# Deep Learning Workshops

Workshops plan:

- **Wednesday, October 12th, 17:00 - 19:00 (theory)**
  - **machine learning fundamentals**
  - **deep neural networks**
- Monday, October 17th, 17:00 - 19:00 (theory + laboratory),
- Wednesday, October 19th, 17:00 - 19:00 (laboratory ?)
- Projects: subsequent Wednesdays
  - in-person: A1 wyb. Wyspiańskiego 27, room 219, 10:00-12:00 or 14:00-16:00
  - or online: zoom

lecturer: jaroslaw.pawlowski@pwr.edu.pl
GitHub repo: https://github.com/jarek-pawlowski/deep-learning-workshops

Why I am here

MedApp

Our offer · About Medapp · Investor relations · News · Contact · Search

**Fast analysis of ECG results boosted by artificial intelligence**

CarnaLife System connects with holters via GSM, WiFi or Bluetooth and enables remote monitoring of the centre's patients. The ECG results from a patient are preliminarily analysed by certified artificial intelligence algorithms, what helps to reduce the time a physician spends on the test analysis and facilitates the diagnostics.

**Certified algorithms detecting irregularities in the ECG signal on the level higher than 95%***

Artificial intelligence algorithms are compliant with the EC-57 standard, confirmed on MIT BIH, NST, AHA reference databases and on clinical data. They provide 98.78 effectiveness in detection and analysis of heart rhythm, 95.5 effectiveness in detection of supraventricular events and atrial fibrillation as well as 95.5 effectiveness in detection of ventricular events. The algorithms are fed by anonymised patient data and thus improve their effectiveness on a permanent basis.

**Assistance in cardiac rehabilitation of patients after myocardial infarction through teleconsultation and post-treatment check ups.**

CarnaLife System enables remote collection of ECG, blood pressure and, optionally, pulse oximetry measurements. The tests are performed under the supervision of a specialist who, in the remote contact with a patent, advises physical exercises, rest or other rehabilitation-related activities.

neurosys

# Rezultaty: precyzja detekcji

Detektor radzi sobie zarówno z typami mikrobów rosnących w małe jak i dużo większe kolonie

**Table 6.** Average precision calculated for the each class of microbes separately at different IoU-thresholds for Faster R-CNN (ResNet-50) with higher-resolution subset. We get overall better performance for smaller colonies.

| | microbe type | | | | |
|---|---|---|---|---|---|
| IoU | *S.aureus* | *B.subtilis* | *P.aeruginosa* | *E.coli* | *C.albicans* |
| 0.5 | 86.0% | 72.4% | 69.9% | 73.1% | 82.0% |
| 0.55 | 85.9% | 69.5% | 67.4% | 68.9% | 82.0% |
| 0.6 | 84.8% | 64.6% | 65.6% | 64.0% | 80.8% |
| 0.65 | 80.9% | 56.9% | 62.4% | 56.6% | 79.6% |
| 0.7 | 77.9% | 48.9% | 57.9% | 47.9% | 77.1% |
| 0.75 | 71.1% | 39.8% | 50.9% | 39.0% | 73.2% |
| 0.8 | 58.4% | 30.0% | 38.0% | 30.2% | 65.5% |
| 0.85 | 37.4% | 17.5% | 23.3% | 19.4% | 49.4% |
| 0.9 | 14.8% | 6.3% | 7.9% | 7.4% | 19.2% |
| 0.95 | 0.6% | 0.4% | 0.7% | 0.5% | 1.8% |
| 0.5:0.95 | 59.8% | 40.6% | 44.4% | 40.7% | **61.1%** |

AP

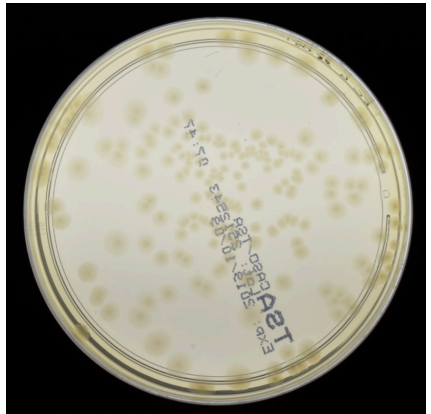
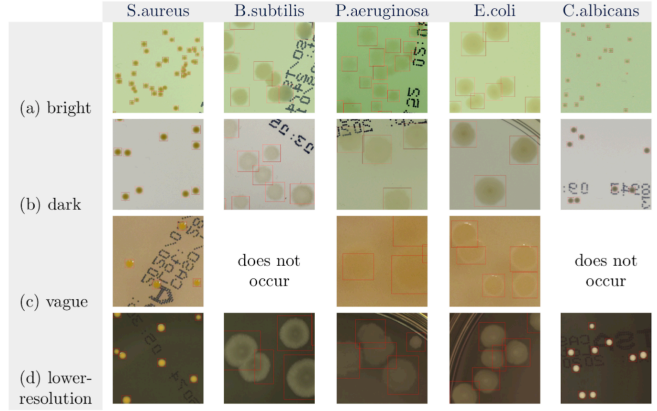
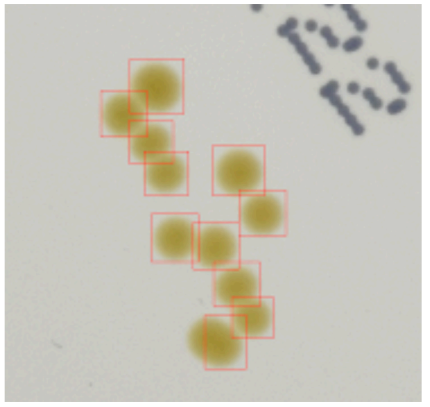


**Table 3.** Examples of Cascade R-CNN with HRNet predictions for different types of microbes grown inside a Petri dish captured in four different setups, as in Table 1. Each image is of 20 × 20 mm size.



# “Machine learning – applications” (last semester @ Big Data Analysis, WPPT PWr)

- advanced machine learning (mainly deep learning but not only)



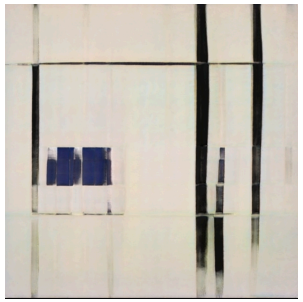




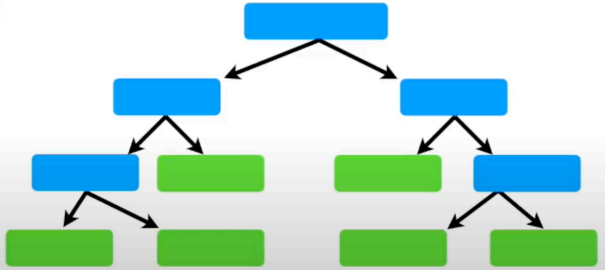

Which one is unreal?

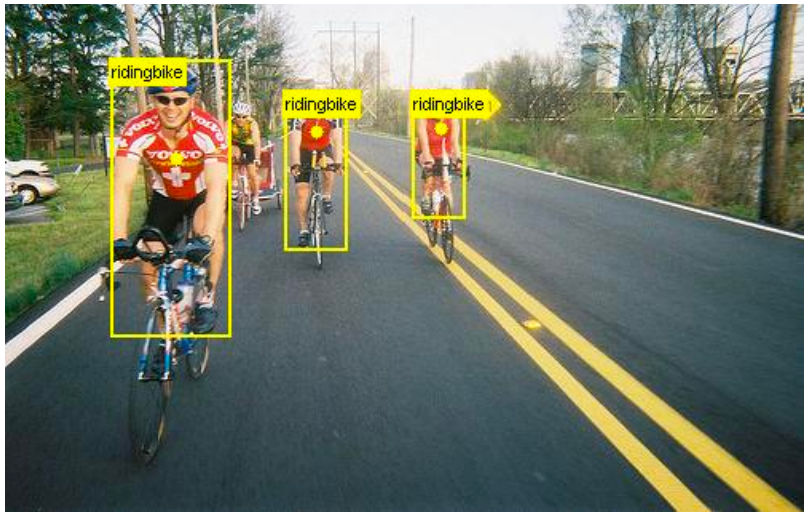
GANs  = Generative Adversarial (neural) Networks

thisxdoesnotexist.com




| Chest Pain | Good Blood Circulation | Blocked Arteries | Heart Disease |
|---|---|---|---|
| No | No | No | No |
| Yes | Yes | Yes | Yes |
| Yes | Yes | No | No |
| Yes | No | ??? | Yes |
| etc… | etc… | etc… | etc… |






# machine learning fundamentals

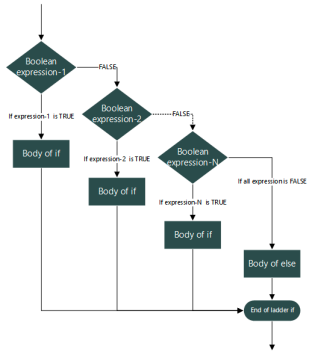
Recommended Literature

- I. Goodfellow , Y. Bengio, and A. Courville, *Deep Learning, Chapter 5: Machine Learning Basics*, https://www.deeplearningbook.org
- Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature **521,** 436 (2015).
- https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf
- https://tomaszgolan.github.io/introduction_to_machine_learning

# What is machine learning?

Family of computer methods (“programs”) that improve automatically through experience (“training”)
- without being *explicitly* designed (“programmed”) to

```
if(finalScore >= 90)
        cout << "You got A!" << endl;
else if(finalScore >= 80 && finalScore < 90)
        cout << "You got B!" << endl;
else if(finalScore >= 70 && finalScore < 80)
        cout << "You got C!" << endl;
else if(finalScore >= 60 && finalScore < 70)
        cout << "You got D!" << endl;
else
        cout << "You got F!" << endl;
```




vs.



<u>What kind of problems we would like (to be able) to solve?</u>
- the early days of AI: problems that are intellectually difficult for humans but straightforward for computers
- nowadays, the *true* challenge to AI are tasks easy for people to perform but **hard to be described formally**
  - like recognizing spoken words or faces on images

# What is machine learning?

How it works?

- Allow computers to learn from experience (“interacting with data”) and understand the world in terms of a *hierarchy of concepts,*
- this approach avoids the need for human to specify the knowledge (“exact algorithms”) that the computer needs,
- this hierarchy of concepts enables the computer to learn complicated concepts by building them out of simpler ones.

If we draw a graph showing how these concepts are built on top of each other, the structure is *deep*, with many layers. For this reason, we call this approach to AI as **deep learning.**

Here each layer is automatically trained to learn and detect different features on the input image.

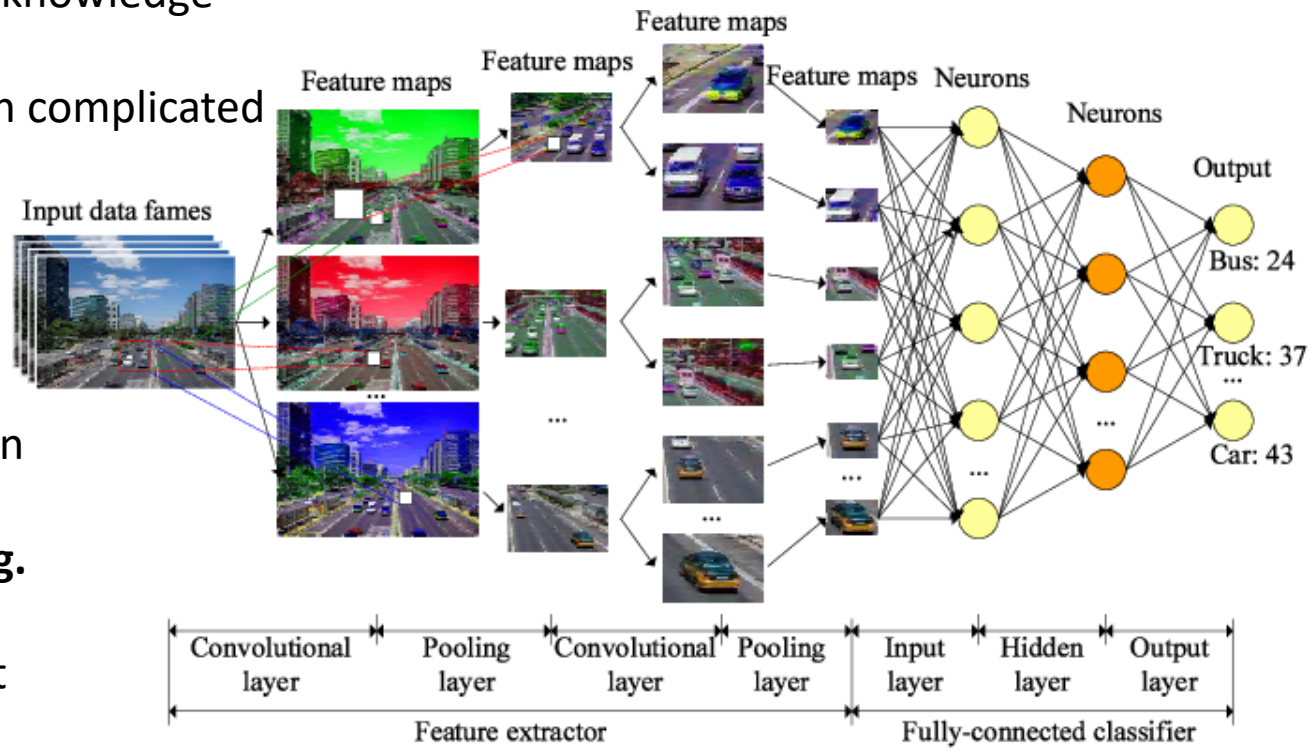


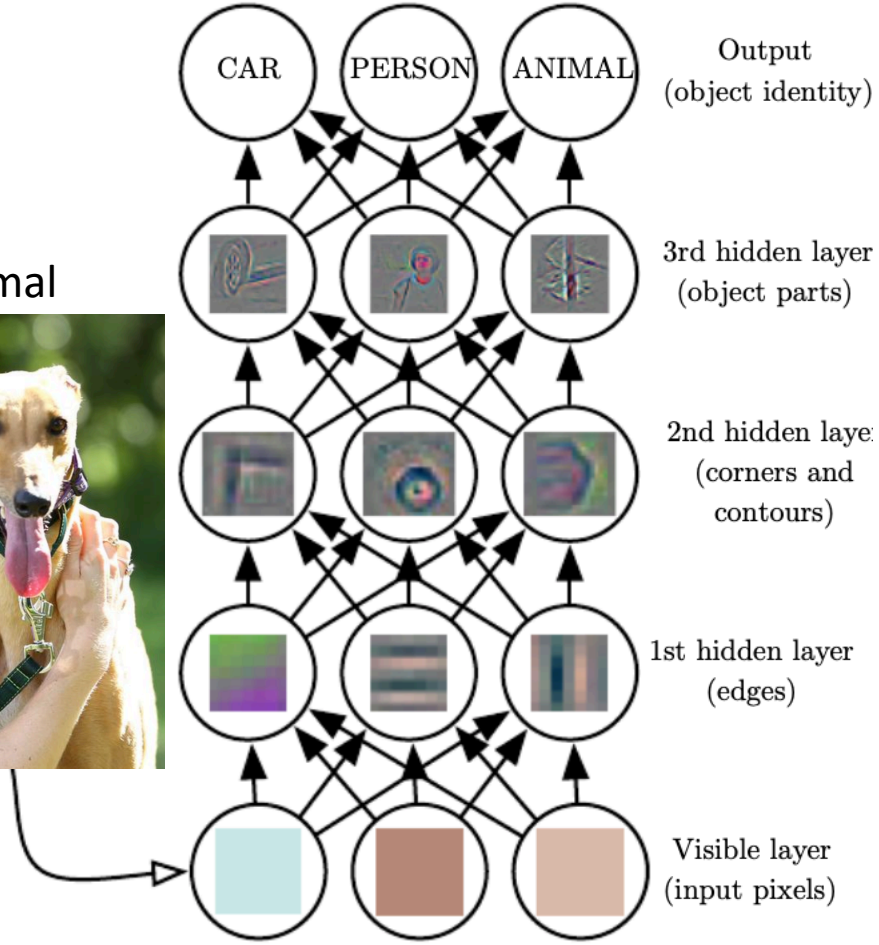

Fig. 3. Example of the CNN model for vehicle classification.

doi:10.1109/TII.2019.2909473

# What is machine learning?

ML methods learn such concepts (“features” or “representations”)
at different levels *automatically*
- by analyzing the data.

person vs animal




To sum up:

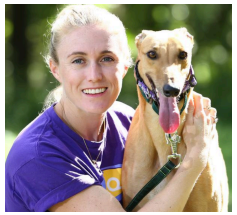
- deep learning solves problems by introducing representations that are expressed in terms of other, simpler representations
- deep learning enables the computer to automatically learn complex representations out of simpler ones

www.deeplearningbook.org/contents/intro.html

# Machine learning tasks

Typical ML tasks:

- classification $f : \mathbb{R}^n \to \{1, \ldots, k\}.$
- regression $f : \mathbb{R}^n \to \mathbb{R}$




- transcription (OCR/ASR), translation
- extract word and whole sentences from written or spoken data

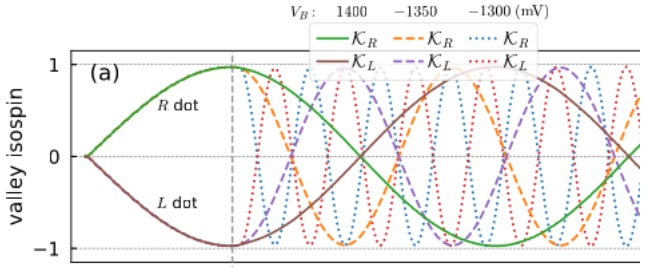
- anomality detection
- detect events or objects that are unusual in a given context

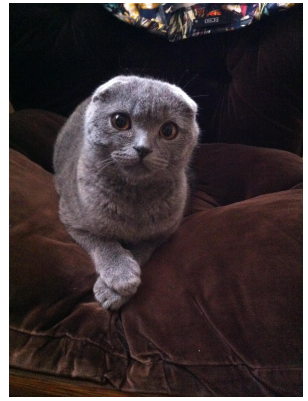
- synthesis and sampling
- generate new examples that are similar to those in the training data

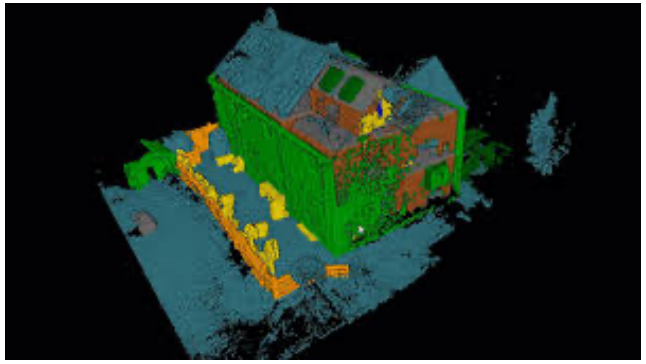
- denoising
- predict clean sample $x$ from the corrupted one $\tilde{x}$

Example data dimensionality:

- 1d signal
- 2d image
- 3d point cloud

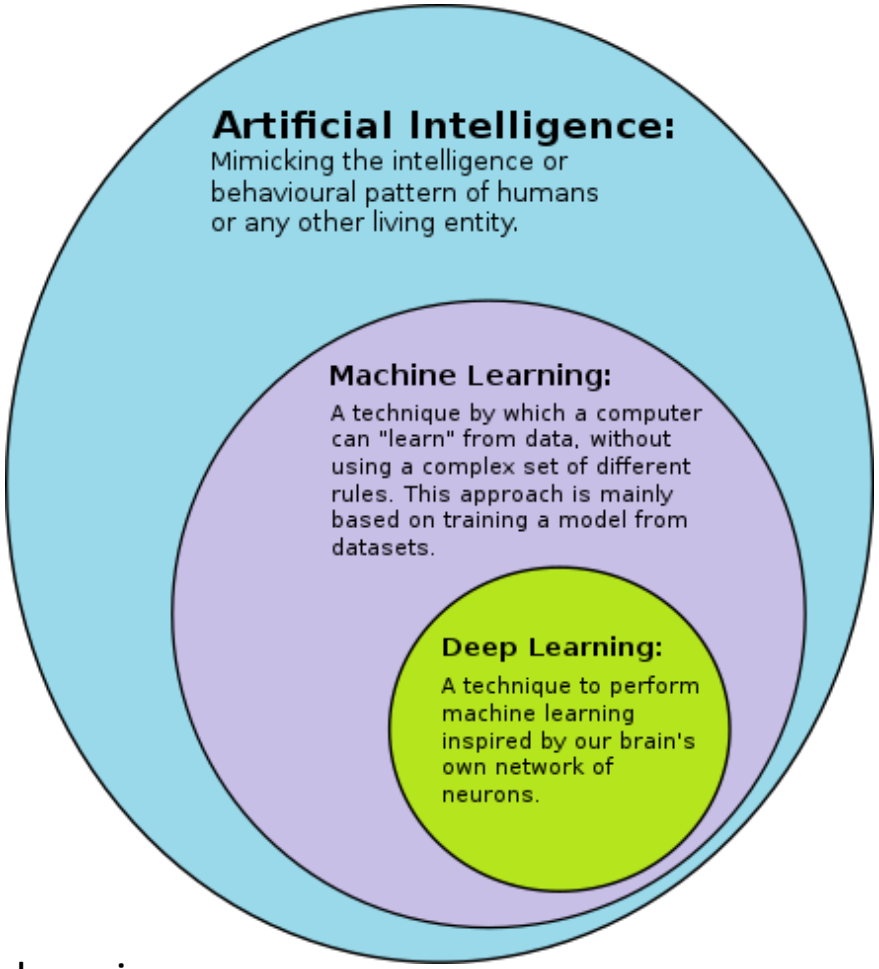


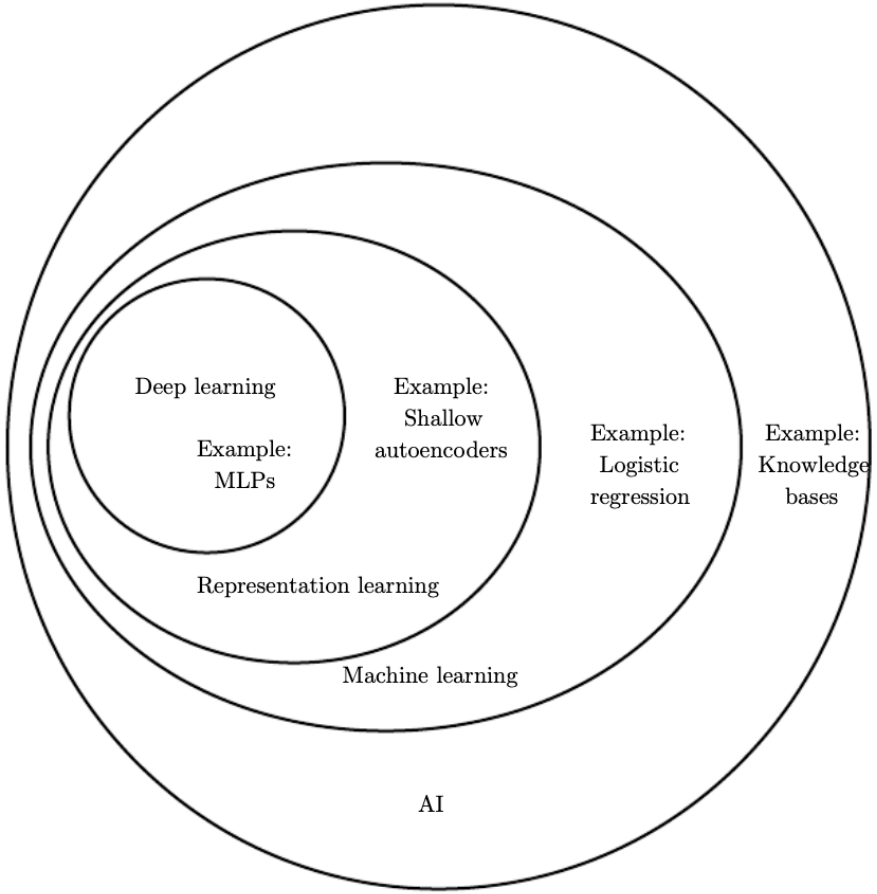





# What is machine learning?

To summarize AI technologies

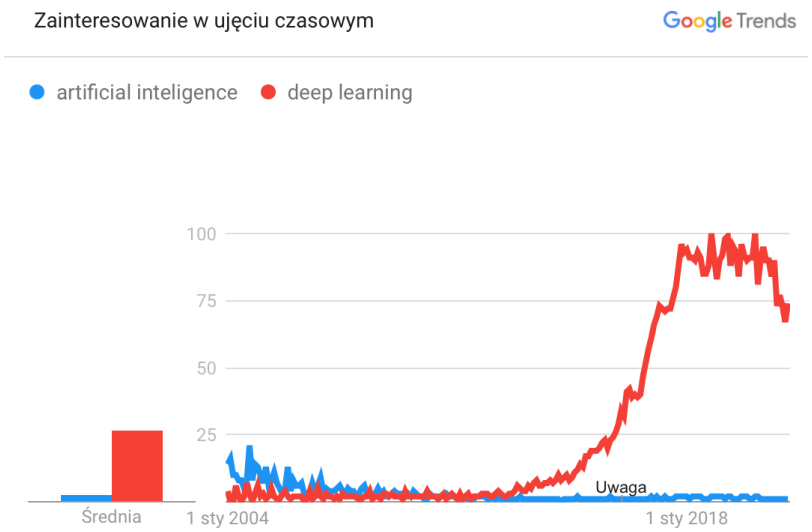



en.wikipedia.org/wiki/Deep_learning

# Why deep learning became so popular?

Why deep learning became a crucial technology in last few years?



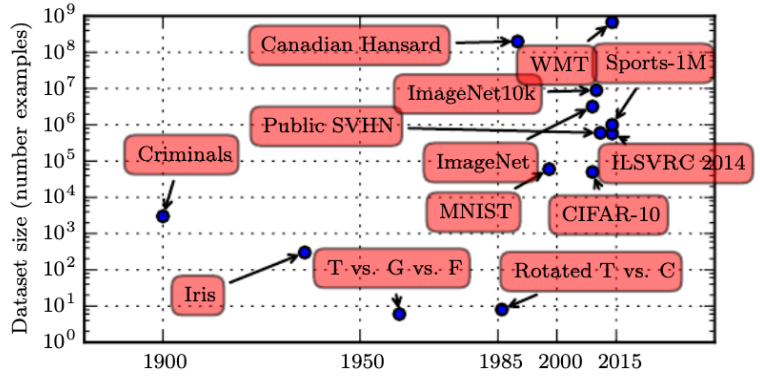




Figure 1.8: Increasing dataset size over time. In the early 1900s, statisticians studied datasets using hundreds or thousands of manually compiled measurements (Garson, 1900;

- Increasing dataset sizes
- a dumb algorithm with lots of data beats a clever algorithm with a modest amount of data

- is connected with increasing size of models

**MNIST Dataset**

https://www.deeplearningbook.org/contents/intro.html

- 60,000 examples, 10 classes
- features: 28x28x1
- http://yann.lecun.com/exdb/mnist/

**ImageNet Dataset**


Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*. [web]

3

image-net.org: 14 M images, 1000 categories

## What is COCO?

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 people with keypoints

cocodataset.org

# Why deep learning became so popular?

How to train such a big models?

- big means that they can't be trained using CPU in a reasonable time

CPU vs GPU

| | Cores | Clock Speed | Memory | Price | Speed |
|---|---|---|---|---|---|
| **CPU** (Intel Core i7-7700k) | 4 (8 threads with hyperthreading) | 4.2 GHz | System RAM | $339 | ~540 GFLOPs FP32 |
| **GPU** (NVIDIA GTX 1080 Ti) | 3584 | 1.6 GHz | 11 GB GDDR5 X | $699 | ~11.4 TFLOPs FP32 |
| **TPU** NVIDIA TITAN V | 5120 CUDA, 640 Tensor | 1.5 GHz | 12GB HBM2 | $2999 | ~14 TFLOPs FP32 ~112 TFLOP FP16 |
| **TPU** Google Cloud TPU | ? | ? | 64 GB HBM | $6.50 per hour | ~180 TFLOP |

**CPU**: Fewer cores, but each core is much faster and much more capable; great at sequential tasks

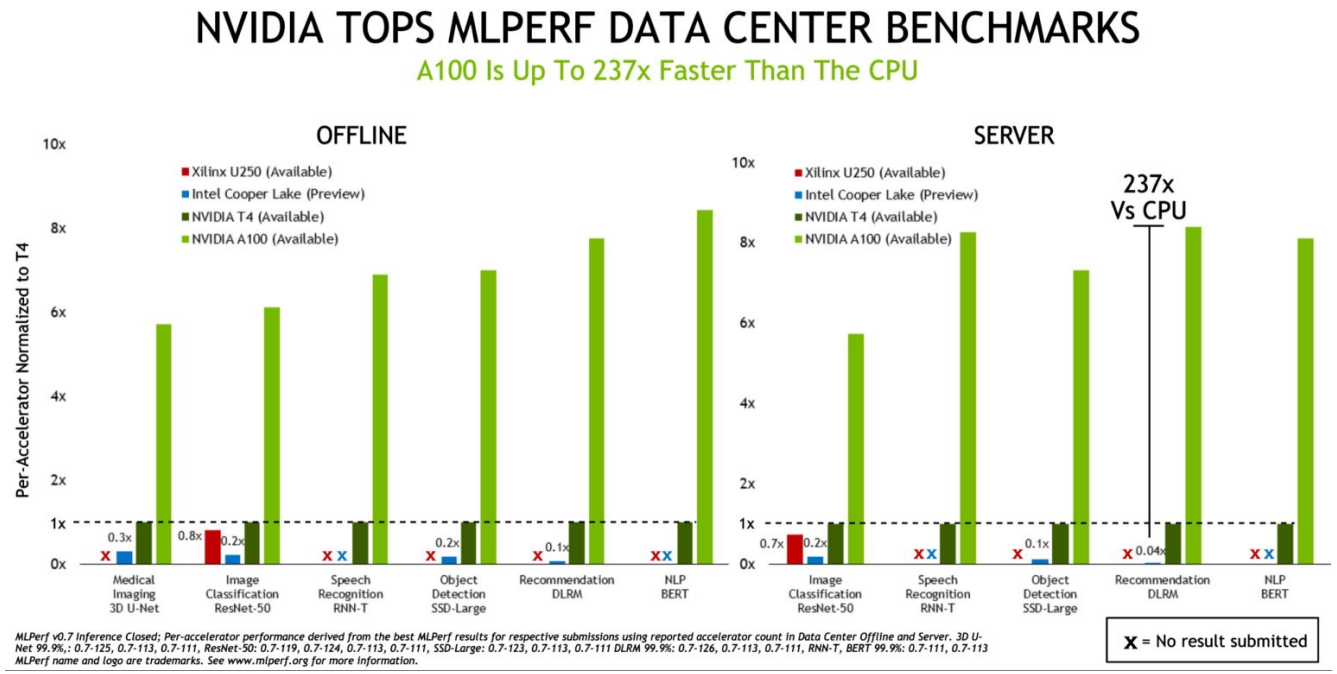
**GPU**: More cores, but each core is much slower and "dumber"; great for parallel tasks

**TPU**: Specialized hardware for deep learning


CPU GPU TPU

ASICs specifically designed for deep learning

GPU vs CPU




source: nvidia.com

https://medium.com/dataseries/cpu-gpu-and-tpu-machine-learning-5033e25b8a0f

# Why deep learning became so popular?

Deep learning models size vs human brain

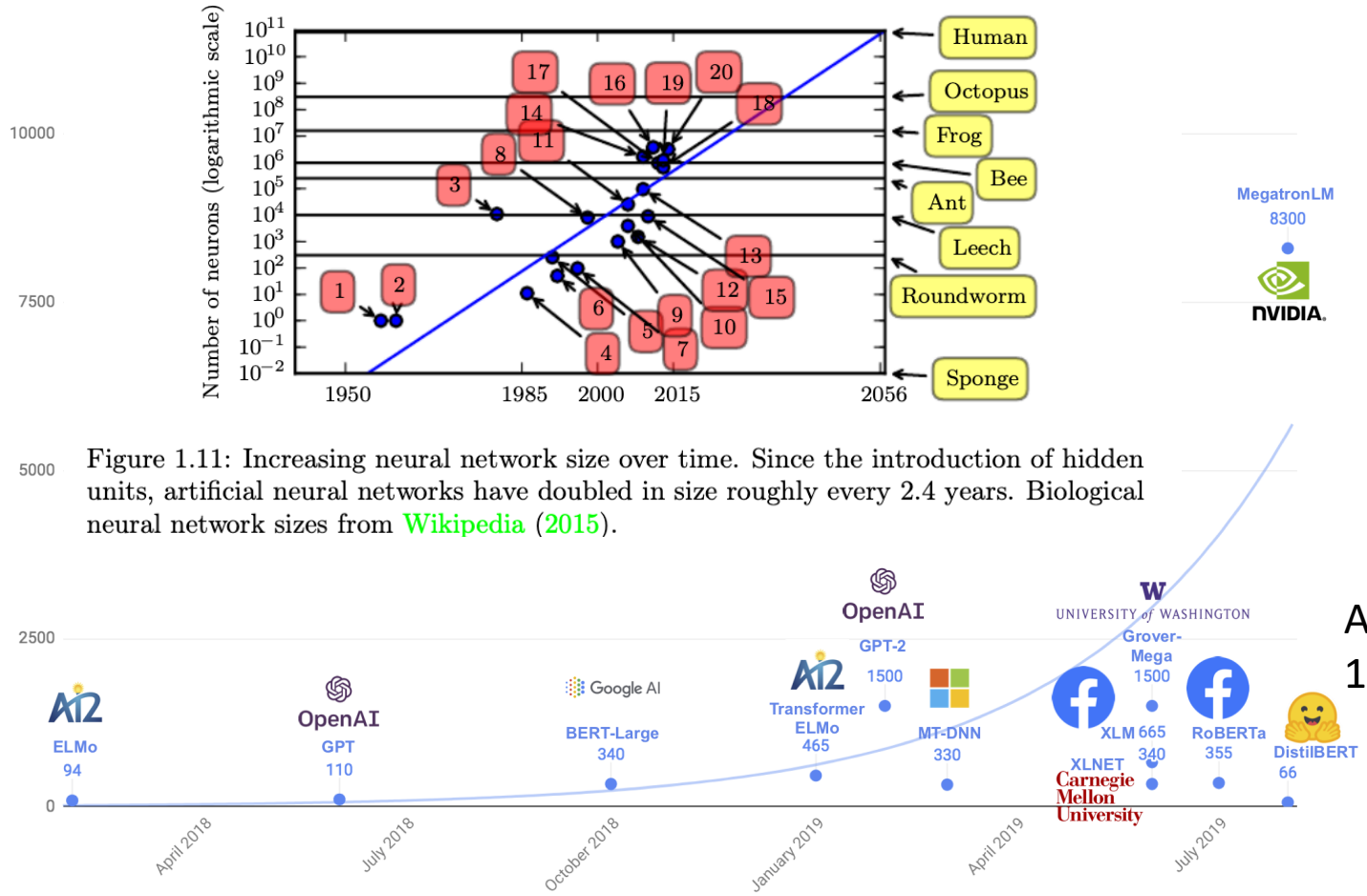






Figure 1.11: Increasing neural network size over time. Since the introduction of hidden units, artificial neural networks have doubled in size roughly every 2.4 years. Biological neural network sizes from Wikipedia (2015).

https://www.deeplearningbook.org/contents/intro.html
https://medium.com/huggingface/distilbert-8cf3380435b5

artificial neural network vs (human) brain
- differences:

- number of connections: human brain has $10^5$ times more connections in total than the biggest ML models
- topology: connections are randomly distributed, artificial networks for layered feedforward structure
- human brain is much faster in generating outputs
- brain structure is much more stable and resilient to errors (artificial networks are susceptible to attacks)
- brains are being in training mode continuously

Actual biggest NLP models:
$10^{11}$ parameters

Image Source: https://timedotcom.files.wordpress.com/2014/05/brain.jpg?w=1100&quality=85

Brain with 1.6 x $10^{10}$ neurons

$10^4$ -$10^5$ connections per neuron

approx. $10^{15}$ connections in total

# Machine learning approaches

**Supervised learning:**

- data is organized in pairs (input, expected output)
- problems: classification, regression
- SVM, decision trees, neural networks
- gaussian discriminant analysis, naive Bayes

"training examples"

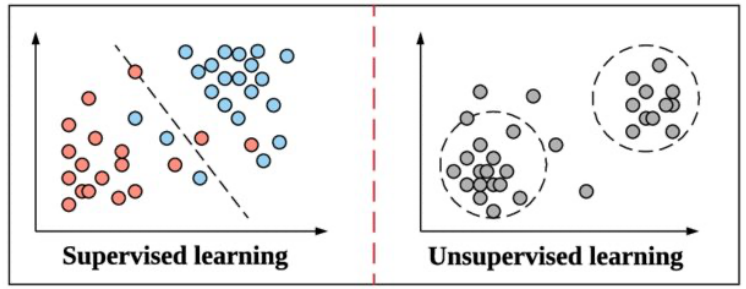
Training set: $\mathcal{D} = \{\langle \mathbf{x}^{[i]}, y^{[i]} \rangle, i = 1, \ldots, n\}$,

Unknown function: $f(\mathbf{x}) = y$

Hypothesis: $h(\mathbf{x}) = \hat{y}$ sometimes

Classification: $h : \mathbb{R}^m \to \mathcal{Y}, \quad \mathcal{Y} = \{1, \ldots, k\}$

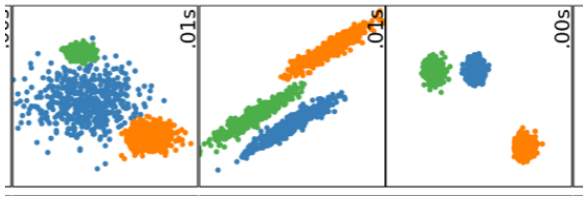
Regression: $h : \mathbb{R}^m \to \mathbb{R}$




www.researchgate.net/project/software-defined-network-3

**Unsupervised learning:**

- in opposite to supervised learning, data is not labeled
- problems: clustering, dimensionality reduction
- PCA, LDA, k-means clustering
- anomaly detection




https://scikit-learn.org

**Reinforcement learning**

Example: Supervised vs Unsupervised

- Having $N$ photos of different animals
- Supervised task (requires labeled data)

> Train an algorithm to recognise given species on a photo.
>
> Output: There is X on a photo.

- Unsupervised task

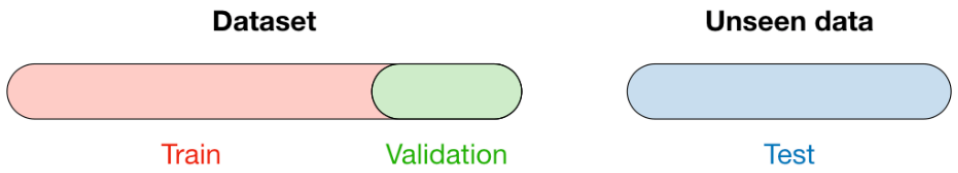
> Train an algorithm to group animals with similar features.
>
> Output: No idea what it is, but it looks similar to these animals.

tomaszgolan.github.io/introduction_to_machine_learning

# Learning scheme

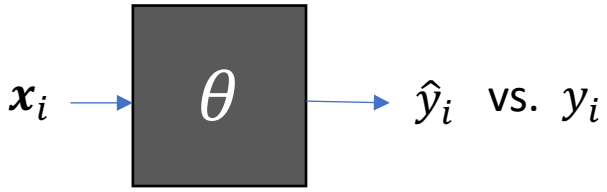
How to perform learning and evaluate model performance

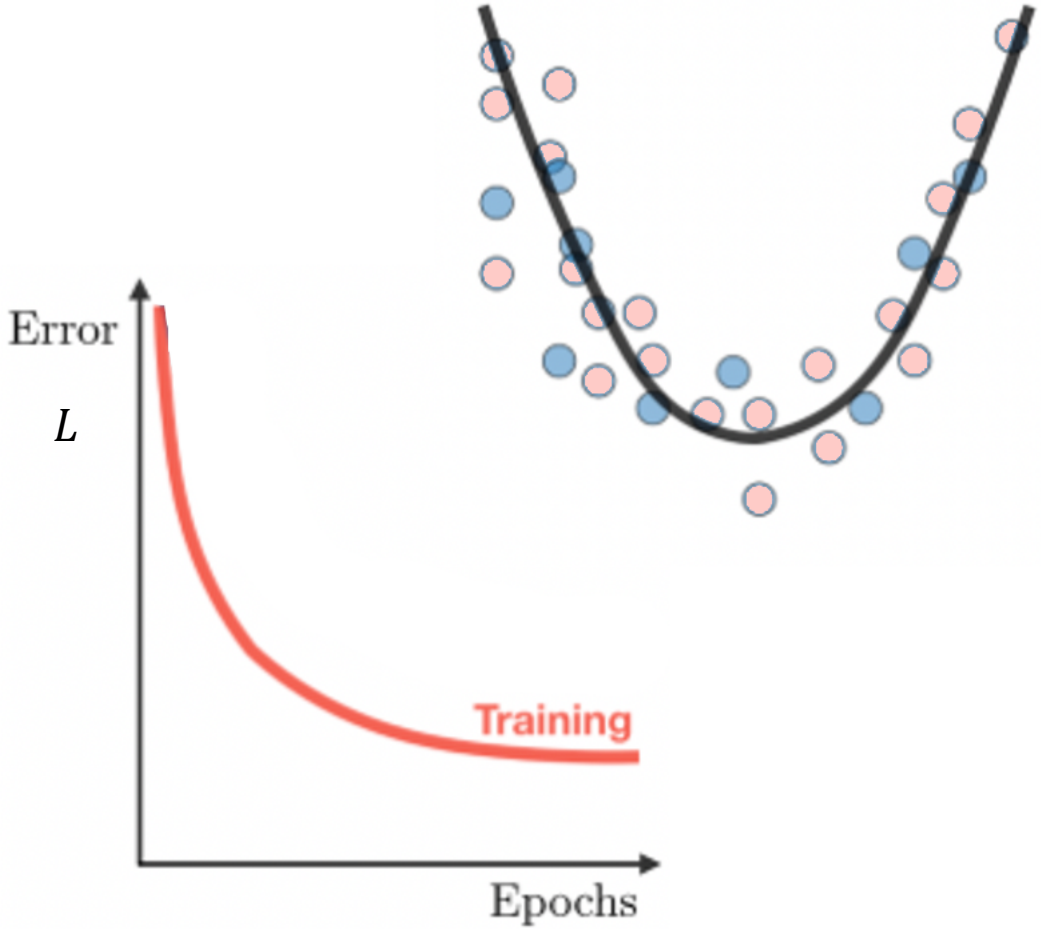
- divide dataset $\{\boldsymbol{x}_i, y_i\}$, into **train** and **test** subsets
  - test should be evaluated on data *unseen* during training




- perform learning process using training subset only
  - during training we minimize *loss function* ("error"):

$$L(\theta, \hat{y}, y) = \frac{1}{n}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$$




- during learning we optimize model to have lowest **training error**,
- we want the generalization error, also called the **test error** to be low as well.

# Bias-variance tradeoff

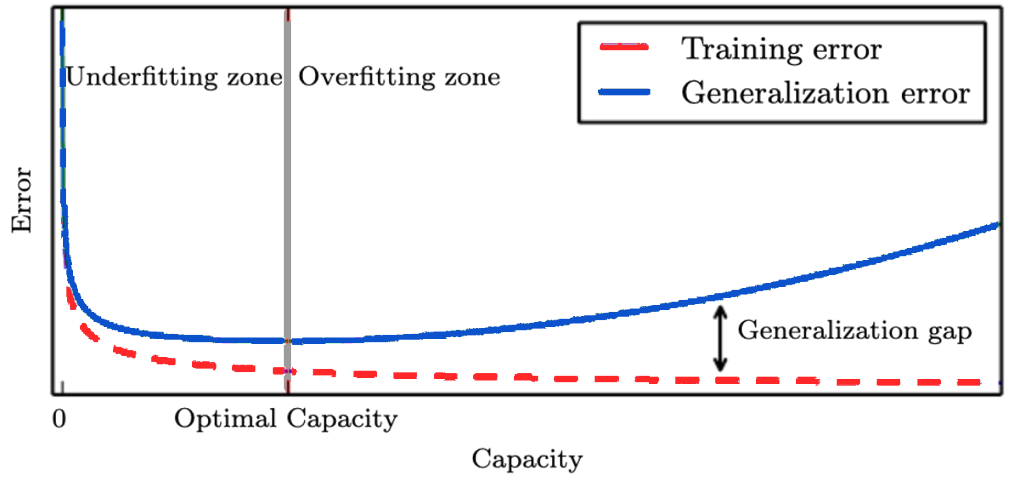
The ability to perform well on previously unobserved inputs ("test samples") is called **generalization**:

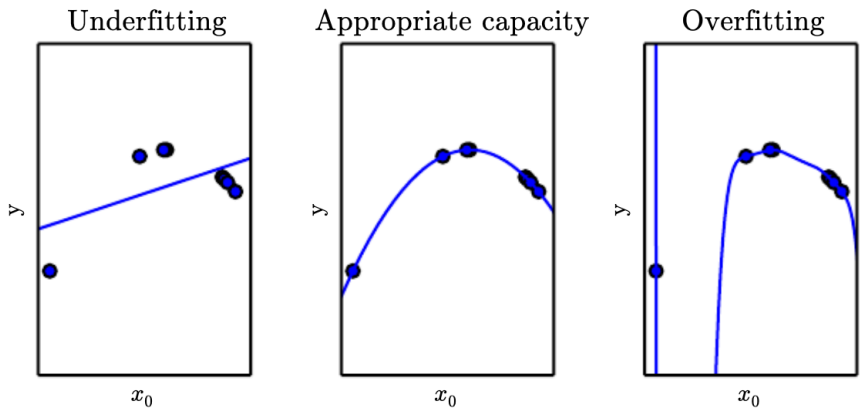
Training and testing error correspond to the two central challenges in machine learning: (model) **underfitting** and **overfitting.**

Underfitting: model is not able to obtain a sufficiently low error on the training set

Overfitting occurs when the distance between the training and test error is too large

We can control whether a model is more likely to overfit or underfit by altering its **capacity**



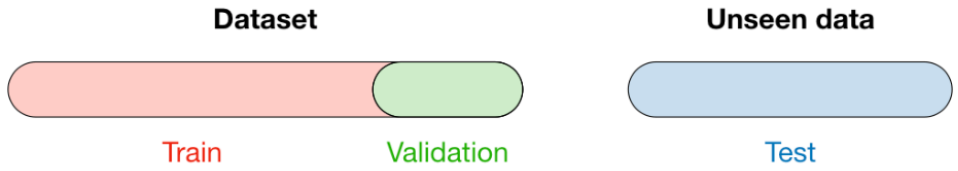

Here capacity means degree of the fitted polynomial:

strongly recommended paper: https://homes.cs.washington.edu/~pedrod/papers/cacm12.pdf

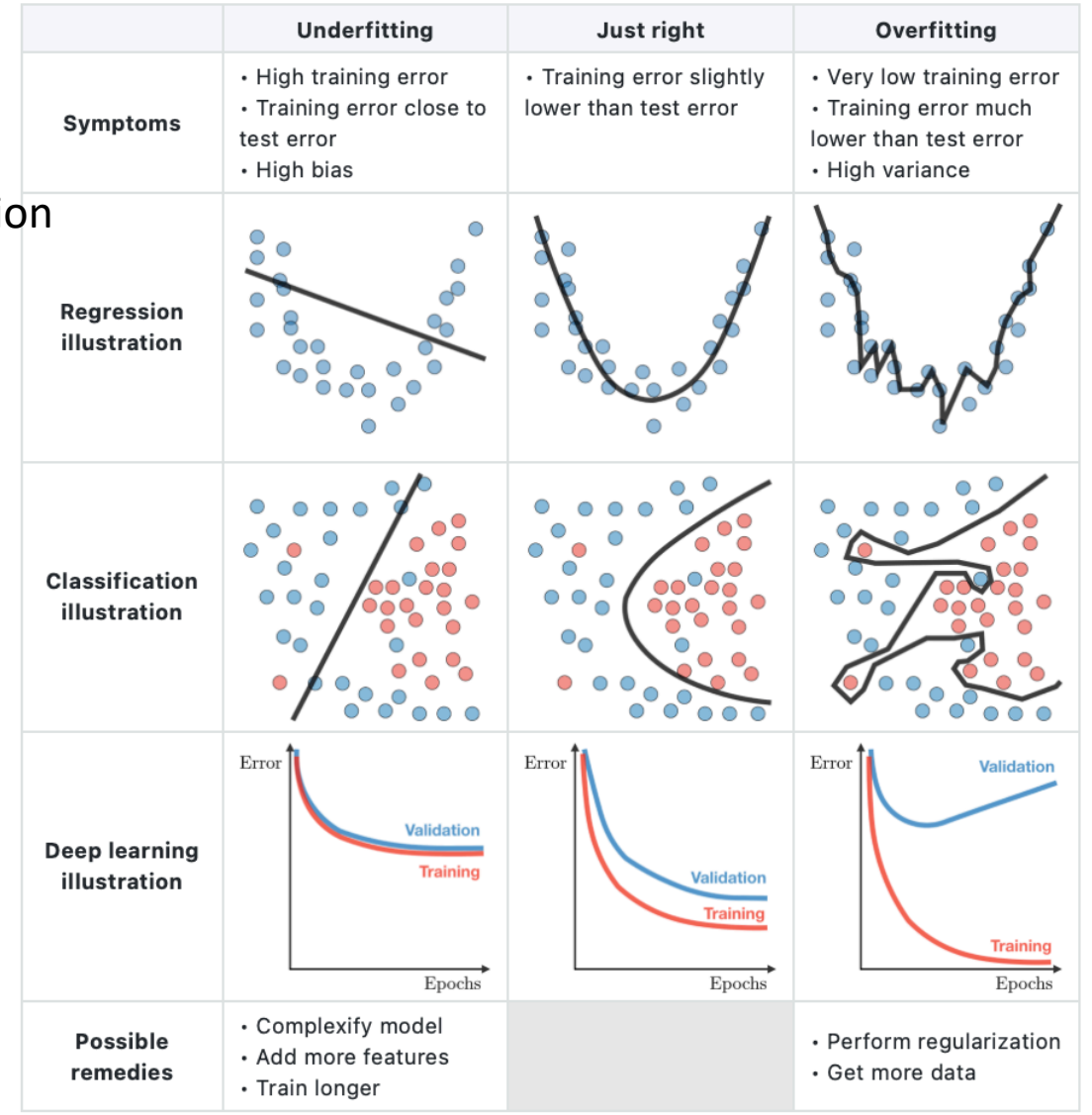
# Avoiding overfitting

How to avoid overfitting

We need another subset (**validation** set) to estimate the generalization error during training, allowing for the model hyperparameters to be updated accordingly.

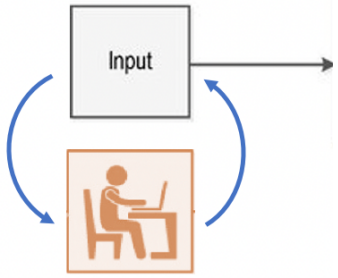



- Do not train to long!
- Augmentation
- Increase dataset size (but new data should be coherent)
  …a dumb algorithm with lots of data beats a clever algorithm with a modest amount of data…



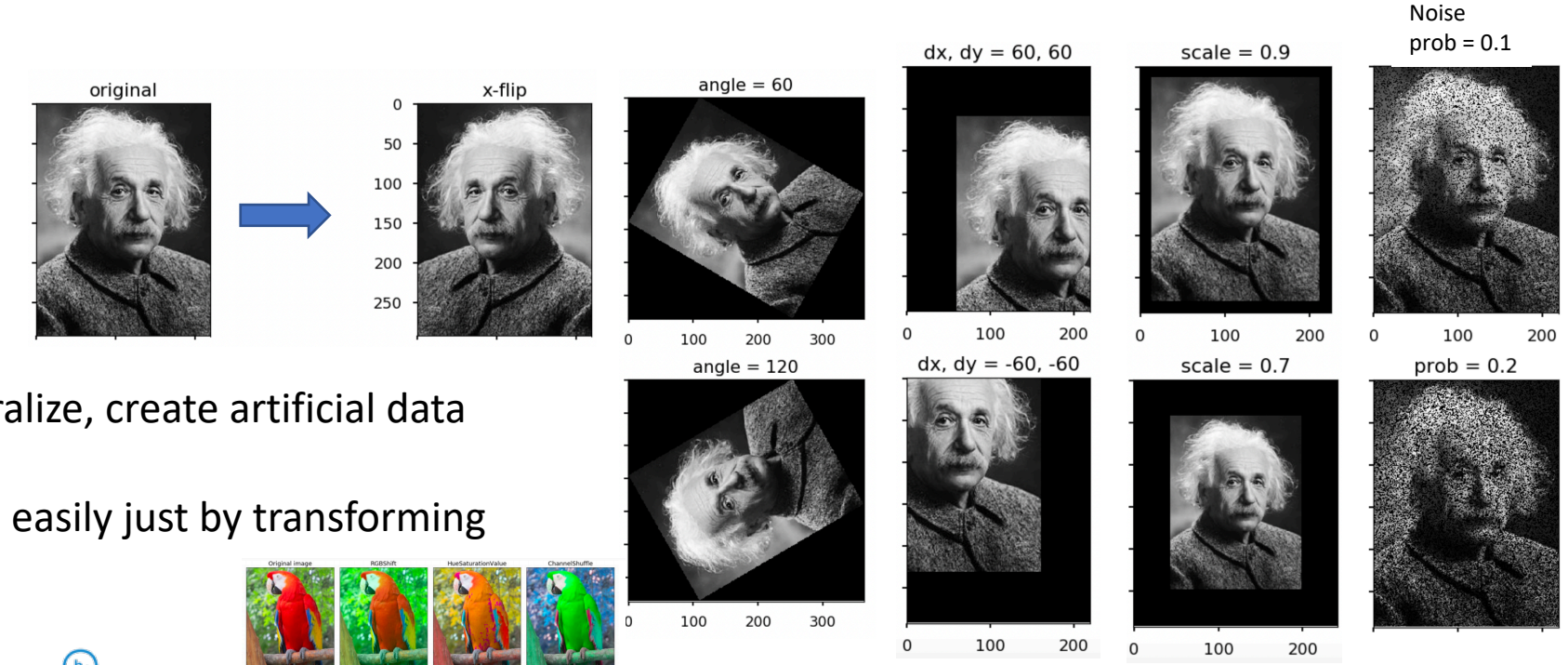

https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks

# Dataset augmentation



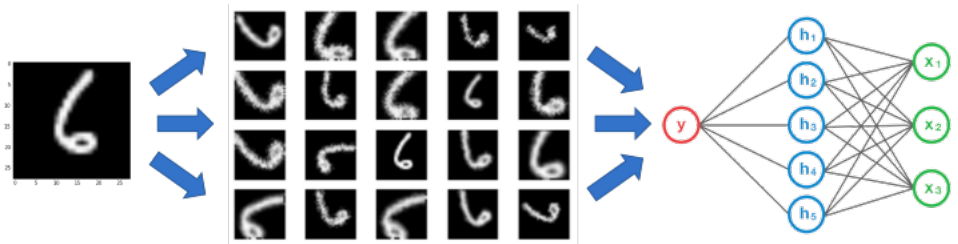

- **dataset augmentation**

Dataset augmentation is a particularly effective technique for a specific classification problem: object recognition

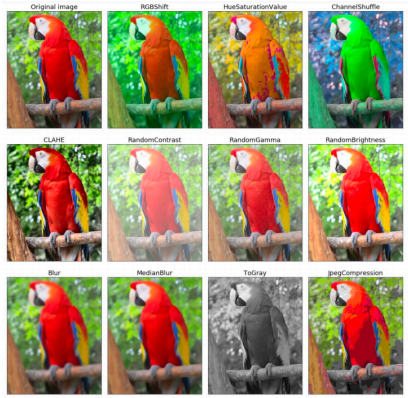



Idea:

- to improve model ability to generalize, create artificial data and add it to the training set
- we can generate new $(X, y)$ pairs easily just by transforming the $X$ inputs in our training set.



nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/



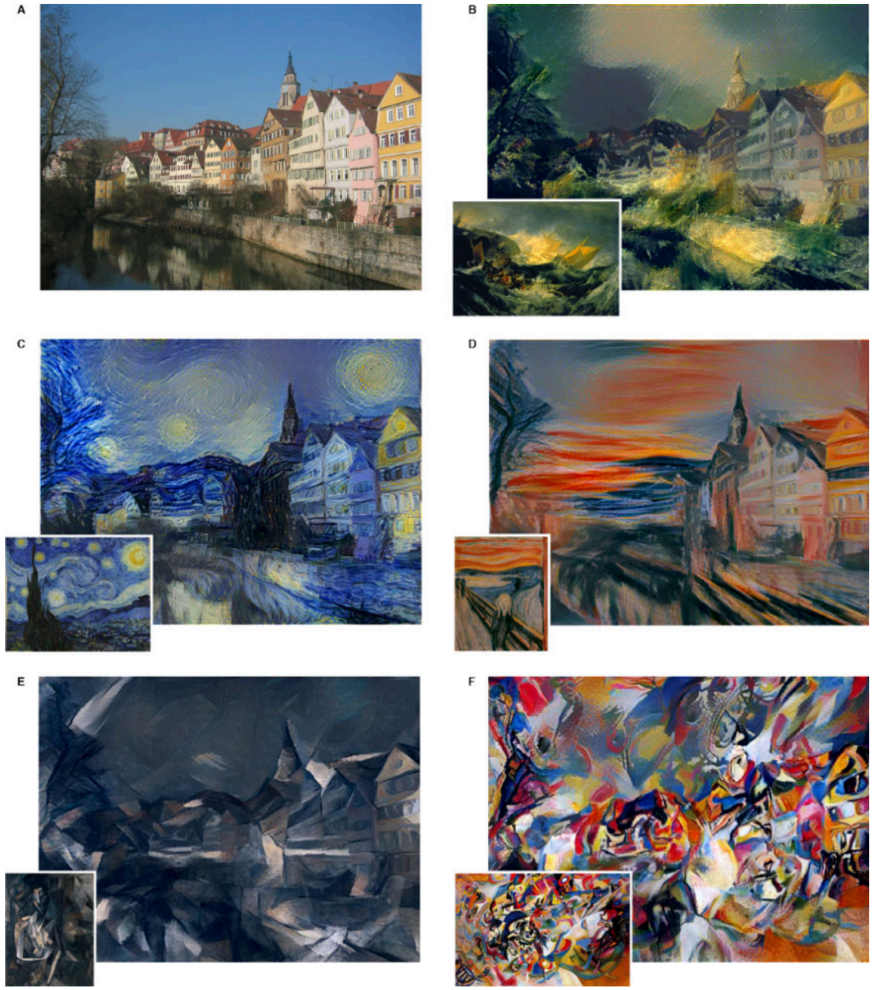
- simple geometrical operations on training images can often greatly improve generalization
- also adding some distractors or noise may help

tomaszgolan.github.io/introduction_to_machine_learning

# ML applications

## ML applications

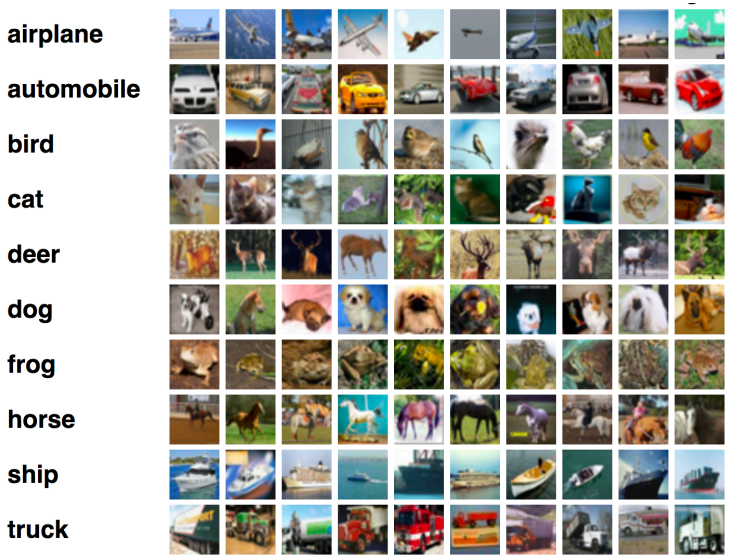
- Image recognition
  - Google Maps - finding licence plates and faces; extracting street names and buildir numbers
  - Facebook - recognising similar faces
- Speech recognition
  - Microsoft - Cortana
  - Apple - Siri
- Natural Language Processing
  - Google Translate - machine translation
  - Next Game of Thrones Book - language modeling
- Misc
  - PayPal - fraud alert
  - Netflix, Amazon - recommendation system




https://arxiv.org/abs/1508.06576

# ML applications: image recognition

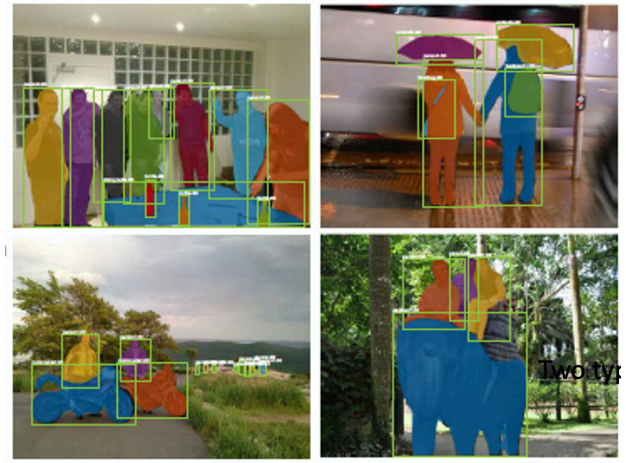
## Image classification



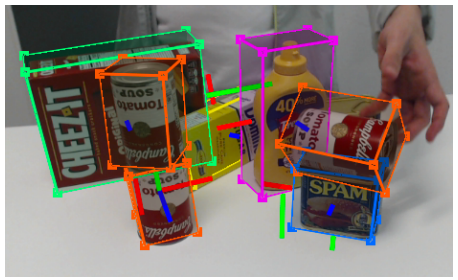

## Object Detection


Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 779-788).

## Object segmentation

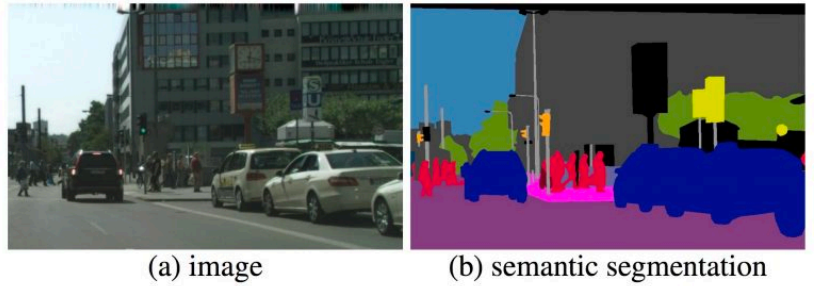


https://arxiv.org/abs/1703.06870

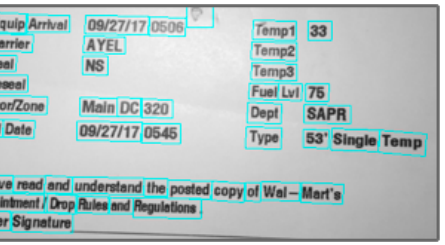



research.nvidia.com

## Types of segmentation:

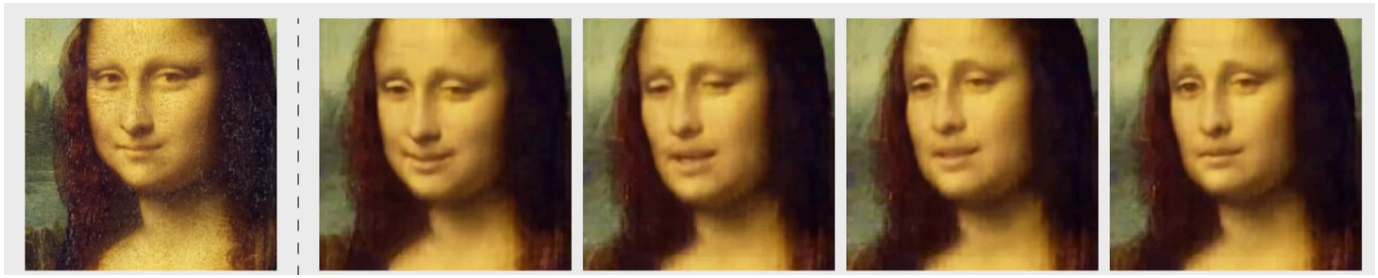



learnopencv.com




- 6D Pose estimation
- Optical character recognition (OCR)
- Face or fingerprint identification

# ML applications: deep fake



Deep ML: potential dangers

Deep fake wars:
Facebook is actively seeking for (ML) tools to detect/eliminate deep fake information

https://miroslawmamczur.pl/

# ML frameworks

ML solution are nowadays extremely easy to implement

- we are able to build quite a complex deep neural network model in a 5 minutes
  (of course learning itself can take 5 days ;-)

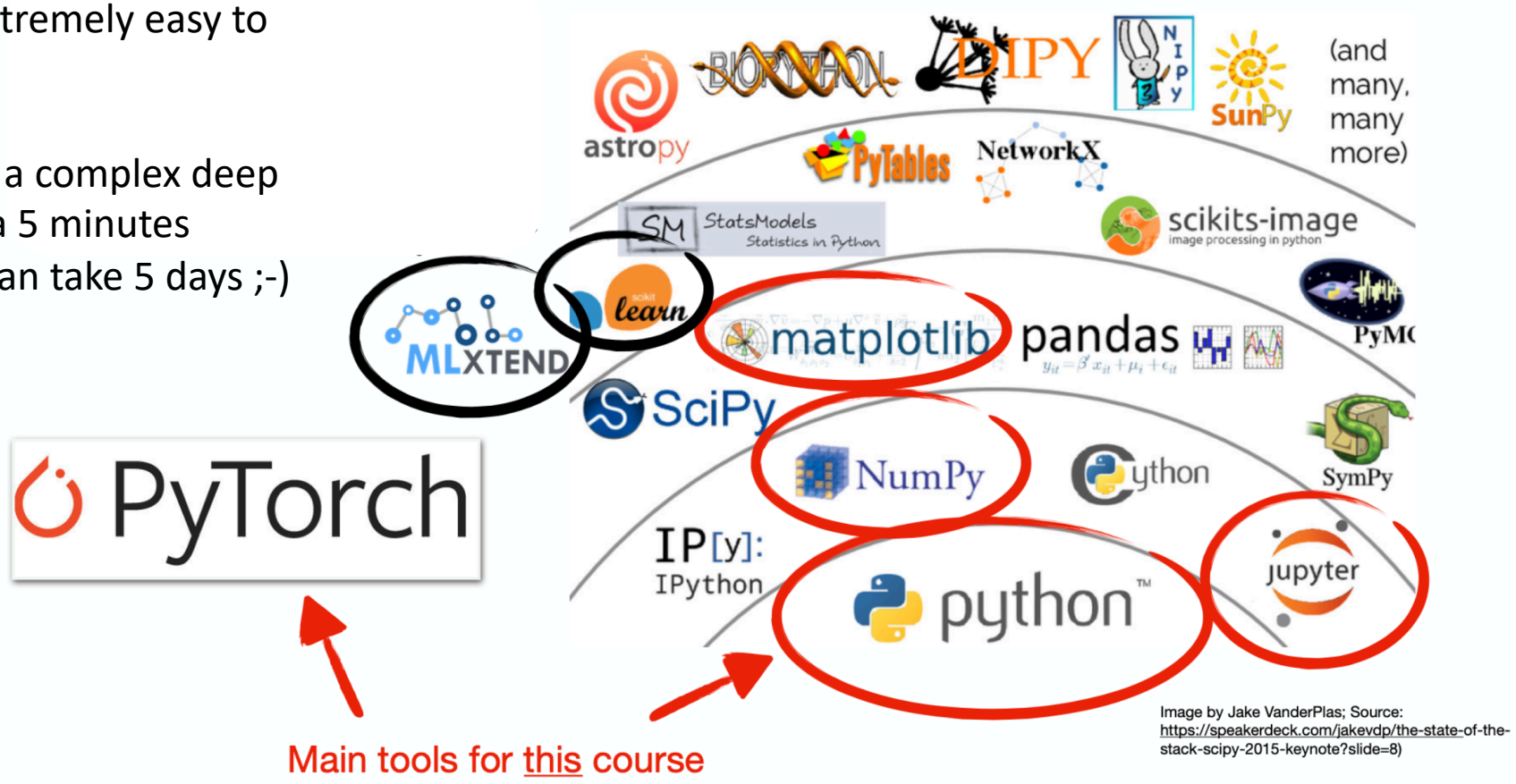



Image by Jake VanderPlas; Source: https://speakerdeck.com/jakevdp/the-state-of-the-stack-scipy-2015-keynote?slide=8)

# Deep neural networks

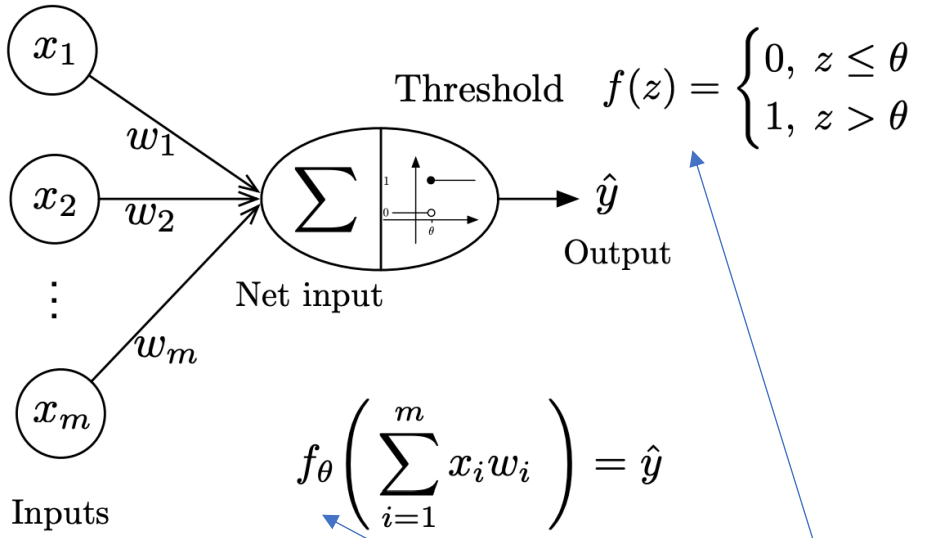
Materials for this Section:

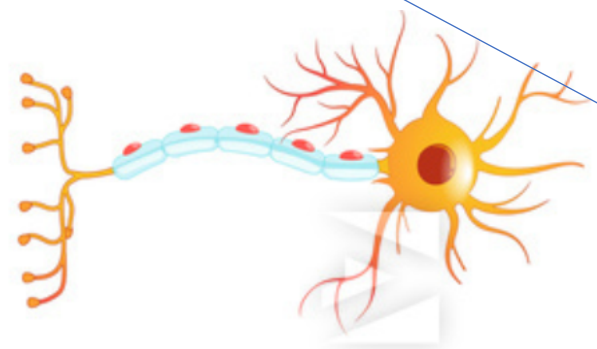
- I. Goodfellow , Y. Bengio, and A. Courville, *Deep Learning, Chapters 6-10:* https://www.deeplearningbook.org
- https://playground.tensorflow.org

# Neural networks

simplest neural network, *perceptron*
-- a computational model of a biological neuron

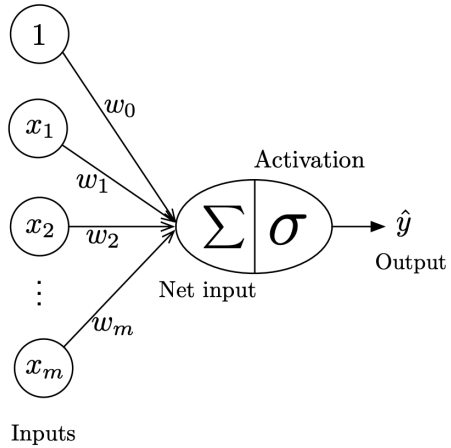
- single input *layer* with multiple input *units* and single output



*activation* with threshold $\theta$

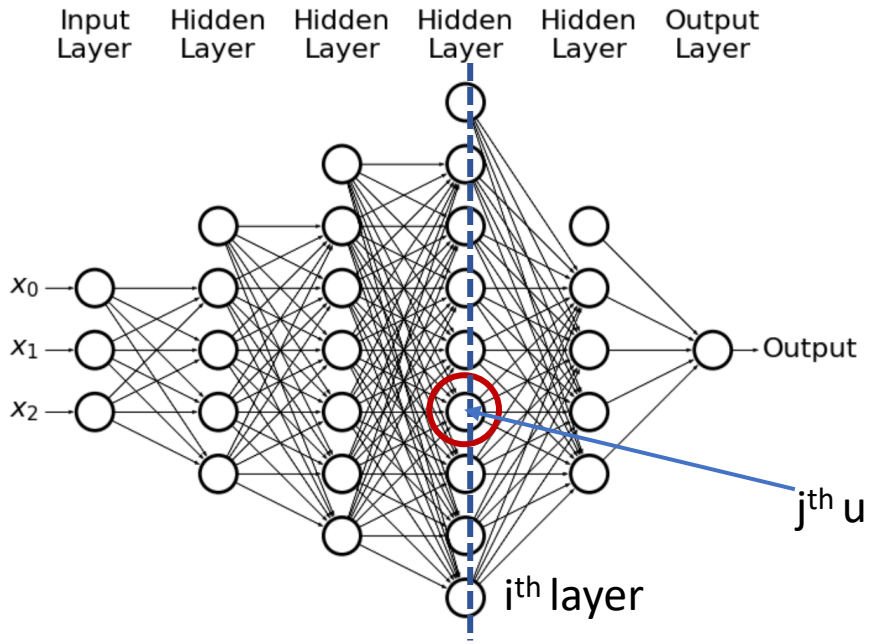
- additional weight = 'bias"



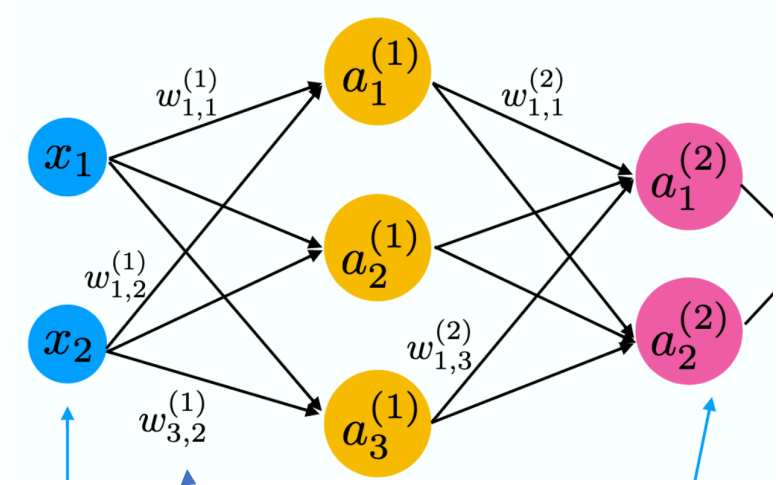

github.com/rasbt/stat453-deep-learning-ss20

# Deep neural networks

- perceptron can not represent XOR function
  -- to do so we need additional *hidden* layers




- fully-connected multi-layer perceptron




- each connection is parametrized by its weight
- each neuron has its own activation function

By noting $i$ the $i^{th}$ layer of the network and $j$ the $j^{th}$ hidden unit of the layer, we have:

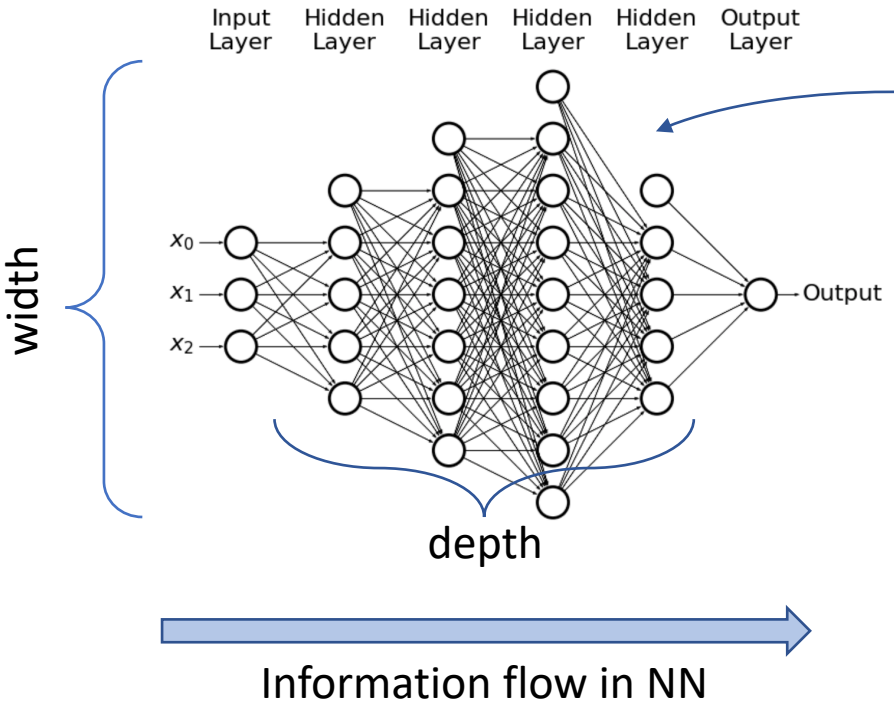
$$x_j^{(i)} = f\left(\sum_k w_k^{(i)} x_k^{(i-1)}\right)$$

where we note $w$, $b$, $z$ the weight, bias and output respectively.

- network is called **deep** when in contains hidden layers
- all weights are fitted during *training*

github.com/rasbt/stat453-deep-learning-ss20

# Deep neural networks




- fully-connected multi-layer perceptron
- however, we could have only a part of connections

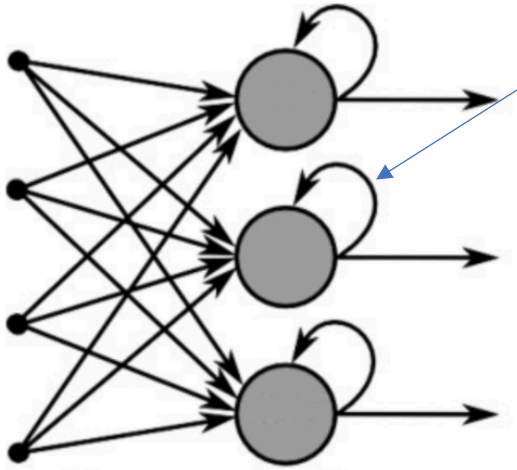
- Deep **feedforward** networks because information flows through the the subsequent layers starting from the input layer and finally to the output layers

  -> network is called deep when contains hidden layers
  -> the dimensionality of hidden layers determines the *width* of the model
  -> number of deep layers gives the *depth* of the model

# Deep neural networks

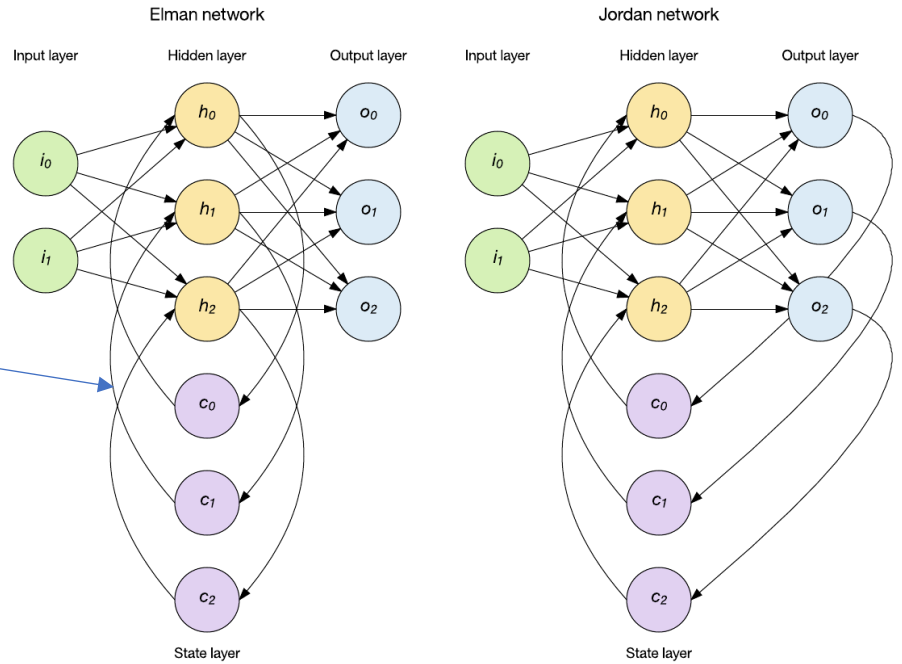
Information flow in NN

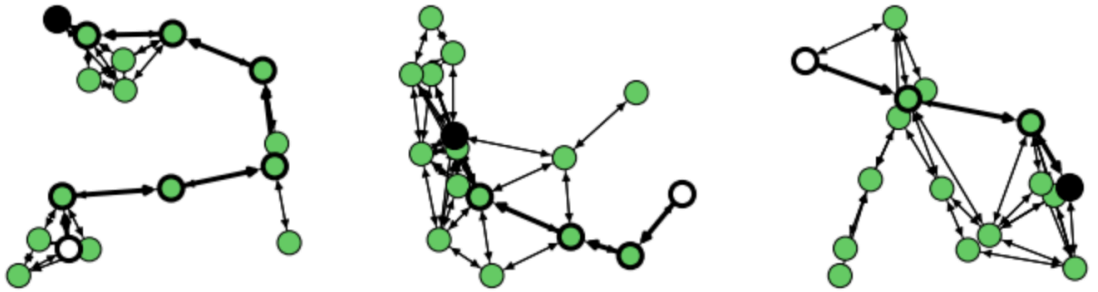
- When feedforward neural networks are extended to include feedback connections, they are called **recurrent** neural networks



Recurrent Neural Network

feedback connections enables information flow in reverse direction



- **graph** neural networks
  -> no distinguished direction of information flow through the network

developer.ibm.com/articles/cc-cognitive-recurrent-neural-networks/

# Deep neural networks: training

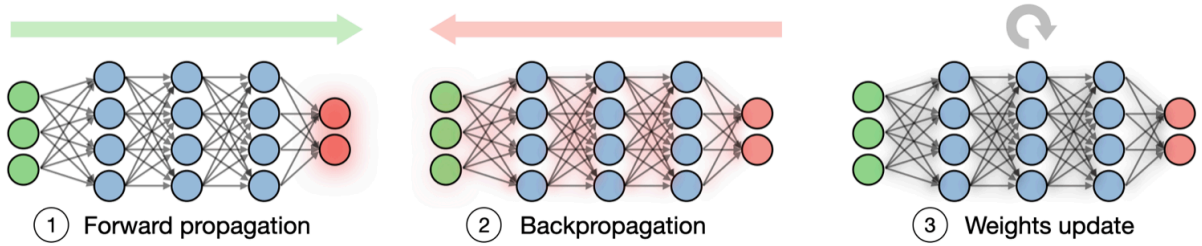
Training NN is just optimization problem:

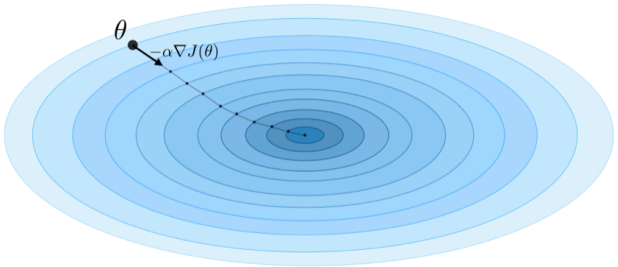
- The task is to minimize loss function in parameters (weights) space

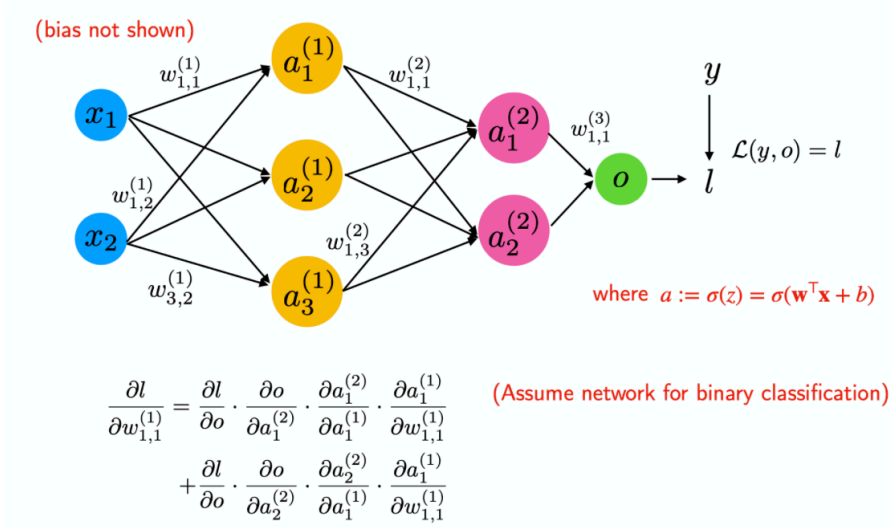
$$w \longleftarrow w - \alpha \frac{\partial L(z, y)}{\partial w}$$

□ **Updating weights** — In a neural network, weights are updated as follows:

- Step 1: Take a batch of training data and perform forward propagation to compute the loss.
- Step 2: Backpropagate the loss to get the gradient of the loss with respect to each weight.
- Step 3: Use the gradients to update the weights of the network.



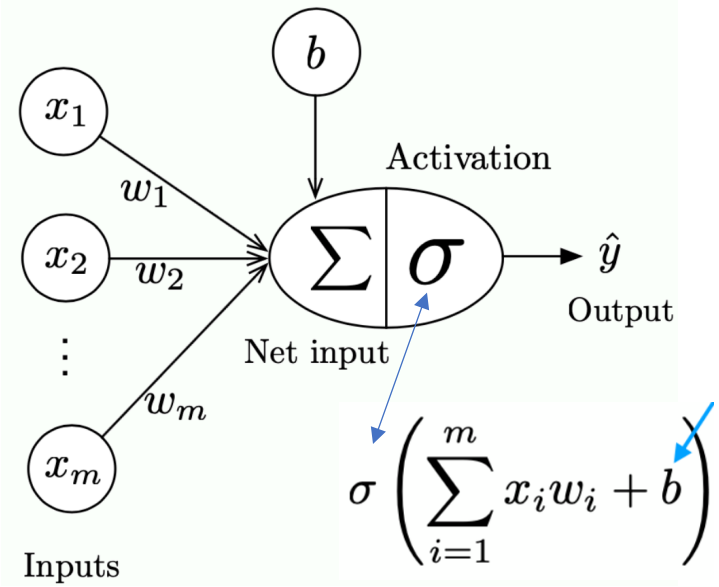




backpropagation utilizes *chain rule*, and reuses partial derivatives calculated before.

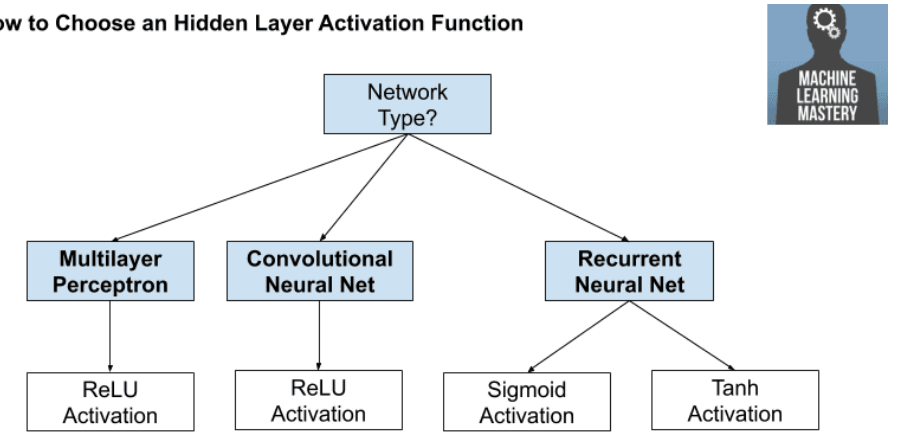
good review: brilliant.org/wiki/backpropagation/

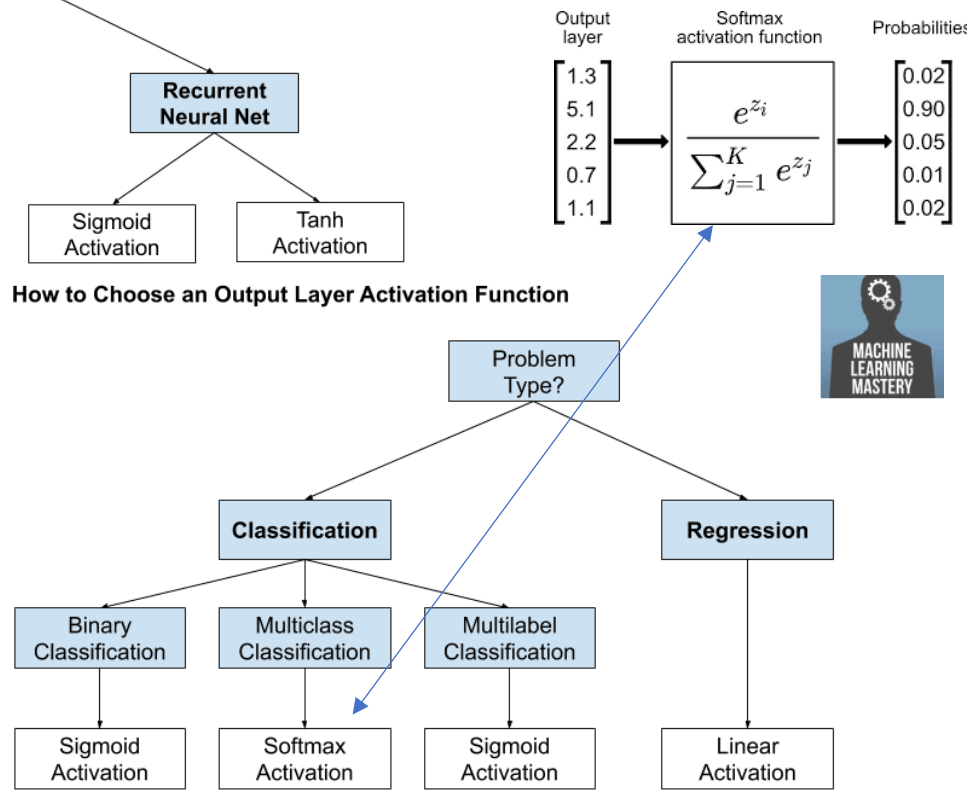
stanford.edu/~shervine/teaching/cs-230/cheatsheet-deep-learning-tips-and-tricks

# Deep neural networks: activation

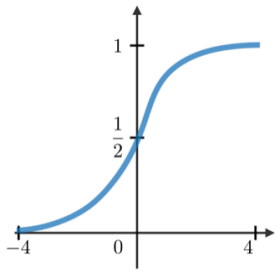
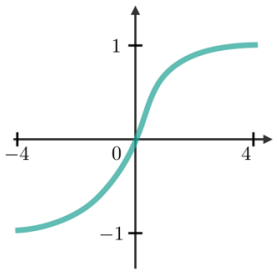
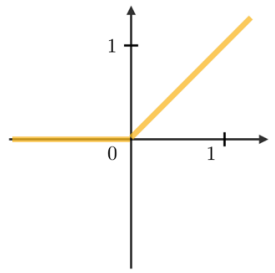
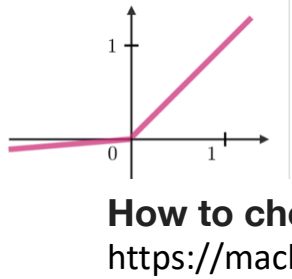



We usually chose activation for hidden units and output layer separately:



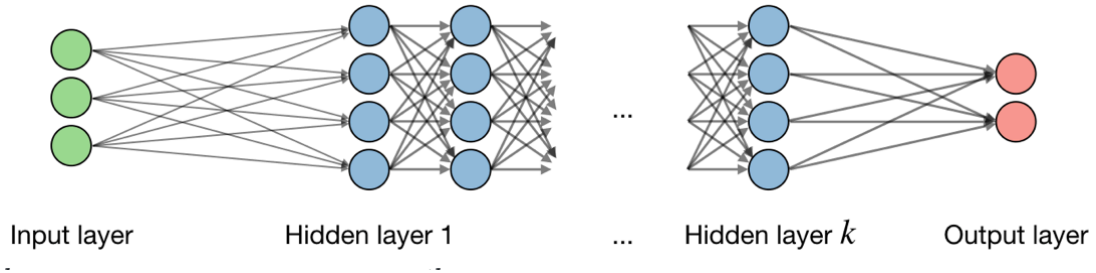

□ **Activation function** — Activation functions are used at the end of a hidden unit to introduce non-linear complexities to the model. Here are the most common ones:

| Sigmoid | Tanh | ReLU | Leaky ReLU |
|---|---|---|---|
| $g(z) = \frac{1}{1+e^{-z}}$ | $g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ | $g(z) = \max(0, z)$ | $g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$ |
|  |  |  |  |

**How to chose Activation for Hidden and output Layers:**
https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

stanford.edu/~shervine/teaching/cs-229/cheatsheet-deep-learning

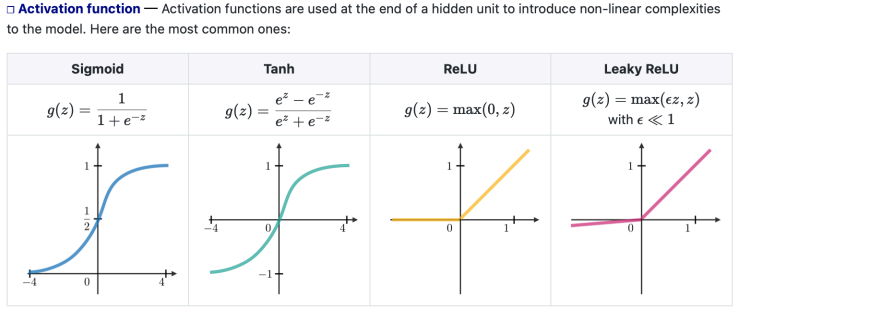
# Deep neural networks: hyperparameters

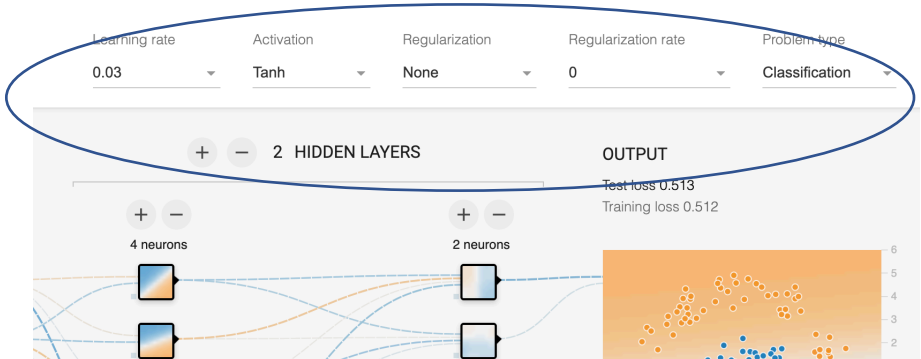



Note, that NN has much more hyperparameters than other ML models:

- number of hidden layers
- number of neurons in layer
- activation functions
- learning algorithm settings: SGD variation, learning rate, number of epochs, momentum, etc.
- batch size in mini-batch gradient descent
- regularization method and its parameters

It is crucial to use cross-validation (at least single validation subset) for hyperparameter tuning.




- Let's play with hyperparameters: **https://playground.tensorflow.org**

# Deep Learning Workshops

Next meeting

- Wednesday, October 12th, 17:00 - 19:00 (theory)
- **Monday, October 17th, 17:00 - 19:00 (theory + applications),**
  - **CNN, GANs**
  - **how to train simple neural network: MLP, CNN, GAN**
- Projects: subsequent Wednesdays (?)
  - in-person: A1 wyb. Wyspiańskiego 27, room 219, 10:00-12:00 or 14:00-16:00
  - or online: zoom