Introduction to Machine Learning

-

How to design machine learning tools for Optics

Workshop

May 6th 2024

Louis Rossignol

LKB

COLLÈGE DE FRANCE 1530

SCIENCES SORBONNE UNIVERSITÉ

ENS Département de Physique École normale supérieure

cnrs

Reference and good ressource

- 3blue1brown on Neural Networks:
- https://youtube.com/playlist?list=PLZHQObOWTQDNU6R1_67000Dx_ZCJB-3pi&si=xvbZnuQsteonsFpn

Outline

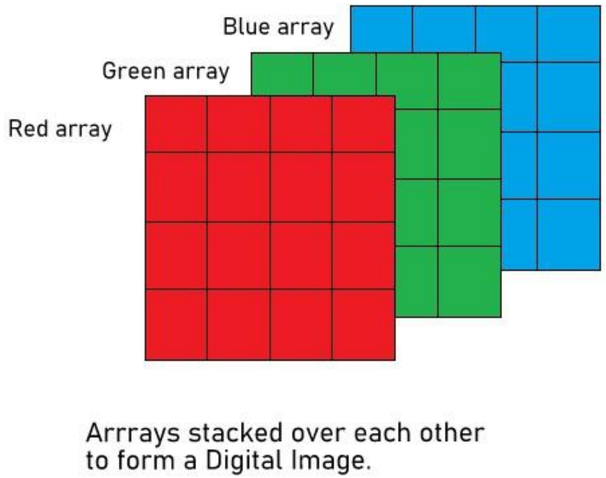
- Introduction
- Machine learning mechanics
  - Back-propagation of Error and Gradient Descent
  - Activation function
  - Loss function
    - Cross entropy loss and Classification
    - Mean Squared Error and Regression
  - Convolutional Neural Networks

Introduction

- Data
- Model
- Loss function

Introduction
-
Data

Image [1]



Pig

Dog

Cat

Sound [2]

Classification

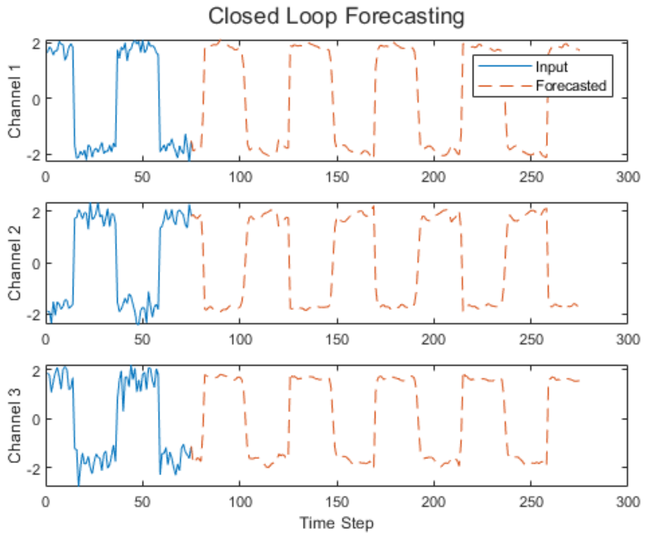
[1] - https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.geeksforgeeks.org%2Fmatlab-rgb-image-representation%2F&psig=AOvVaw3nGrGp2TvHzoNbSHI8GIK8&ust=1714476975492000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCNDEicqq54UDFQAAAAAdAAAAABAE

[2] - https://www.google.com/url?sa=i&url=https%3A%2F%2Fpngtree.com%2Fso%2Fblack-and-white-sound-waves&psig=AOvVaw3ZKMWpx1T6W2Yw4UVqrS33&ust=1714476738077000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCID-oNmp54UDFQAAAAAdAAAAABAK

Introduction
-
Data

Image [1]

Sound [2]



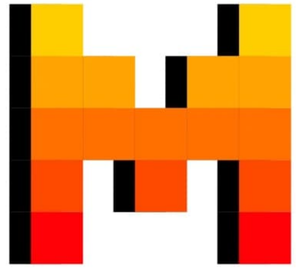
Forecasting

[1] - https://www.google.com/url?sa=i&url=https%3A%2F%2Fstock.adobe.com%2Ffr%2Fsearch%2Fimages%3Fk%3Dsequence&psig=AOvVaw3FXo8WKCjXAvoFTo8OsmGb&ust=1714477066240000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCMDZ4fWq54UDFQAAAAAdAAAAABAE

[2] - https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.mathworks.com%2Fhelp%2Fdeeplearning%2Fug%2Ftime-series-forecasting-using-deep-learning.html&psig=AOvVaw35v6rvPsQNiSFWjHxQY4-i&ust=1714477261772000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCNCR9tKr54UDFQAAAAAdAAAAABAJ

Introduction
-
Data

Text



Text prediction

[1] - https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.iemoji.com%2Fview%2Femoji%2F1068%2Fproposed%2Fpages&psig=AOvVaw2xFcqlKl-W9YG-DS6hZKb5&ust=1714479822672000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCKiygZi154UDFQAAAAAdAAAAABAE

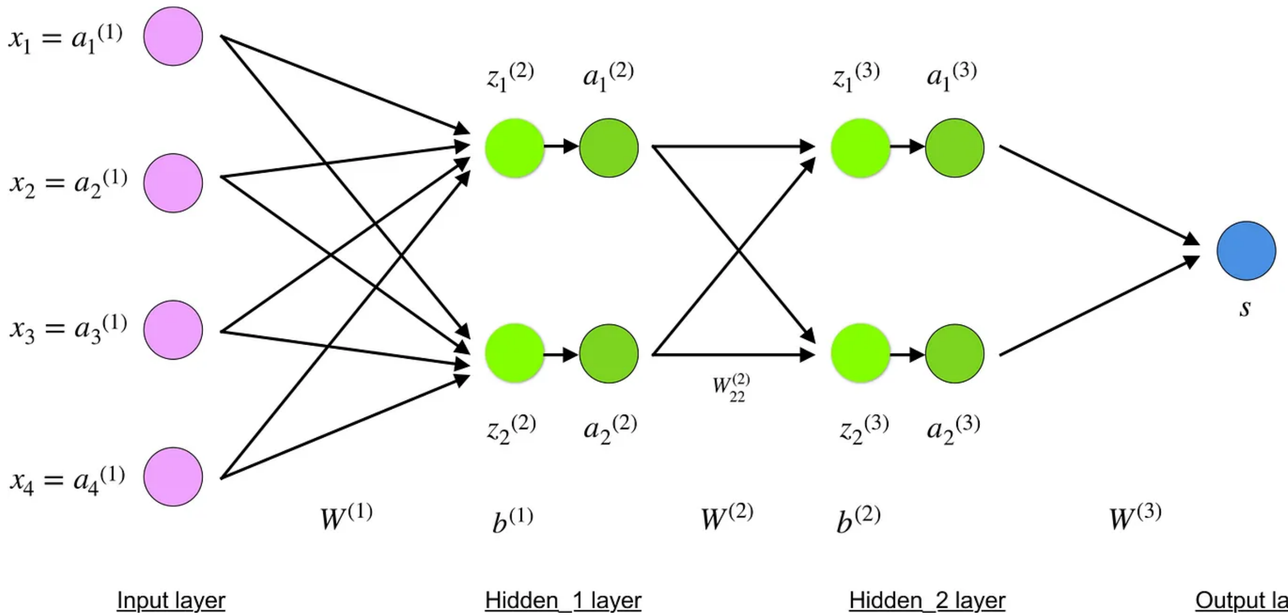
[2] - https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.geeky-gadgets.com%2Fchat-gpt-login%2F&psig=AOvVaw0zJUSVrrfJX2mgnomf1Yvv&ust=1714479965542000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCPDb5aS254UDFQAAAAAdAAAAABAE
https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.jaimelesstartups.fr%2Fmistral-ai-ia-generative-llm%2F&psig=AOvVaw3HoeZ6G6buYd5ZIUq6MDgK&ust=1714479988170000&source=images&cd=vfe&opi=89978449&ved=0CBIQjRxqFwoTCNCNzqO254UDFQAAAAAdAAAAABAK
https://plugins.jetbrains.com/plugin/17718-github-copilot
https://www.google.com/url?sa=i&url=https%3A%2F%2Fca.linkedin.com%2Fcompany%2Fblackboxtech&psig=AOvVaw363p8kpYRiUvRuvnxwoGb9&ust=1714480207899000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCNiZoM-254UDFQAAAAAdAAAAABAE

Important

- Generalization
- Data formatting
- **Model**

Introduction
-
Model

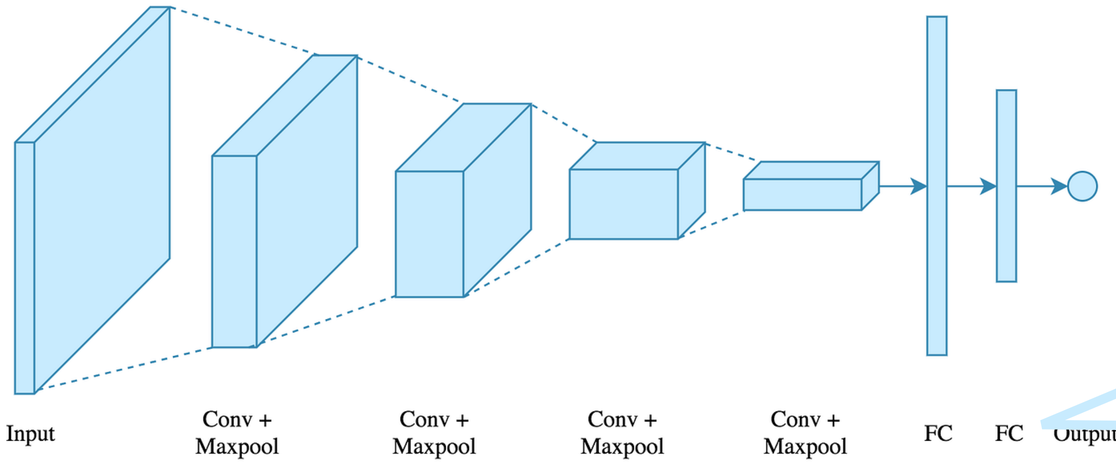
Fully connected layer [1]



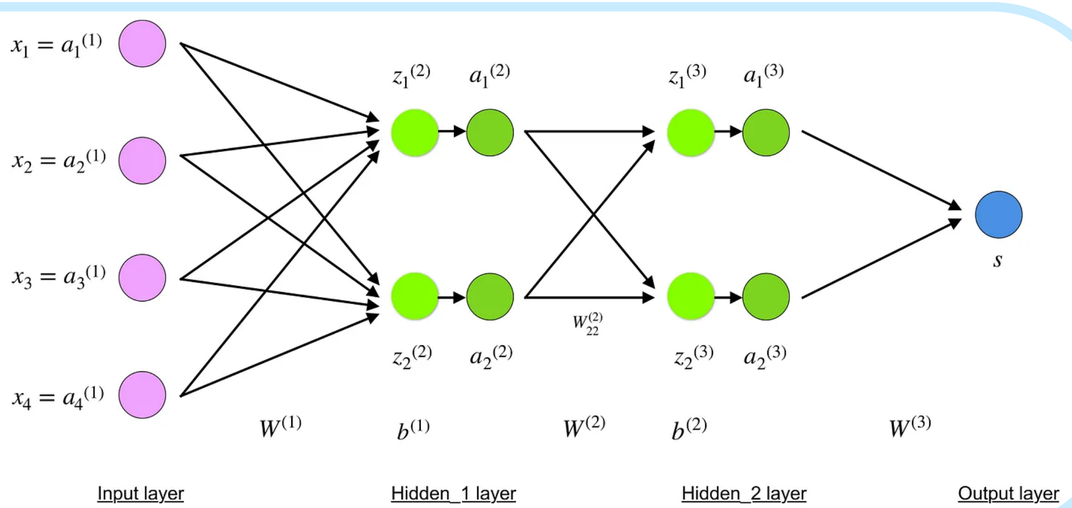
[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

Introduction
-
Model

Convolutional Neural Network [1]



Fully connected layer [2]



[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-Wo1WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

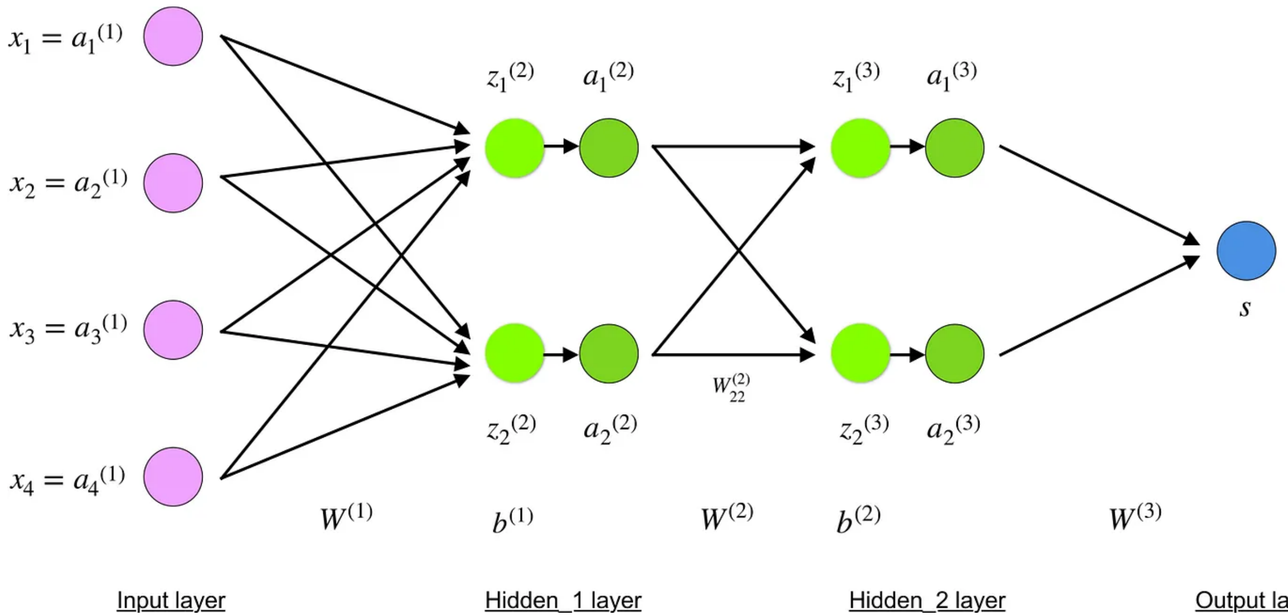
[2] - https://towardsdatascience.com/understanding-backpropagation-algorithm-7bb3aa2f95fd

Important

- The proper model for data
- Mindful of the memory and computing power
- **Loss function**

Introduction
-
Model

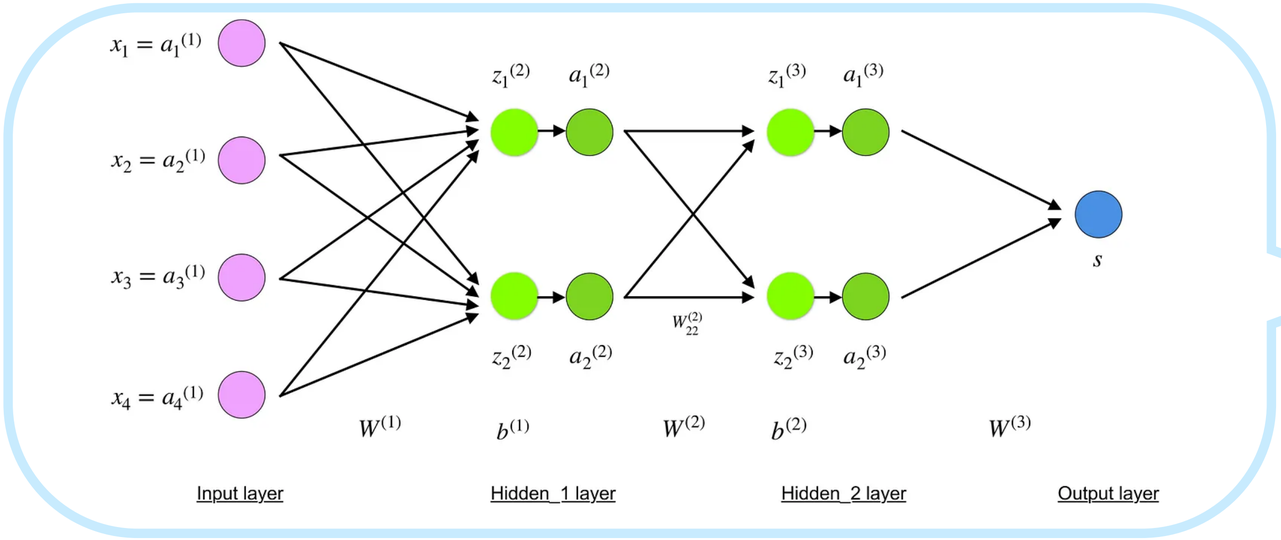
Fully connected layer [1]



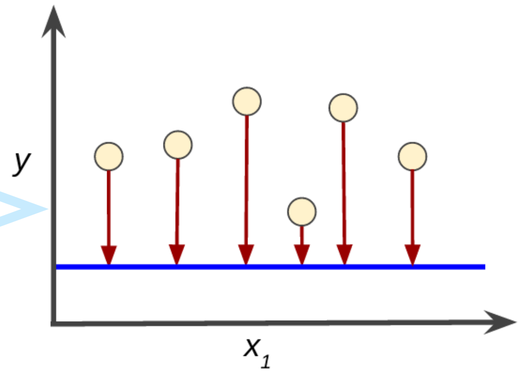
[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

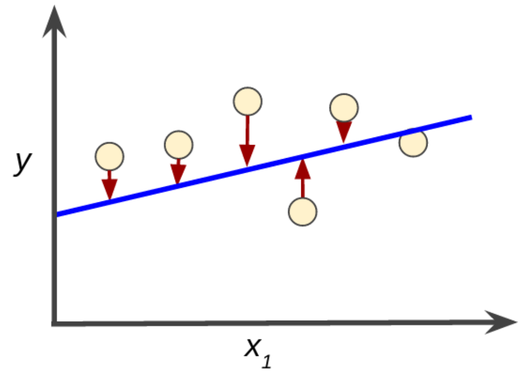
Introduction
-
Loss function

Fully connected layer [1]



Loss Function [1]





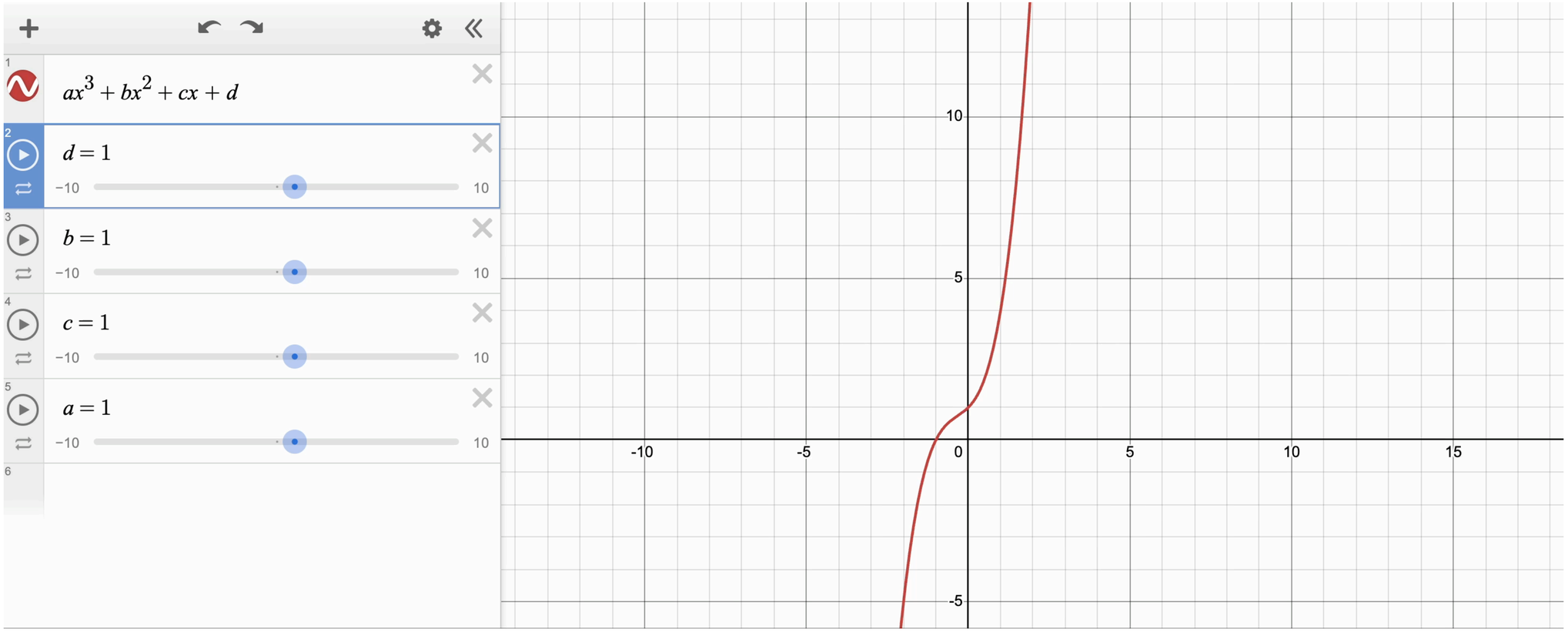
[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40omayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

[2] - https://developers.google.com/machine-learning/crash-course/descending-into-ml/training-and-loss

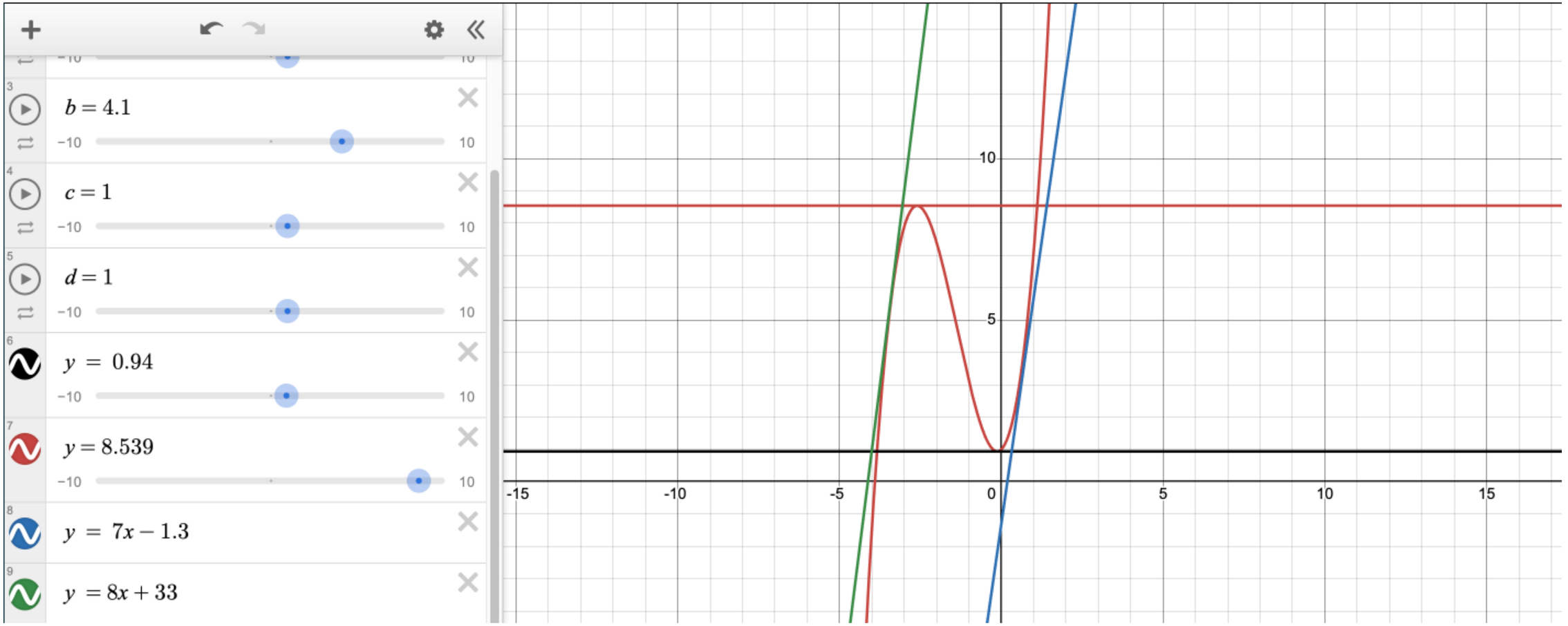
Machine learning mechanics

- Back-propagation of Error and Gradient Descent
- Activation function
- Convolutional Neural Networks
- Loss function

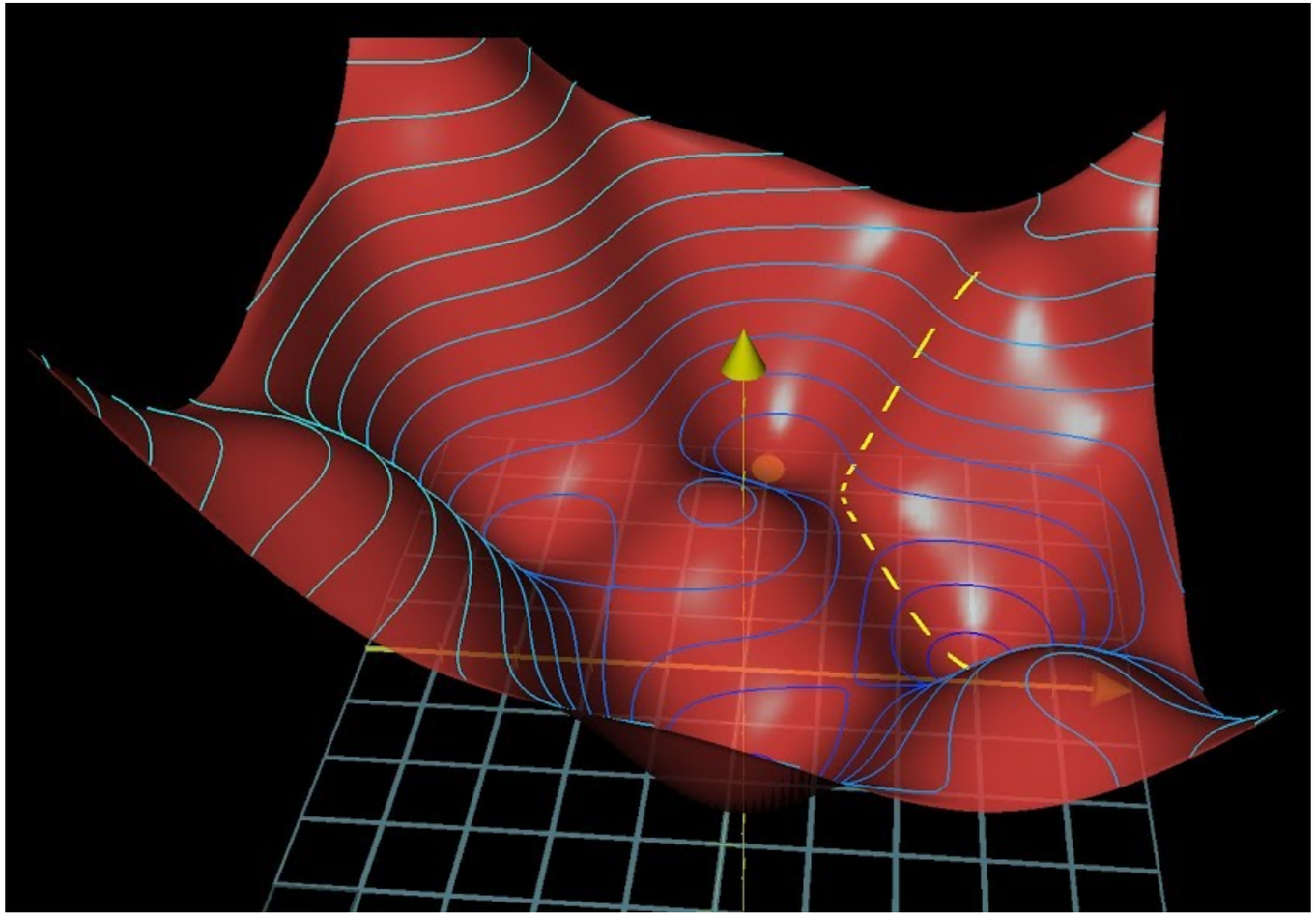
Machine learning mechanics
-
Back-propagation of Error and Gradient Descent



Machine learning mechanics
-
Back-propagation of Error and Gradient Descent



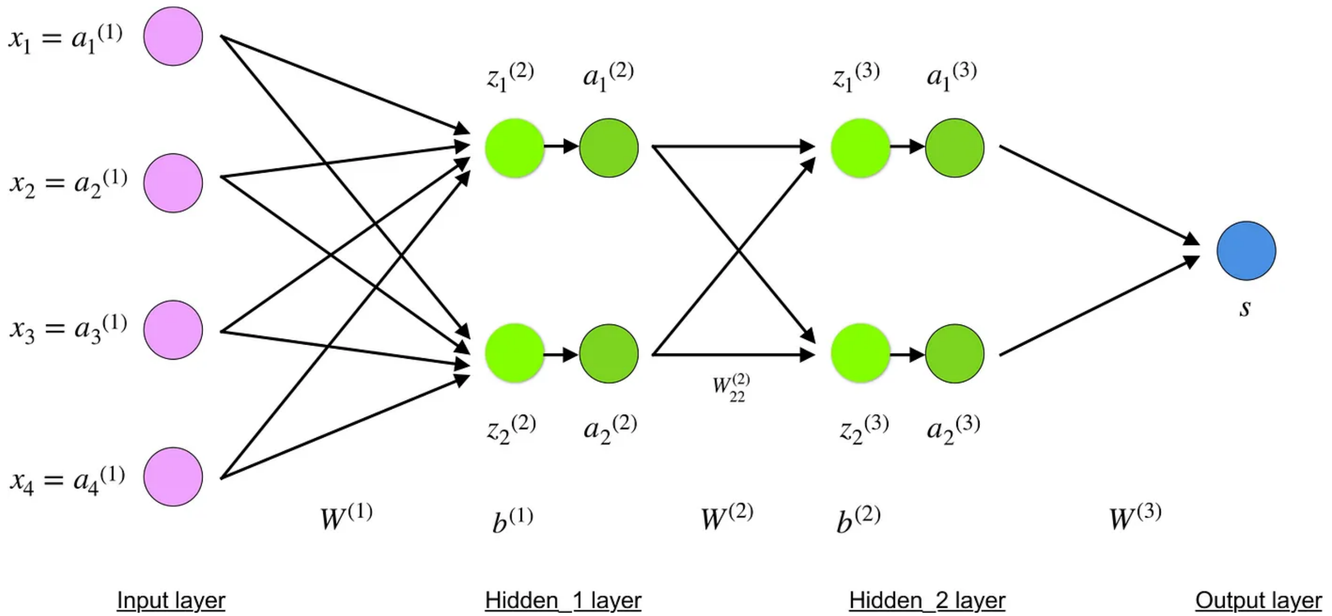
Machine learning mechanics
-
Gradient Descent



https://www.google.com/url?sa=i&url=https%3A%2F%2Fm.youtube.com%2Fwatch%3Fv%3DIHZwWFHWa-w&psig=AOvVaw0Cn41Px9Jc3_dY_qNyZEfg&ust=1714573793103000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCND01qCT60UDFQAAAAAdAAAAABAE

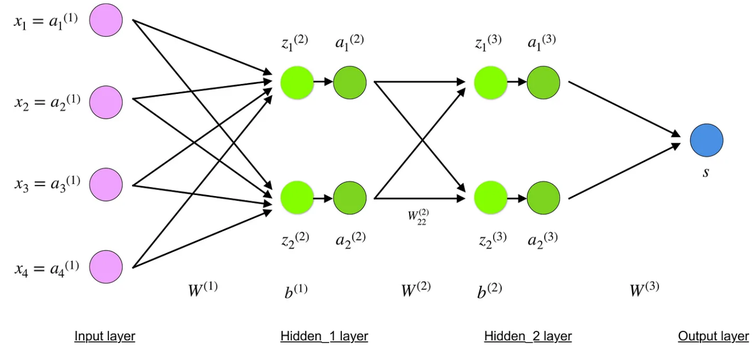
Machine learning mechanics
-
Back-propagation of Error and Gradient Descent

Fully connected layer [1]



[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.researchgate.net%2Ffigure%2FMulti-Layer-Perceptron-MLP-diagram-with-four-hidden-layers-and-a-collection-of-single_fig1_334609713&psig=AOvVaw29hcG3fO4ZfJHPdcwvvhqg&ust=1714481773623000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCLiQqLq854UDFQAAAAAdAAAAABAE

Machine learning mechanics
-
Back-propagation of Error and Gradient Descent

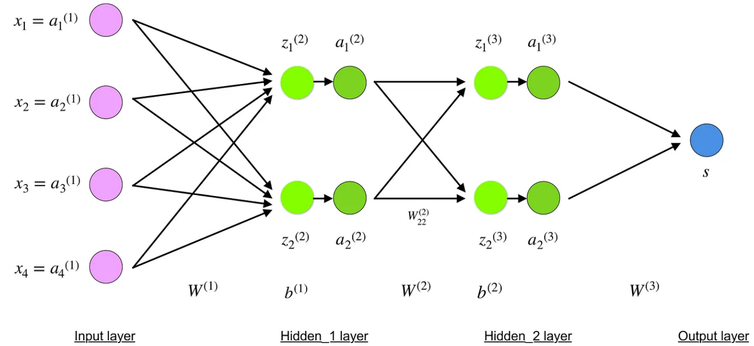


$$W^{(1)} = \begin{bmatrix} W_{11}^{(1)} & W_{12}^{(1)} & W_{13}^{(1)} & W_{14}^{(1)} \\ W_{21}^{(1)} & W_{22}^{(1)} & W_{23}^{(1)} & W_{24}^{(1)} \end{bmatrix} \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad b^{(1)} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(1)} \end{bmatrix}$$

$$z^{(2)} = \begin{bmatrix} W_{11}^{(1)}x_1 + W_{12}^{(1)}x_2 + W_{13}^{(1)}x_3 + W_{14}^{(1)}x_4 \\ W_{21}^{(1)}x_1 + W_{22}^{(1)}x_2 + W_{23}^{(1)}x_3 + W_{24}^{(1)}x_4 \end{bmatrix} + \begin{bmatrix} b_1^{(1)} \\ b_2^{(1)} \end{bmatrix}$$

[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

Machine learning mechanics
-
Forward Propagation



$$x = a^{(1)} \qquad Input\ layer$$

$$z^{(2)} = W^{(1)}x + b^{(1)} \qquad neuron\ value\ at\ Hidden_1\ layer$$

$$a^{(2)} = f(z^{(2)}) \qquad activation\ value\ at\ Hidden_1\ layer$$

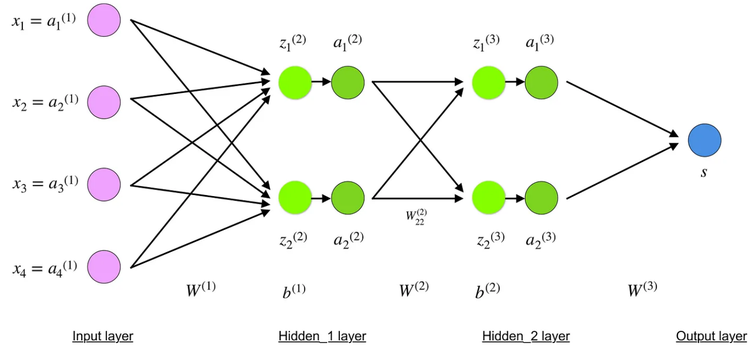
$$z^{(3)} = W^{(2)}a^{(2)} + b^{(2)} \qquad neuron\ value\ at\ Hidden_2\ layer$$

$$a^{(3)} = f(z^{(3)}) \qquad activation\ value\ at\ Hidden_2\ layer$$

$$s = W^{(3)}a^{(3)} \qquad Output\ layer \qquad ==> \qquad C = cost(s, y)$$

[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

Machine learning mechanics
-
Back Propagation



$$z_j^l = \sum_{k=1}^{m} w_{jk}^l a_k^{l-1} + b_j^l \qquad by\ definition$$

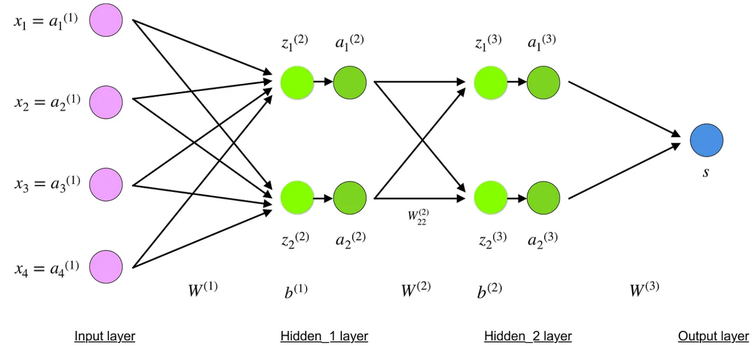
$$m\ -\ number\ of\ neurons\ in\ l-1\ layer$$

$$\frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l}\frac{\partial z_j^l}{\partial w_{jk}^l} \qquad chain\ rule \qquad \frac{\partial z_j^l}{\partial w_{jk}^l} = a_k^{l-1} \qquad by\ differentiation\ (calculating\ derivative)$$

$$\frac{\partial C}{\partial w_{jk}^l} = \frac{\partial C}{\partial z_j^l} a_k^{l-1} \qquad final\ value$$

[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

Machine learning mechanics
-
Back Propagation



$$z_j^l = \sum_{k=1}^{m} w_{jk}^l a_k^{l-1} + b_j^l \qquad by\ definition$$

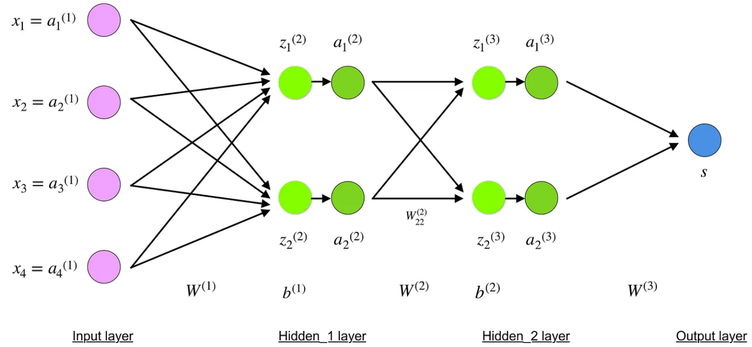
$$m\ -\ number\ of\ neurons\ in\ l-1\ layer$$

$$\frac{\partial C}{\partial b_j^l} = \frac{\partial C}{\partial z_j^l}\frac{\partial z_j^l}{\partial b_j^l} \qquad chain\ rule \qquad \frac{\partial z_j^l}{\partial b_j^l} = 1 \qquad by\ differentiation\ (calculating\ derivative)$$

$$\frac{\partial C}{\partial b_j^l} = \frac{\partial C}{\partial z_j^l}1 \qquad final\ value$$

[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

Machine learning mechanics
-
Back Propagation



$$while\ (termination\ condition\ not\ met)$$

$$w := w - \epsilon \frac{\partial C}{\partial w}$$

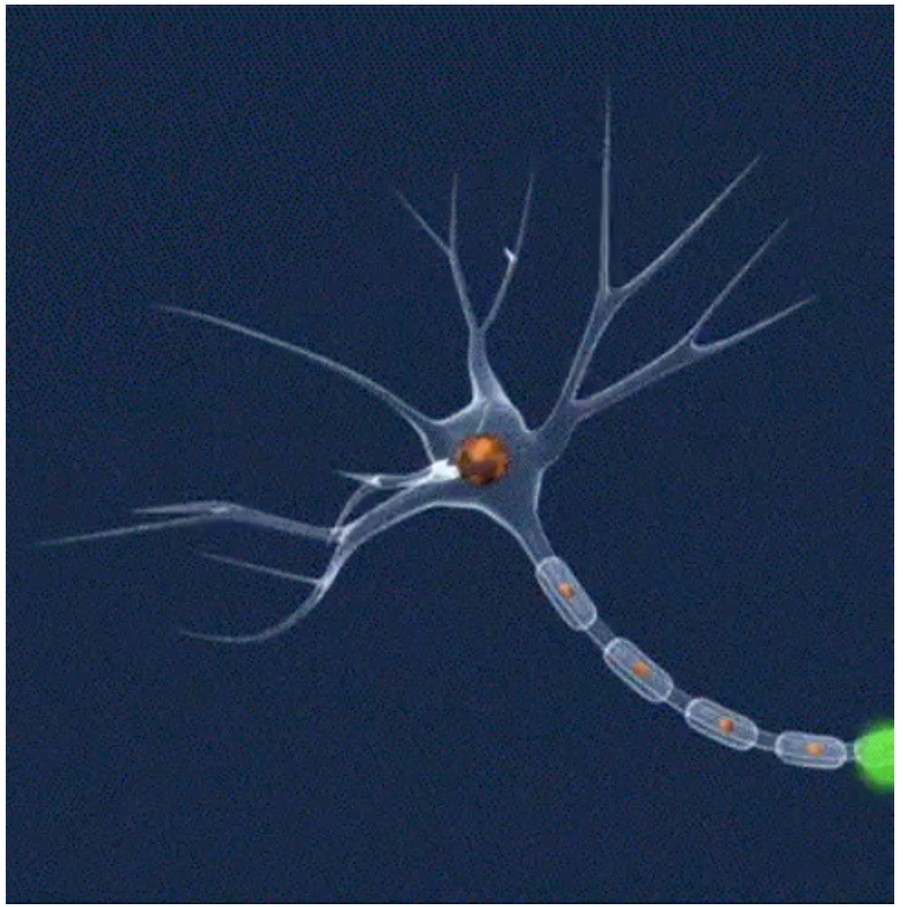
$$b := b - \epsilon \frac{\partial C}{\partial b}$$

$$end$$

Learning rate ϵ

[1] -https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2F%40mayankverma05032001%2Fbinary-classification-using-convolution-neural-network-cnn-model-6e35cdf5bdbb&psig=AOvVaw2y5GRO7aT-W01WYOiBYh3j&ust=1714480373134000&source=images&cd=vfe&opi=89978449&ved=0CBQQjhxqFwoTCJCdup6354UDFQAAAAAdAAAAABAE

Machine learning mechanics
-
Activation Functions



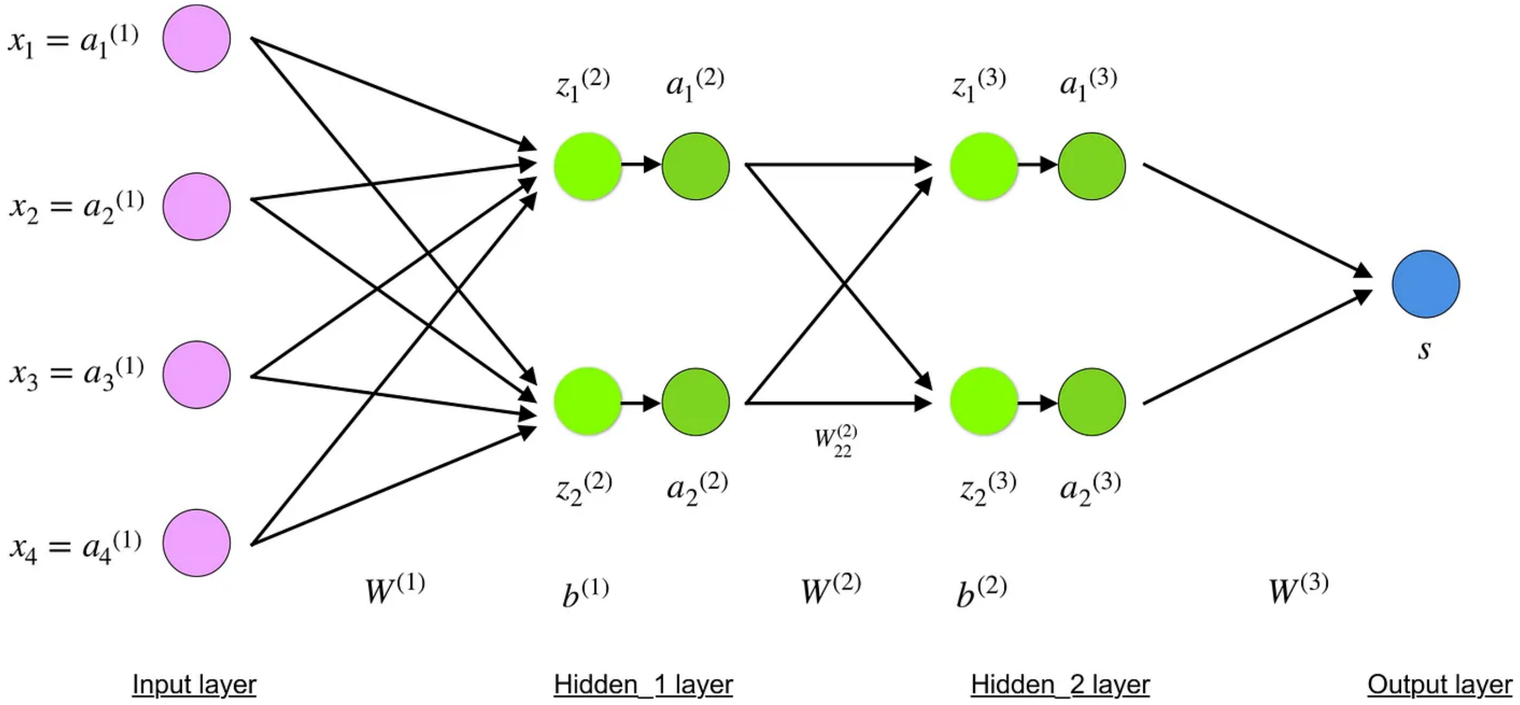
https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

https://www.nichd.nih.gov/newsroom/releases/031813-backwards-neurons

Machine learning mechanics

-

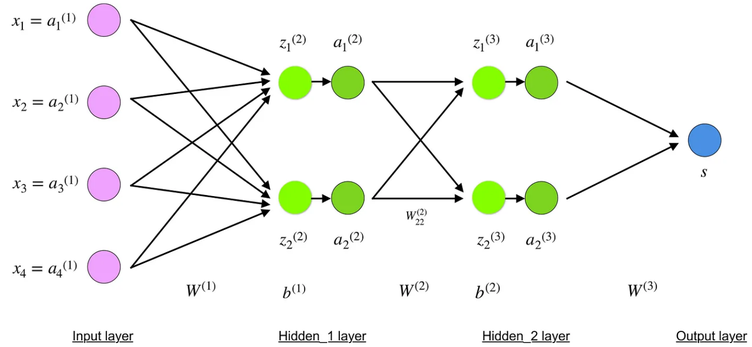
Activation Functions



$$Y = \sum (weight * input) + bias$$

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

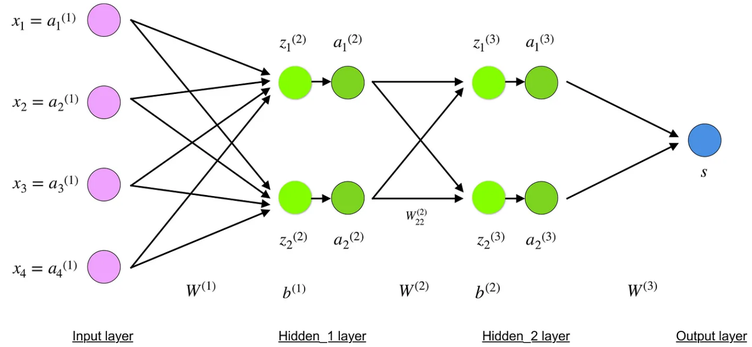
Machine learning mechanics
-
Activation Functions



$$Y = \sum (weight * input) + bias$$

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

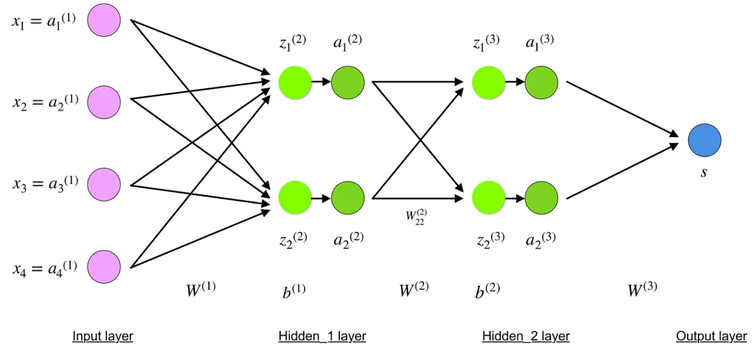
Machine learning mechanics
-
Activation Functions



$$Y \quad \in [-\infty; \infty]$$

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

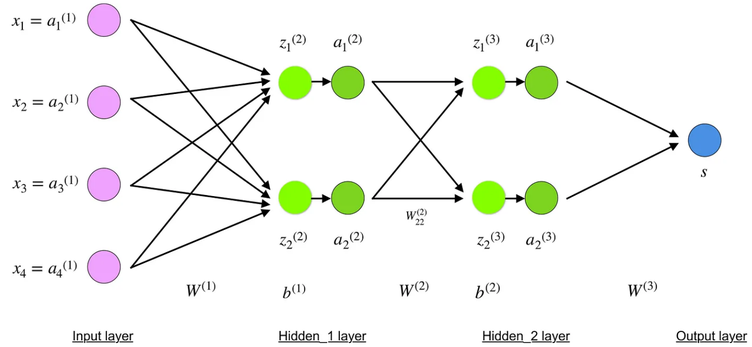
Machine learning mechanics
-
Activation Functions

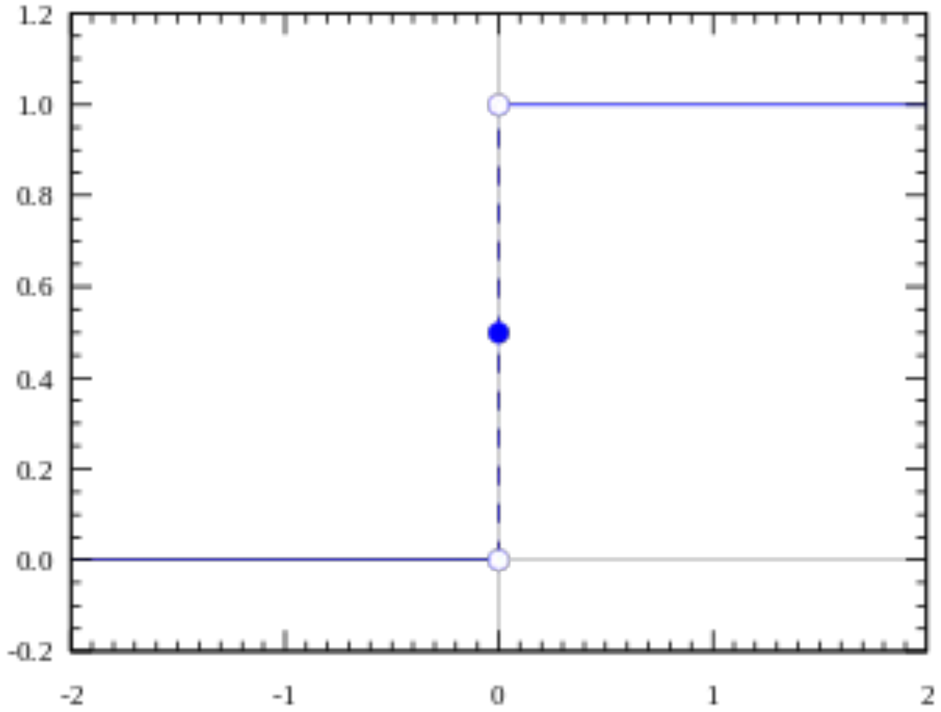


$$f(Y) \quad \in [\text{min} ; \text{max}]$$

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

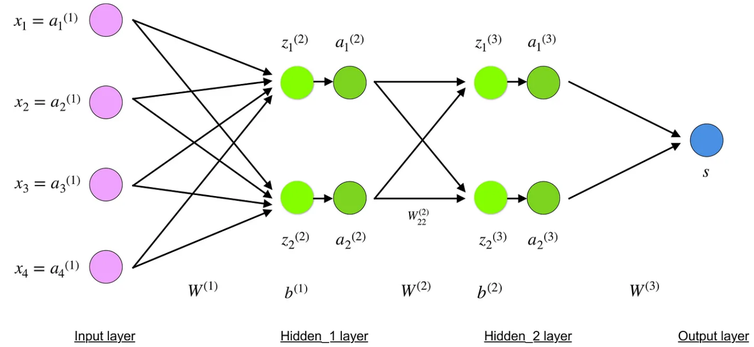
Machine learning mechanics
-
Activation Functions
-
Step function

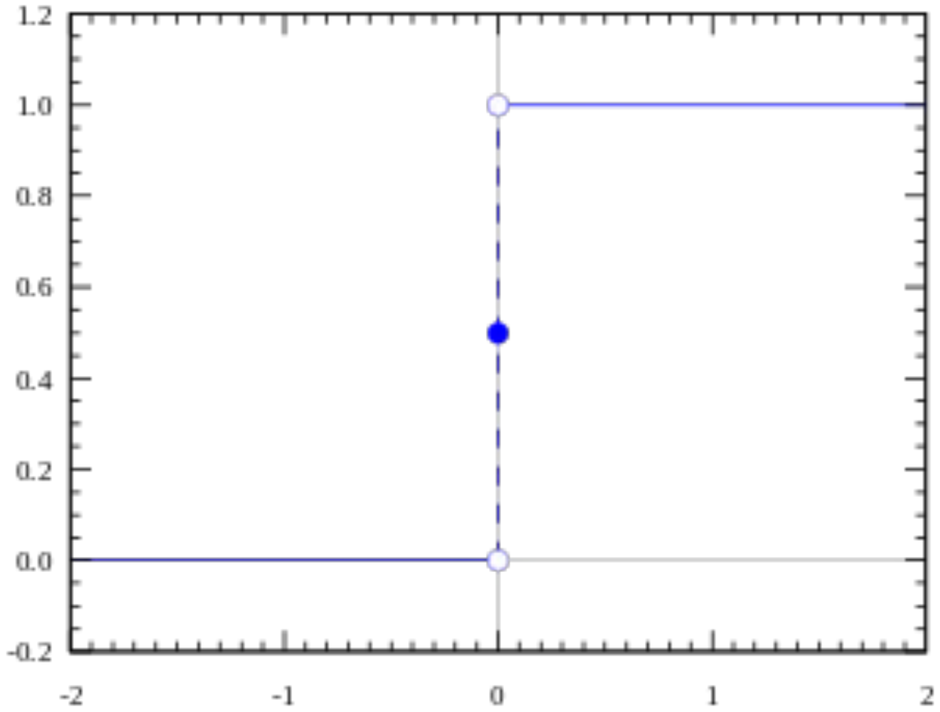




https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Machine learning mechanics
-
Activation Functions
-
Step function

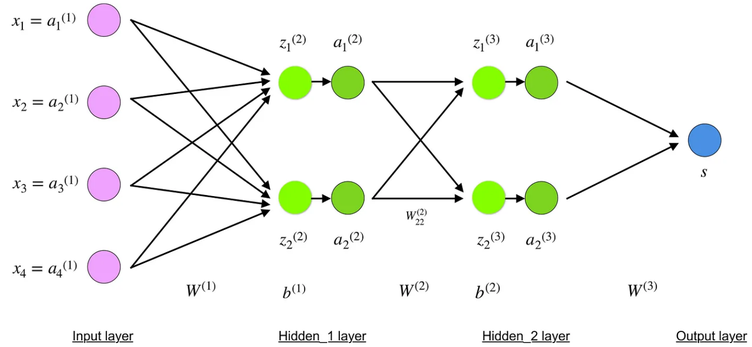




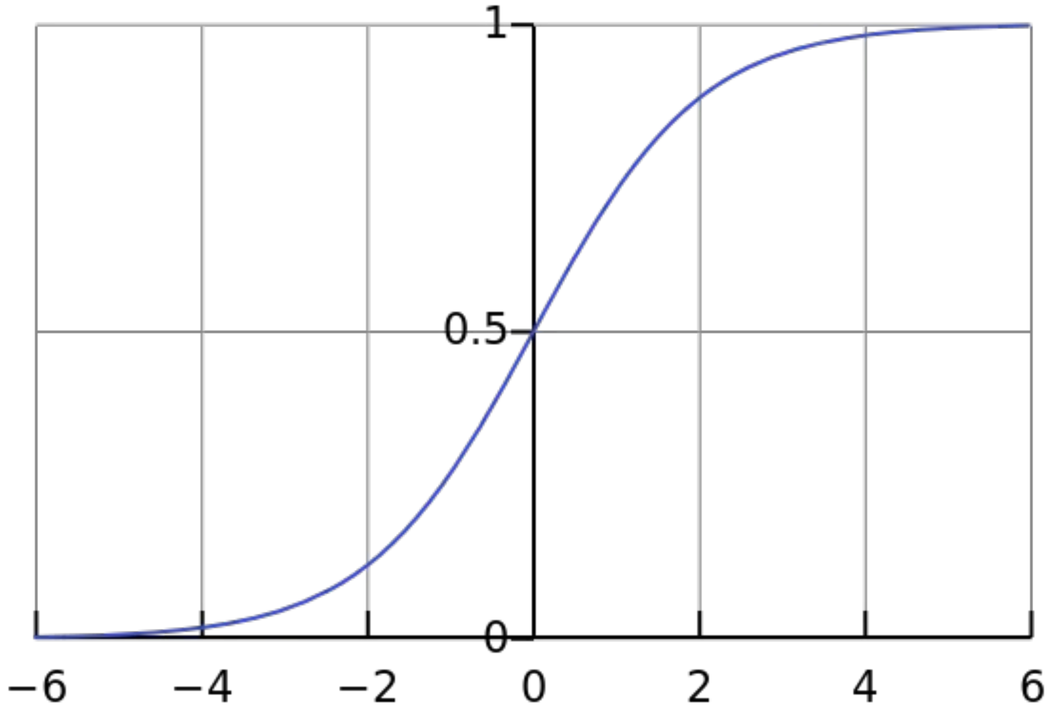
- Pros:
  - Mimics the neuron
  - $Y \in [0; 1]$
- Cons:
  - No smooth gradient
  - Binary

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Machine learning mechanics
-
Activation Functions
-
Sigmoid function

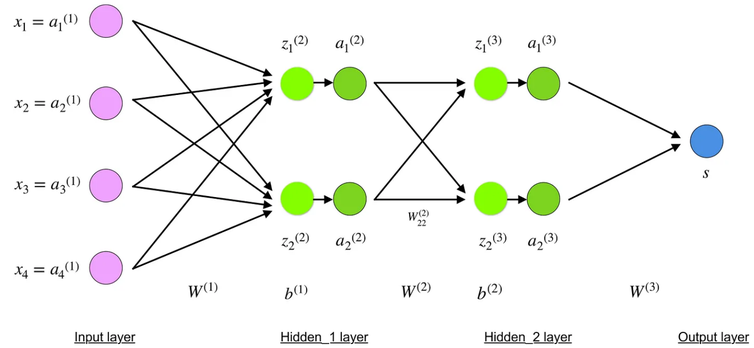


$$A = \frac{1}{1+e^{-x}}$$

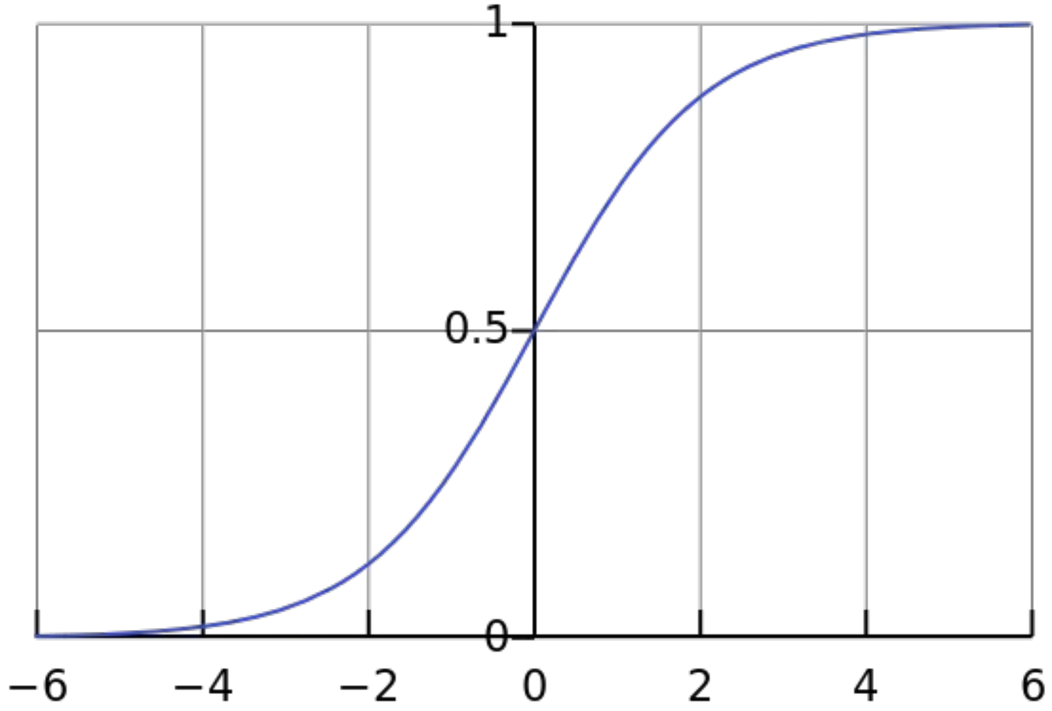


https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Machine learning mechanics
-
Activation Functions
-
Sigmoid function



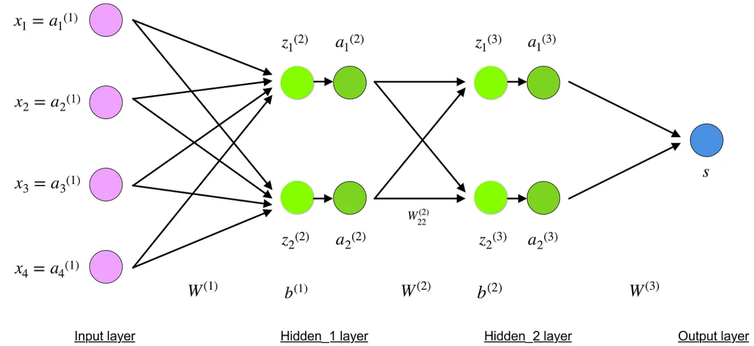
$$A = \frac{1}{1+e^{-x}}$$



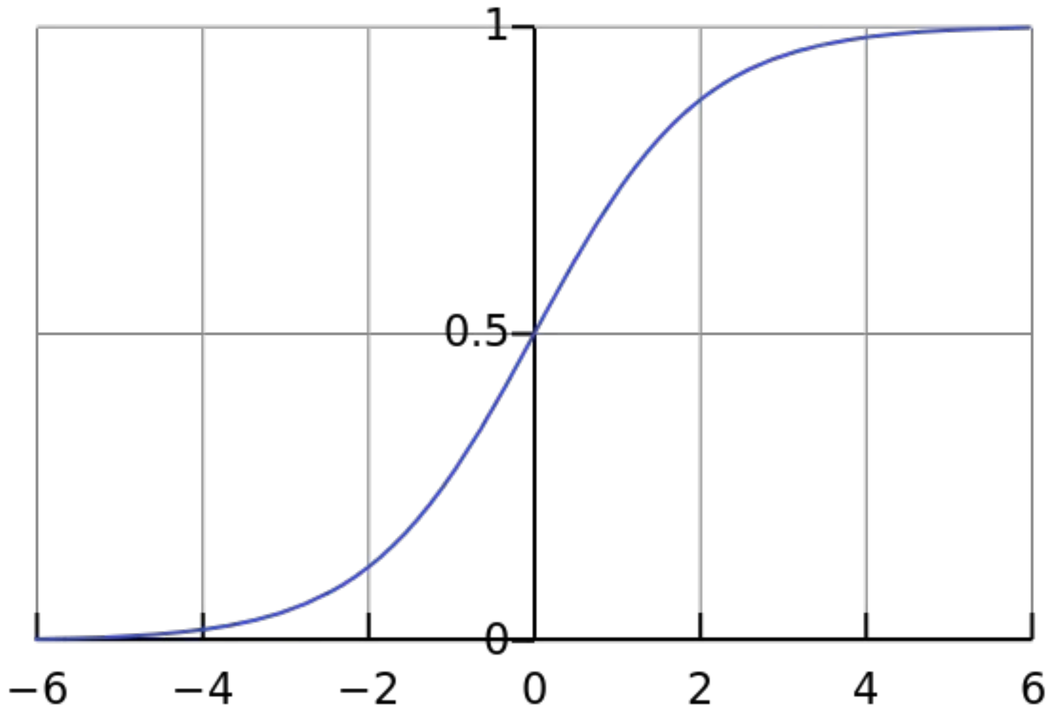
- Pros:
  - Smooth step
  - Smooth gradient
  - Not Binary
  - $Y \in [0; 1]$
- Cons:
  - Small gradient at the top and bottom
  - Computationally intensive

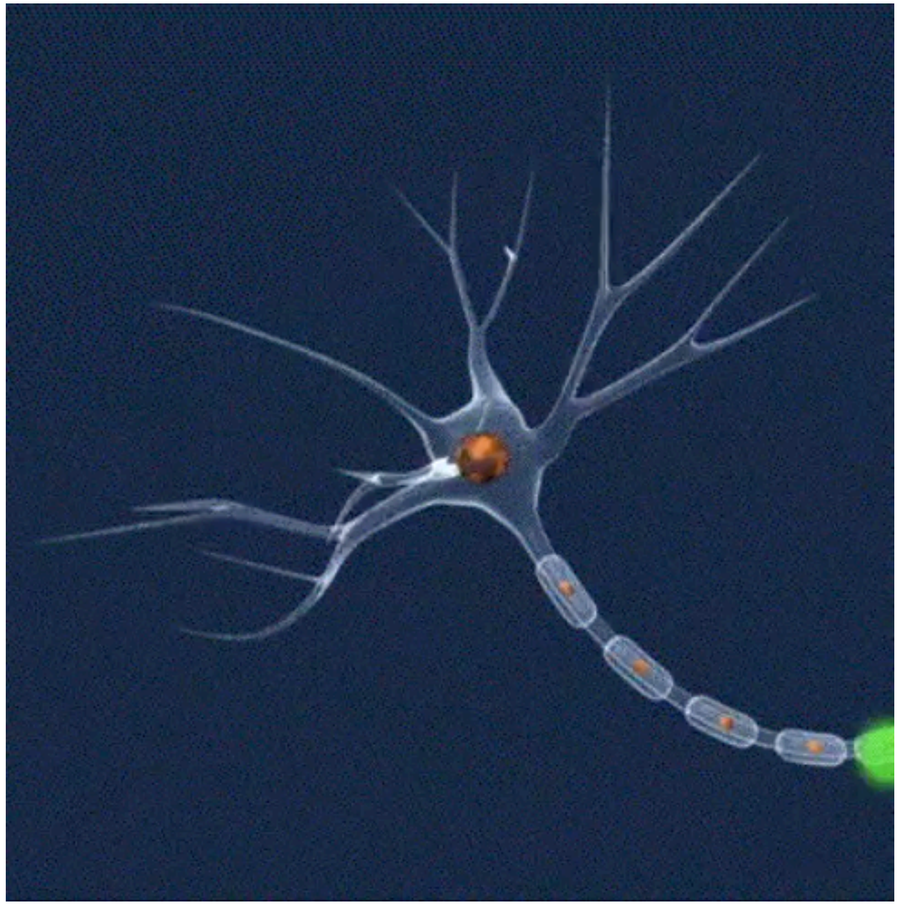
https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Machine learning mechanics
-
Activation Functions
-
Sigmoid function



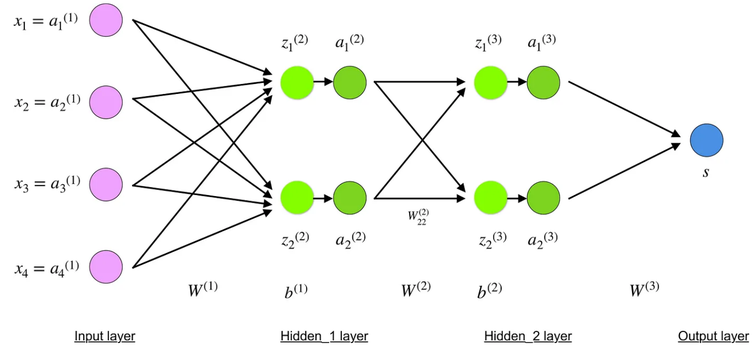
$$A = \frac{1}{1+e^{-x}}$$



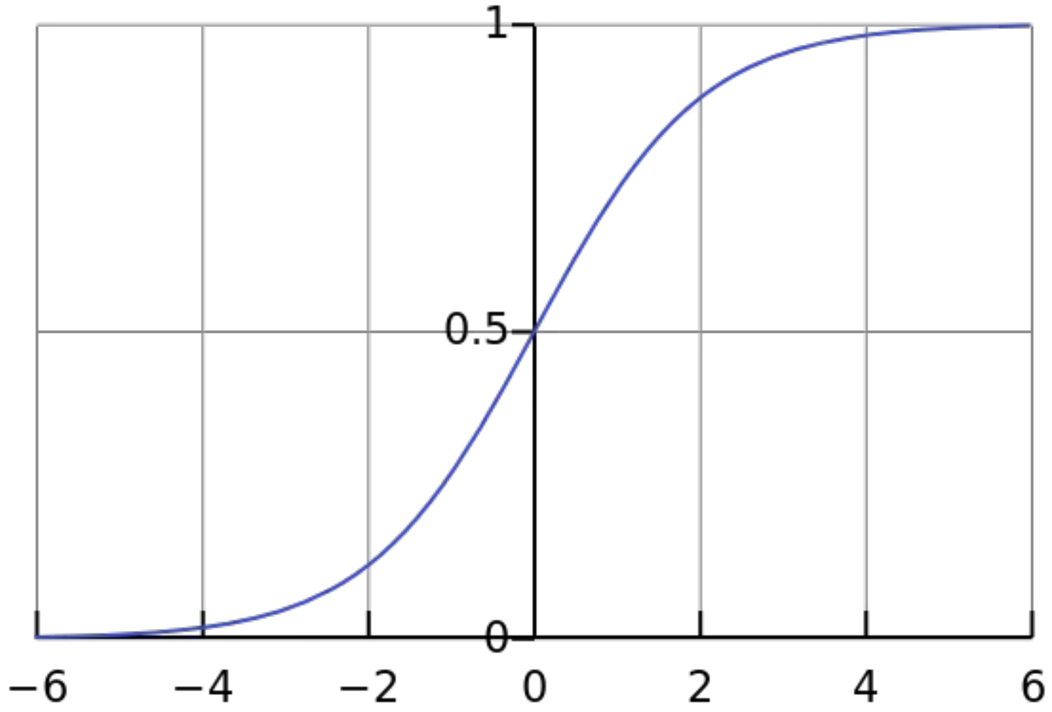


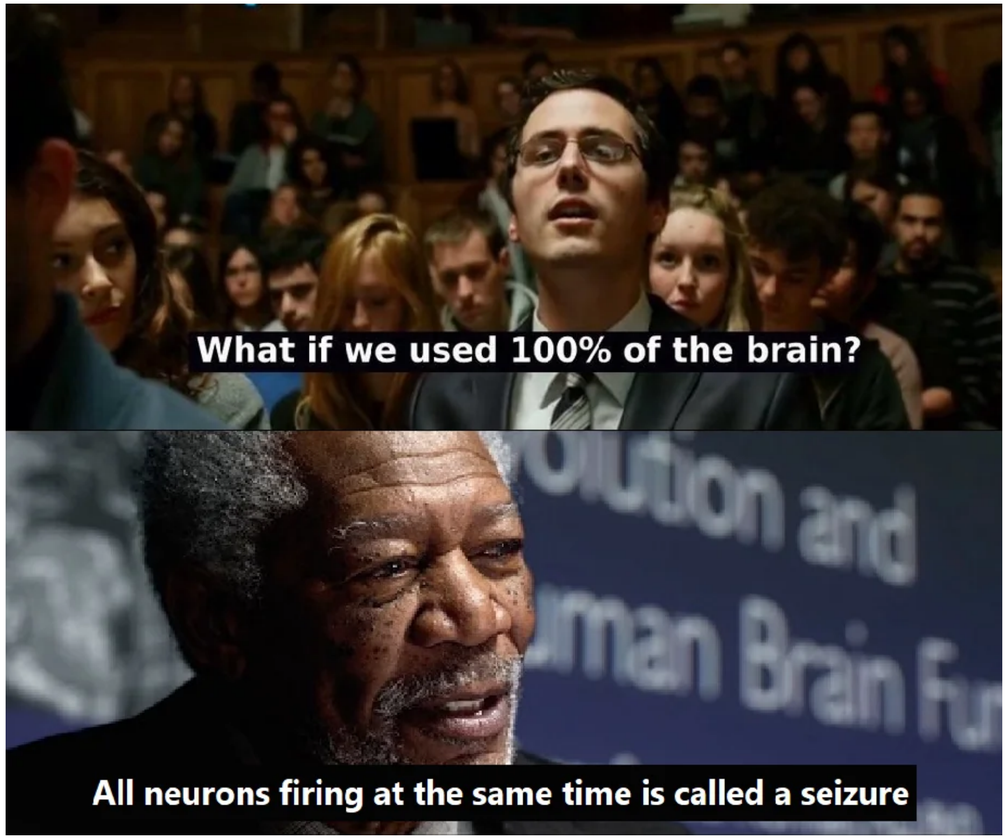
https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Machine learning mechanics
-
Activation Functions
-
Sigmoid function



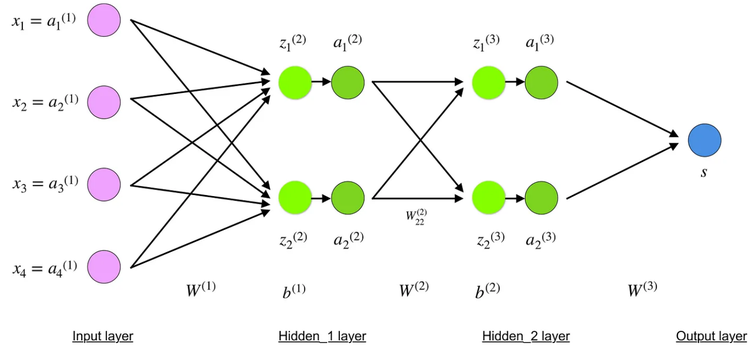
$$A = \frac{1}{1+e^{-x}}$$



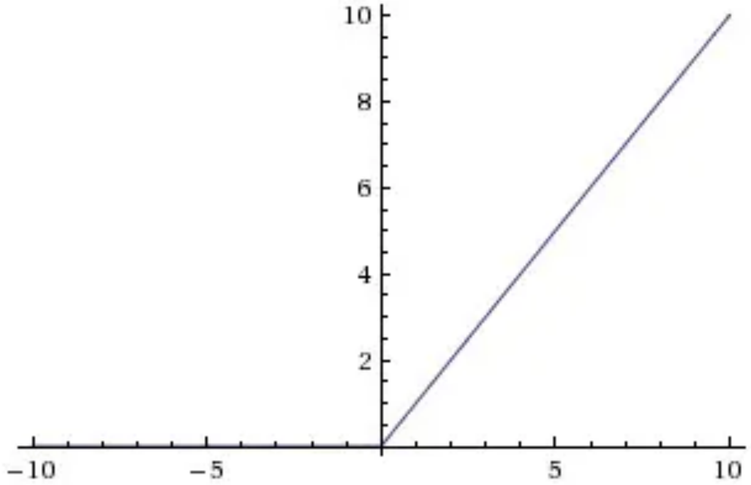


https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0
So people with epilepsy are actually geniuses : r/memes

Machine learning mechanics
-
Activation Functions
-
ReLu function



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



https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Machine learning mechanics

-

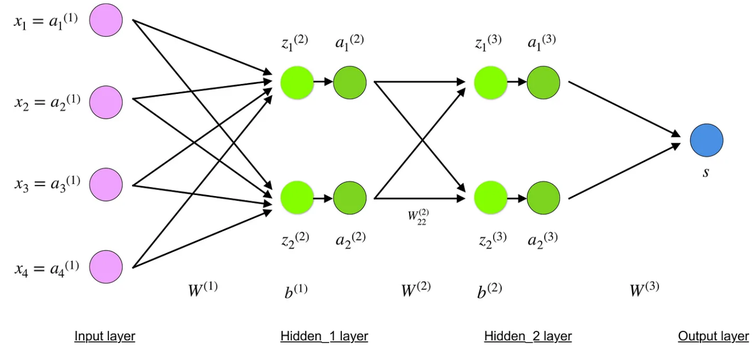
Activation Functions

-

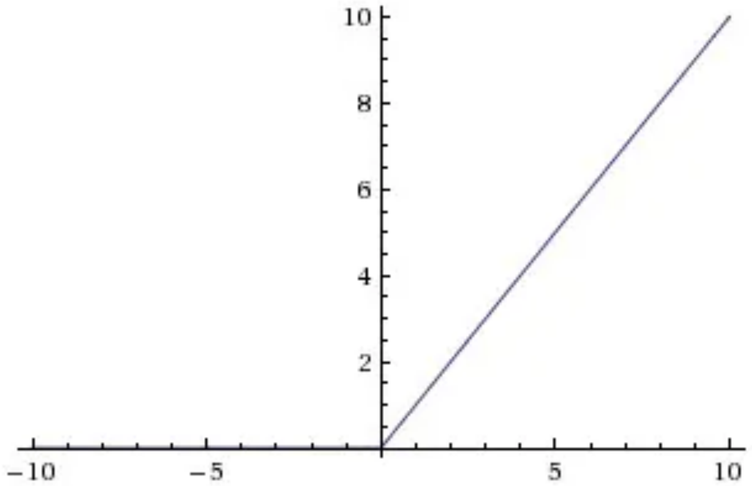
Rectified Linear Unit (ReLU)

-

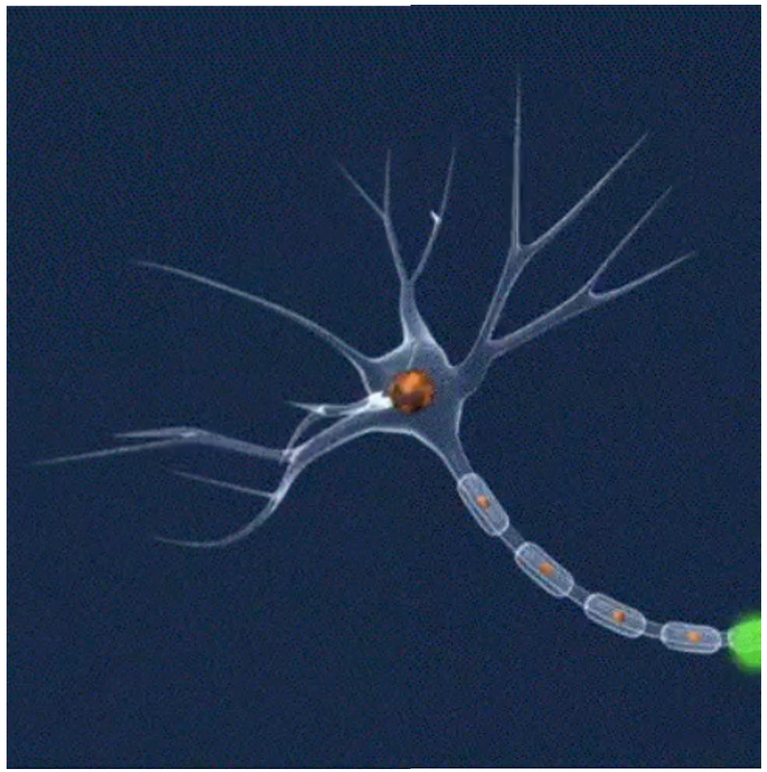
Sparsity of the activation



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



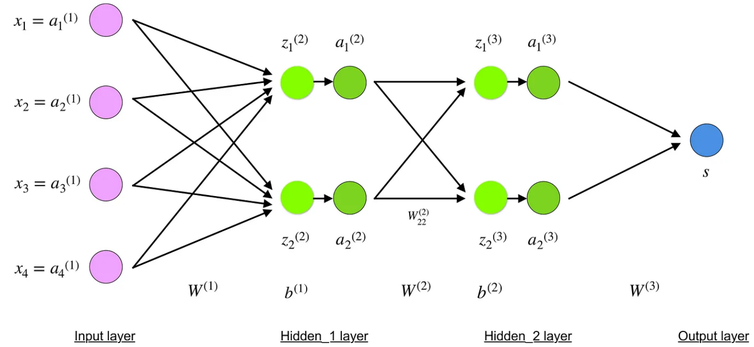
$$Y \in [0; \infty]$$



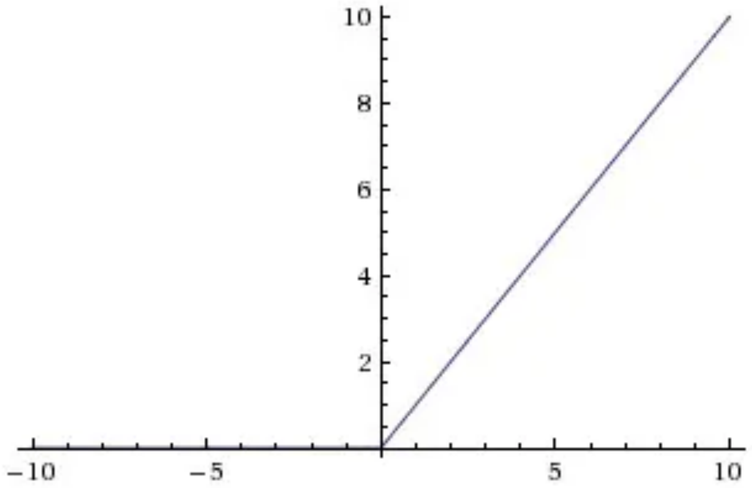
https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

https://www.nichd.nih.gov/newsroom/releases/031813-backwards-neurons

Machine learning mechanics
-
Activation Functions
-
Rectified Linear Unit (ReLU)



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



$$Y \in [0; \infty]$$

- Pros:
  - Less computationally intensive
  - Not Binary
  - Constant gradient
- Cons:
  - Dying ReLU
  - Can Blow up

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

Machine learning mechanics

-

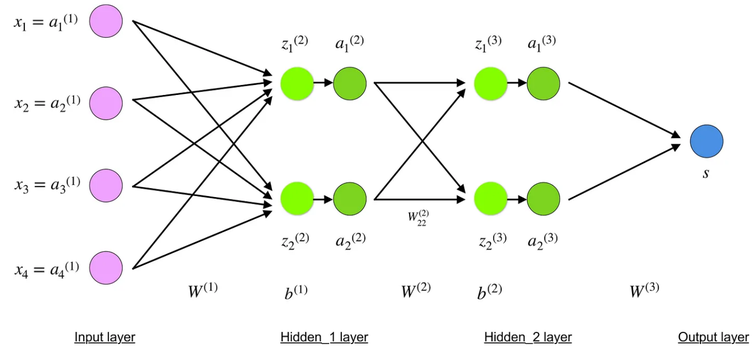
Activation Functions

-

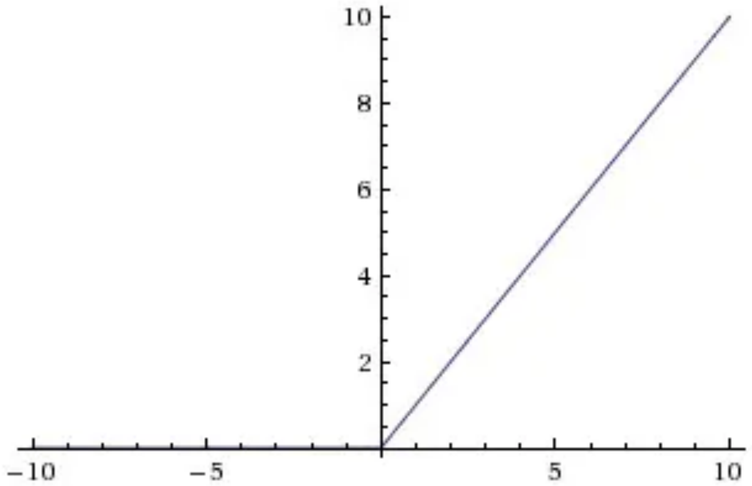
Rectified Linear Unit (ReLU)

-

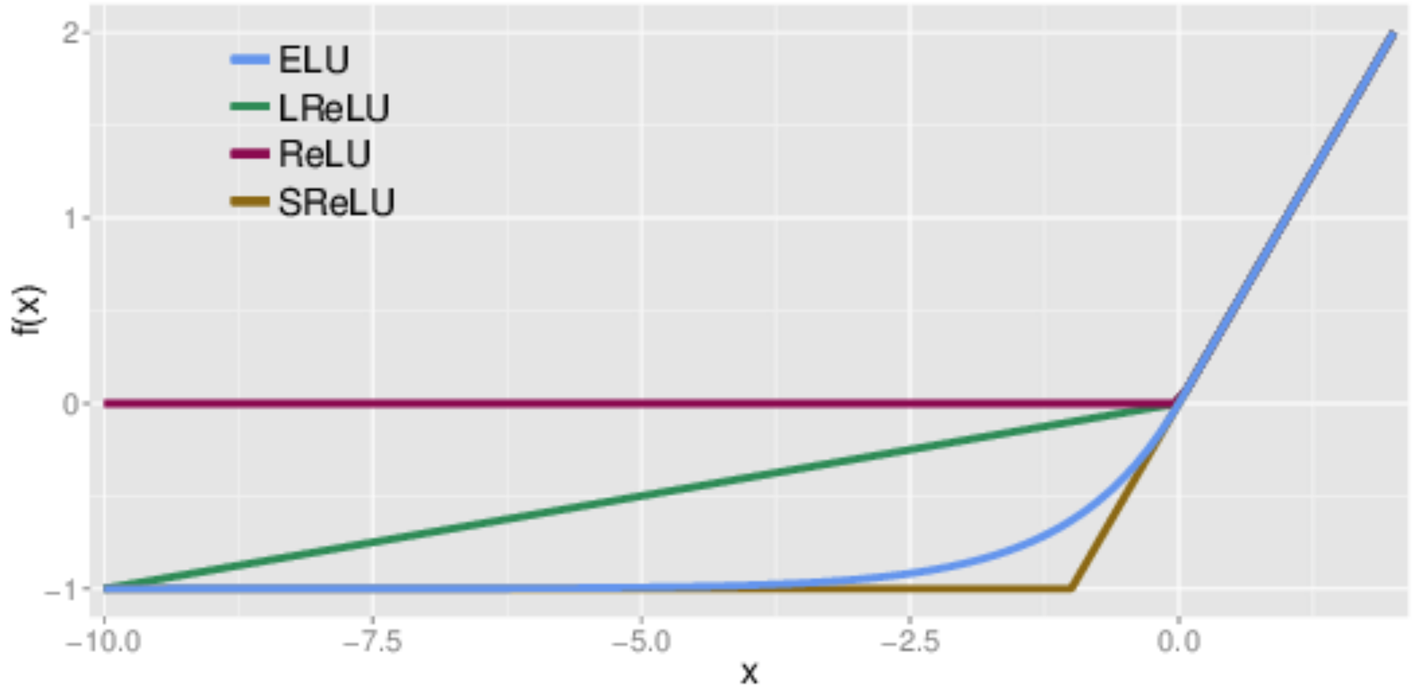
Dying ReLu Problem



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



$$Y \in [0; \infty]$$



https://medium.com/the-theory-....ng/understanding-activation-functions-in-neural-networks-9491262884e0

https://www.nichd.nih.gov/newsroom/releases/031813-backwards-neurons

Which one to choose ?

- Depends on the size of the model
- A good and balanced combination of both 😯

Machine learning mechanics
-
Loss Functions

- Metric to characterize how far the model is from the label
  - Classification Losses:
    - Discrete (Ex: Character recognition, Animals, (n_2, I_{sat}))
  - Regression Losses:
    - Continuous (Ex: Someone's age, (n_2, I_{sat}))

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

https://www.nichd.nih.gov/newsroom/releases/031813-backwards-neurons

Machine learning mechanics

-

Loss Functions

-

Classification Losses

Mean Square Error: $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$ (better gradients, penalizes off values)

Mean Absolute Error: $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} \mid Y_i - \hat{Y}_i \mid$ (less penalizes off values)

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

https://www.nichd.nih.gov/newsroom/releases/031813-backwards-neurons

Machine learning mechanics

-

Loss Functions

-

Regression Losses

Mean Square Error: $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$ (better gradients, penalizes off values)

Mean Absolute Error: $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} \mid Y_i - \hat{Y}_i \mid$ (less penalizes off values)

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

https://www.nichd.nih.gov/newsroom/releases/031813-backwards-neurons

Machine learning mechanics

-

Loss Functions

-

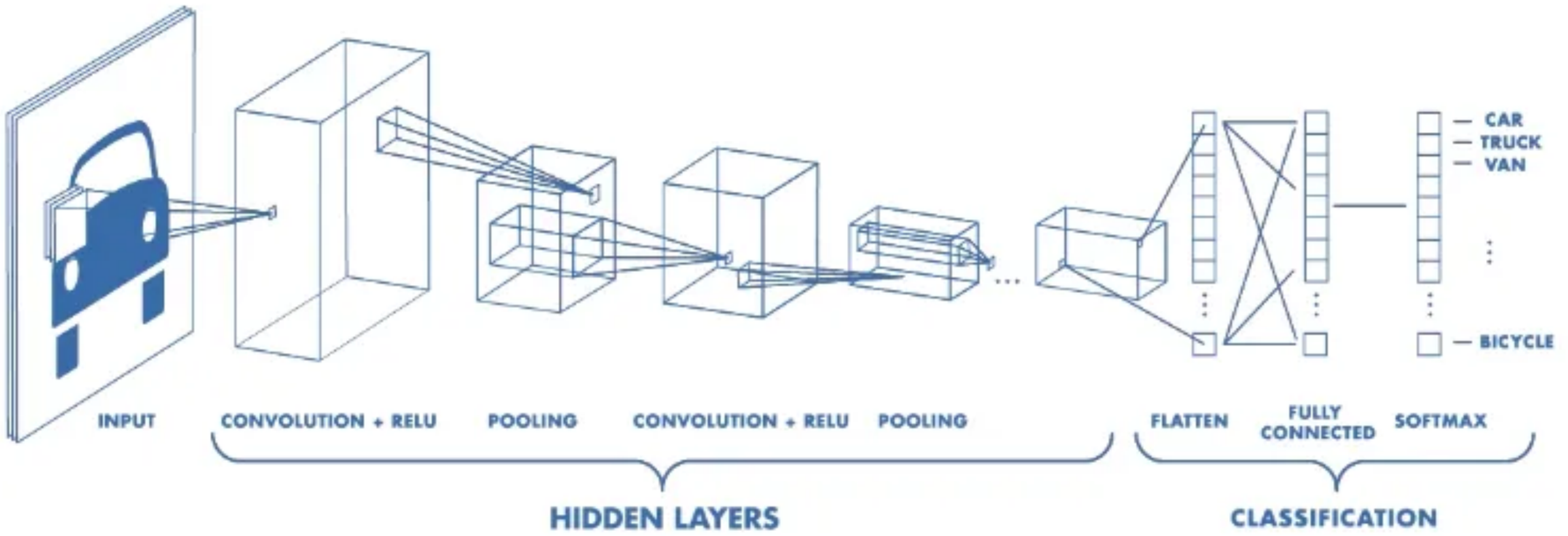
Classification Losses

$$CrossEntropyLoss = -(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i))$$

https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

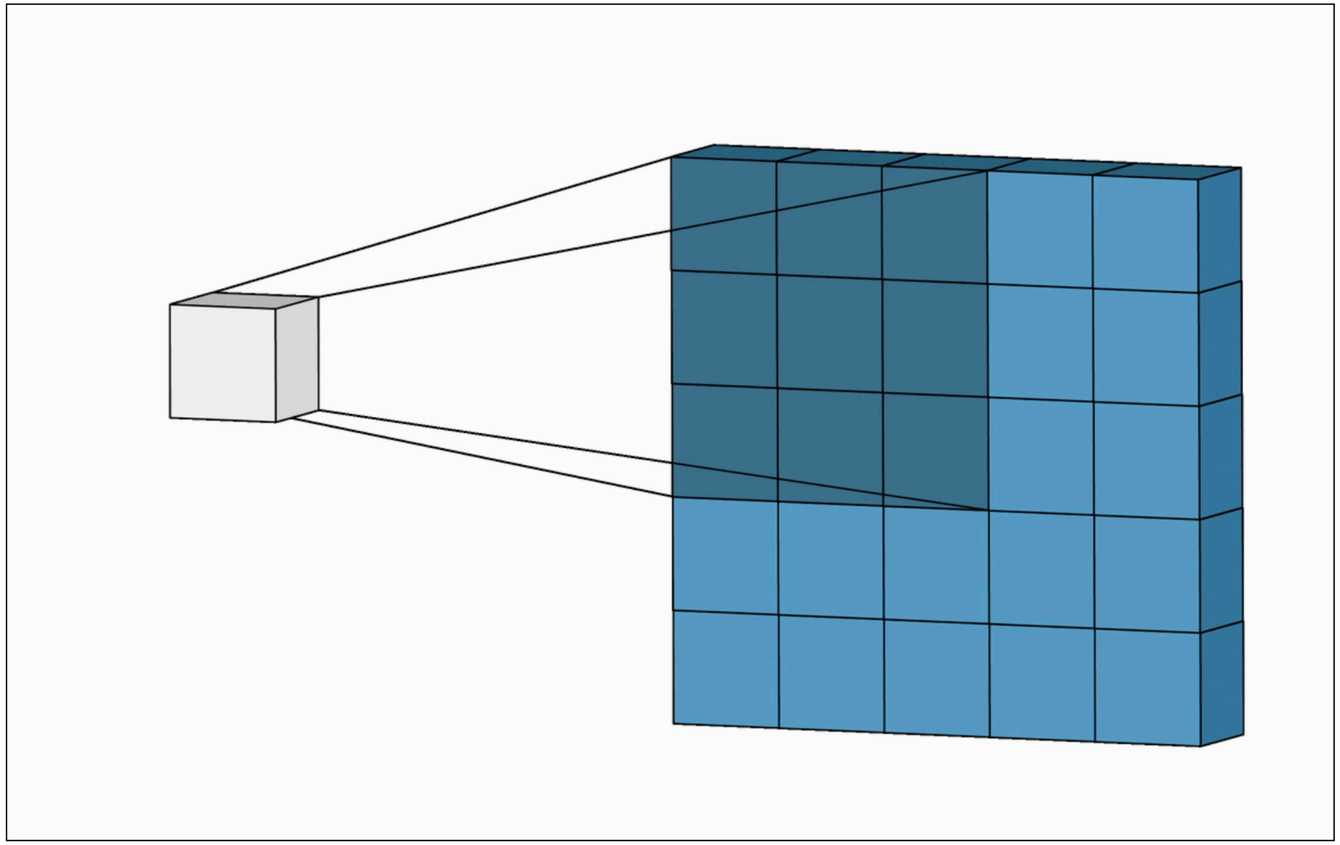
https://www.nichd.nih.gov/newsroom/releases/031813-backwards-neurons

Machine learning mechanics
-
Convolutional Neural Networks



https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

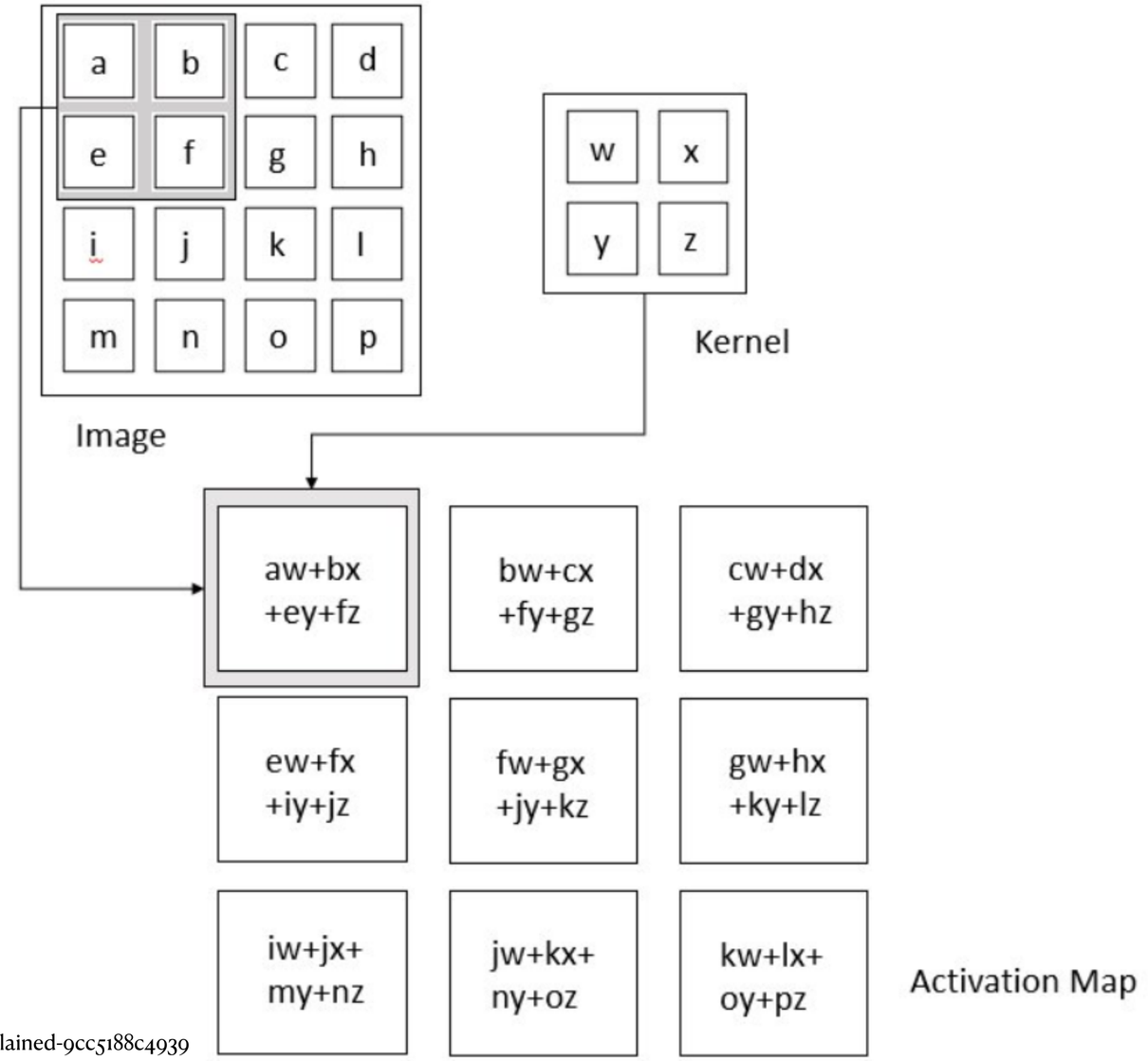
Machine learning mechanics
-
Convolution



https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

Machine learning mechanics
-
Convolution

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

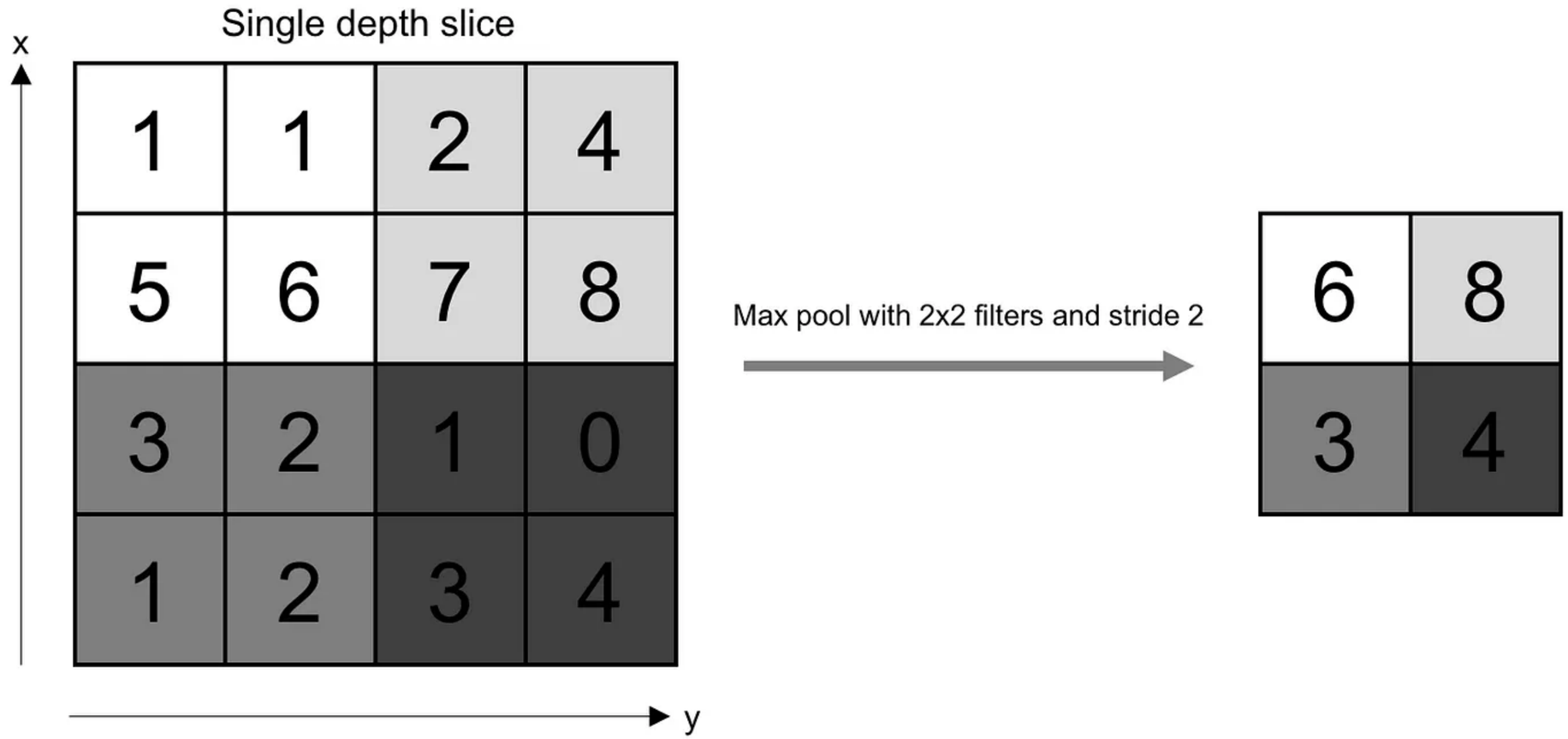


https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

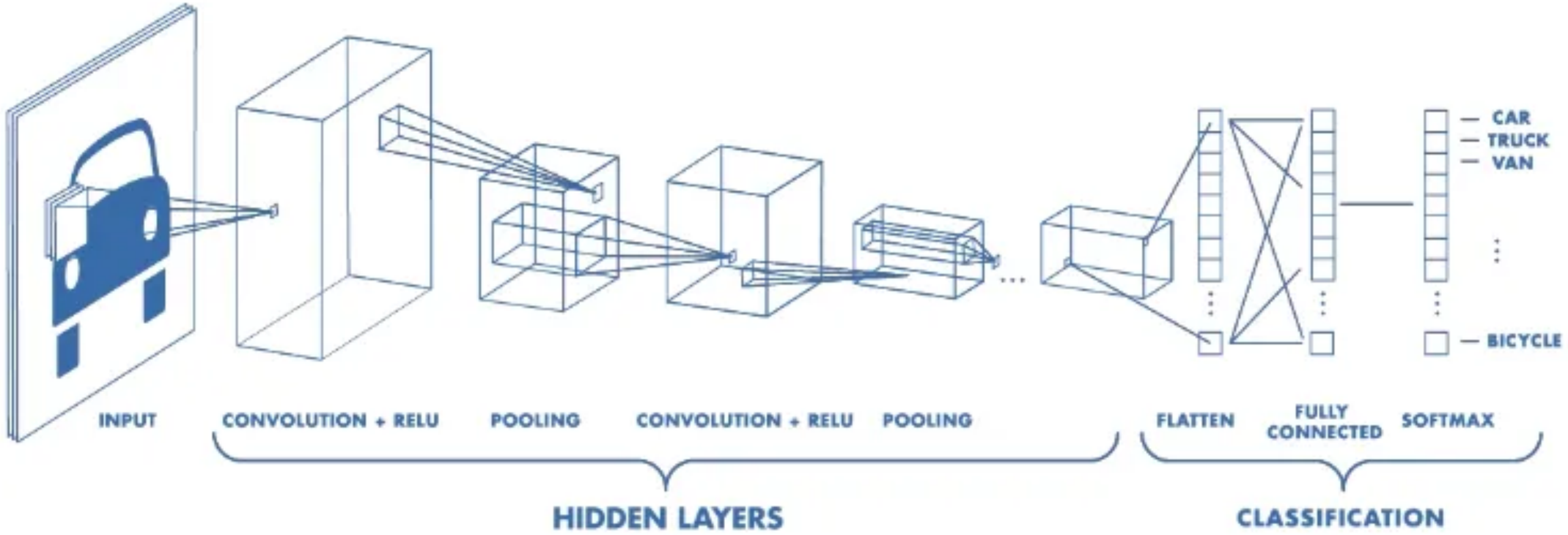
Machine learning mechanics

-

Pooling Layer



Machine learning mechanics
-
Convolutional Neural Networks



https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939

Why CNN ?

- Deals with signals and images as a whole
- Parameter sharing
- Better generalization

Thank You