Deep learning overview, representation learning methods in detail (sammons map, t-sne), the backprop algorithm in detail, and regularization and its impact on optimization.

- (30min) What is deep learning overview (Slides)
    - Define supervised and self-supervised prob perspective
    - How to approach problems (use sklearn)
    - Examples of go-to methods: logistic regression, decision tree etc (use sklearn)
- (45min) Backprop in more detail (Slides)
    - Work through an example of manually performing the algorithm
    - Backpropagation (visualizing the chain rule)
    - Intuition for applying gradient updates for arbitrary functions
- break
- (1hr) Representation learning (Slides)
    - Non-linear dim reduction
    - word2vec
    - Sammons map (tutorial code)
    - t-SNE
    - Regularization

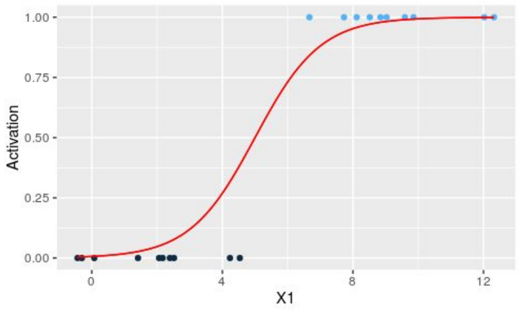
Intro to Deep Learning

3 lectures
Ask questions

What is Deep Learning?

Logistic Regression

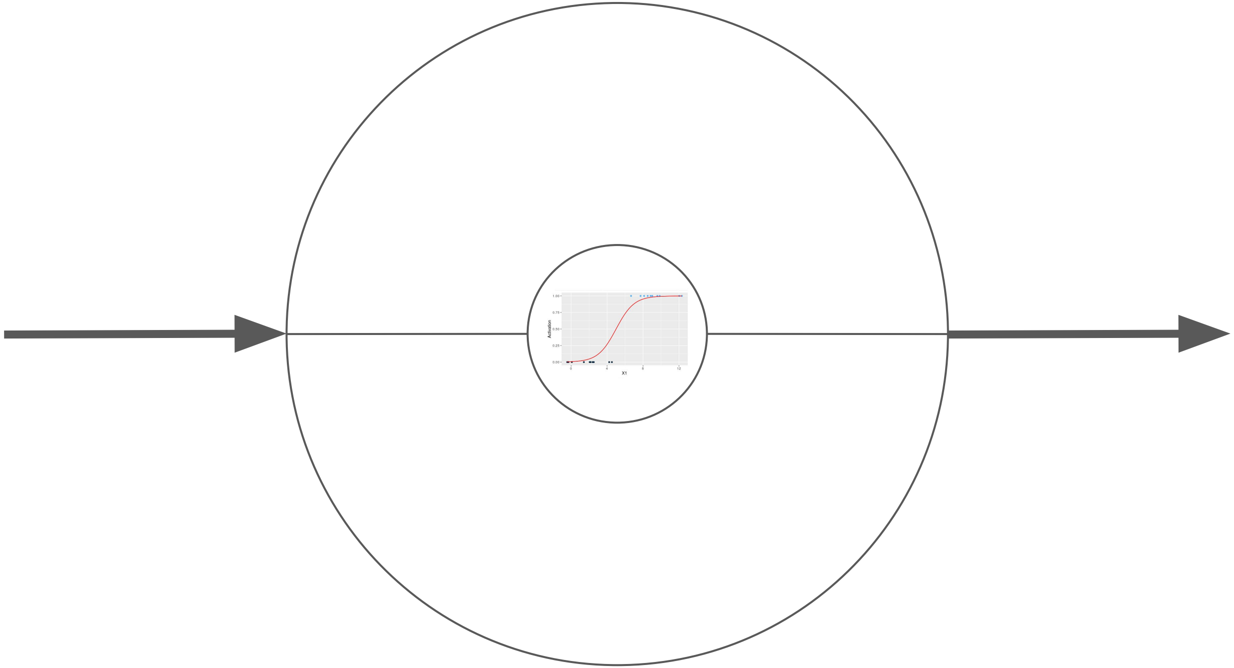
[200, 45, -4, 80]



[1, 1, 0, 1]

What is Deep Learning?

Logistic Regression



What is Deep Learning?

Logistic Regressions

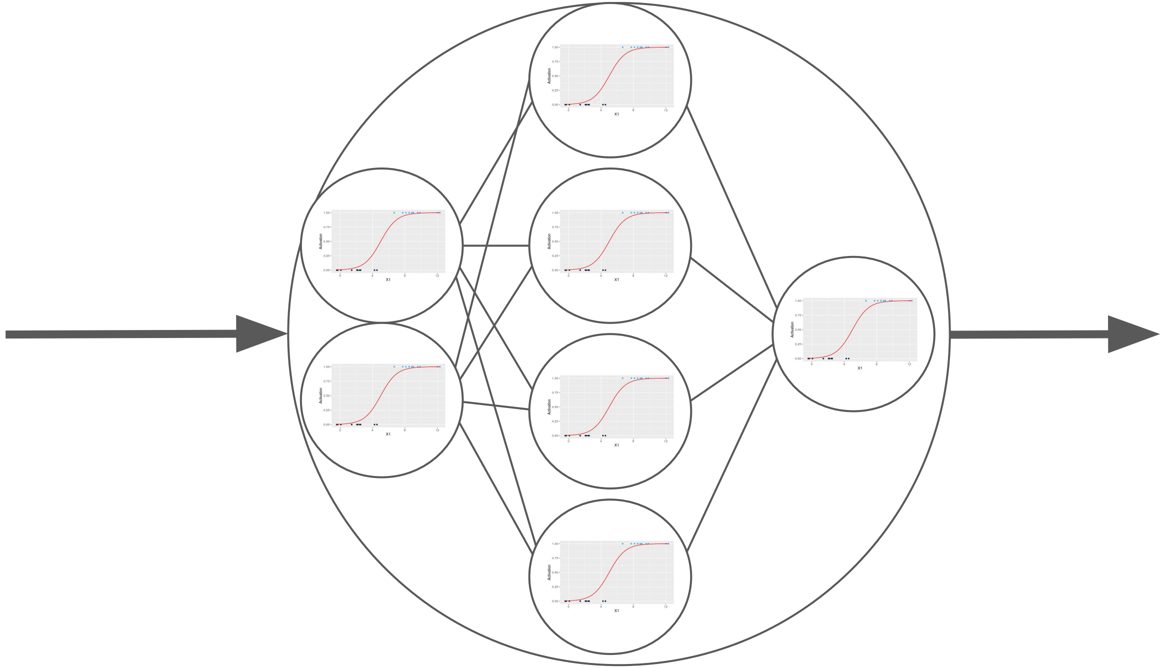


Image credit Pascal Vincent

What is Deep Learning?

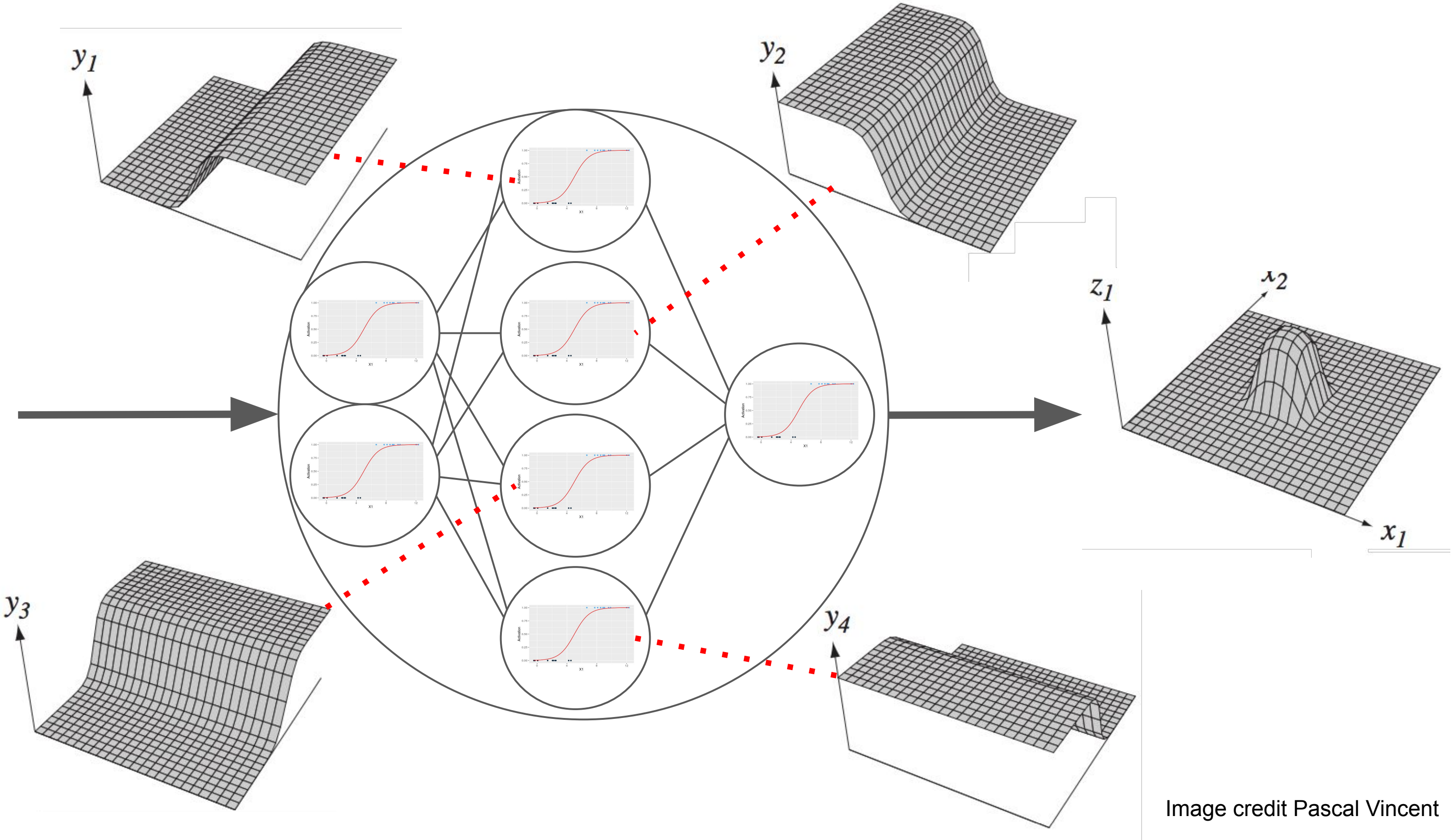
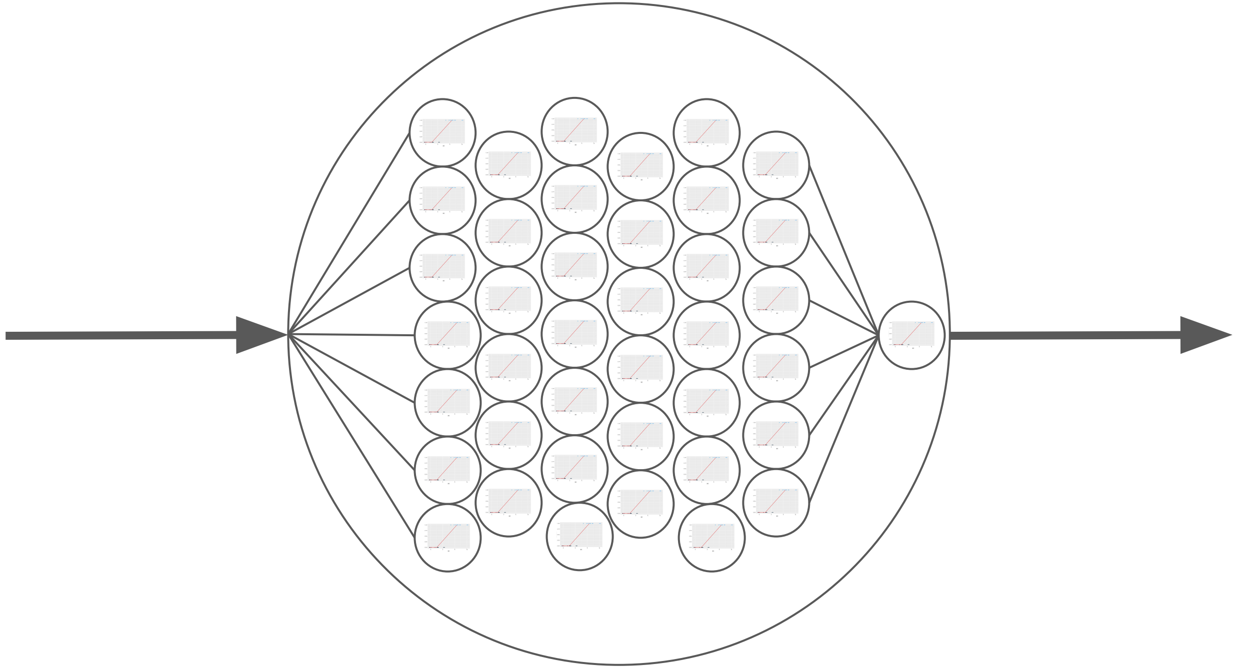


Image credit Pascal Vincent

What is Deep Learning?

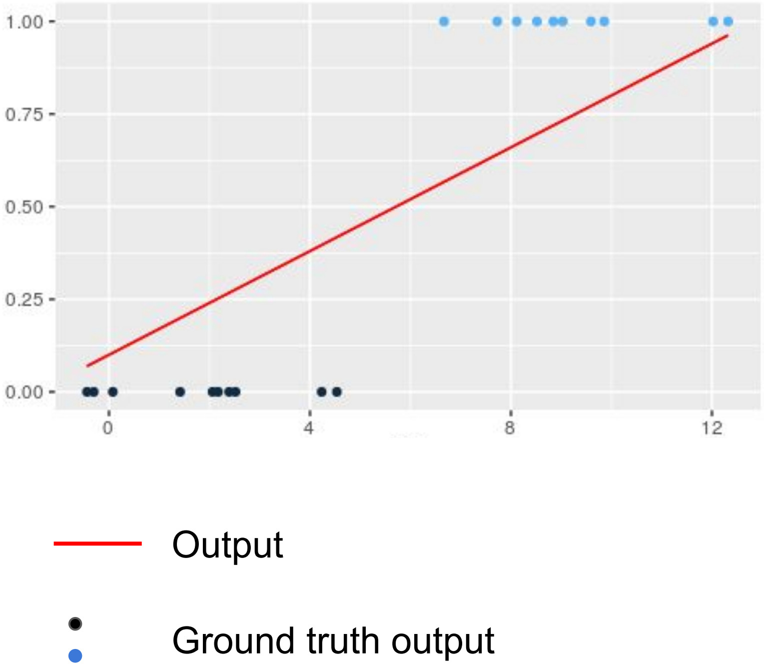
Logistic Regressions



Neural Networks: Architectures

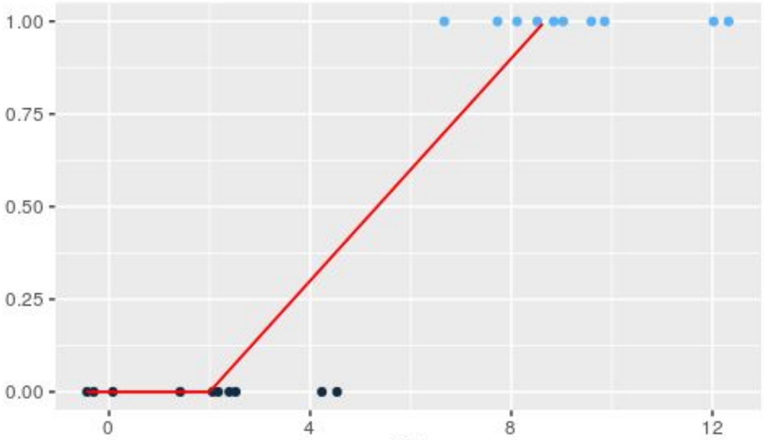
$$\sum_i w_i x_i + b$$

$$\sum_i w_i x_i + b$$



$$f\left(\sum_i w_i x_i + b\right)$$

$$f(x) = max(0, x)$$

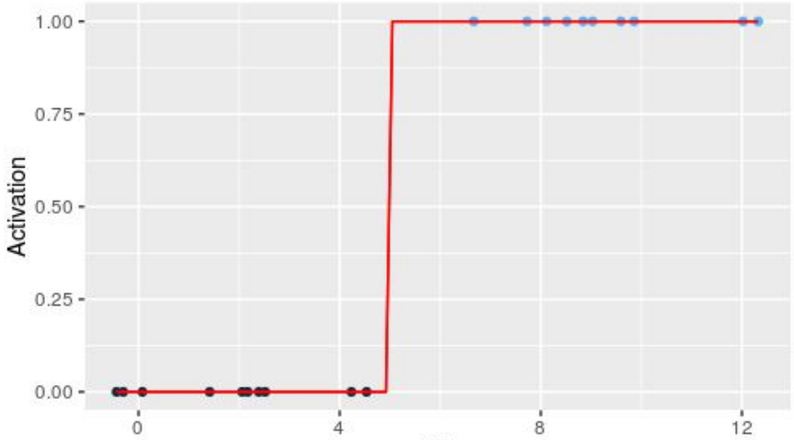


Output

Ground truth output

$$f\left(\sum_i w_i x_i + b\right)$$

$$f(x) = \begin{cases} 1 & \text{if } (x > 0) \\ 0 & otherwise \end{cases}$$

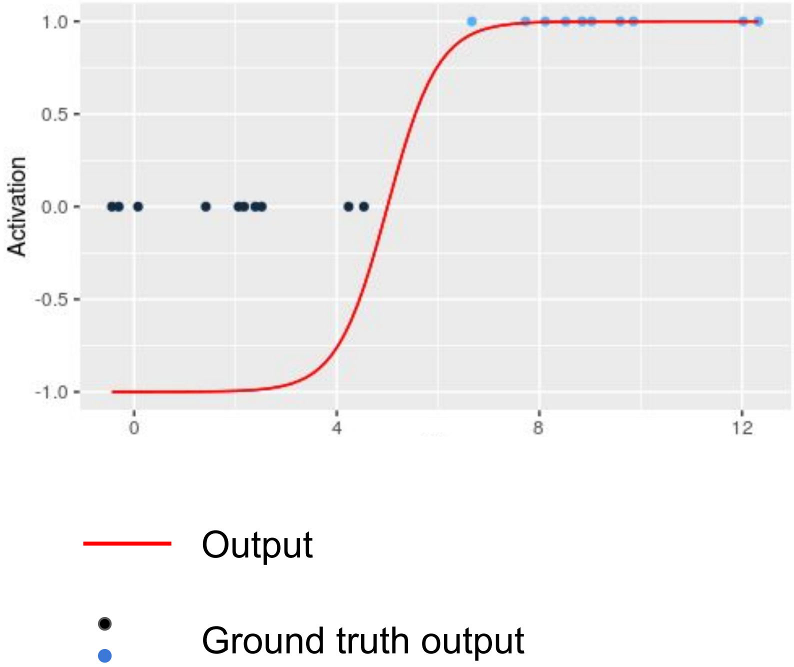


Output

Ground truth output

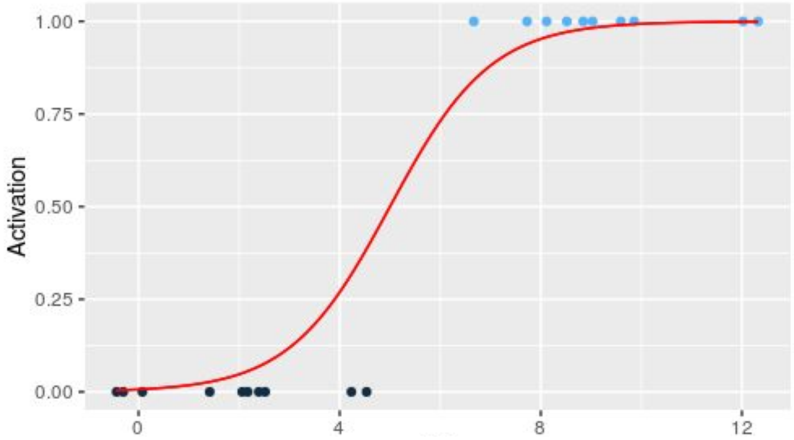
$$f\left(\sum_i w_i x_i + b\right)$$

$$f(x) = tanh(x)$$



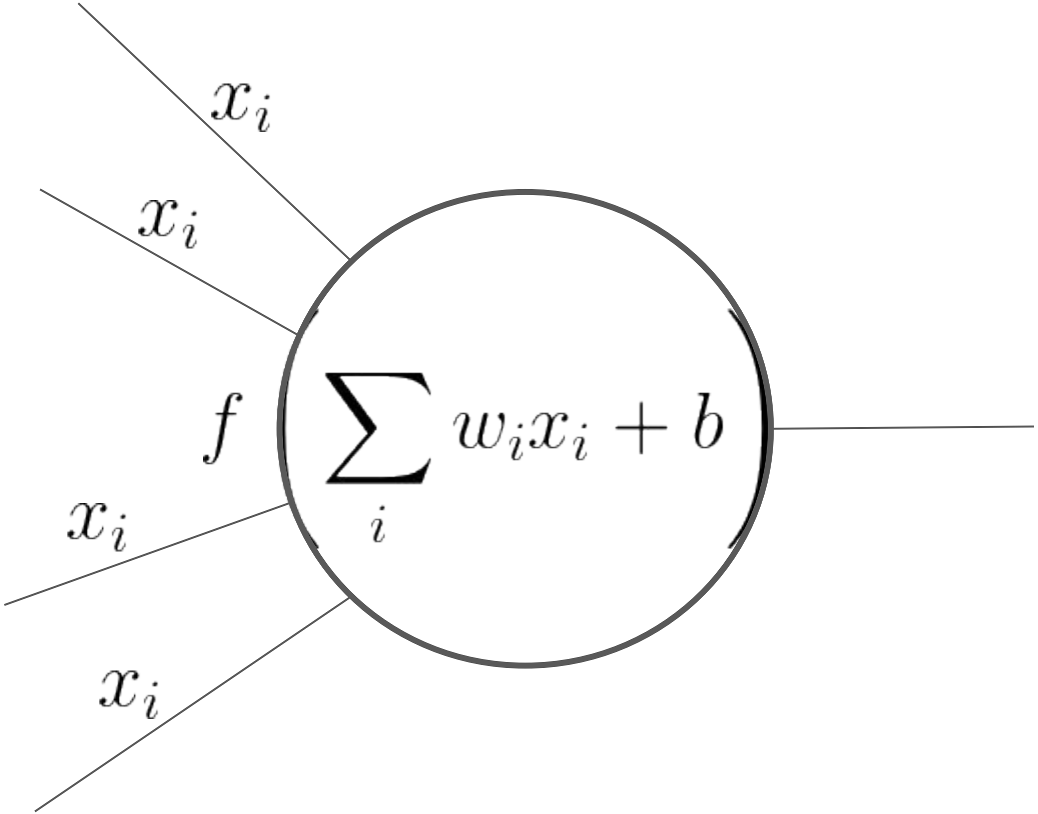
$$f\left(\sum_i w_i x_i + b\right)$$

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

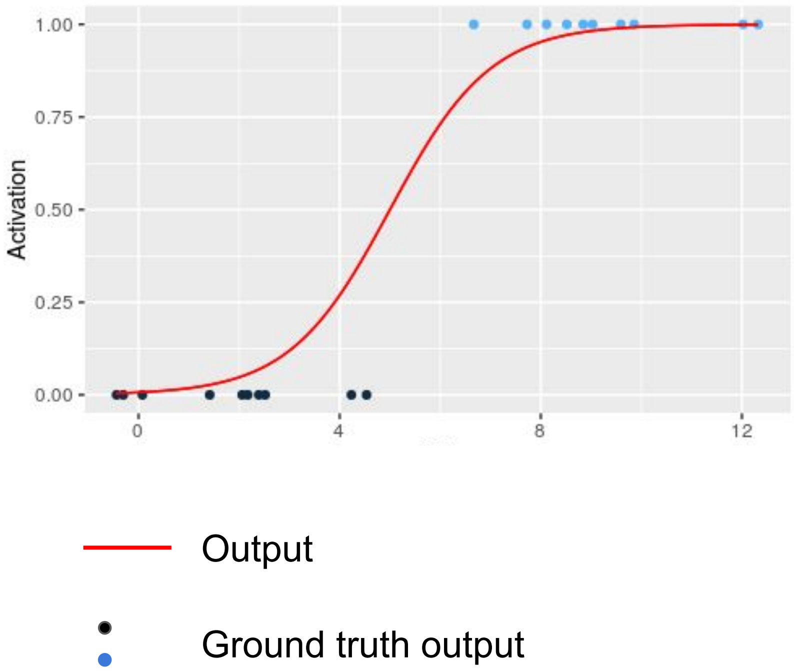


Output

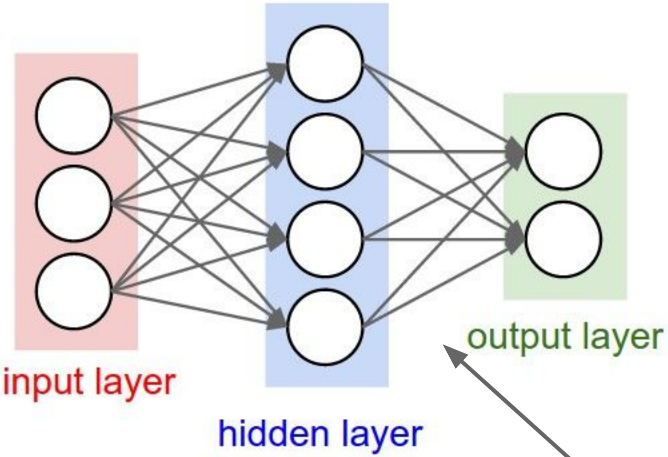
Ground truth output

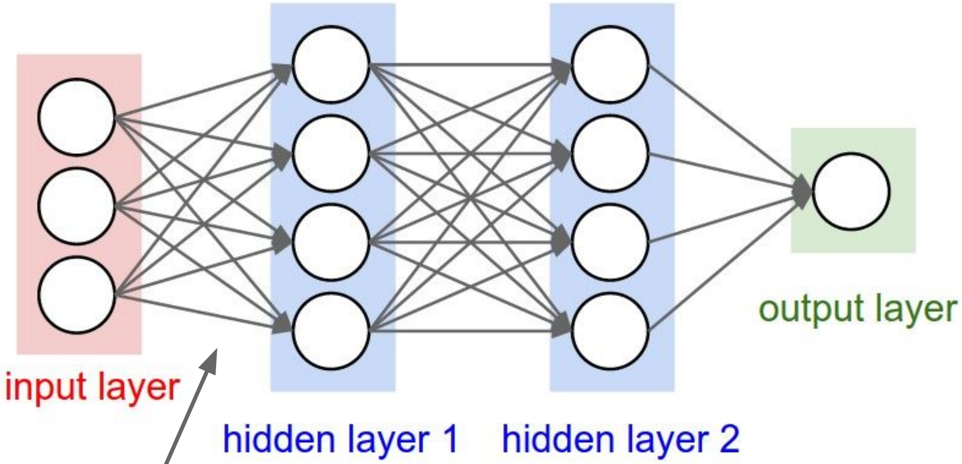

x_i
x_i
$f\left(\sum_i w_i x_i + b\right)$
x_i
x_i

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


1.00
0.75
0.50
0.25
0.00
Activation
0
4
8
12
Output
Ground truth output

Neural Networks: Architectures





"2-layer Neural Net", or
"1-hidden-layer Neural Net"

"3-layer Neural Net", or
"2-hidden-layer Neural Net"

"Fully-connected" layers

Multi-class classification

Approach: at the end of a network output a vector with some number of units as classes.

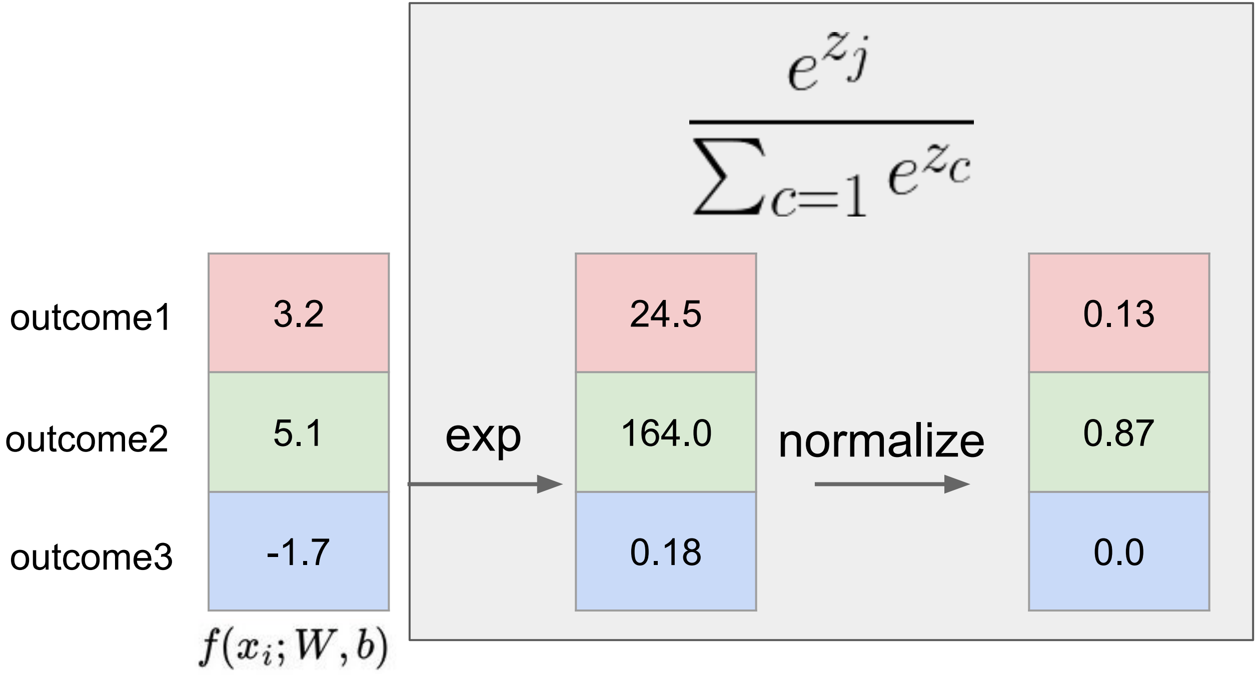
Normalize these outputs to be a probability distribution.

One way is to use a softmax function.

$$\frac{e^{z_j}}{\sum_{c=1} e^{z_c}}$$

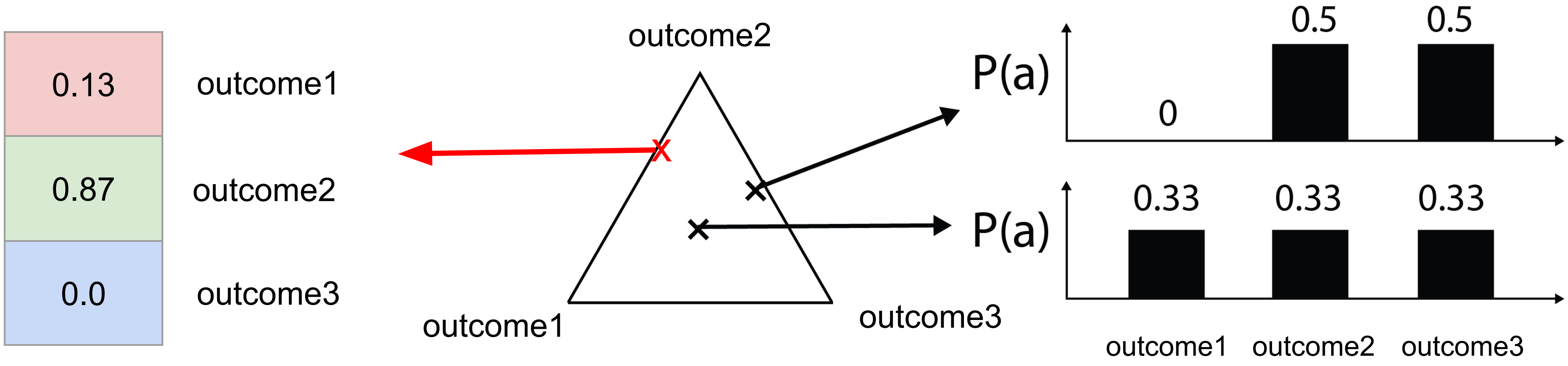
Softmax function

A note on the softmax function



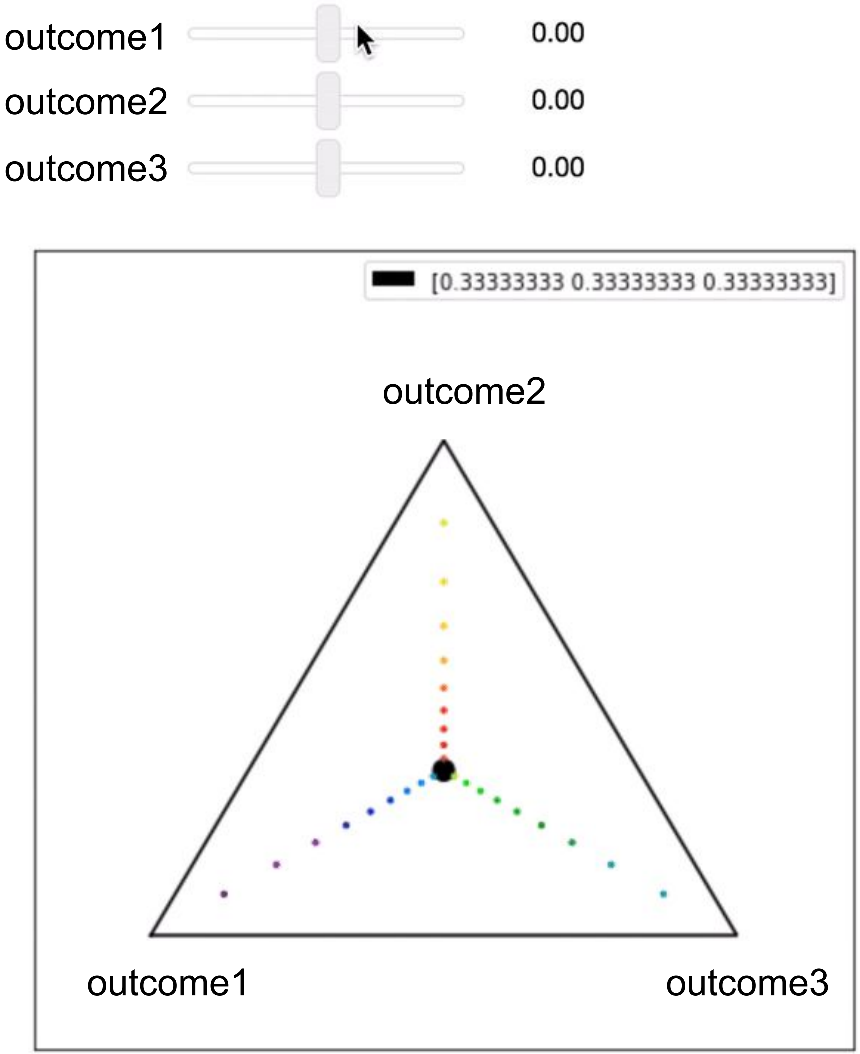
To predict multiple classes we project to a probability distribution

Simplex



Because it is on a simplex; the correction of one term impacts all

Image credits: http://gureckislab.org/



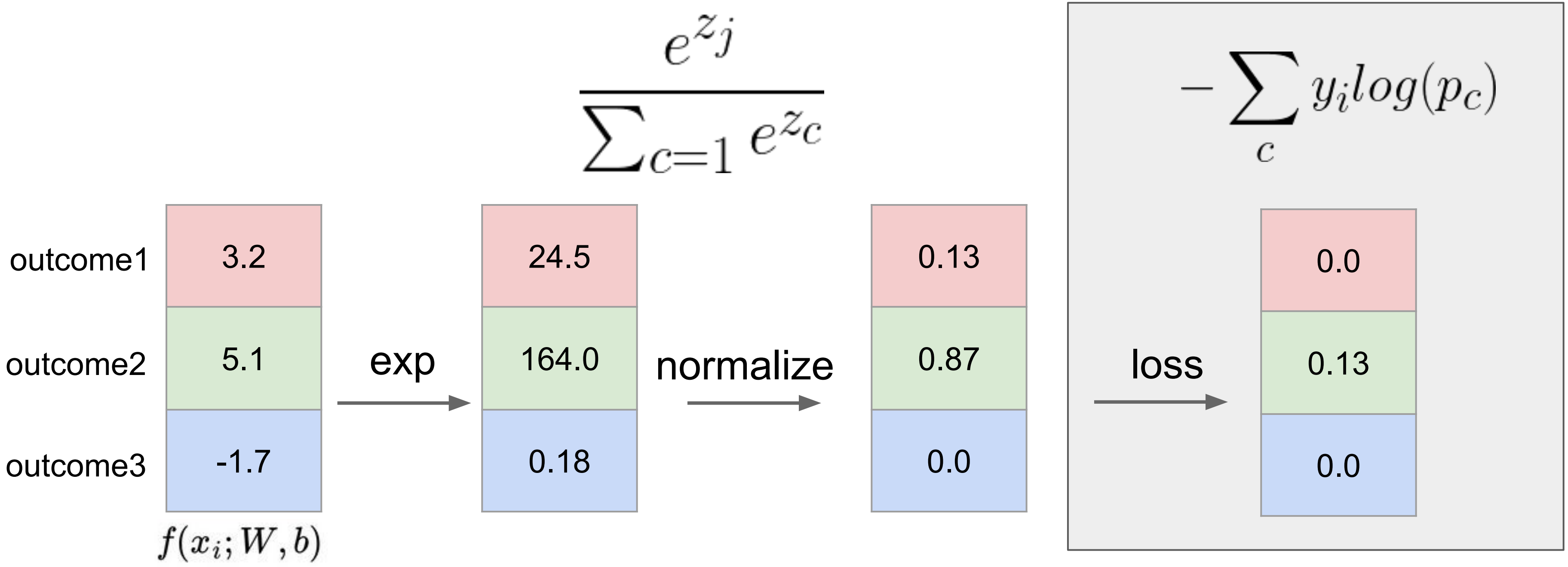
Infinite ways to generate the same output.

A correction of one sends gradients to others

We can learn unseen classes through a process of elimination.

https://github.com/ieee8023/NeuralNetwork-Examples/blob/master/general/simplex-softmax.ipynb

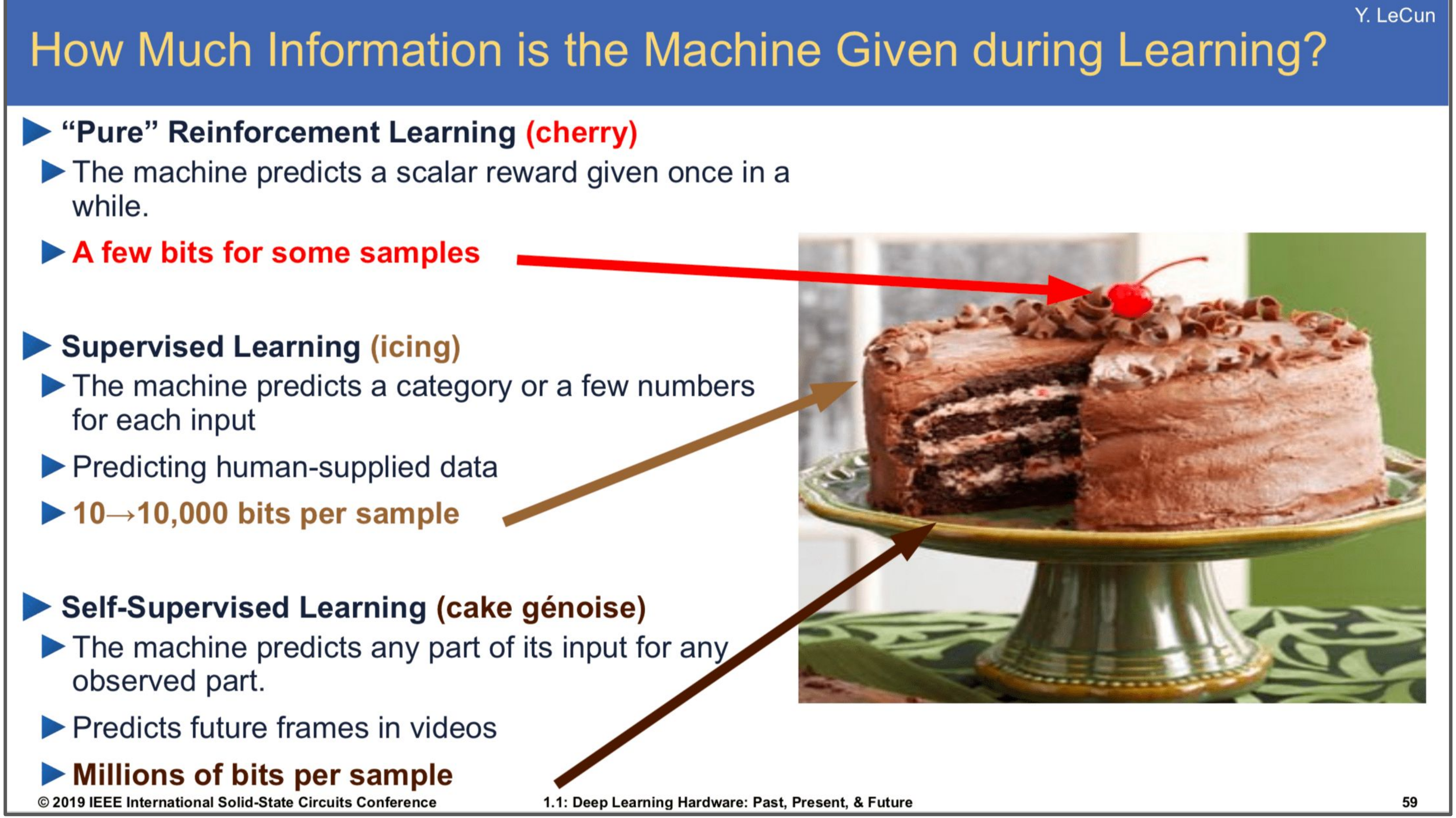
Softmax and Cross-entropy loss



To predict multiple class we can project the output onto a simplex and compute the loss there.

Types of learning

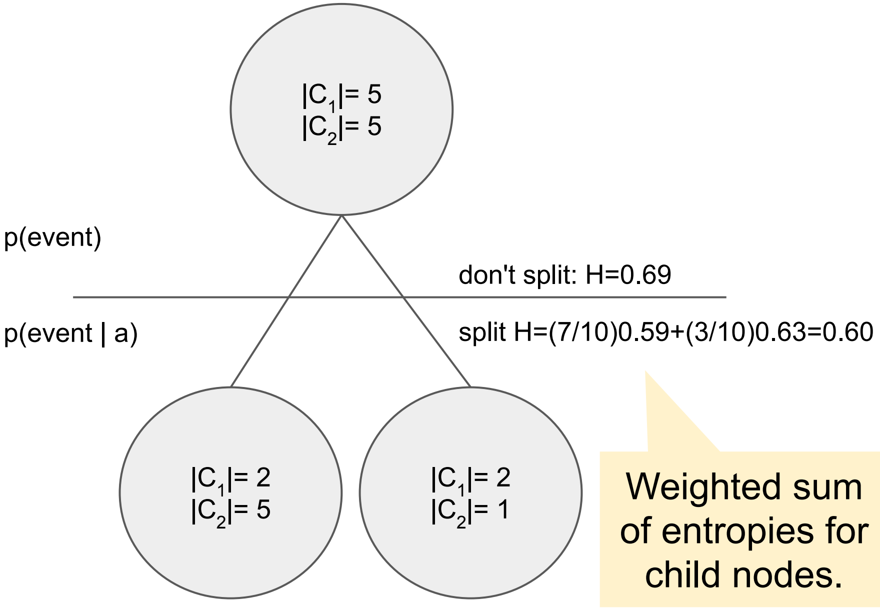
Supervised Clear training signal related to goal e.g. classification, regression
Self-supervised/unsupervised Using the data itself as a training signal e.g. clustering, autoencoders



Yann Lecun's cake slide

A quick intro to decision trees

One approach to build them using "information gain" where a node is split into two if the entropy reduction is the most.



$$IG(Y, A) = H(Y) - H(Y|A)$$

$$H(Y) = -\sum_{y \in Y} p(Y = y) \log p(Y = y)$$

$$H(Y|A) = \sum_{a \in A} p(A = a) H(Y|A = a)$$

Sklearn Code Example: https://colab.research.google.com/drive/1IrWtY0g3tP1w14PEW2egze6mJ9T5dnk8

Homework

Create an example split which should not be split given the IG criteria.

Can something be supervised and unsupervised? Think of examples.