ĐẠI HỌC ĐÀ NẴNG

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN

VIETNAM - KOREA UNIVERSITY OF INFORMATION AND COMMUNICATION TECHNOLOGY

한-베정보통신기술대학교

Nhân bản – Phụng sự – Khai phóng

Chapter 3

Regression Techniques

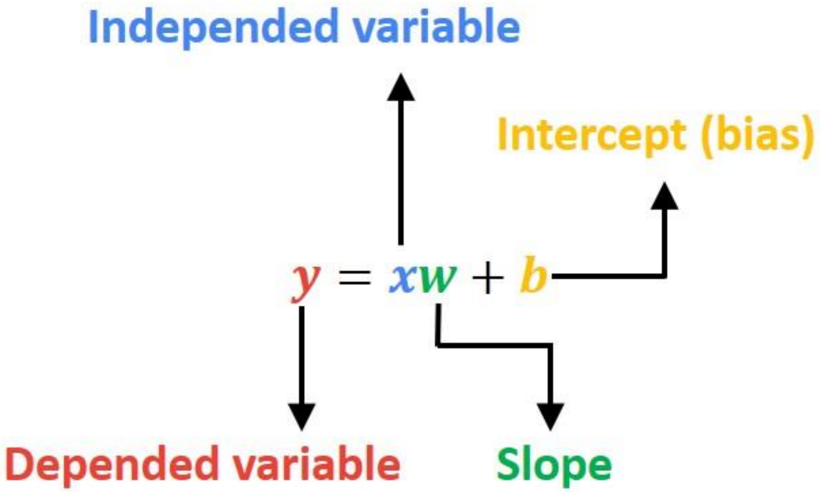
Machine Learning

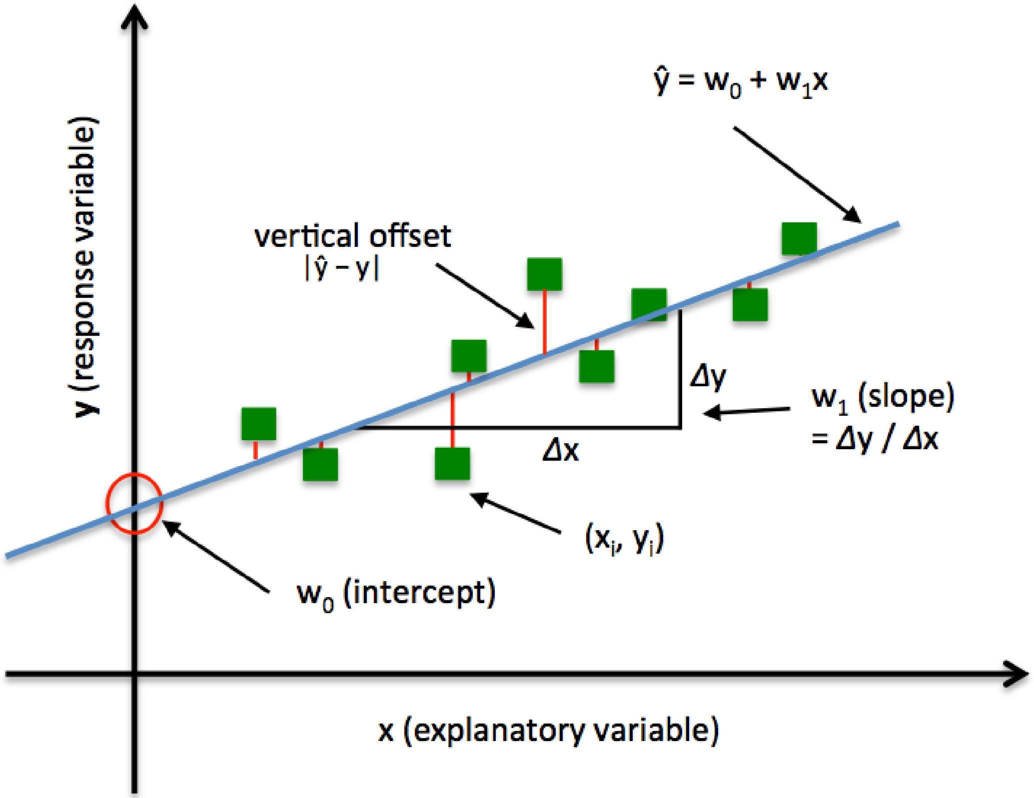
CONTENTs

- Linear Regression
  - Linear Problems
  - Gradient Descent

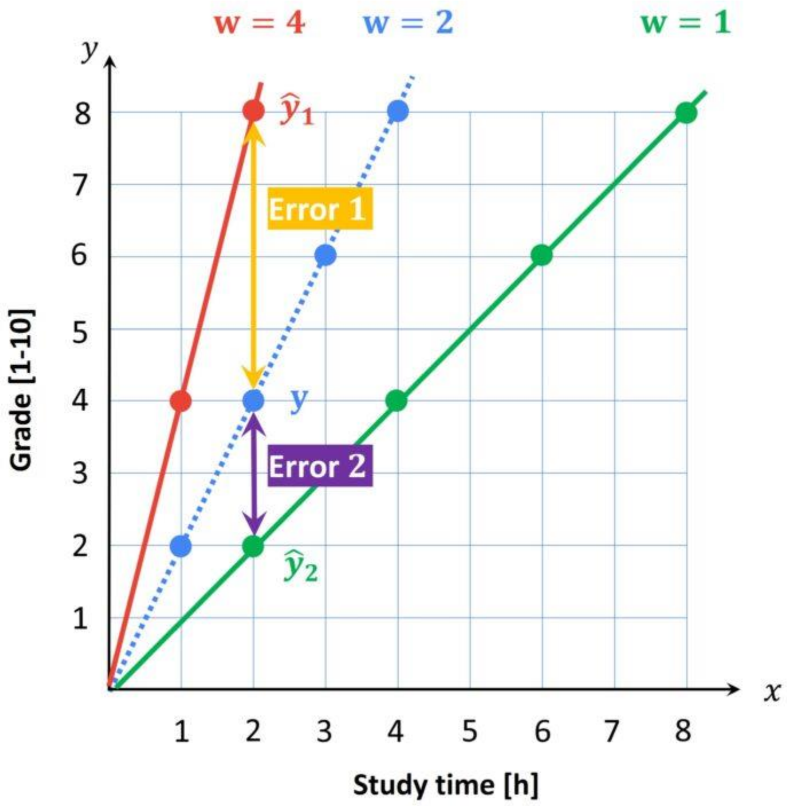
- Linear Problems
  - Linear Regression
  - Nonlinear Regression
  - Derivatives and Finding Extreme Points
- Gradient Descent

- A linear function is a systematic or sequential increase or decrease represented by a straight line.
- Example : Linear Regression





Searching minimal loss



$$\hat{y} = xw + b$$

$$loss = (\hat{y} - y)^2 = (xw - y)^2$$

Error $= \hat{y} - y$

Error 1 $= \hat{y}_1 - y = 4$

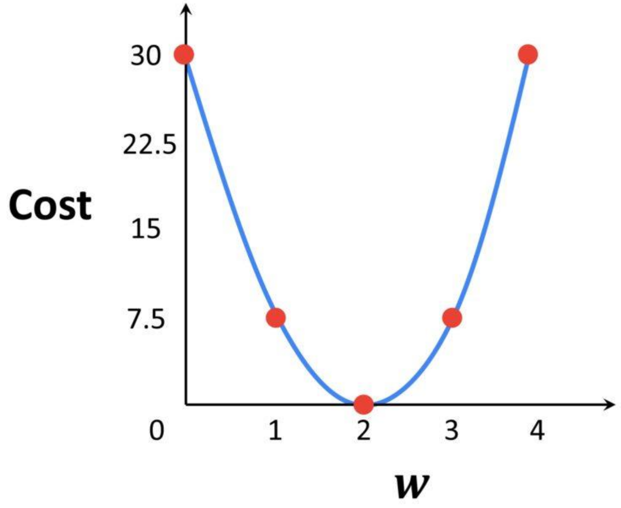
Error 2 $= \hat{y}_2 - y = 2$

Error 2 $<$ Error 1

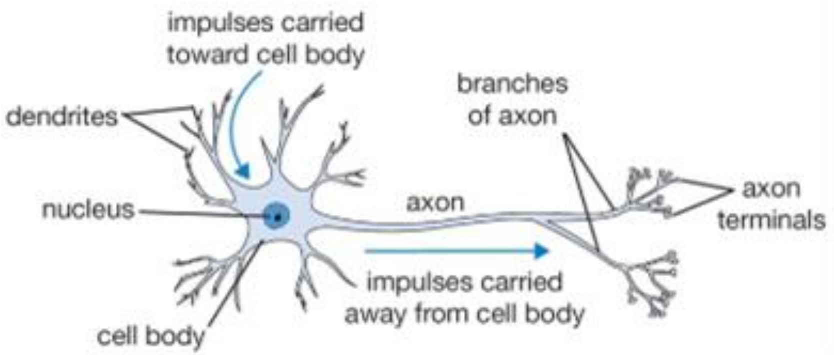
Loss function

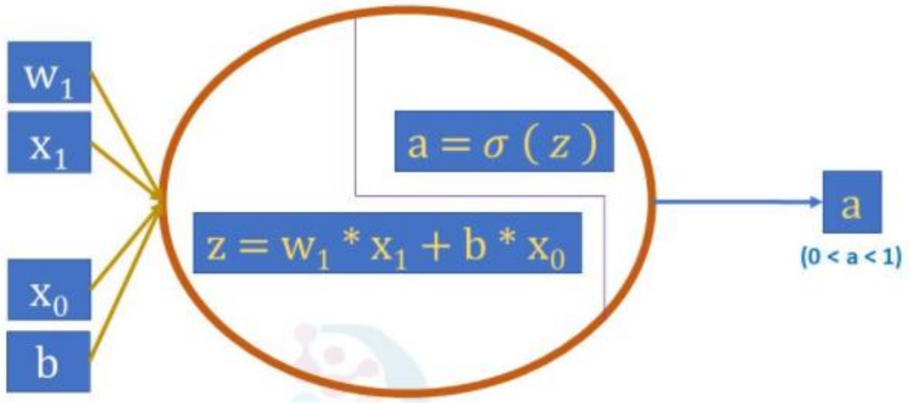
$$cost = \frac{1}{N}\sum_{n=1}^{N} loss_n$$

$$cost = \frac{1}{N}\sum_{n=1}^{N} (\hat{y}_n - y_n)^2$$

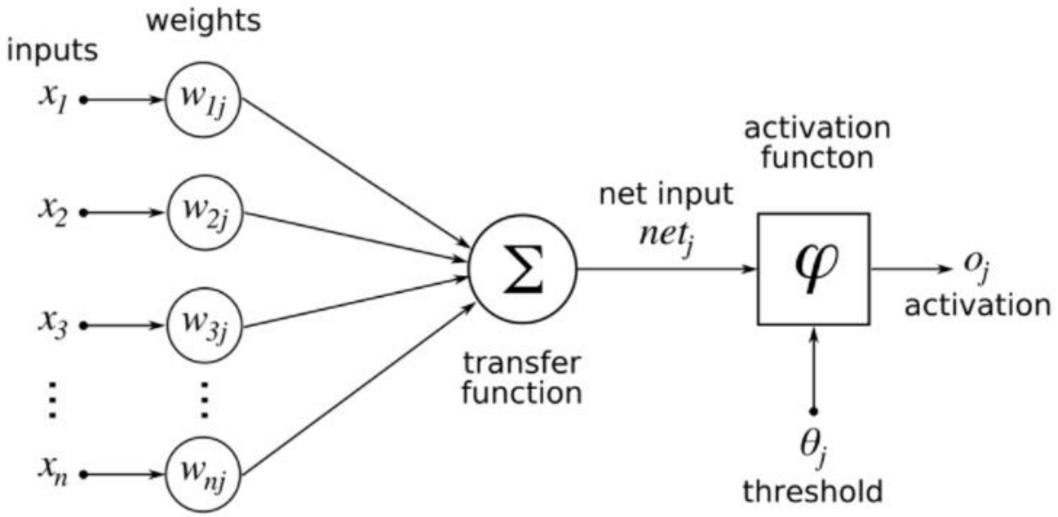


- A non-linear function is a function where the data does not increase or decrease in a systematic or sequential way.
- Activation function is an important concept in machine learning, especially in deep learning. They basically decide whether a neuron should be activated or not and introduce non-linear transformation to a neural network. The main purpose of these functions is to convert an input signal of a neuron and produce an output to feed in the next neuron in the next layer
- Example: Activation Functions

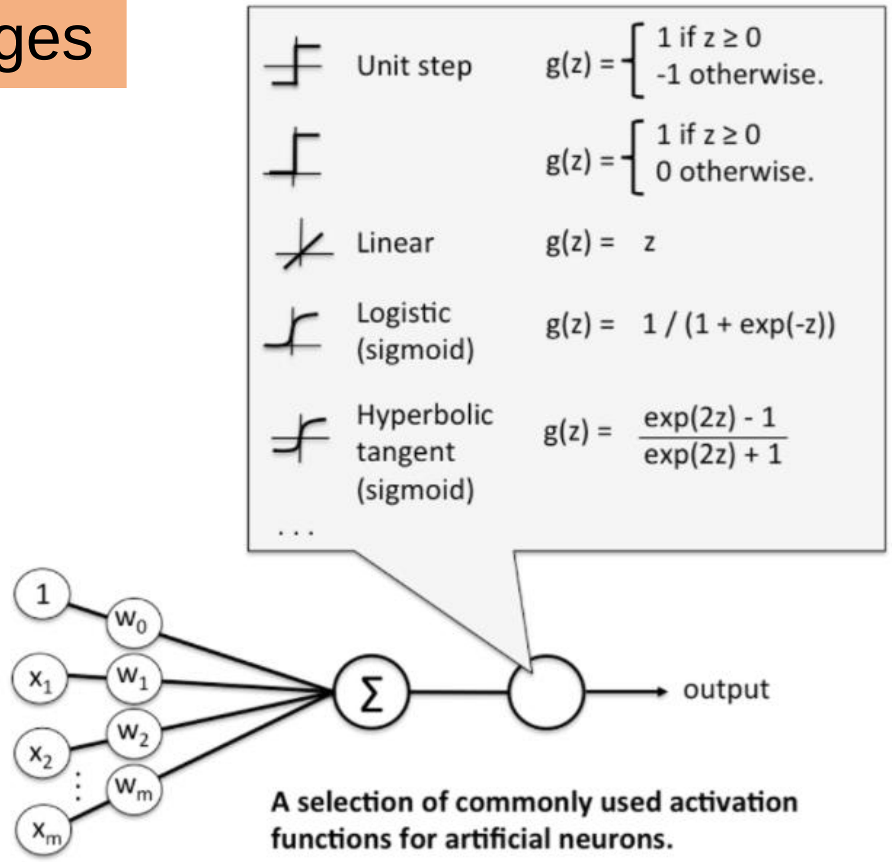




Activation Functions Advantages



Source: Isaac Changhau



A selection of commonly used activation functions for artificial neurons.

- **Sigmoid Activation Function:**
  - Range from [0,1]
  - Not Zero Centered
  - Have Exponential Operation
- **Hyperbolic Tangent Activation Function(tanh):**
  - Ranges Between [-1,1]
  - Zero Centered
- **Rectified Linear Unit Activation Function (ReLU):**
  - It doesn’t Saturate
  - It converges faster than some other activation functions

Sigmoid

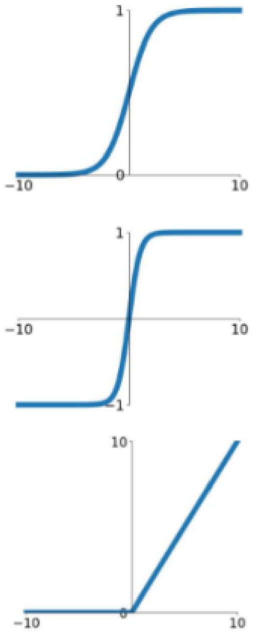
$\sigma(x) = \frac{1}{1+e^{-x}}$

tanh

$\tanh(x)$

ReLU

$\max(0, x)$

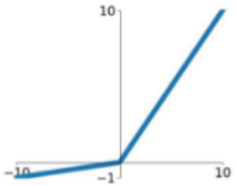


Linear and Nonlinear Functions

- **Leaky ReLU:**
  - Leaky ReLU improvement over ReLU Activation function.
  - It has all properties of ReLU
  - It will never have dead ReLU problem.
- **Maxout:**
  - It has property of Linearity in it
  - it never saturates or die
  - But is Expensive as it doubles the parameters.
- **ELU(Exponential Linear Units):**
  - No Dead ReLU Situation.
  - Closer to Zero mean Outputs than Leaky ReLU
  - More Computation because of Exponential Function

Leaky ReLU

$\max(0.1x, x)$

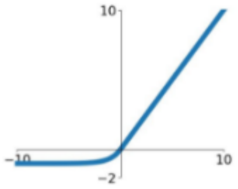


Maxout

$\max(w_1^T x + b_1, w_2^T x + b_2)$

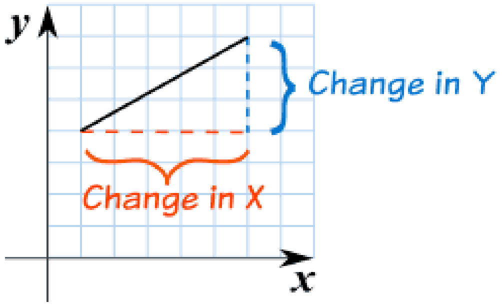
ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



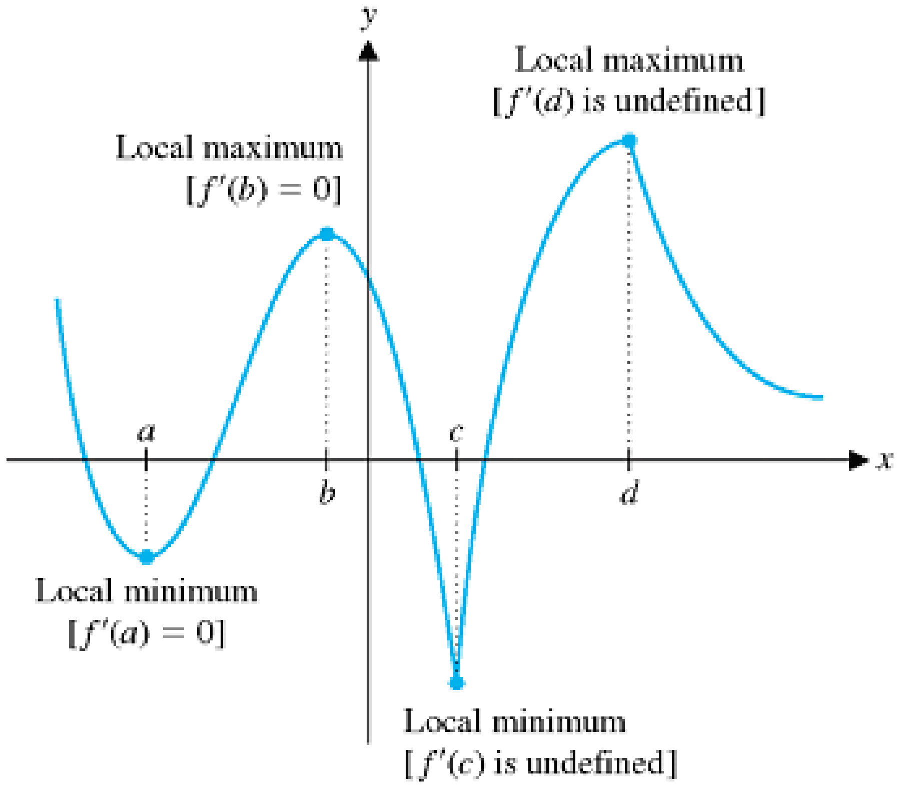
- Suppose we have a function y = f(x) which is dependent on x then the derivation of this function means the rate at which the value y of the function changes with change in x.
- In geometry slope represents the steepness of a line. It answers the question: how much does y or f(x) change given a specific change in x?
- Using this definition we can easily calculate the slope between two points. But what if I asked you, instead of the slope between two points, what is the slope at a single point on the line? In this case there isn't any obvious "rise-over-run" to calculate. Derivatives help us answer this question

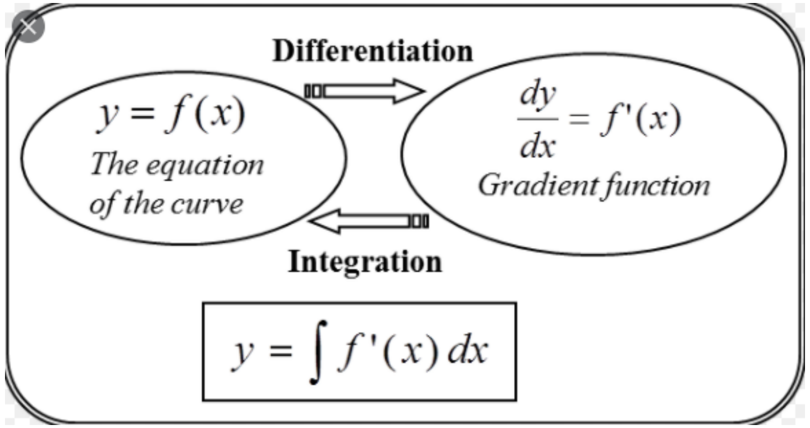
$$\text{Slope} = \frac{\text{Change in Y}}{\text{Change in X}}$$

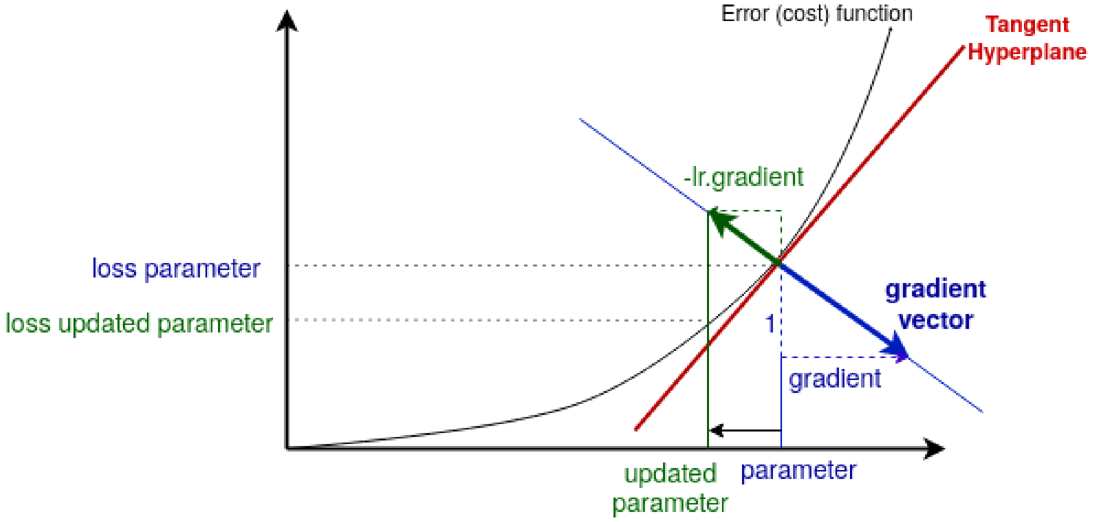


$$f'(a) = \lim_{h \to 0} \frac{f(a+h) - f(a)}{h}$$

Finding Extreme Points



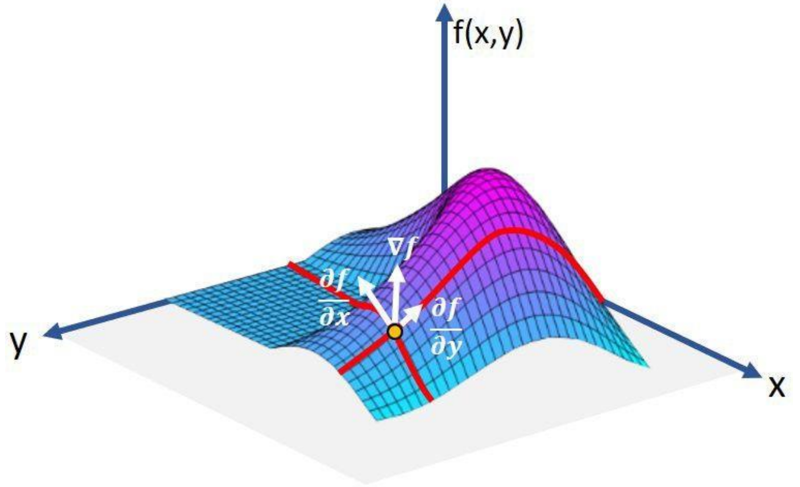




Partial derivative

$$f_x = \frac{\partial f}{\partial x} = \lim_{h \to 0} \frac{f(x+h, y) - f(x, y)}{h}$$

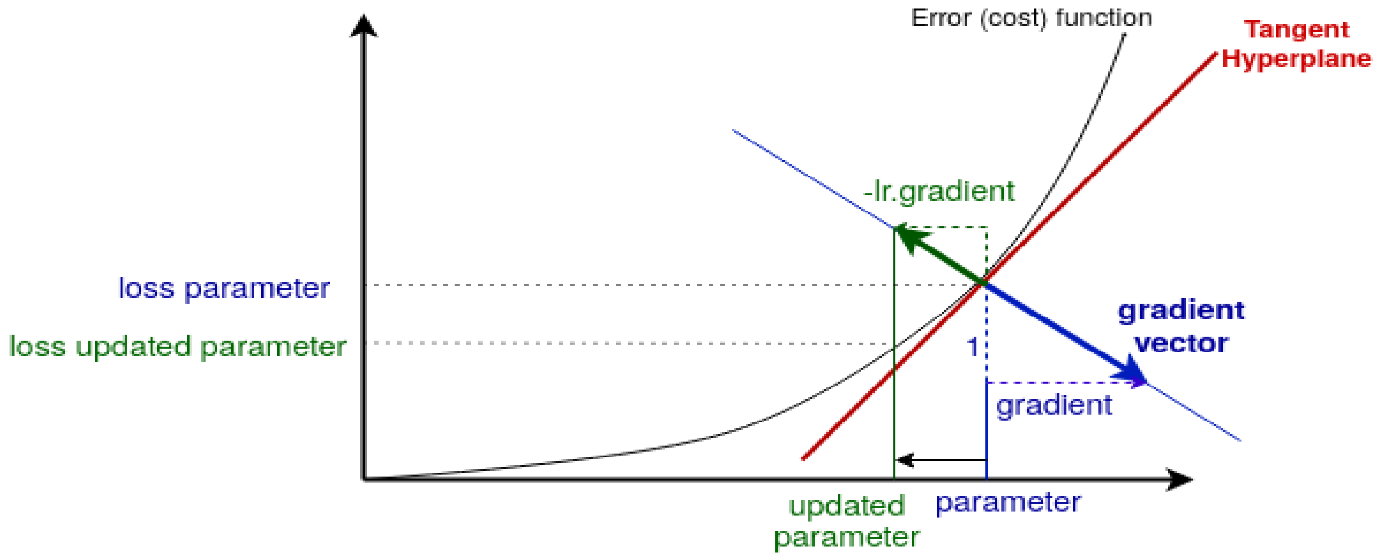
$$f_y = \frac{\partial f}{\partial y} = \lim_{h \to 0} \frac{f(x, y+h) - f(x, y)}{h}$$



$$\text{Jacobian matrix}: J = \begin{pmatrix} \frac{\partial f_1}{\partial M_1} & \cdots & \frac{\partial f_1}{\partial M_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial M_1} & \cdots & \frac{\partial f_n}{\partial M_n} \end{pmatrix}$$

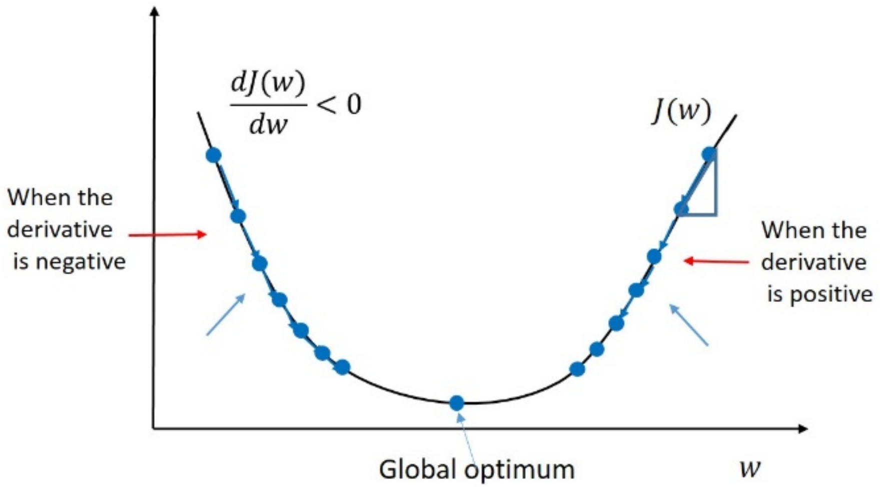
$$\nabla f = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} \quad \text{and} \quad \nabla^2 f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

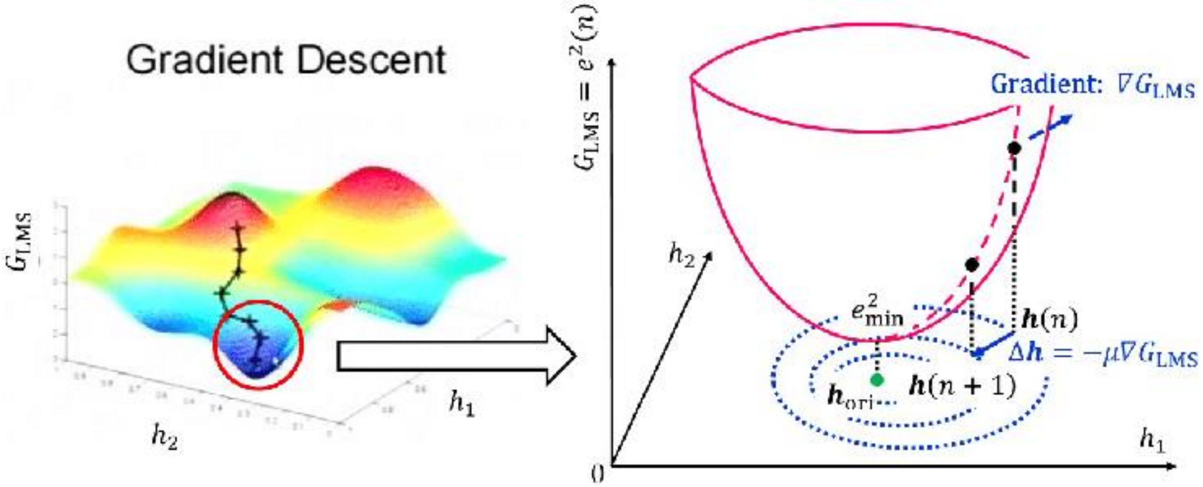
- A gradient is a vector that stores the partial derivatives of multivariable functions. It helps us calculate the slope at a specific point on a curve for functions with multiple independent variables.
- The gradient vector is the vector generating the line orthogonal to the tangent hyperplane. Then you take the opposite of this vector (hence “descent”), multiply it by the learning rate lr.

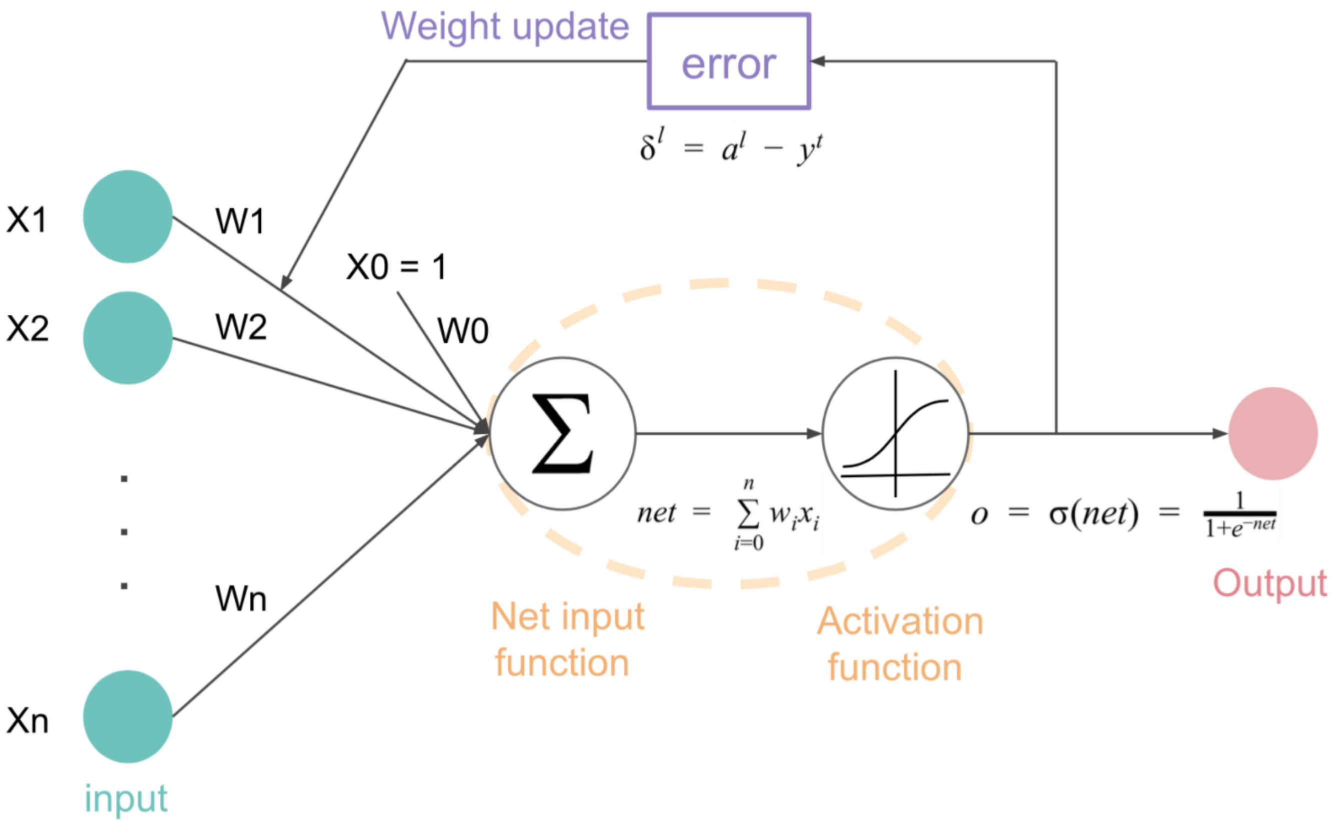


- The projection of this vector on the parameter space (here: the x-axis) gives you the new (updated) parameter. Then you repeat this operation several times to go down the cost (error) function, with the goal of reaching a value for w where the cost function is minimal.
- The parameter is thus updated as follow at each step:

parameter <-- parameter - lr*gradient







Derivative of Sigmoid function

$$y = \frac{1}{1+e^{-x}}$$

$$\frac{dy}{dx} = -\frac{1}{(1+e^{-x})^2}(-e^{-x}) = \frac{e^{-x}}{(1+e^{-x})^2}$$

$$= \frac{1}{1+e^{-x}}\left(1 - \frac{1}{1+e^{-x}}\right) = y(1-y)$$

$$Loss(y, \hat{y}) = \sum_{i=1}^{n} (y - \hat{y})^2$$

$$\frac{\partial\, Loss(y,\hat{y})}{\partial W} = \frac{\partial Loss(y,\hat{y})}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z} * \frac{\partial z}{\partial W} \quad \text{where } z = Wx + b$$

$$= 2(y - \hat{y}) * \text{derivative of sigmoid function} * x$$

$$= 2(y - \hat{y}) * z(1\text{-}z) * x$$

Optimization problem:

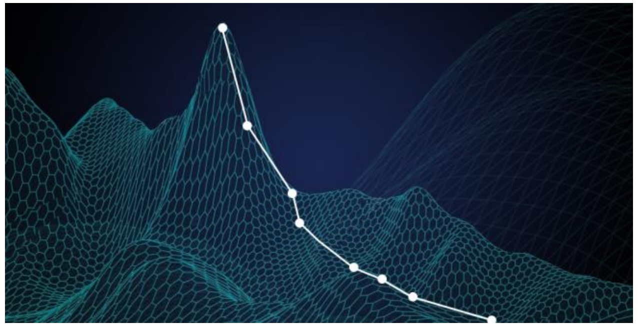
$$L(w) = \sum_{i=1}^{\ell} L(w; x_i, y_i) \to \min_w$$

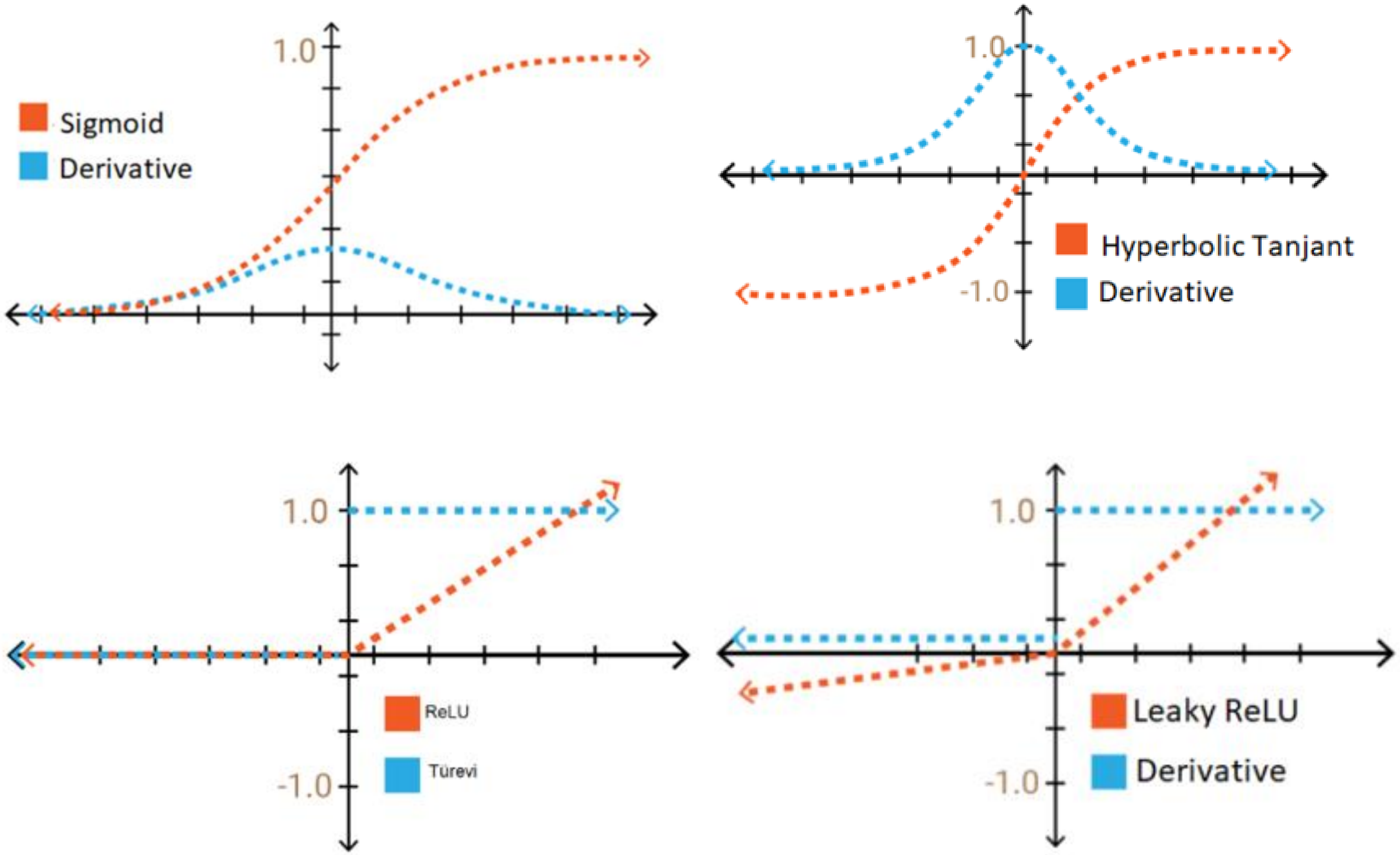
w^0 — initialization

while True:

$$w^t = w^{t-1} - \eta_t \nabla L(w^{t-1})$$

if $\|w^t - w^{t-1}\| < \epsilon$ then break





SUMMARY

- Introduction
- Applications of ML
- Types of ML Systems
- Main Challenges of ML
- Testing & Validating

Enjoy the Course…!