Advanced Workshop on Earthquake Fault Mechanics: Theory, Simulation and Observation (Trieste, 2019)

a gentle introduction INTO MACHINE LEARNING

DOGS vs Cats

Distinguishing features?

- Fur?
- Number of eyes? Number of legs?
- Whiskers?
- Shape of ears?

Additional complications

- Multiple objects

Additional complications

- Multiple objects
- Incomplete data

Additional complications

- Multiple objects
- Incomplete data
- Conflicting information

Additional complications

- Multiple objects
- Incomplete data
- Conflicting information
- Uncategorised objects

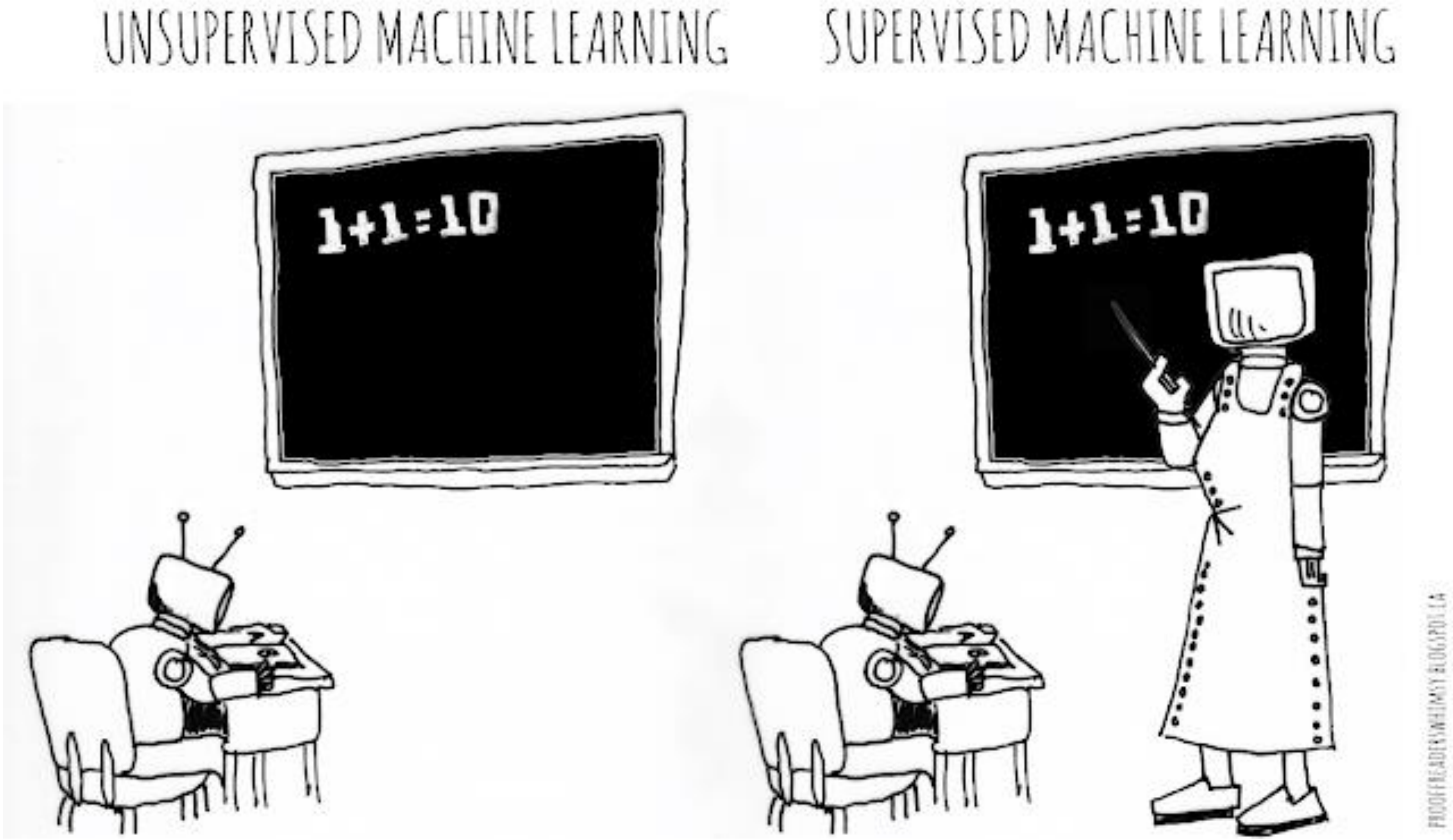
Advantage of Machine Learning

In traditional approaches you have to be specific

In Machine Learning the specifics are learned

Today’s/Tomorrow’s Menu

1. General overview of ML (history, examples, concepts)
   + basic hands-on tutorial
2. ML in Geosciences (examples & techniques)
   + advanced hands-on tutorial
3. Best practices and pitfalls
   + real-world exercise
4. Random Forests
   + real-world exercise



Part 1:
General overview of ML

AI vs. ML vs. DL

Artificial Intelligence (AI)

- Chess computers
- Computer games
- Robotics
- Decision policies

Machine Learning (ML)

- Random Forests
- Support Vector Machines

Deep Learning (DL)

Neural Networks with many (up to hundreds) of “layers”

Machine Learning Is Not New!

- 1957: Perceptrons
- 1970 (Dark Age): Backpropagation
- 1982: Recurrent Neural Network
- 1989: Reinforcement Learning
- **Mid-late 90's: "Silver Age"**
- **1995: Random Forests, SVM**
- **1997: IBM Deep Blue beats Kasparov**
- **1997: LSTM**
- **1998: CNN, handwriting analysis**
- **2009: ImageNet, "Golden Age"**
- **2010: Xavier initialisation**
- **2014: DeepMind, GAN**
- **2015: TensorFlow**
- **2016: AlphaGo**
- **2019: Neural ODE**

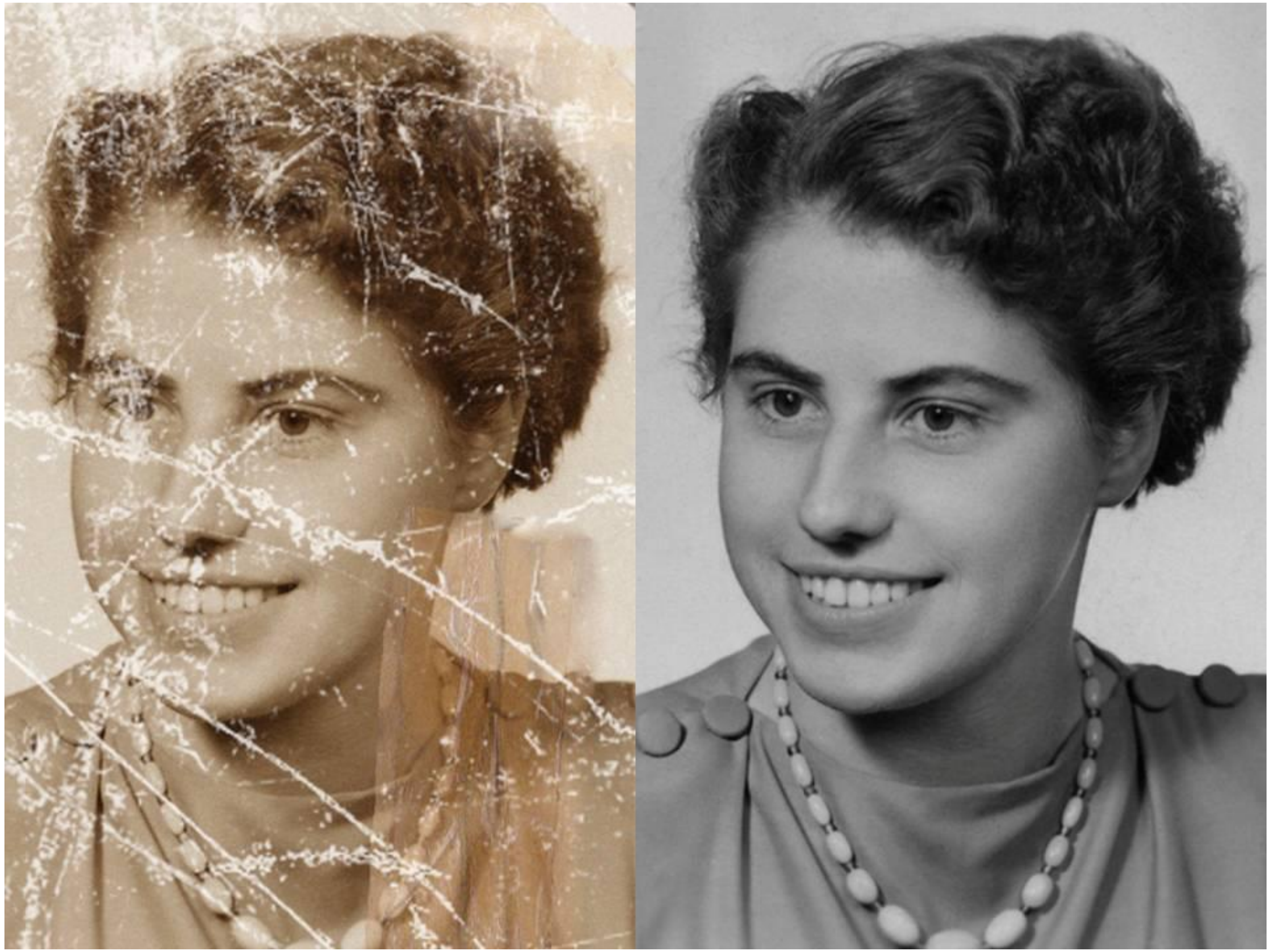
Machine Learning Is Not New!

Examples of ML (Deep Learning)



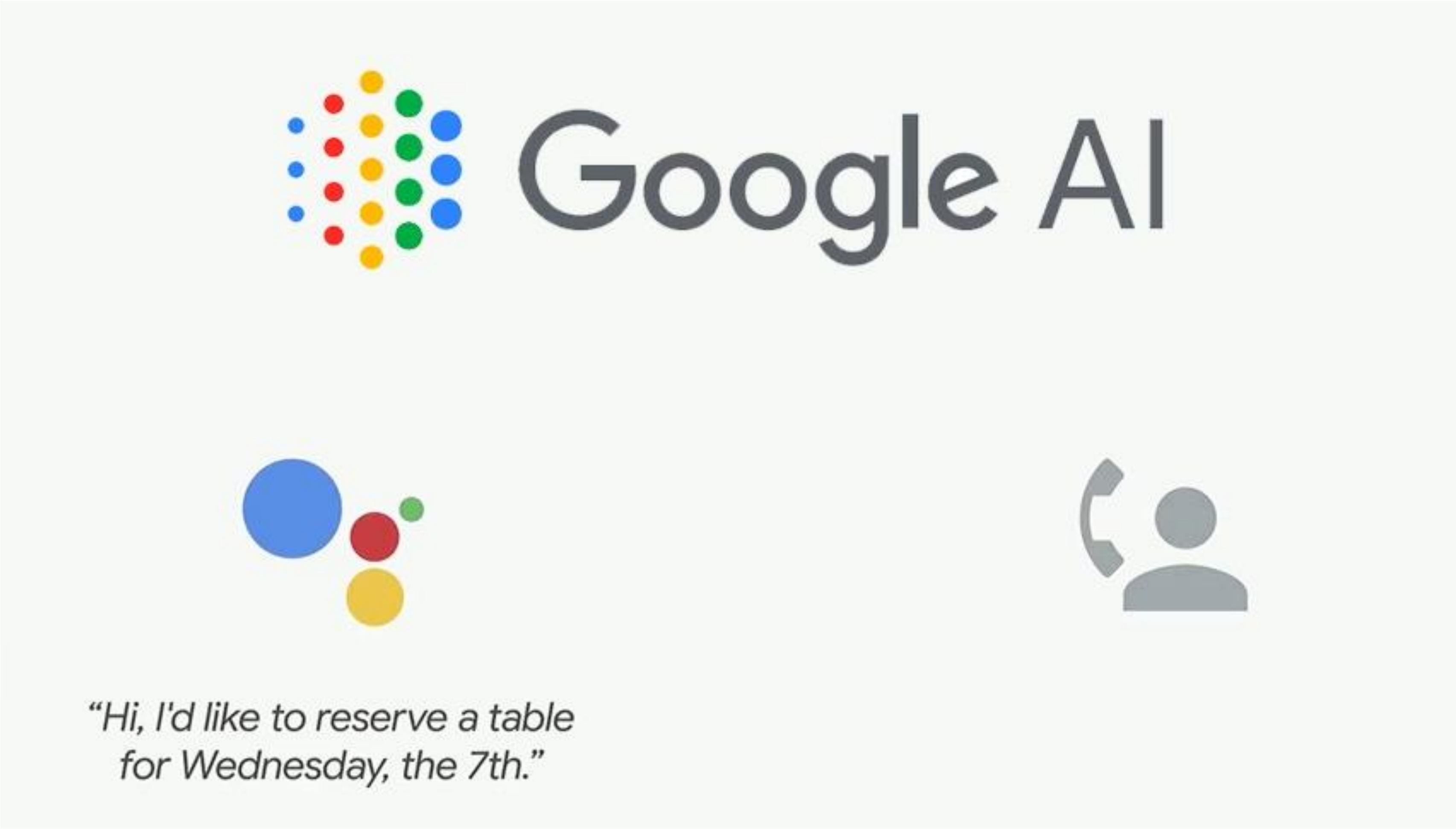
ThisPersonDoesNotExist.com

Examples of ML (Deep Learning)

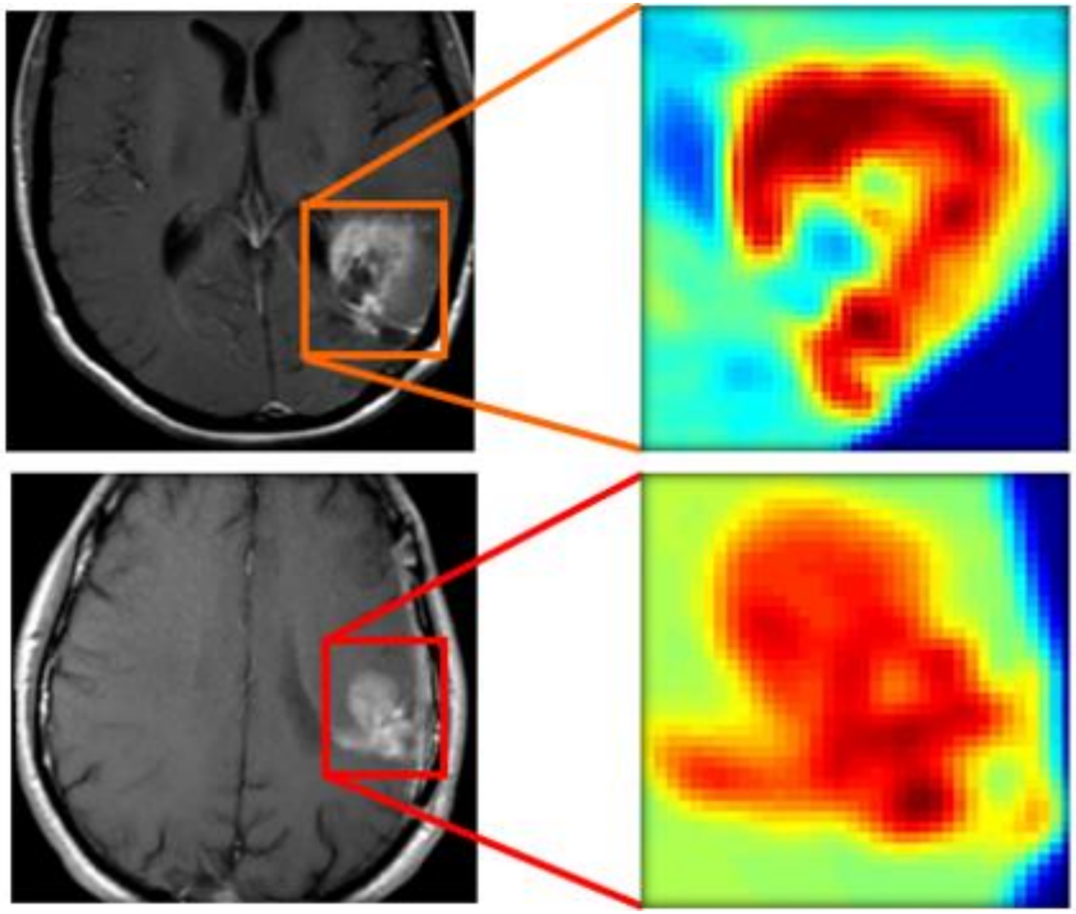


https://github.com/WenchenLi/PhotoRestoration

Examples of ML (Deep Learning)



Examples of ML (Deep Learning)

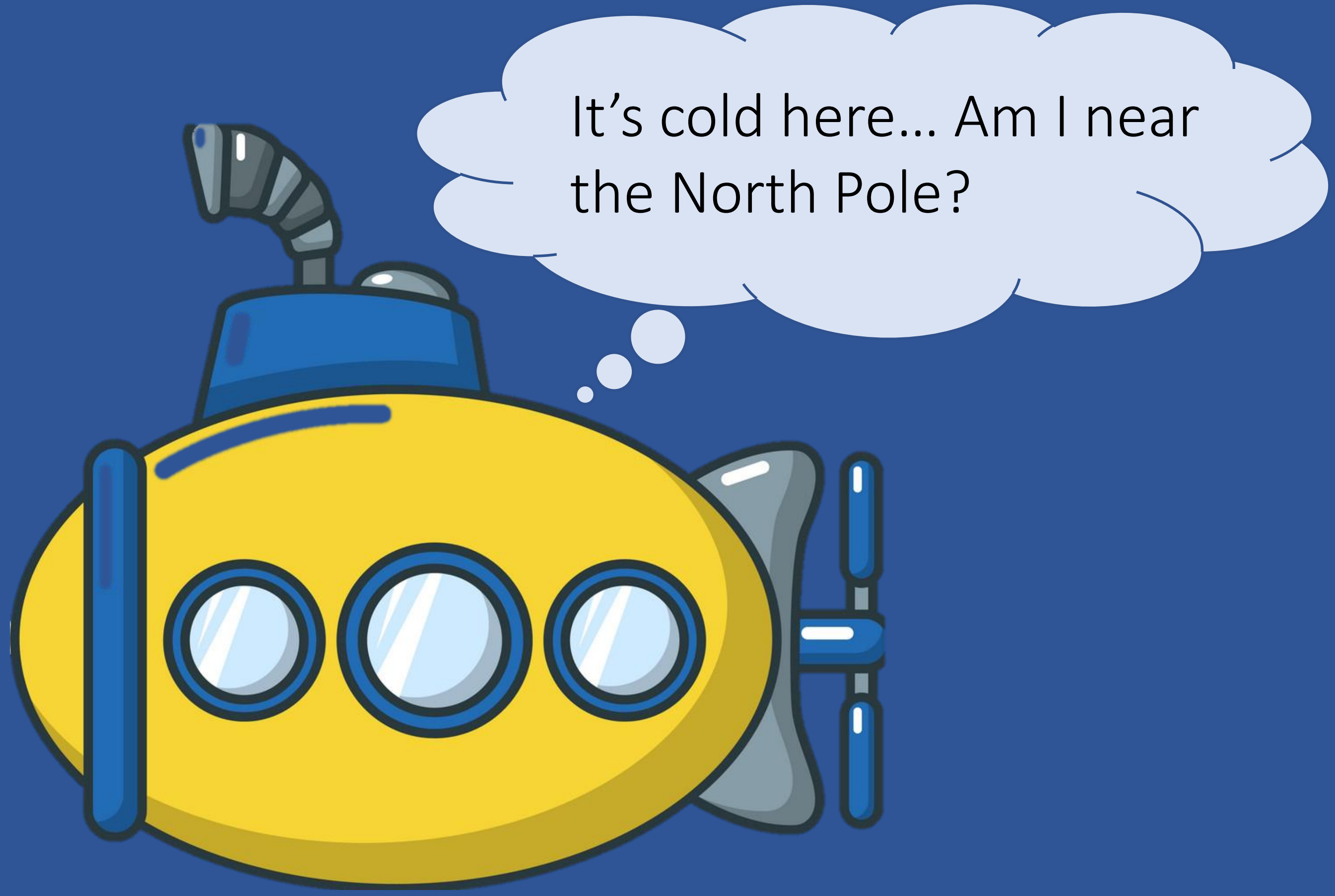


Case Engineering

Basic Theory of Neural Networks

Intuition

- Neural Nets find common patterns in data => generalisation
- Learning from examples (*supervised machine learning*)
- Apply generalised representation to new data
- Analogy:
    - Students doing exercises in class
    - Learning general patterns and approach
    - Extrapolating during exam (same problem, different numbers)


It’s cold here… Am I near the North Pole?

Basic Theory of Neural Networks

Simple objective

"For input x, give prediction y"

Input $\longrightarrow$ Output

Basic Theory of Neural Networks

<u>More precise objective</u>

"Find $f: x \to y'$"

Input $\longrightarrow$ $f(x)$ $\longrightarrow$ Output

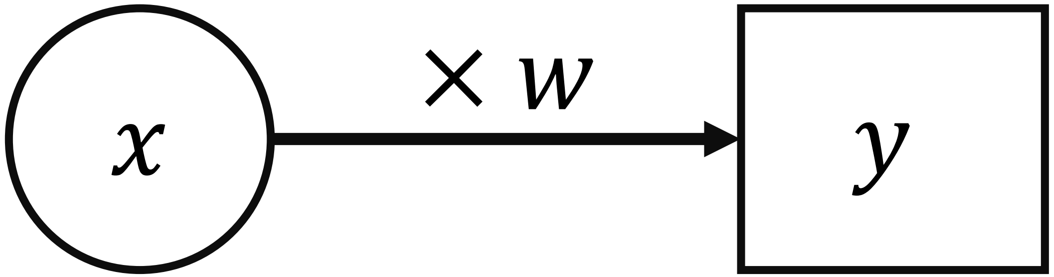
Basic Theory of Neural Networks

<u>More precise objective</u>

“Find optimal $\tilde{f}: x \to y \approx y'$*”*

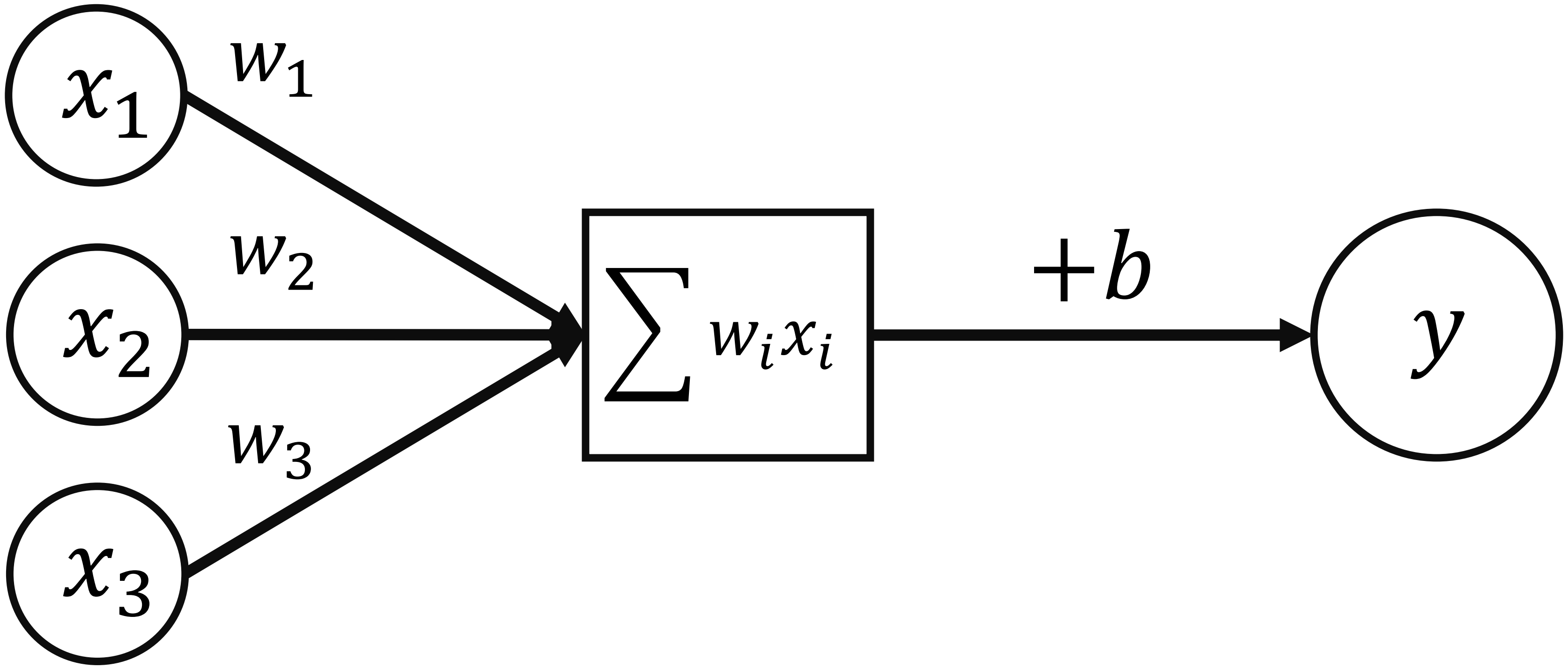
Input $\longrightarrow$ $\tilde{f}(x)$ $\longrightarrow$ Output

Single Neuron - Single Input



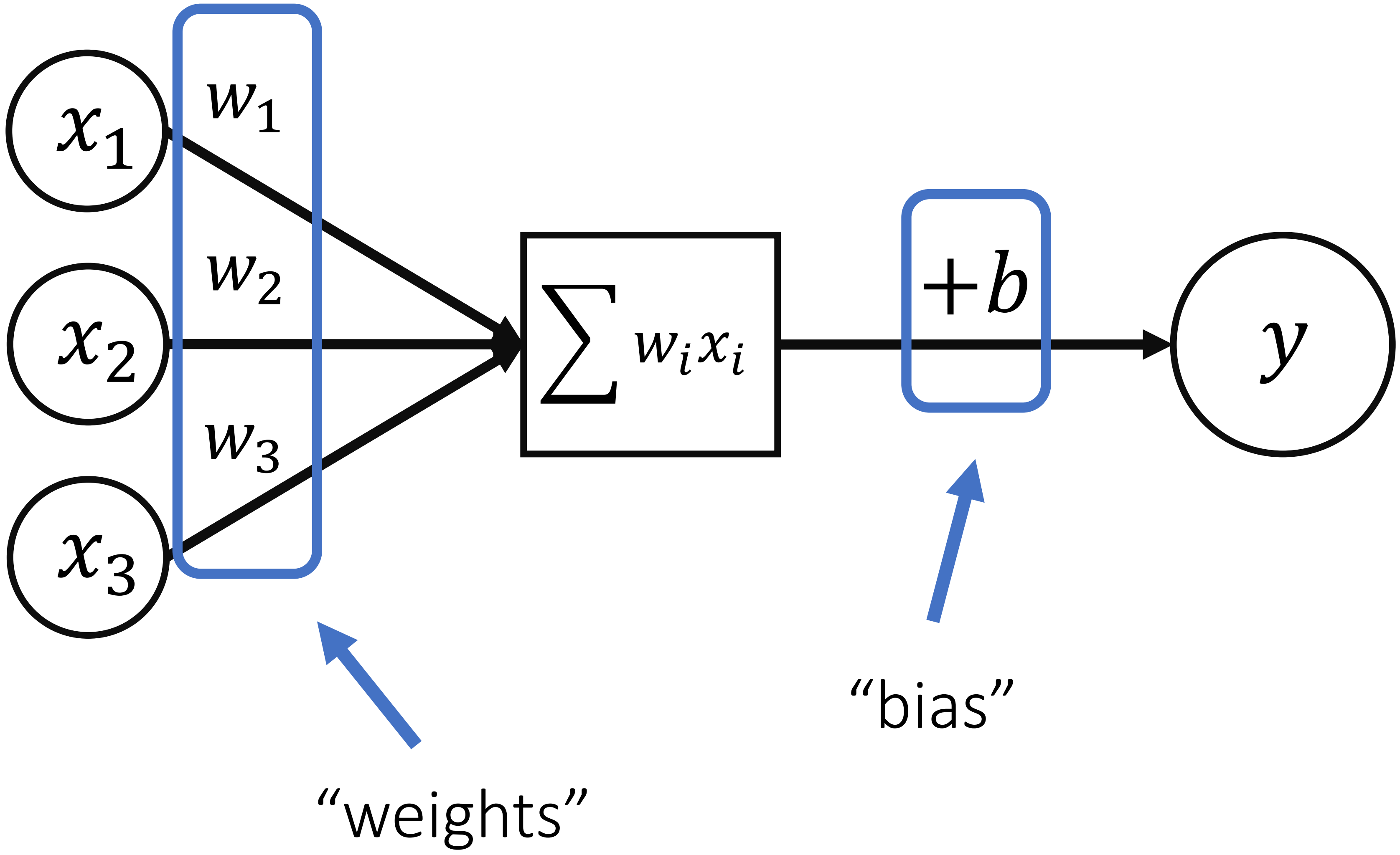
$$y = wx$$

Single Neuron - Multi Input

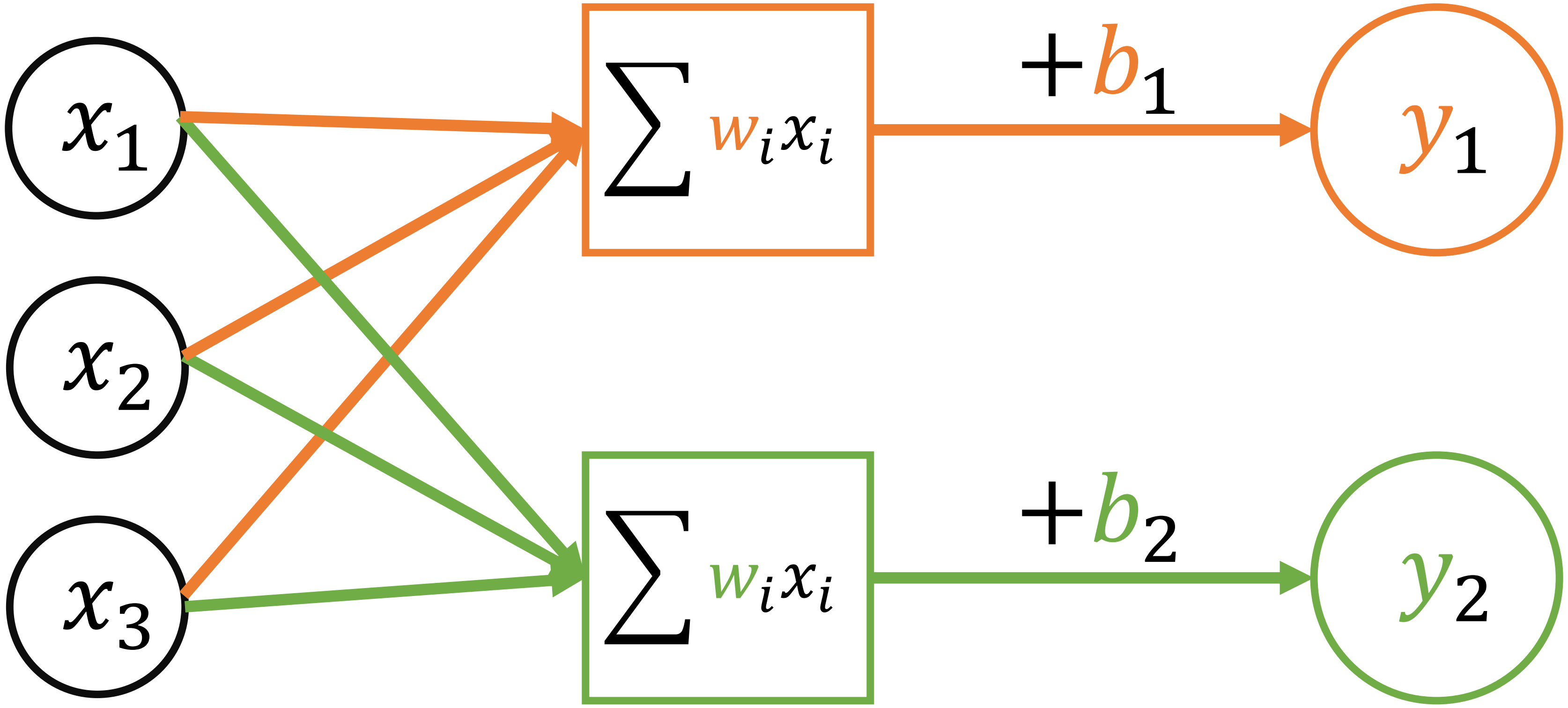


$$y = w_i x_i + b$$

Single Neuron - Multi Input

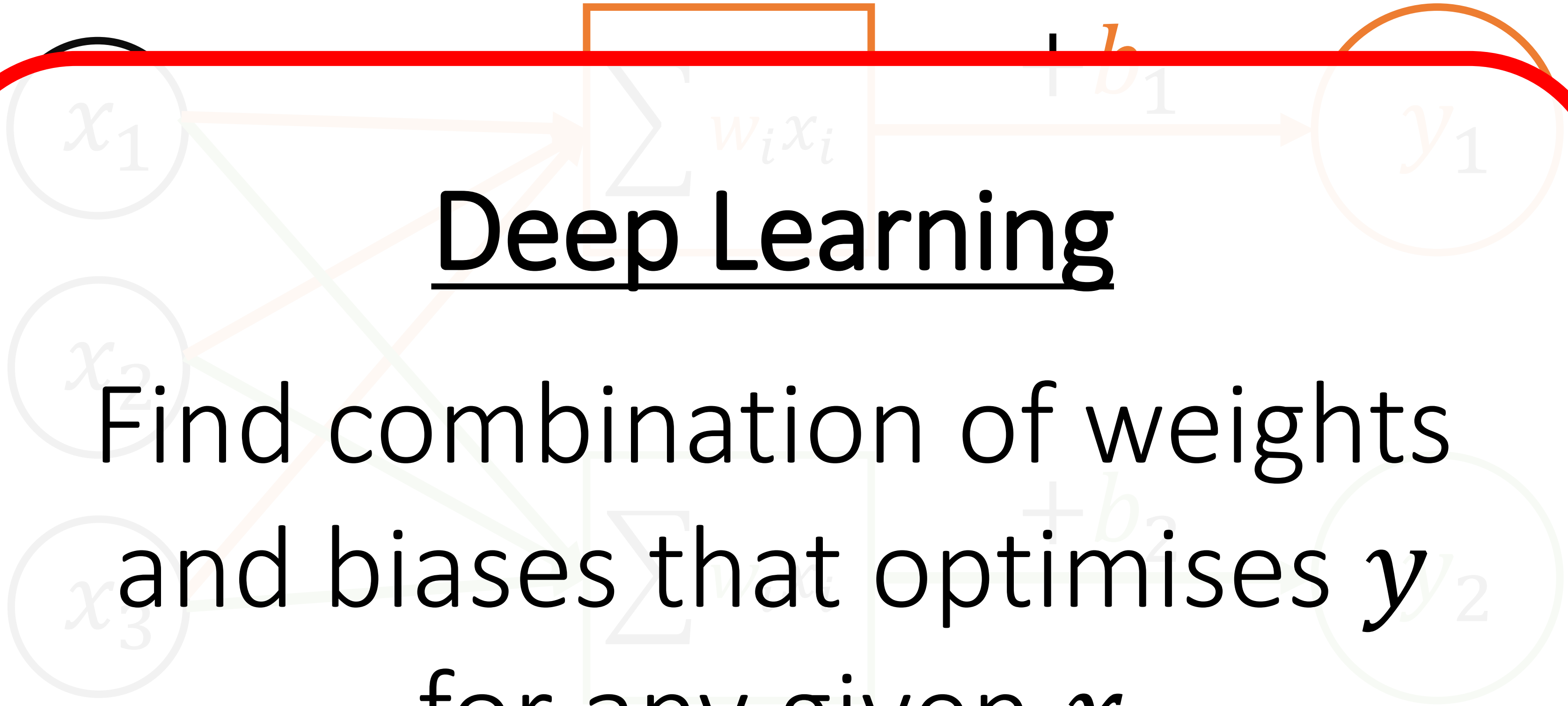


Single Layer of Neurons



$$y_i = W_{ij} x_j + b_i$$

Single Layer of Neurons

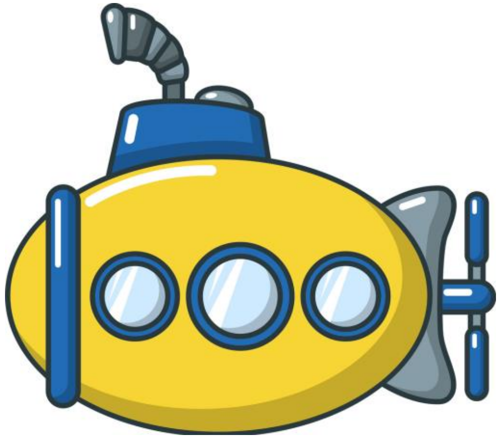


<u>Deep Learning</u>

Find combination of weights and biases that optimises y for any given x

$$y_i = W_{ij}x_j + b_i$$

Single Layer - Example



Input

- Temperature
- Pressure
- pH
- Salinity
- Light-intensity
- ...

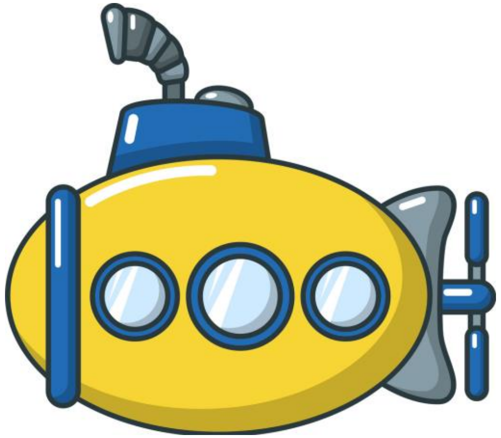
Neural Network

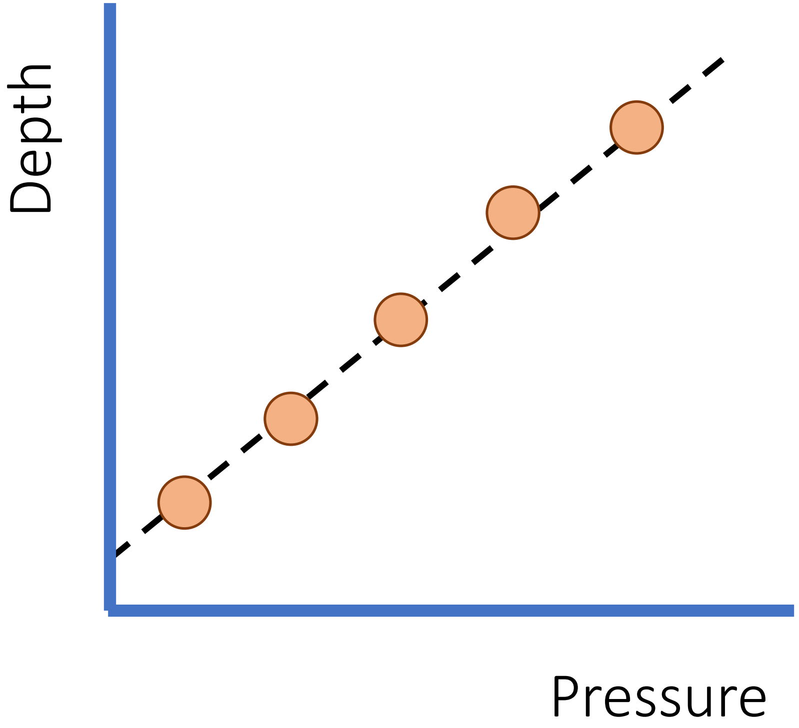
$$y_i = W_{ij}x_j + b_i$$

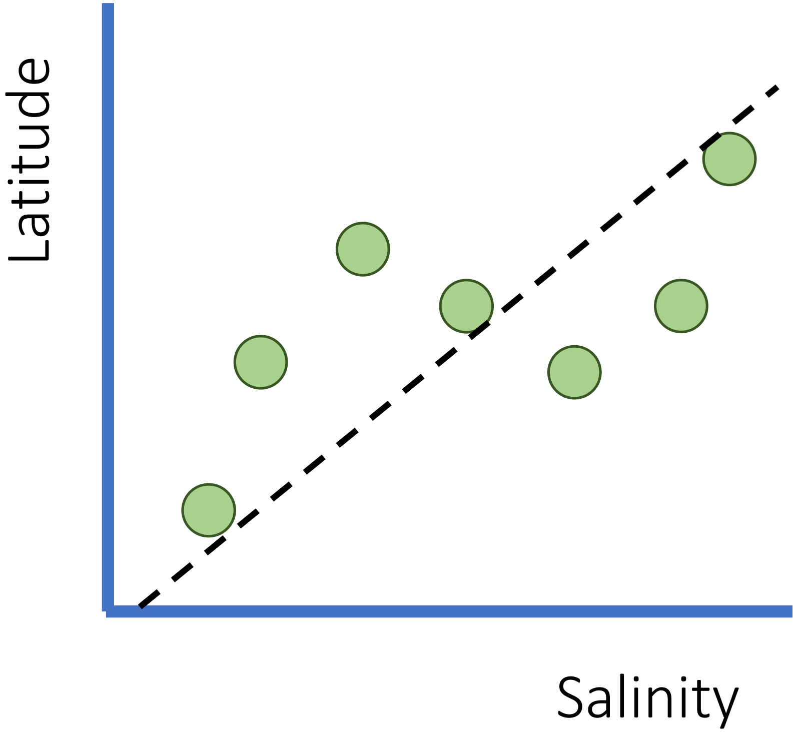
Prediction

- Latitude
- Longitude
- Depth

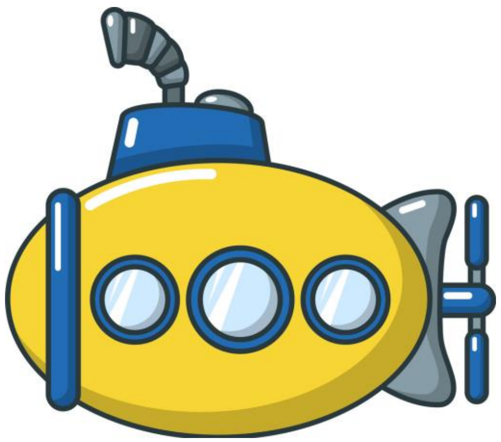
Single Layer - Example

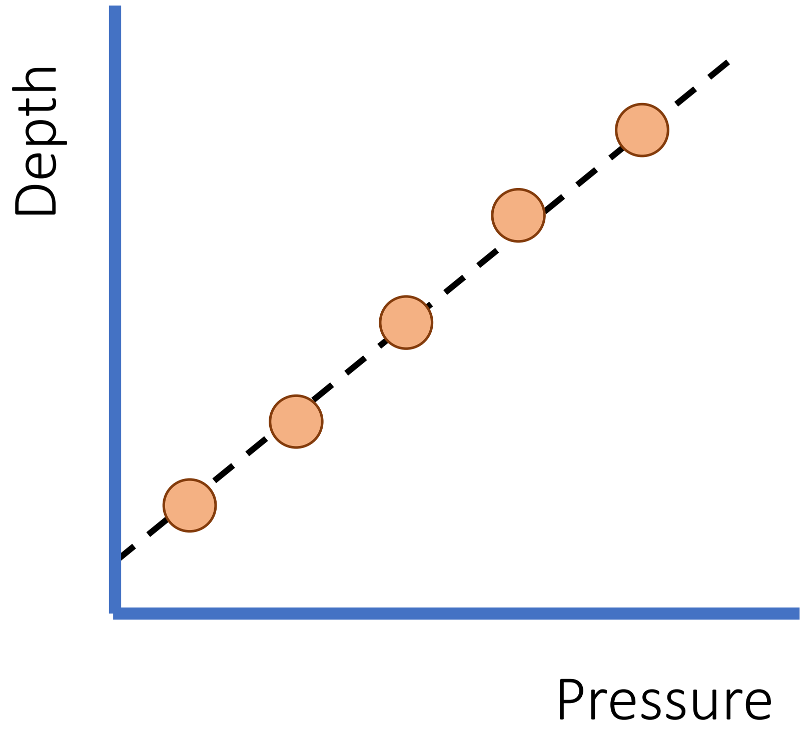


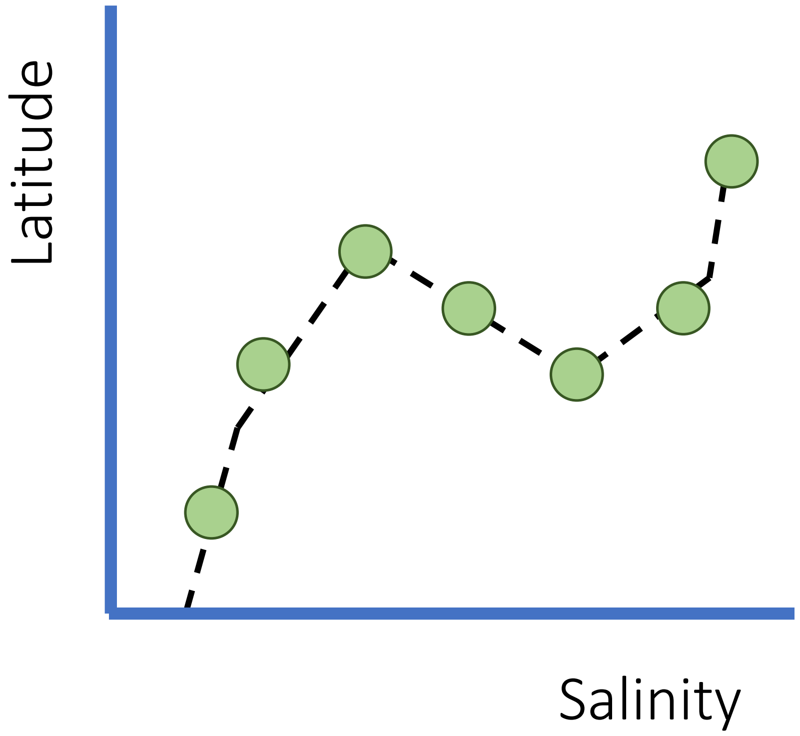




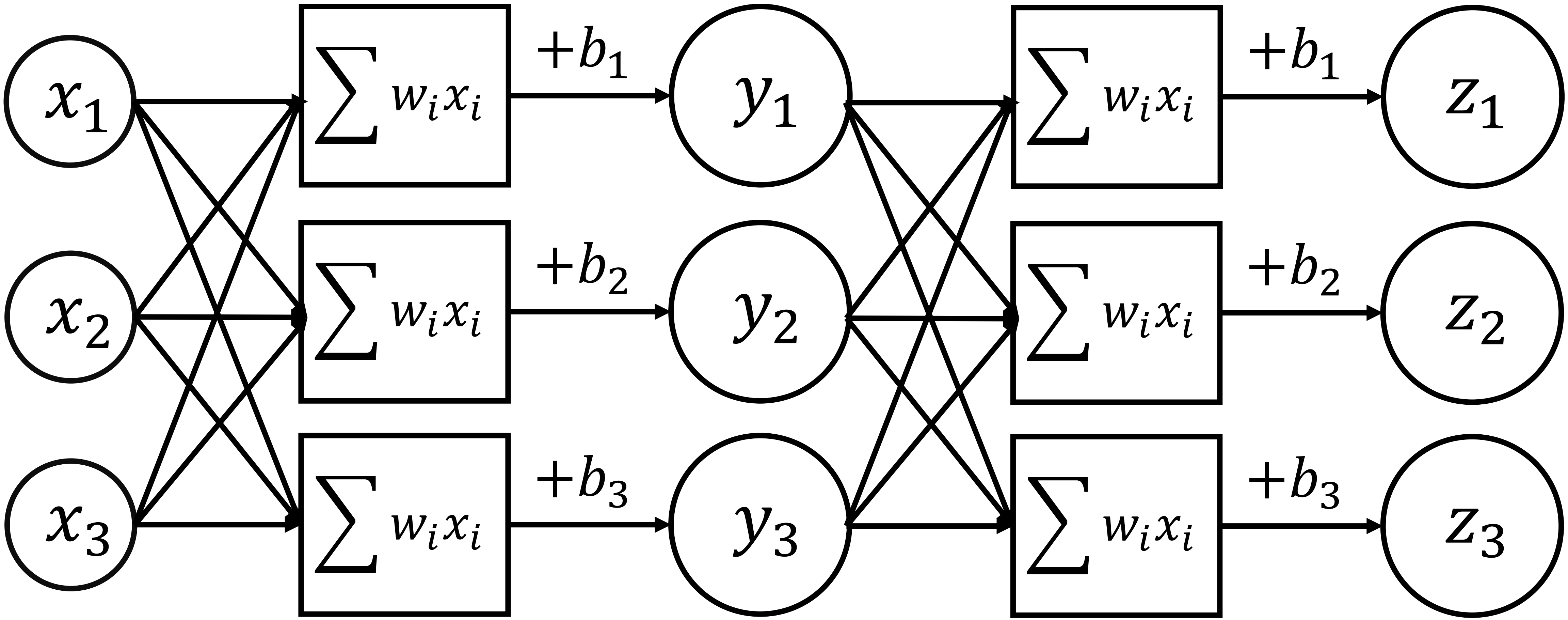
Single Layer - Example







Multiple Layers of Neurons



Multiple Layers of Neurons

$$
\begin{gathered}
x^{(1)} = W^{(1)} x^{(0)} + b^{(1)} \\
x^{(2)} = W^{(2)} x^{(1)} + b^{(2)} \\
x^{(3)} = W^{(3)} x^{(2)} + b^{(3)} \\
\vdots \\
x^{(n)} = W^{(n)} x^{(n-1)} + b^{(n)}
\end{gathered}
$$

Quiz

$$x^{(n)} = W^{(n)} x^{(n-1)} + b^{(n)}$$

a) Multiple layers = better
b) Multiple layers = worse
c) Makes no difference

Multiple Layers of Neurons

$$x^{(2)} = W^{(2)}x^{(1)} + b^{(2)}$$

$$= W^{(2)}\left[W^{(1)}x^{(0)} + b^{(1)}\right] + b^{(2)}$$

$$= W^{*}x^{(0)} + b^{*}$$

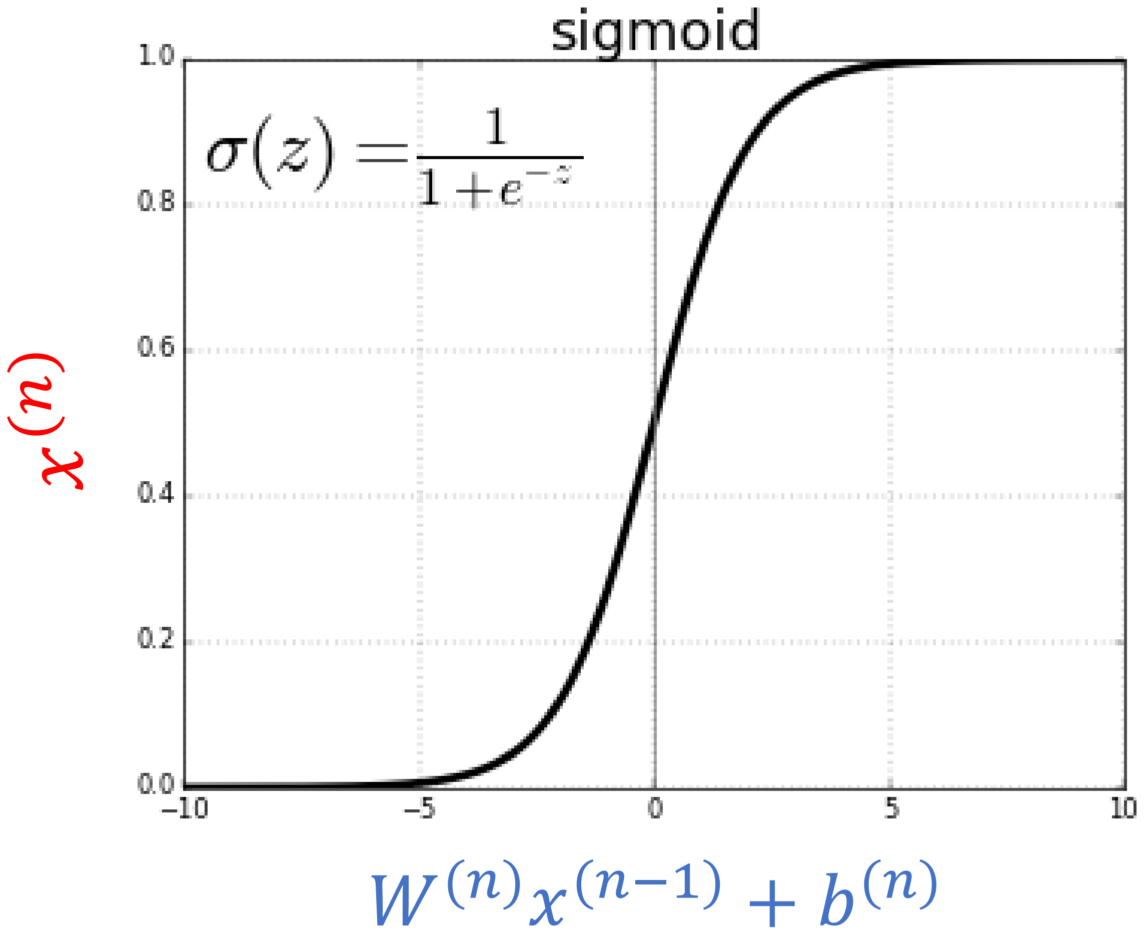
Non-linearity

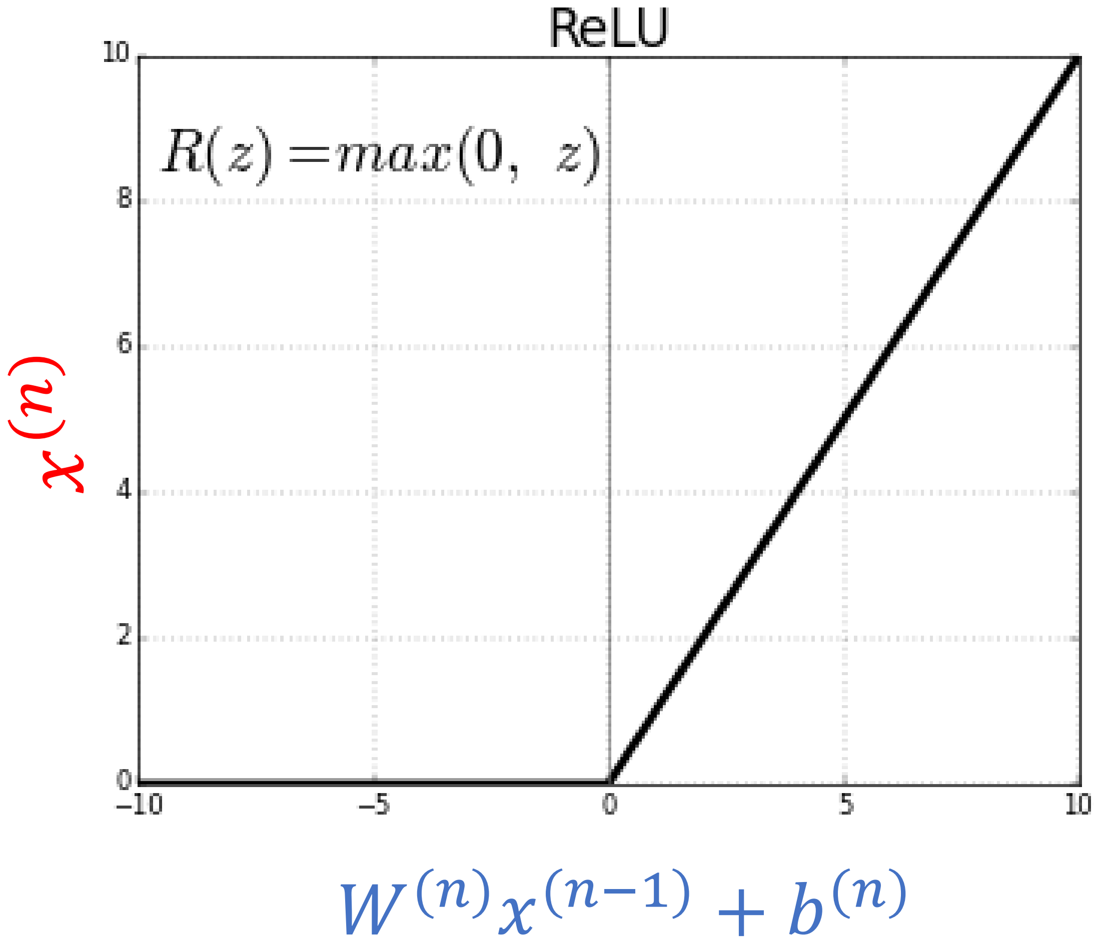
$$x^{(n)} = f\left(W^{(n)} x^{(n-1)} + b^{(n)}\right)$$

Non-linear function:
"Activation"

Activation Functions

$$x^{(n)} = f(W^{(n)} x^{(n-1)} + b^{(n)})$$





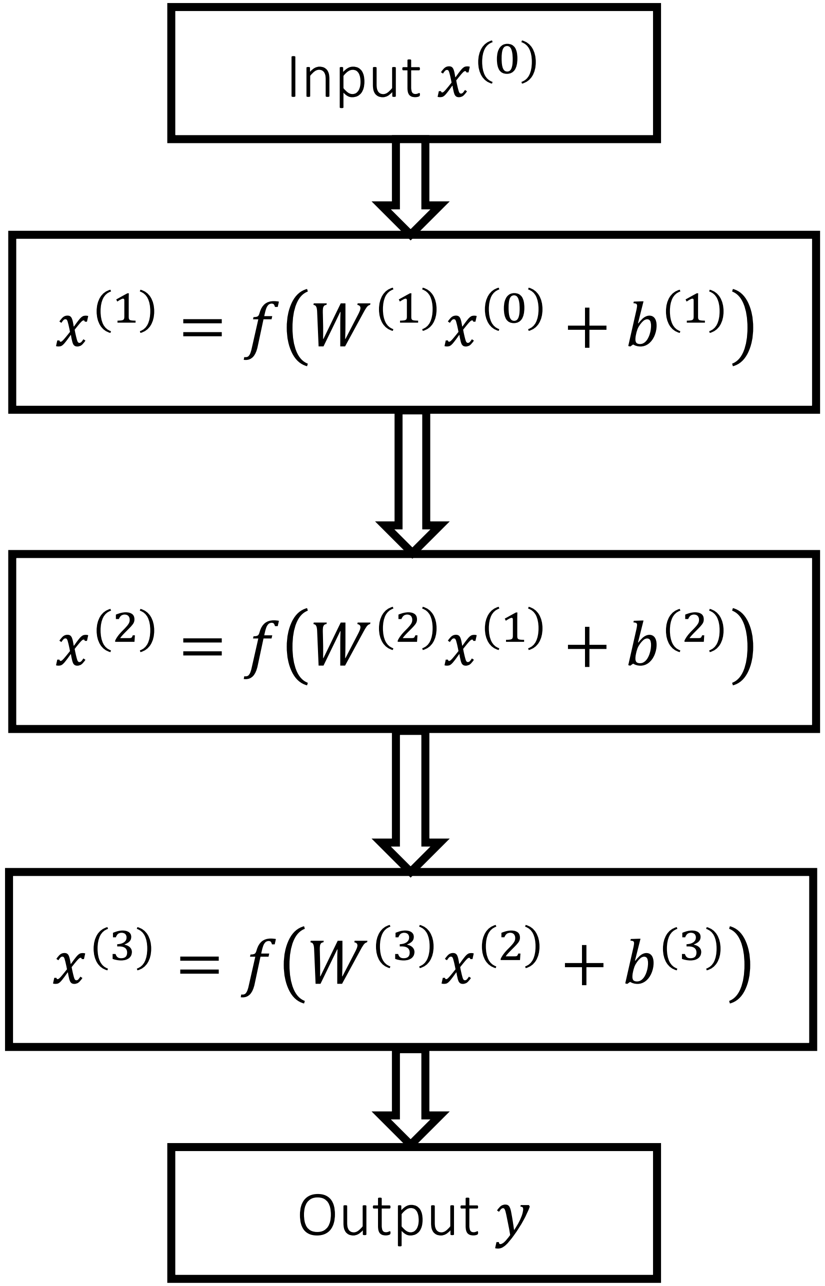
Neural Network

We have a prediction y, and reality ("ground truth") y'.

Performance of the neural net is given by a loss function: $\mathcal{L}(y, y')$

<u>Optimisation problem:</u>

Find all $W^{(n)}$ and $b^{(n)}$ that minimise $\mathcal{L}(y, y')$



Loss Functions

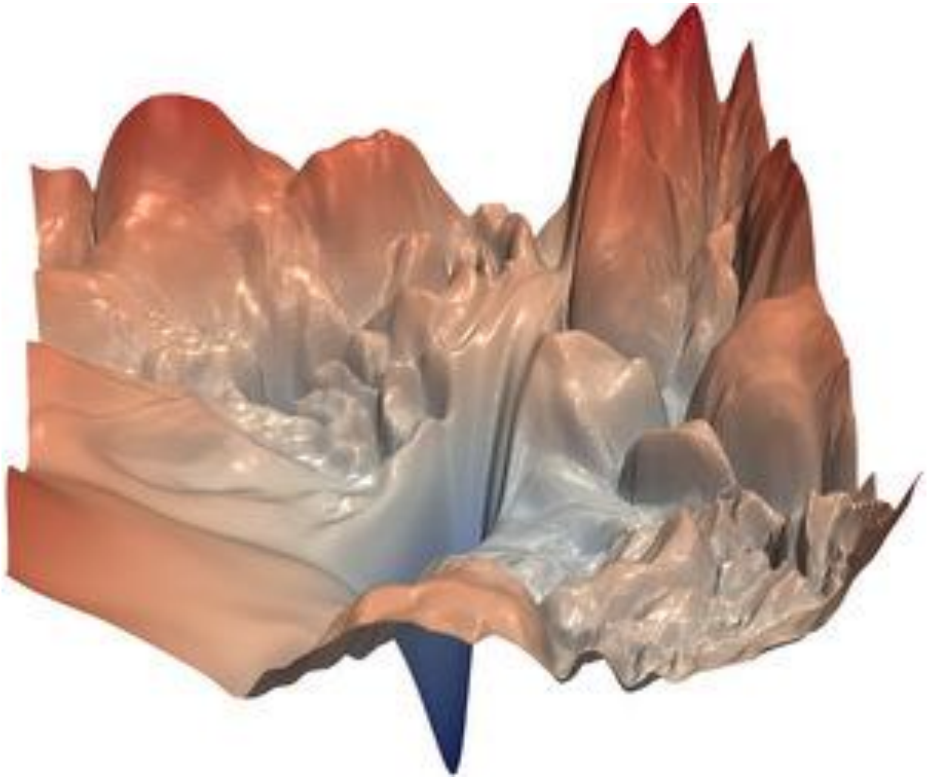
Mean-squared error:

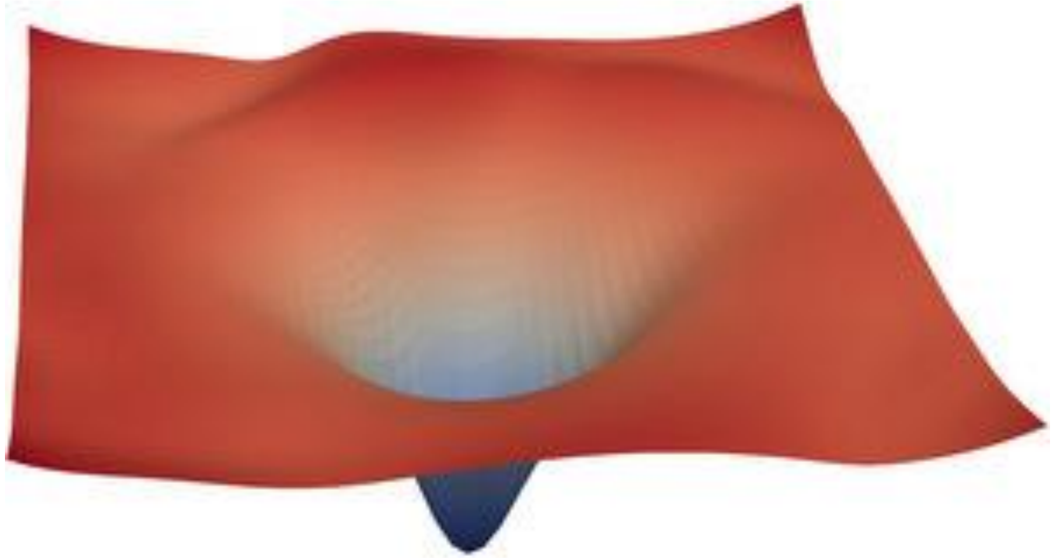
$$\mathcal{L}(y, y') = \frac{1}{N} \sum_{i=1}^{N} (y_i - y_i')^2$$

Binary cross-entropy:

$$\mathcal{L}(y, y') = -\sum_{i=1}^{N} y_i' \log y_i + (1 - y_i') \log(1 - y_i)$$

Loss Landscape





Li et al. (2018), *Visualizing the loss landscape of neural nets*

Training = Optimisation

- Create a training dataset with input x and ground truth y'
  - Submarine example: $x = [T, P, \mathrm{pH}, \dots]$, $y' = [\mathrm{lat}, \mathrm{lon}, \mathrm{depth}]$
- Feed x into the neural network, get prediction y
- Calculate loss $\mathcal{L}(y, y')$
- Update all weights $W^{(n)}$ and biases $b^{(n)}$ to decrease $\mathcal{L}$
- Repeat

Model Performance (classification)

Accuracy: how many predictions are correct?

$$A = \frac{\text{correct predictions}}{\text{total predictions}} = \frac{\text{true positives} + \text{true negatives}}{\text{total predictions}}$$

Model Performance (classification)

Precision: how reliable are the predictions?

$$P = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How often is the model fooled?

Model Performance (classification)

Recall: how complete are the predictions?

$$R = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$

How many things did
the model miss?

Performance Cat Detection

Dataset: 998 dogs, 2 cats

Prediction: 999 dogs, 1 cat

Accuracy: (1 + 998) / 1000 = 99.9% ← most of the predictions were correct

Precision: 1 / 1 = 100% ← all of the cat detections were correct

Recall: 1 / (1 + 1) = 50% ← 50% of the cats were not detected!

Other Metrics

- Area Under Curve (AUC): balance between recall and precision
- F1-score: “ordered” classes
- Mean Absolute Error (MAE)
- Mean Squared Error (MSE)
- Intersection Over Union (IOU)

Getting Your Hands Dirty

- Open terminal
- Navigate to the ML tutorials directory
- Command: `conda activate ICTP_EQ`
- Command: `jupyter notebook`
- Browser should open a new tab with notebooks