Deep Learning Made Easy

The Basics

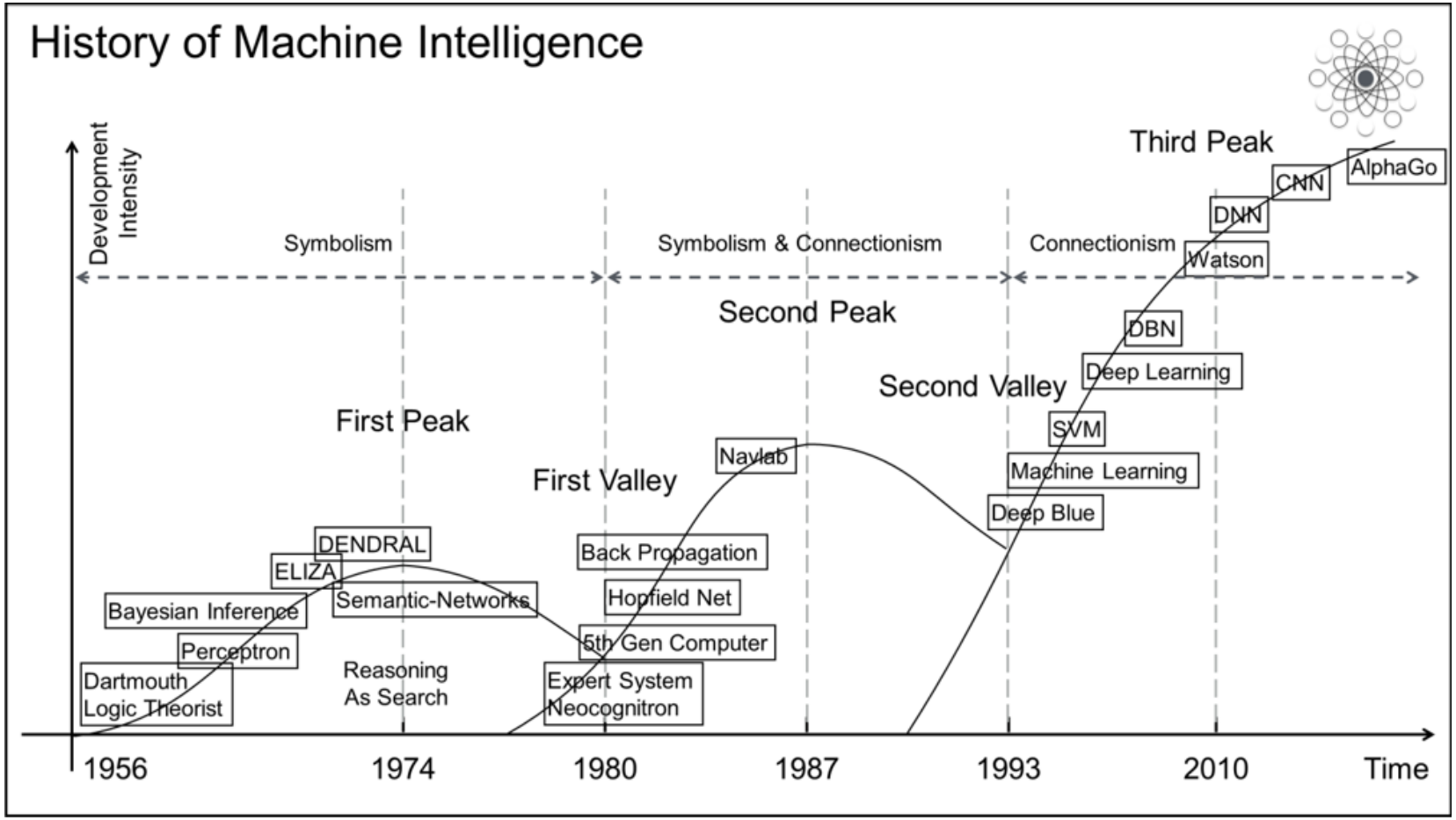
Schedule

week	Date	Topic
1	02.11	Introduction
2	02.18	The basics: python, math, and AI development
3	02.25	AI history & Perceptron
4	03.04	Training: forward propagation & backpropagation
5	03.11	CNN
6	03.18	Metrics
7	03.25	Word embedding
8	04.01	RNN
9	04.08	Autoencoder & GAN
10	04.15	Project presentation

Today's Agenda

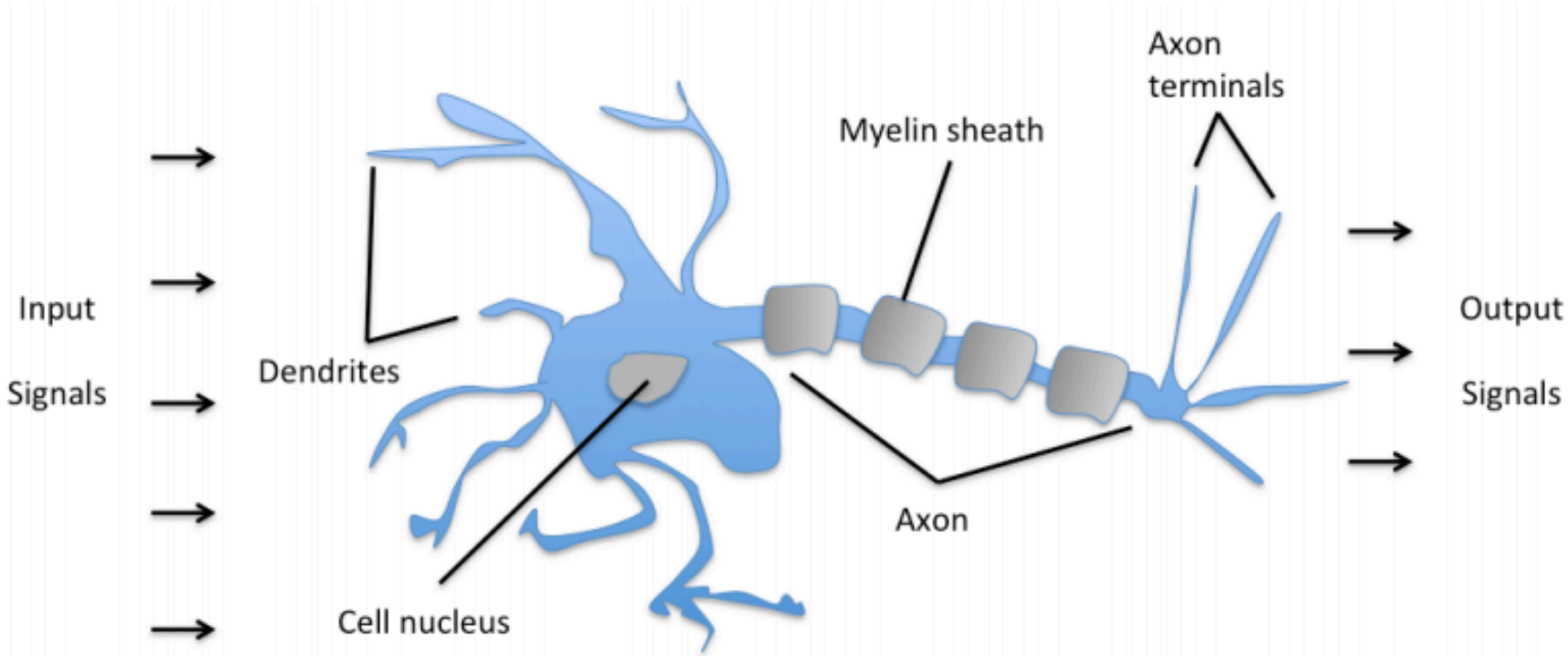
- Demo:
  - Teachable Machine
  - Peltarion
- Perceptron:
  - Y = WX + b
  - Two inputs: x1, x2
  - One output: y
  - Linear regression
- XOR problem
  - Linear regression can't solve the X
- Multi-layer neural network
  - Feedforward
  - Backpropagation
- Assignment

A Brief History of Machine Intelligence



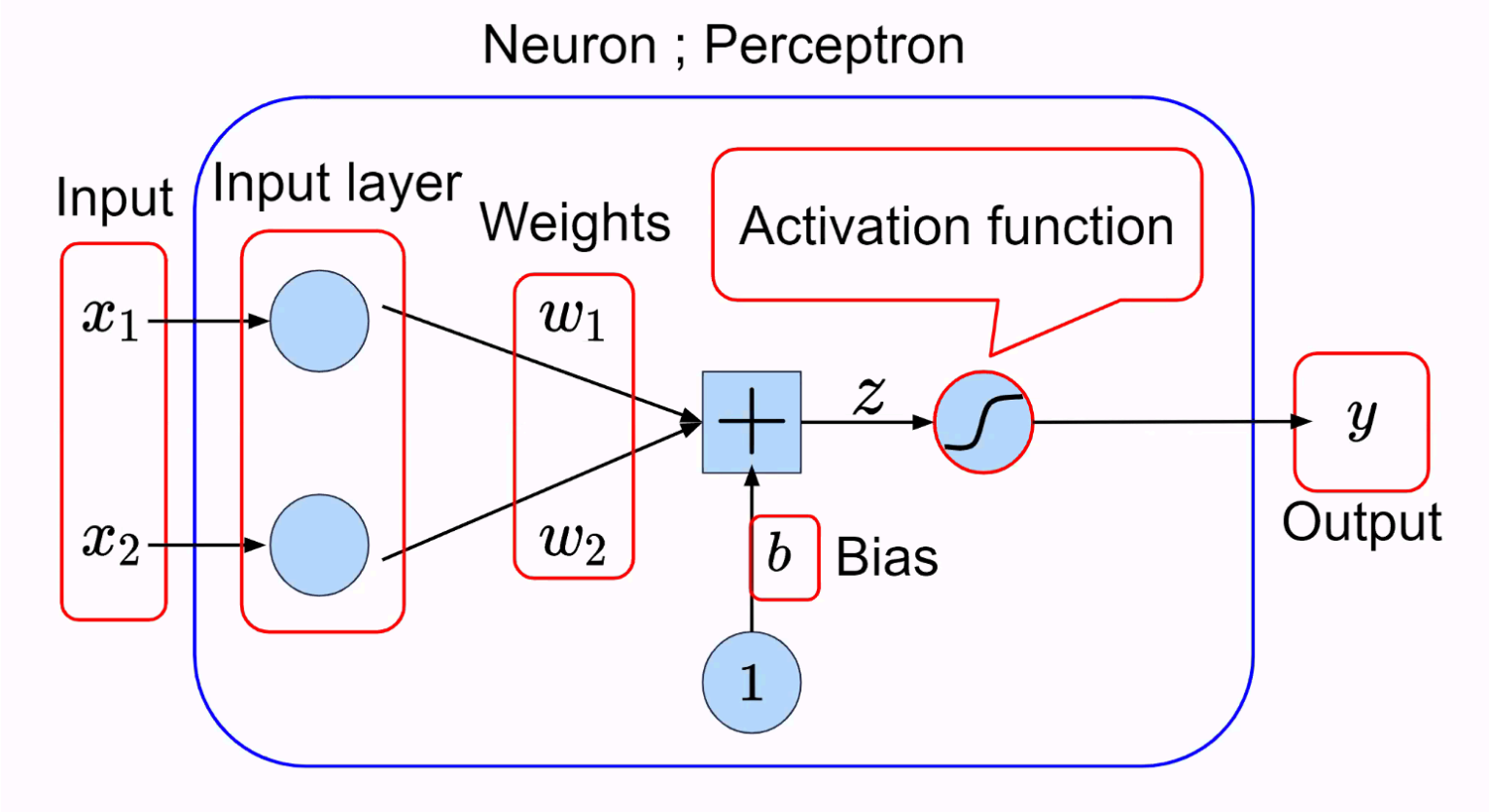
https://www.alibabacloud.com/blog/looking-forward-will-machine-intelligence-be-able-to-outsmart-hackeers_595907

Neural Network



Schematic of a biological neuron.

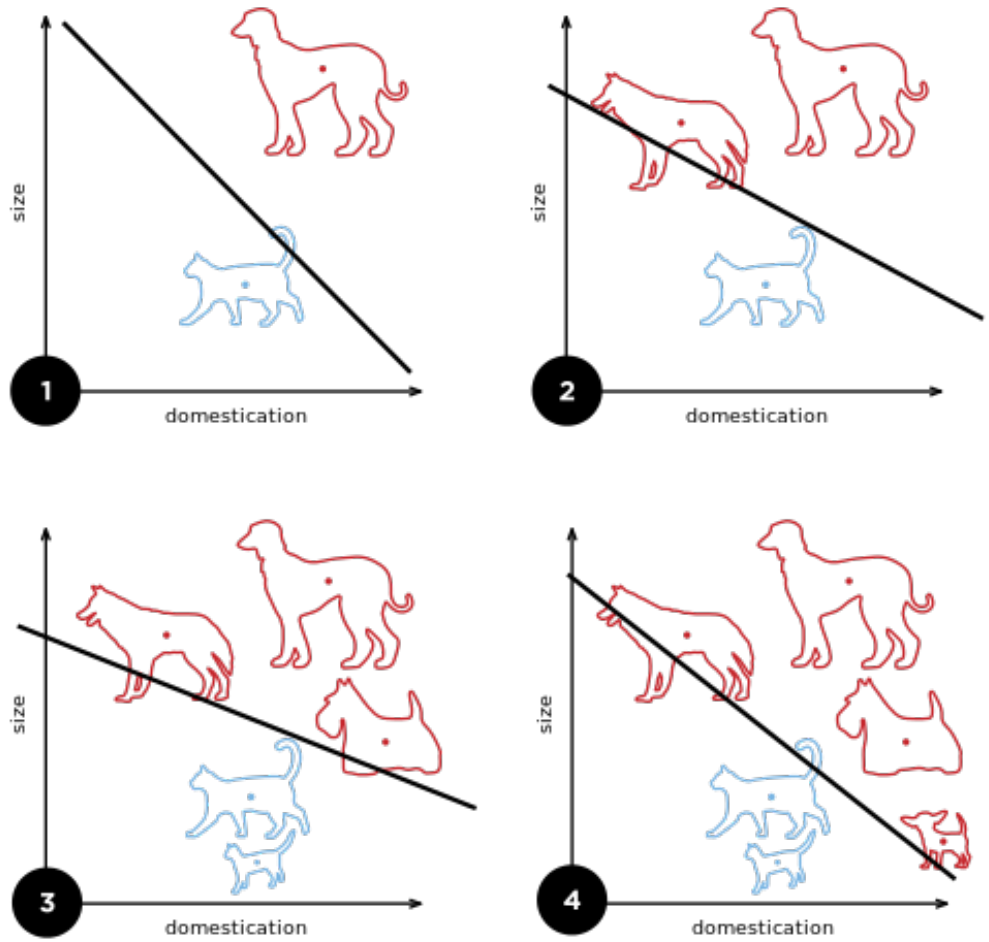
Perceptron



Developed by Frank Rosenblatt at the Cornell Aeronautical Laboratory in 1958 (https://en.wikipedia.org/wiki/Perceptron)

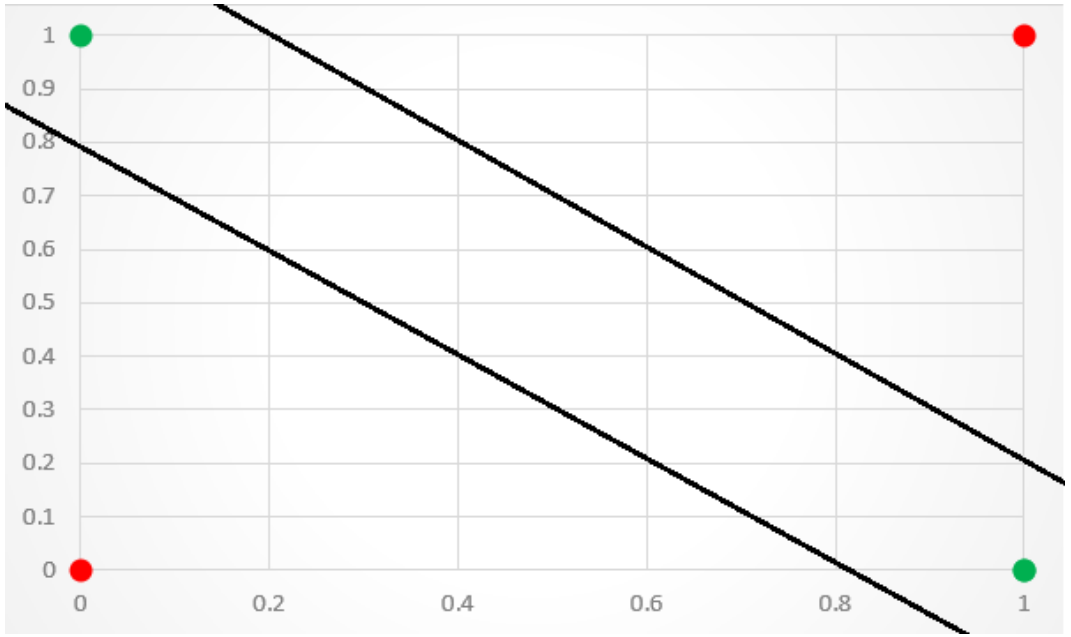
Perceptron for classification problems

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{otherwise} \end{cases}$$



XOR Problem

Input1	input2	Output
0	0	0
1	0	1
0	1	1
1	1	0



- Minksy and Papert (1969)
- Solution: Multilayer NN + Backpropagation

What

- Dataset: data for training and testing
  - Requires preprocessing
- Model: What the network learns
  - Training, validation, and testing
- Inference: Model in action
  - Predicting based on the learned model

Rank, dimension, axes, and shape

- Rank: number of dimensions
- Dimension: 2D, 3D, etc.
- Axes: indices of a dimension
- Shape: number of elements in each dimension
  - a scalar has a rank 0 and an empty shape ()
  - a vector has rank 1 and a shape of (D0)
  - a matrix has rank 2 and a shape of (D0, D1) and so on

How to train a model

- Define input and output
- Decide on the input features
- Build layers of the network: hyperparameters
  - Number of layers
  - Learning rate
  - Number of epochs
  - Etc.
- Train the model: parameters
  - Weights and biases
  - Variables in TensorFlow
- Verify the model:
  - Using verification data

Models and Functions

- Hypothesis:

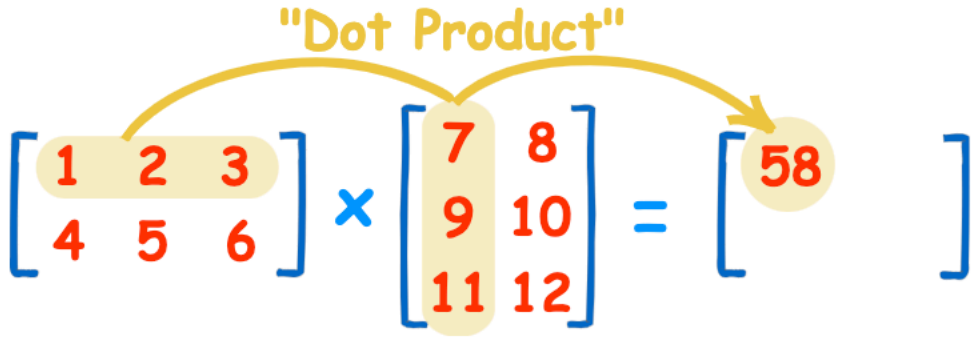
$$H(x_1, x_2, x_3) = w_1 x_1 + w_2 x_2 + w_3 x_3 + b$$

- Activation:
  - Sigmoid, ReLU, LeakyReLU, etc.

- Cost:

$$cost(W, b) = \frac{1}{m} \sum_{I=1}^{m} (H(x_1^{(i)}, x_2^{(i)}, x_3^{(i)}) - y^{(i)})^2$$

Matrix multiplication



The "Dot Product" is where we **multiply matching members**, then sum up:

$$(1, 2, 3) \bullet (7, 9, 11) = 1\times7 + 2\times9 + 3\times11$$
$$= 58$$

https://www.mathsisfun.com/algebra/matrix-multiplying.html

Functions using matrix

- Hypothesis
  - Y = WX + b

$$\begin{pmatrix} x_1 & x_2 & x_3 \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix} = \begin{pmatrix} x_1 w_1 + x_2 w_2 + x_3 w_3 \end{pmatrix}$$

- Activation function
- Cost function

$$cost(W, b) = \frac{1}{m} \sum_{I=1}^{m} (H(x_1^{(i)}, x_2^{(i)}, x_3^{(i)}) - y^{(i)})^2$$

Activation functions

- Introduces non-linearity
- Normalizes the output: activation functions are also called Normalization functions
- Different kinds
  - Step function: $$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{otherwise} \end{cases}$$
  - Sigmoid:

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

$S(x)$ = sigmoid function

e = Euler's number

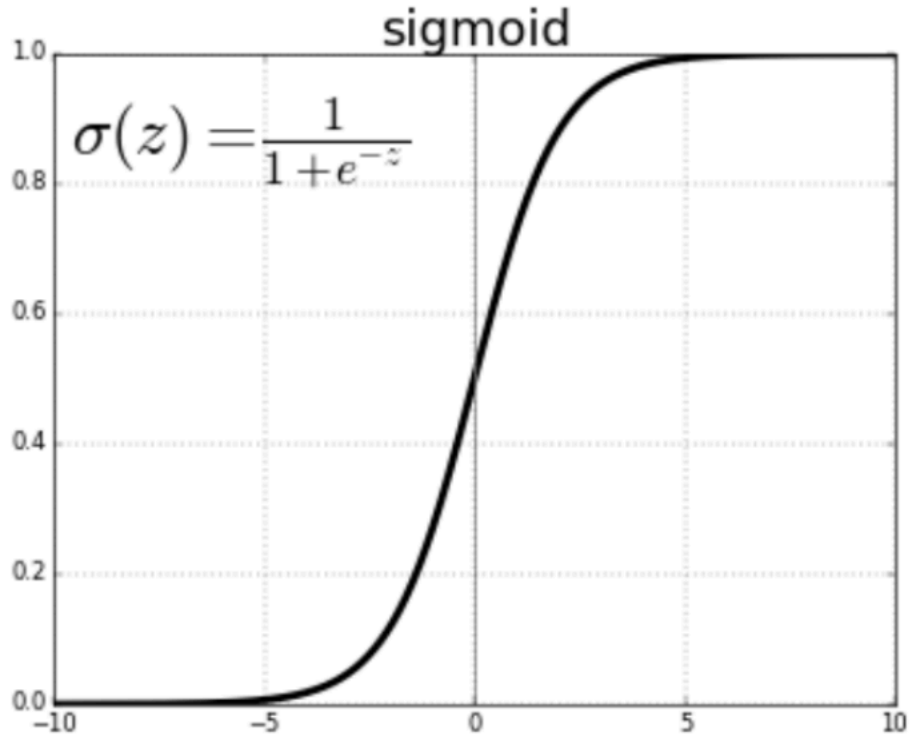
Activation function: Sigmoid

- Sigmoid:

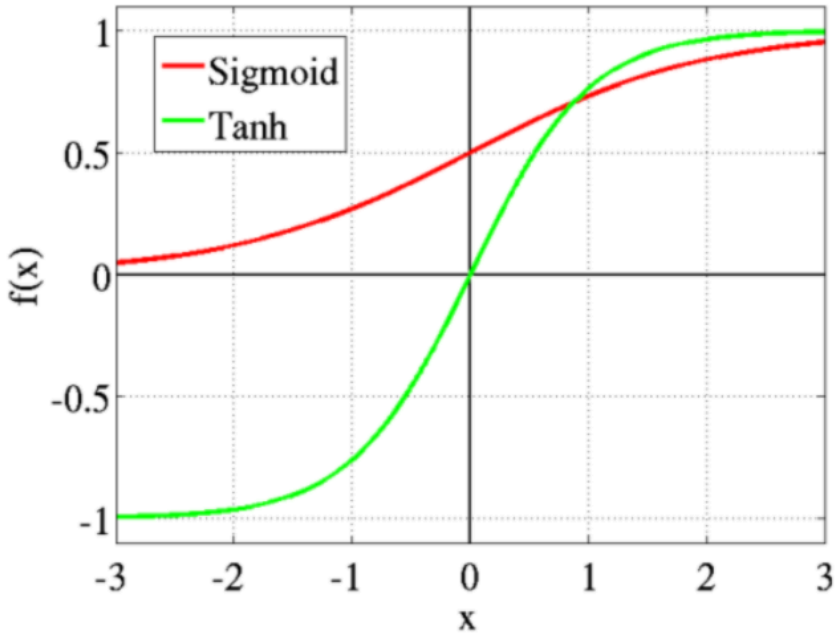
$$S(x) = \frac{1}{1 + e^{-x}}$$

$S(x)$ = sigmoid function

e = Euler's number



Activation functions: Sigmoid and Tanh



https://www.neuronactivator.com/blog/what-even-is-activation-function

Activation Functions: ReLU and Leaky ReLU

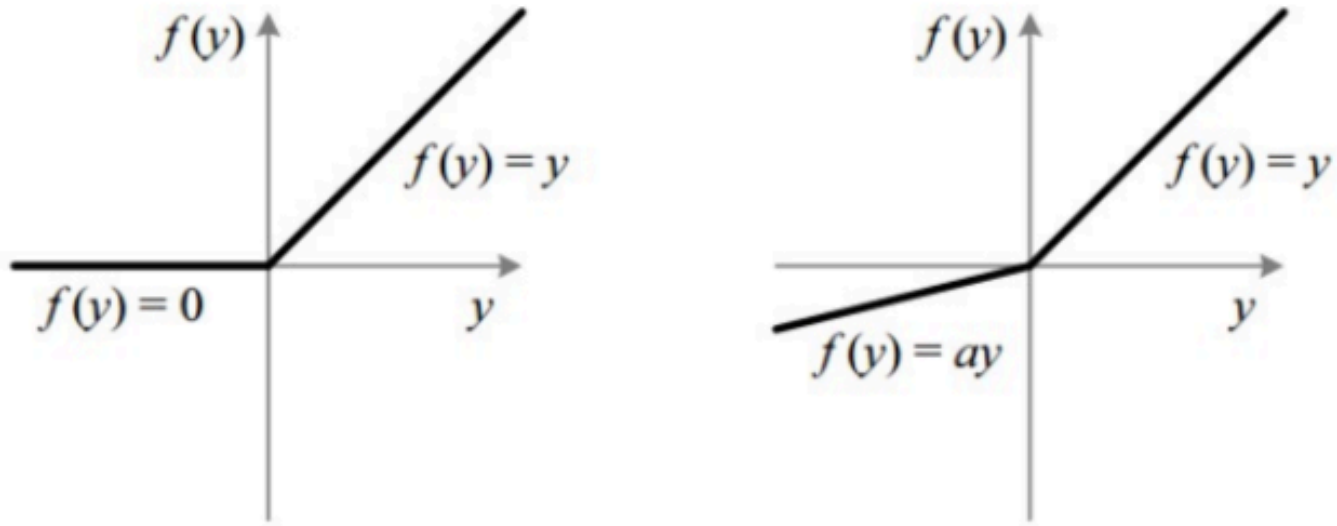


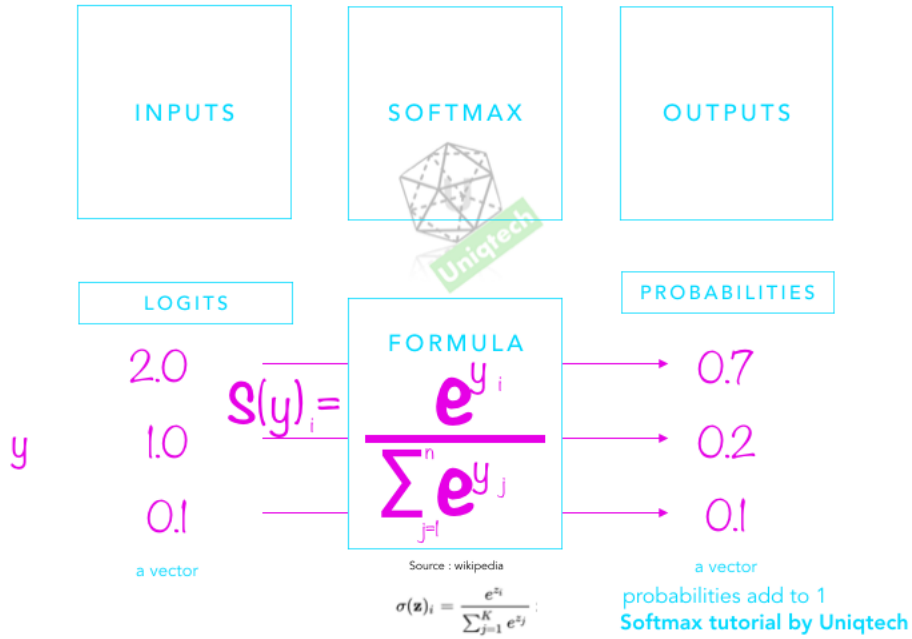
Fig : ReLU v/s Leaky ReLU

https://www.neuronactivator.com/blog/what-even-is-activation-function

Activation Function: Softmax

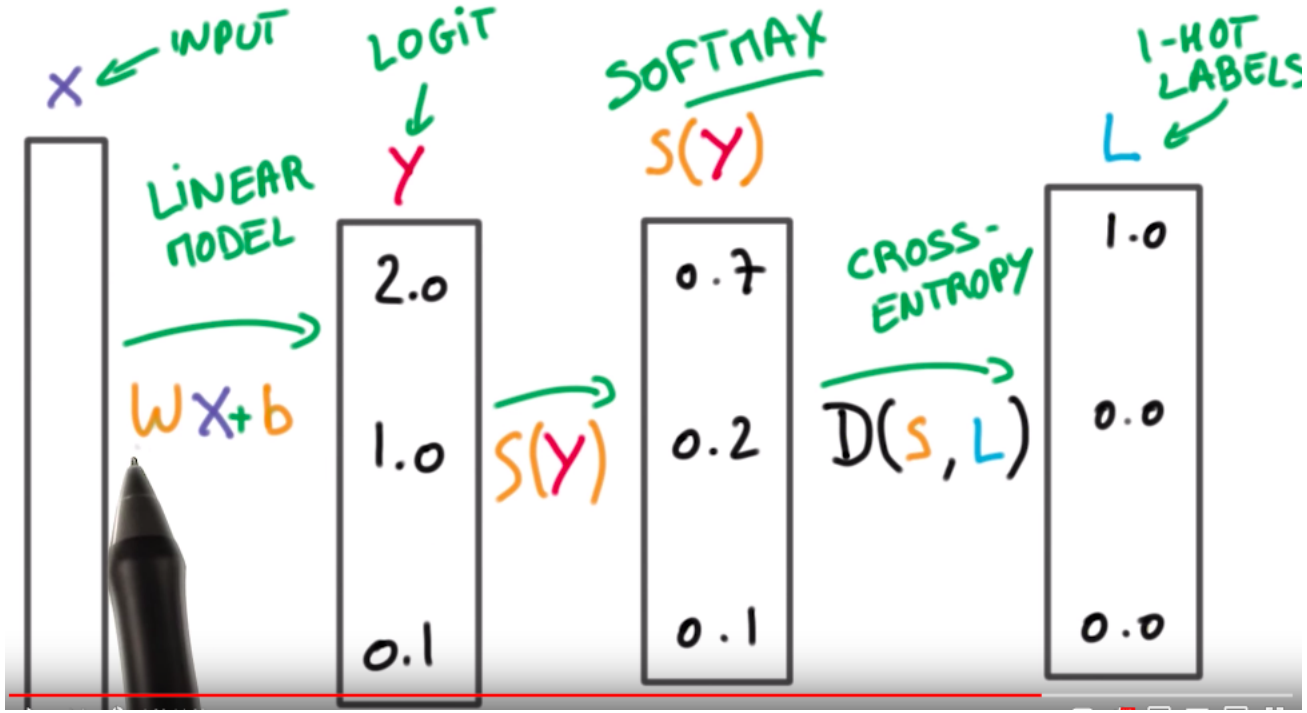
- The softmax function is often used in the final layer of a neural network-based classifier.
- All probabilities sum to one
- Often used with a log loss (or cross-entropy) cost function
- To solve a non-linear variant of multinomial logistic regression.

Activation Function: Softmax



https://medium.com/data-science-bootcamp/understand-the-softmax-function-in-minutes-f3a59641e86d

Loss Function: Cross Entropy

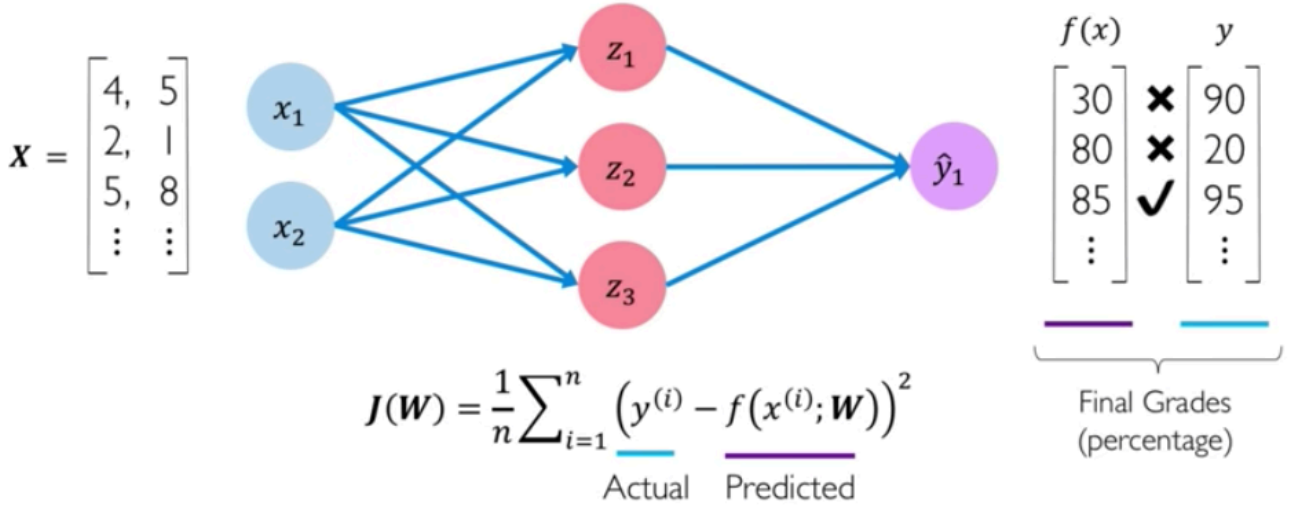


https://medium.com/data-science-bootcamp/understand-cross-entropy-loss-in-minutes-9fb263caee9a

Loss Function: Mean Squared Error

Mean Squared Error Loss

Mean squared error loss *can be used with regression models that output continuous real numbers*




```
loss = tf.reduce_mean( tf.square(tf.subtract(y, predicted)) )
```

http://introtodeeplearning.com/

Training is minimizing the cost

- Training a neural network is basically the problem of minimizing the cost function.
- Gradient descent is the most popular approach.
- For a given cost function, minimize cost(W, b)

Backpropagation

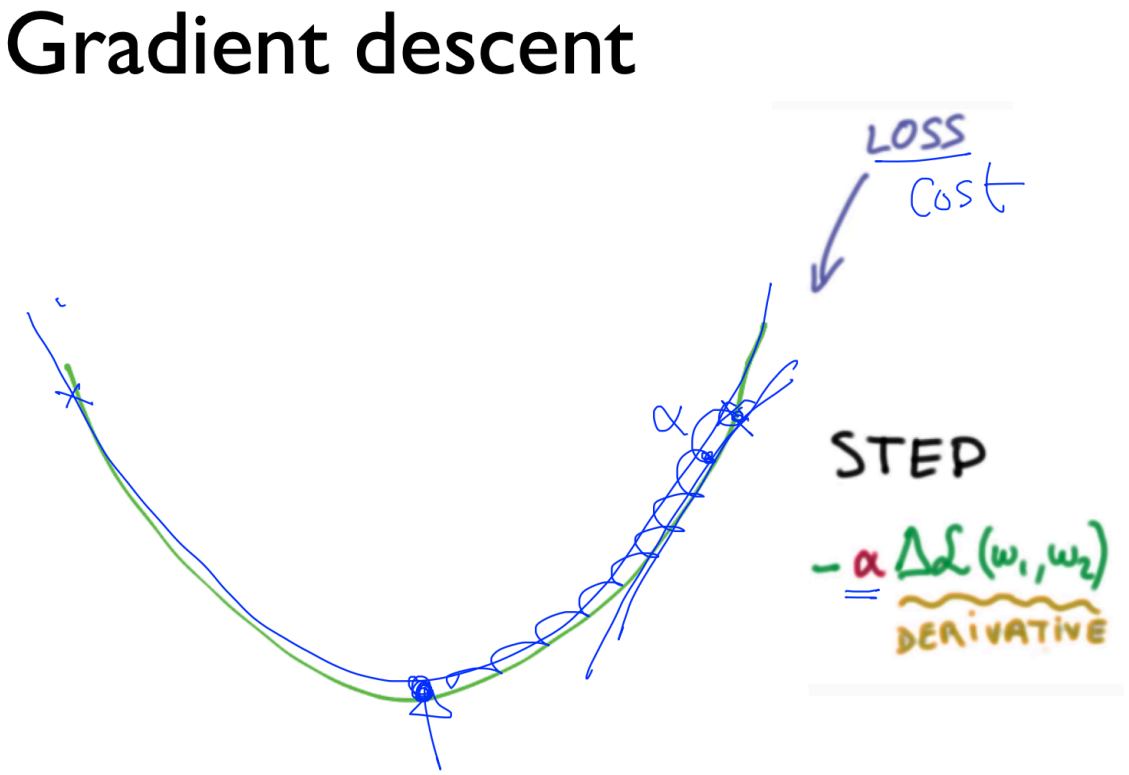
1. Use Calculus: the study of change
2. Batches, mini-batches, and stochastic batches

Steps:

1. Take the derivative: the slope of a tangent line at a specific point in time
2. Partial derivative
3. The chain rule: composite functions

- Backpropagation of errors: Updating weights using gradient descent

Backpropagation using gradient descent



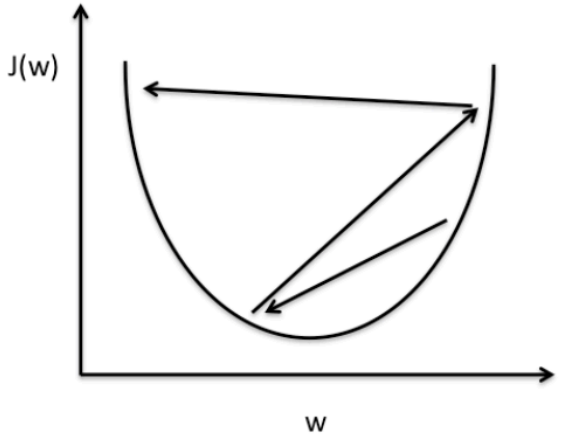
https://www.udacity.com/course/viewer#!/c-ud730/l-6370362152/m-6379811827

Problems with training

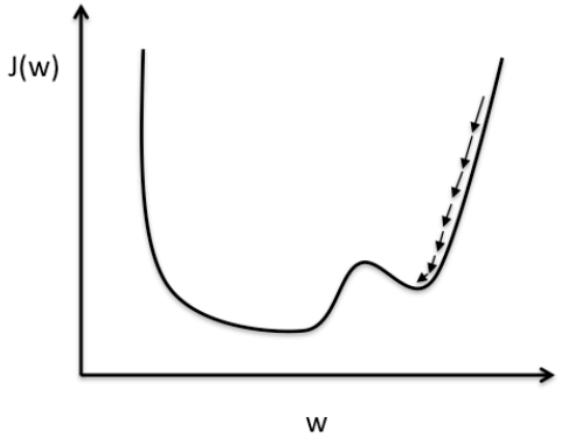
- Initial weights: random means you can’t predict
- Vanishing/exploding gradients
- Local minima
- Overfitting & underfitting
- Hyperparameters: learning rate, number of layers, etc. require human intelligence!

Learning rate

Learning rate: NaN!



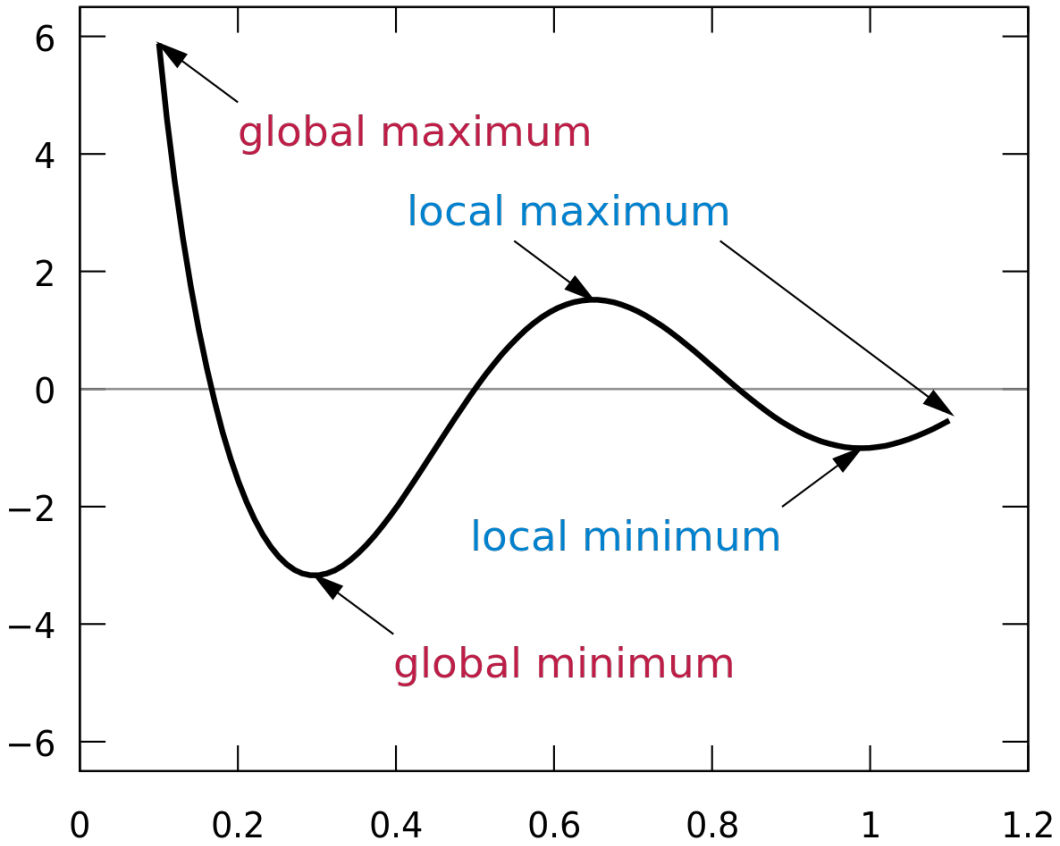
Large learning rate: Overshooting.



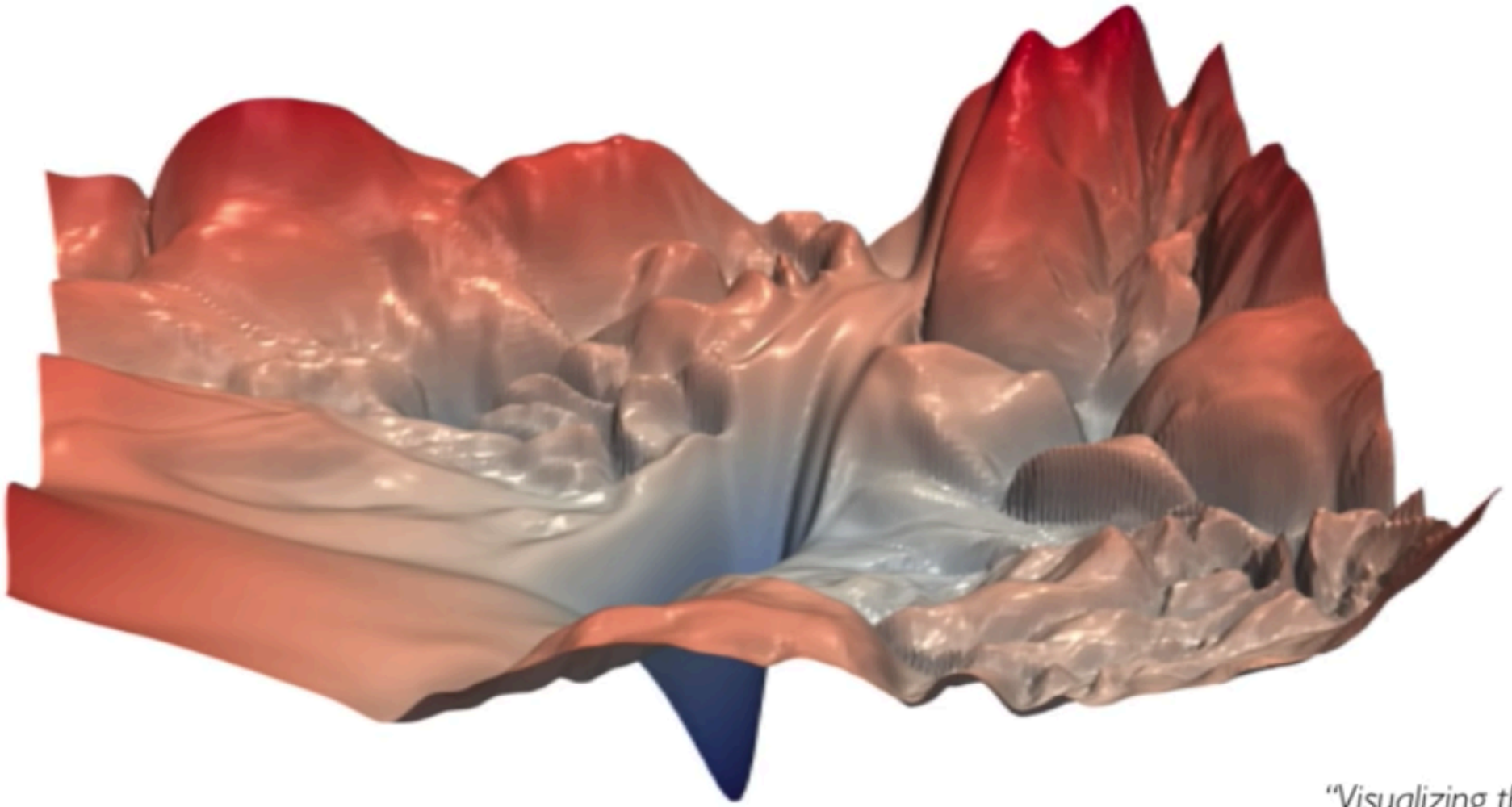
Small learning rate: Many iterations until convergence and trapping in local minima.

http://sebastianraschka.com/Articles/2015_singlelayer_neurons.html

Local minima



Training Neural Networks is Difficult



"Visualizing the loss landscape of neural nets". Dec 2017.

Assignment

- Train & demo an AI model using either Teachable Machine or Peltarion
  - You are welcome to use the ”flag classification” problem outlined in: https://github.com/changsin/DeepLearningMadeEasy/blob/main/flag_classification/Problem-Definition.ipynb or you can choose a different topic
  - This is an individual assignment, but you are welcome to discuss in the forum and among your classmates.
  - Record a video clip for your demo

Lab time

- To clone: from your terminal
  - >git clone https://github.com/changsin/DeepLearningMadeEasy.git
- Or use google colab to open the git hub repository
- Git is an open source version control system
  - Github is a host service using git.