

- We have to apply the *chain rule* to loss or error to solve the derivative because of functional dependencies.

$E = f(\mathrm{e})$	given $E = e^2$
$e = f(\mathrm{y})$	given $e = (d - y)$
$y = f(\mathrm{a})$	function of activation
$a = f(w)$	given $a = \sum_{\mathrm{i}=1}^{\mathrm{n}} x_i w_i$

when we are computing partial derivative w.r.t. weight

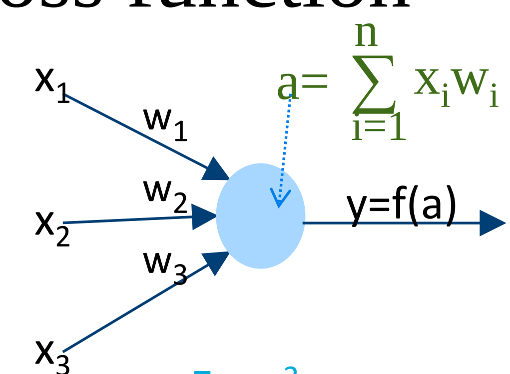
$$\frac{\partial E}{\partial w_k} = \frac{\partial E}{\partial e} \cdot \frac{\partial e}{\partial y} \cdot \frac{\partial y}{\partial a} \cdot \frac{\partial a}{\partial w_k}$$

Computation of Weight Adjustment

- Weights are adjusted based on how each weight changes error and computed as $\dfrac{\partial E}{\partial w_k}$
- Compute –ve of the gradient of loss function

$\Delta w_k = -\dfrac{\partial E}{\partial w_k}$ and applying

the *chain rule*, $\dfrac{\partial E}{\partial w_k} = \dfrac{\partial E}{\partial e}.\dfrac{\partial e}{\partial a}.\dfrac{\partial a}{\partial w_k}$



$E = e^2$

$e = d - a$

$$-\frac{\partial E}{\partial w_k} = -\frac{\partial e^2}{\partial e}.\frac{\partial (d-a)}{\partial a}.\frac{\partial a}{\partial w_k}$$

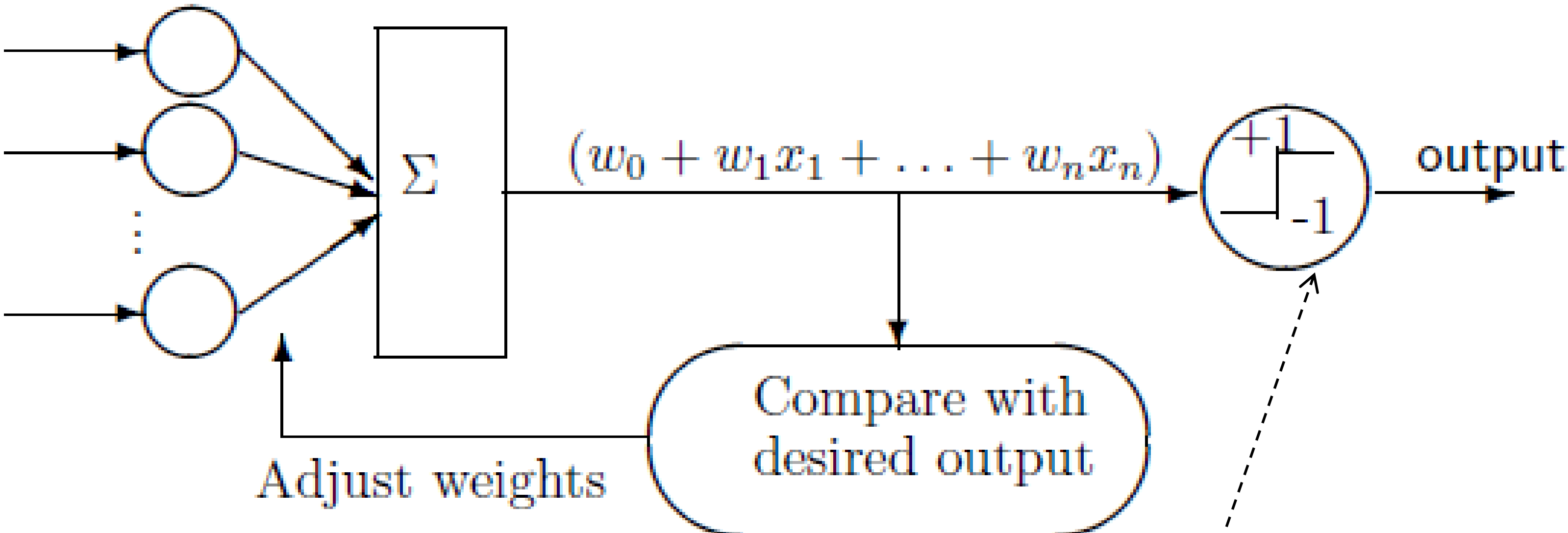
$$= -2(d-a).(-1).\frac{\partial (w_1 x_1 + \cdots + w_k x_k + \cdots + w_n x_n)}{\partial w_k}$$

$$= c(d-a).x_k$$

c = a small +ve constant learning rate

Feedback Loop in Adaline

$$x_j = \begin{cases} +1 & \text{if } a > 0 \\ -1 & \text{if } a \leq 0 \end{cases}$$



See example 2.6 in the book. You will see that the solution applies correction only when a data point is incorrectly classified (error> threshold) to reduce the computation.

An output function can be used to generate custom output after training