AI Research – Criteo AI Lab

An Introduction to Machine Learning

Liva Ralaivola, Director of AI Research,
Criteo AI Lab
@CriteoAILab, @LivaRalaivola

INRAE Jan. 31st, 2022

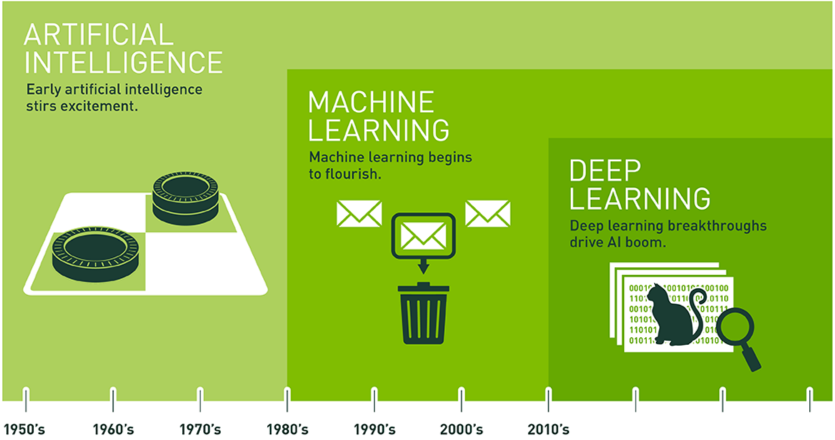
CRITEO

Outline

Outline

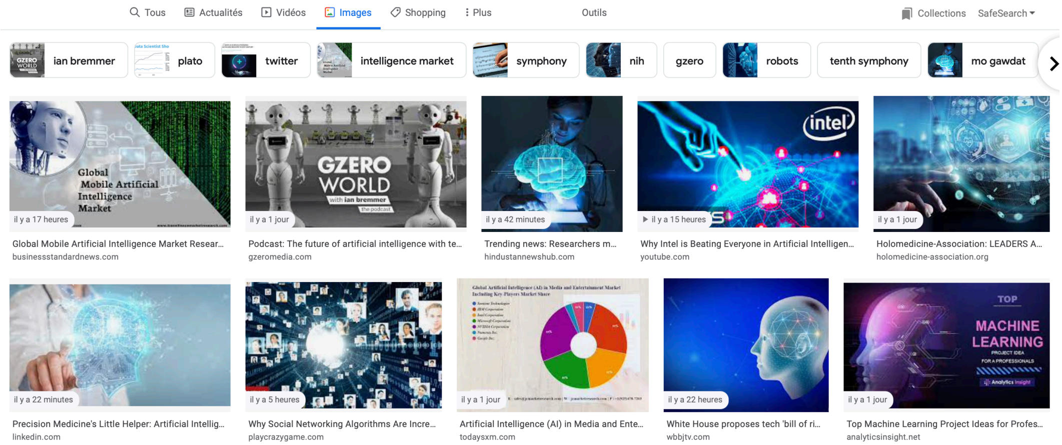
AI, Machine Learning, Deep Learning

Today: data, software, computing power



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

In the news... as of Oct. 10th, 2021



Annotation/Image decoding

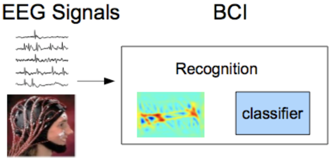
(from Farabet et al, 2013)

P300 Speller

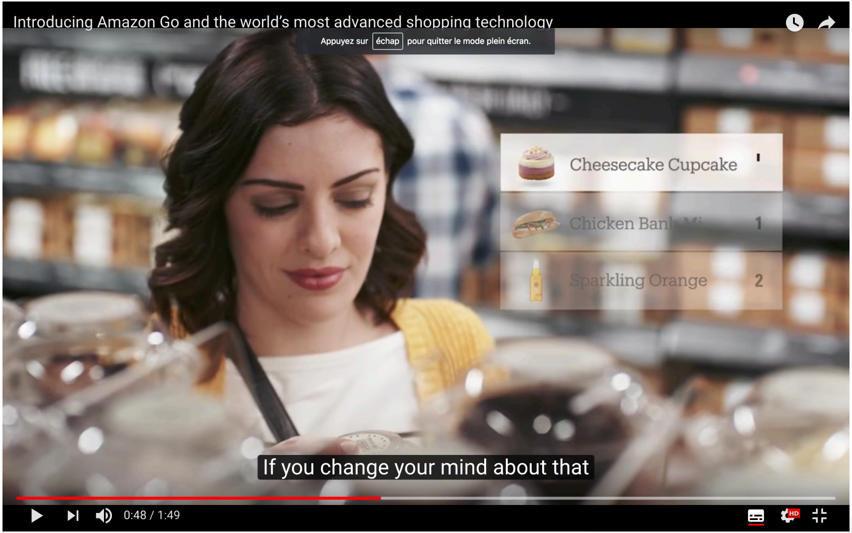
Vintage P300 Speller

(from Breaking bad)

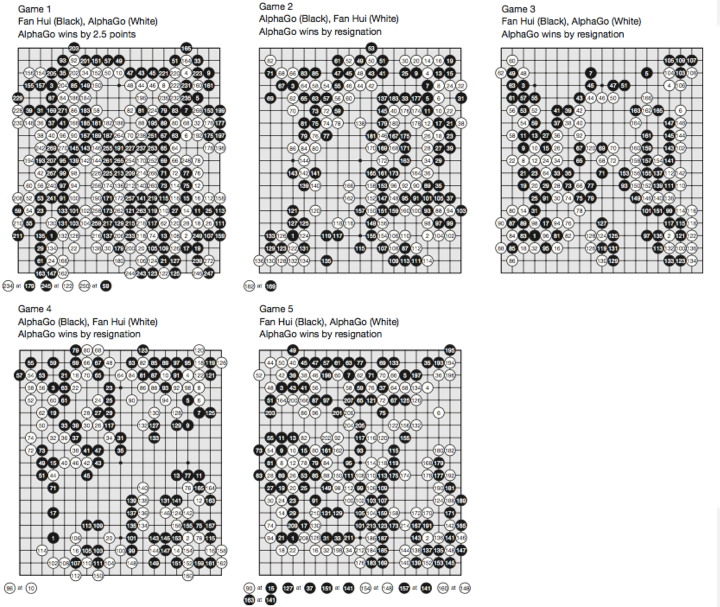
Modern P300 Speller (pictures from A. Rakotomamonjy, video from Robo Doc)



ML-cashing Amazon shops

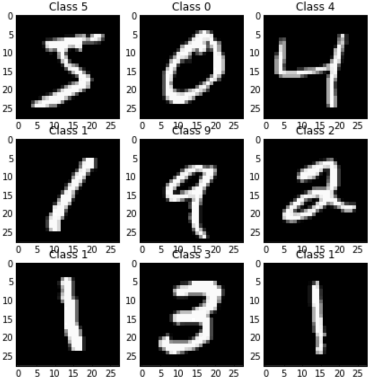


AlphaGo (Silver et al. 2016)

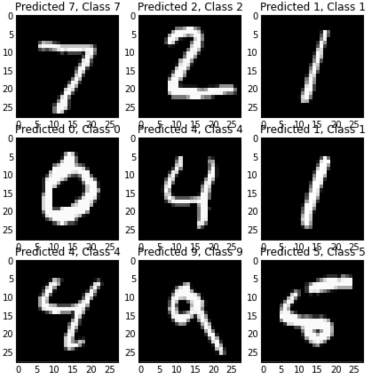


Core goal in supervised learning: generalization

Training set



Test set

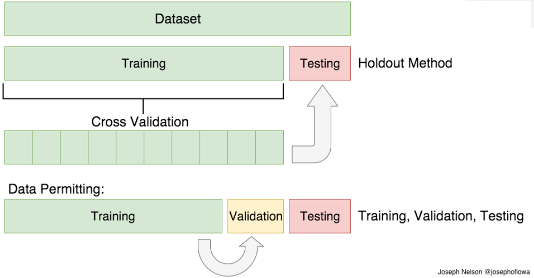


(from Keras Mnist Tutorial)

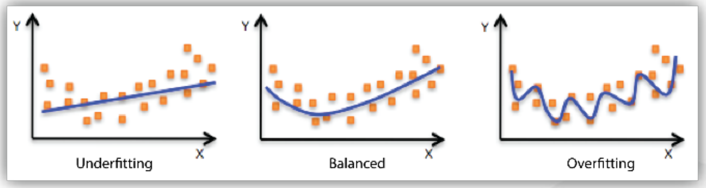
Generalization: from the training set to beyond

Design algorithms capable from pairs (measure, target), to create a predictors which, given a measure, estimates the corresponding target

Core goal in supervised learning: generalization... in practice



(from Train/Test Split and Cross Validation in Python)

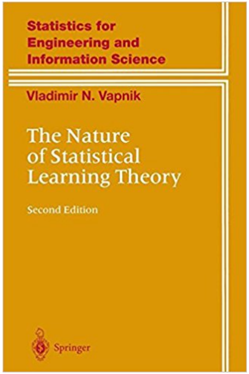


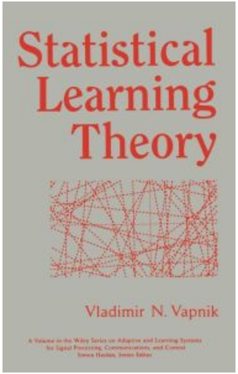
(from Amazon AWS)

Outline

Positioning

V. Vapnik sets, at the end of the 70's, the mathematical basis of machine/statistical learning, at the intersection of computer science, statistics, and optimization

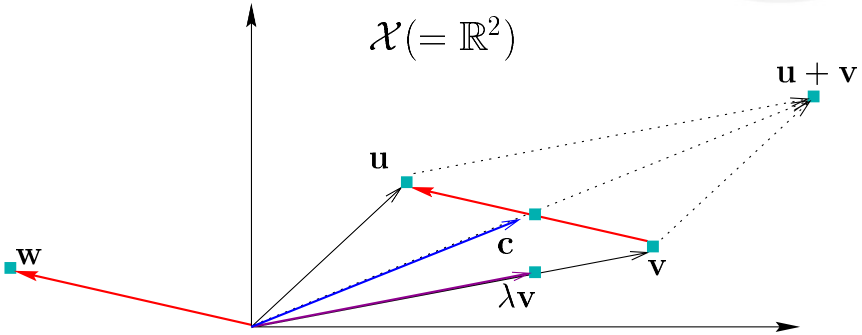




"ML is the study of computer algorithms that improve automatically through experience."

T. Mitchell, 1997

Warm-up: a first handcrafted classifier



- $\mathbf{u}, \mathbf{v}, \mathbf{w}, \mathbf{c}$ are vectors
- $\mathbf{w} = \mathbf{u} - \mathbf{v}$ (red arrows)
- $\mathbf{c} = \frac{1}{2}(\mathbf{u} + \mathbf{v})$
- Here: $0 < \lambda < 1$

Warm-up: a first handcrafted classifier

Inner product $\langle \cdot, \cdot \rangle : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

- symmetric: $\langle \mathbf{u}, \mathbf{v} \rangle = \langle \mathbf{v}, \mathbf{u} \rangle$
- bilinear: $\langle \lambda \mathbf{u}_1 + \gamma \mathbf{u}_2, \mathbf{v} \rangle = \lambda \langle \mathbf{u}_1, \mathbf{v} \rangle + \gamma \langle \mathbf{u}_2, \mathbf{v} \rangle$
- positive: $\langle \mathbf{u}, \mathbf{u} \rangle \geq 0$
- definite: $\langle \mathbf{u}, \mathbf{u} \rangle = 0 \Rightarrow \mathbf{u} = 0$

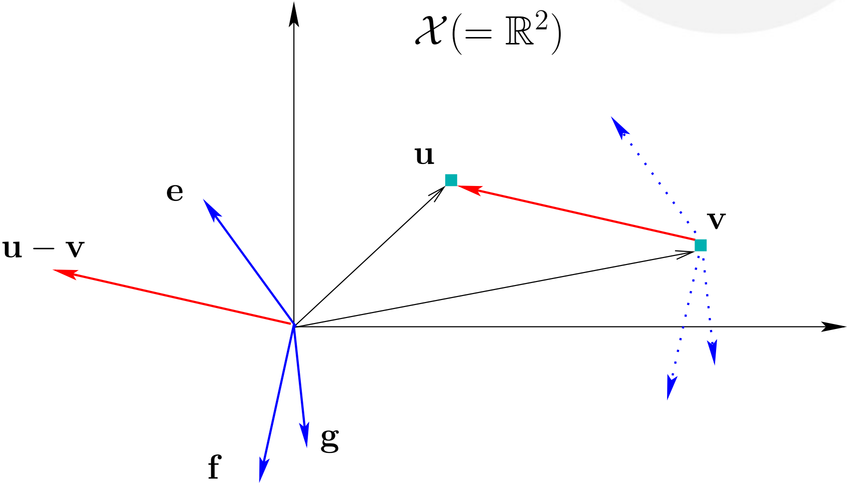
Inner product

- provides $\mathcal{X}$ with a structure
- can be viewed as a 'similarity'
- defines a norm $\| \cdot \|$ on $\mathcal{X}$: $\|\mathbf{u}\| = \sqrt{\langle \mathbf{u}, \mathbf{u} \rangle}$

In $\mathbb{R}^2$

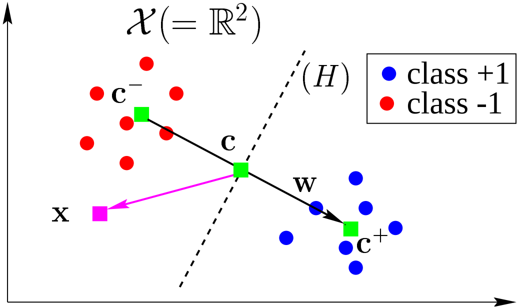
- $\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$, $\mathbf{v} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}$: $\langle \mathbf{u}, \mathbf{v} \rangle = u_1 v_1 + u_2 v_2$

Warm-up: a first handcrafted classifier



- $\langle \mathbf{u}-\mathbf{v}, \mathbf{e}\rangle > 0$: $\mathbf{u}-\mathbf{v}$ and $\mathbf{e}$ point to the 'same direction'
- $\langle \mathbf{u}-\mathbf{v}, \mathbf{f}\rangle = 0$: $\mathbf{u}-\mathbf{v}$ and $\mathbf{f}$ are orthogonal
- $\langle \mathbf{u}-\mathbf{v}, \mathbf{g}\rangle < 0$: $\mathbf{u}-\mathbf{v}$ and $\mathbf{g}$ point to 'opposite directions'

Warm-up: a first handcrafted classifier



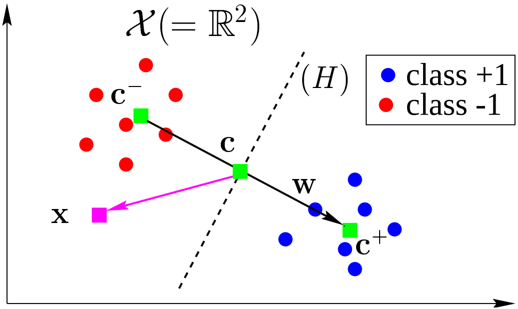
- $\mathbf{c}^+ = \frac{1}{n^+} \sum_{\{i:y_i=+1\}} \mathbf{x}_i$
- $\mathbf{c}^- = \frac{1}{n^-} \sum_{\{i:y_i=-1\}} \mathbf{x}_i$
- $\mathbf{c} = \frac{1}{2}(c^+ + c^-)$
- $\mathbf{w} = c^+ - c^-$

Decision function

Classify points $\mathbf{x}$ according to which of the two class means $\mathbf{c}^+$ or $\mathbf{c}^-$ is closer:

- for $\mathbf{x} \in \mathcal{X}$, it is sufficient to take the sign of the inner product between $\mathbf{w}$ and $\mathbf{x} - \mathbf{c}$
- if $h(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} - \mathbf{c} \rangle$, we have the classifier $f(\mathbf{x}) = \text{sign}(h(\mathbf{x}))$
- the (dotted) hyperplane (H), of normal vector $\mathbf{w}$, is the decision surface

Warm-up: a first handcrafted classifier



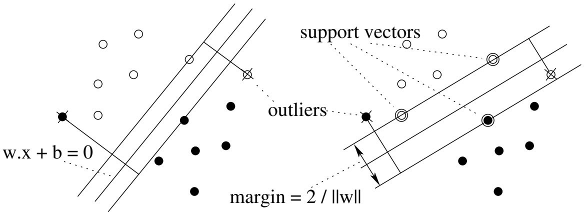
- $\mathbf{c}^+ = \frac{1}{n^+} \sum_{\{i: y_i = +1\}} \mathbf{x}_i$
- $\mathbf{c}^- = \frac{1}{n^-} \sum_{\{i: y_i = -1\}} \mathbf{x}_i$
- $\mathbf{c} = \frac{1}{2}(c^+ + c^-)$
- $\mathbf{w} = c^+ - c^-$

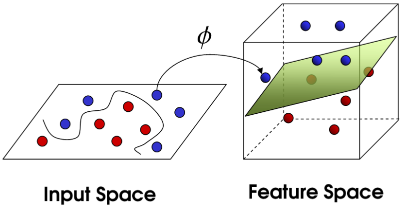
On evaluating $h(\mathbf{x})$

$$h(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} - \mathbf{c} \rangle = \langle \mathbf{w}, \mathbf{x} \rangle - \langle \mathbf{w}, \mathbf{c} \rangle = \dots$$
$$= \sum_{i=1,\dots,m} \alpha_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b, \quad \text{with } b \text{ a real constant}$$

Inner products are sufficient (remember that)

Kernel methods: graceful methods

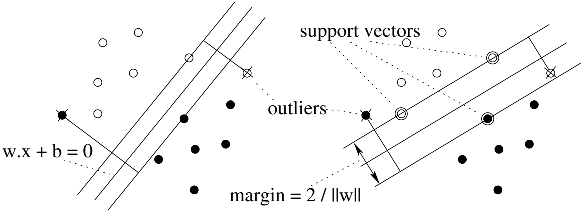


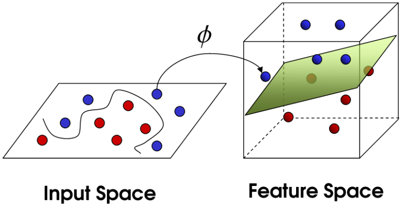


Silk methods

- Thereotical guarantees
- Convex optimization
- Nonlinearity handled through the kernel trick
- Success stories: structured data classification, ranking, scoring, theory

Kernel methods: graceful methods





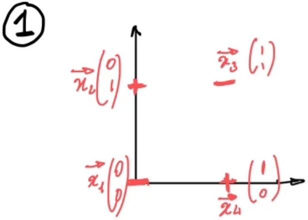
Kernelizing the handcrafted classifier

$h(\cdot) = \sum_{i=1,\dots,m} \alpha_i \langle \mathbf{x}_i, \cdot \rangle + b$ simply turns into

$$h(\mathbf{x}) = \sum_{i=1,\dots,m} \alpha_i \mathbf{k}(\mathbf{x}_i, \mathbf{x}) + b, \quad \text{with } b \text{ a real constant}$$

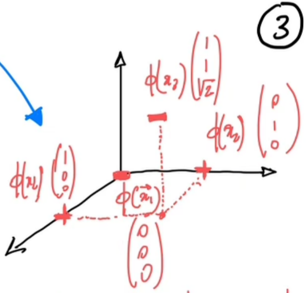
where $k(\cdot,\cdot)$ has replaced $\langle \cdot, \cdot \rangle$ and computes an inner product on the nonlinear embedding of its arguments

Example: 2nd degree polynomial kernel



2

$$\phi : \vec{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \longmapsto \phi(x) = \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2}\, x_1 x_2 \end{bmatrix}$$



And the magic is here:

4

$$\langle \phi(\vec{u}), \phi(\vec{v}) \rangle = \left\langle \begin{bmatrix} u_1^2 \\ u_2^2 \\ \sqrt{2}\, u_1 u_2 \end{bmatrix}, \begin{bmatrix} v_1^2 \\ v_2^2 \\ \sqrt{2}\, v_1 v_2 \end{bmatrix} \right\rangle$$

$$= u_1^2 v_1^2 + u_2^2 v_2^2 + 2 u_1 u_2 v_1 v_2$$

$$= u_1^2 v_1^2 + u_2^2 v_2^2 + 2 (u_1 v_1)(u_2 v_2)$$

$$= (u_1 v_1 + u_2 v_2)^2 = \underbrace{\langle \vec{u}, \vec{v} \rangle^2}_{\text{This is a } \boxed{\text{kernel}}}$$

There exists a linear separating hyperplane in $\mathbb{R}^3$.

Adaboost: combining weak learners

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in \mathscr{X}$, $y_i \in \{-1, +1\}$.
Initialize: $D_1(i) = 1/m$ for $i = 1, \ldots, m$.
For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : \mathscr{X} \to \{-1, +1\}$.
- Aim: select h_t with low weighted error:

$$\varepsilon_t = \Pr_{i \sim D_t}\left[h_t(x_i) \neq y_i\right].$$

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1 - \varepsilon_t}{\varepsilon_t}\right)$.
- Update, for $i = 1, \ldots, m$:

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$

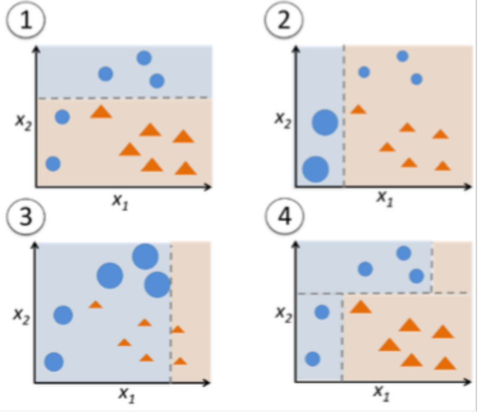
  where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$

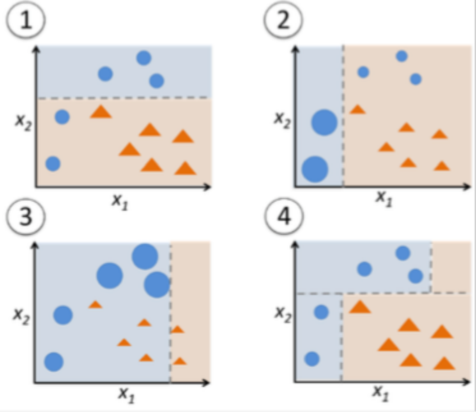
(from Freund and Schapire, 1997, 2012)

Adaboost: combining weak learners



(from Raschka, https://sebastianraschka.com/faq/docs/bagging-boosting-rf.html)

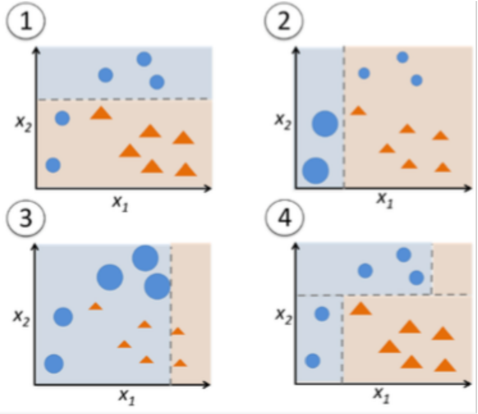
Adaboost: combining weak learners



(from Raschka, https://sebastianraschka.com/faq/docs/bagging-boosting-rf.html)

- Algorithmic simplicity, effectiveness
- Theoretical results
- Gödel price 2003

Adaboost: combining weak learners



(from Raschka, https://sebastianraschka.com/faq/docs/bagging-boosting-rf.html)

Find an illustrative example of Adaboost running

Bandits: exploration vs. exploitation dilemma

How to make the best use of your budget and bet?

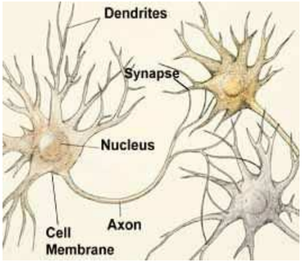
Features

- Problem easy to pose, many variations
- Exploration/exploitation dilemma
- Success stories: ad placement, recommendation, Go

Outline

Perceptron, binary case (Rosenblatt, 1958)

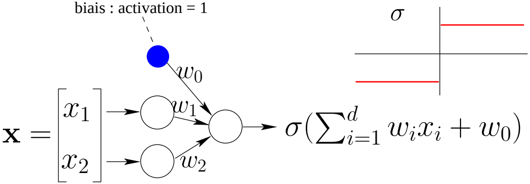
Inspiration: (real) neural networks



Biological motivations

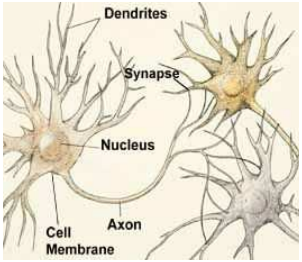
- Learning systems made of several simple computational units connected to each other
- Memory capacity / plasticity of these systems

Perceptron: a linear classifier, $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{-1, +1\}$



Perceptron, binary case (Rosenblatt, 1958)

Inspiration: (real) neural networks



Biological motivations

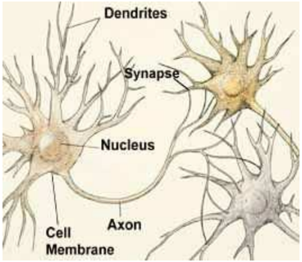
- Learning systems made of several simple computational units connected to each other
- Memory capacity / plasticity of these systems

Perceptron: a linear classifier, $\mathcal{X} = \mathbb{R}^d$, $\mathcal{Y} = \{-1, +1\}$

- Classifier parameters: $\mathbf{w} \in \mathbb{R}^d$
- Prediction of the classifier: $f(\mathbf{x}) = \text{sign}\langle \mathbf{w}, \mathbf{x} \rangle$
- Question: how to learn $\mathbf{w}$ from observations?

Perceptron, binary case (Rosenblatt, 1958)

Inspiration: (real) neural networks



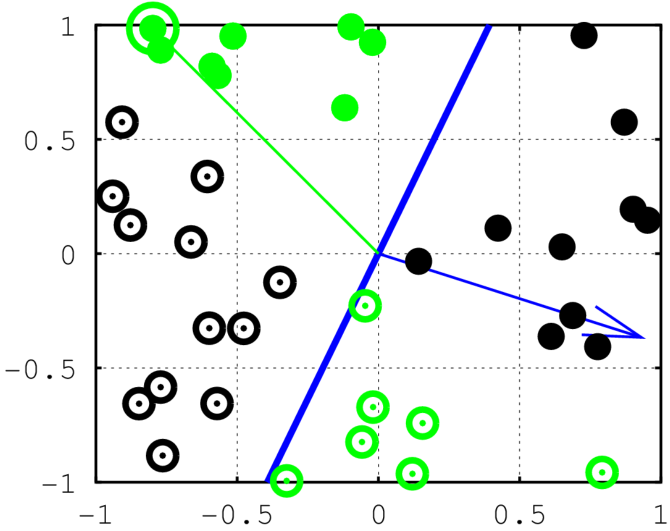
Biological motivations

- Learning systems made of several simple computational units connected to each other
- Memory capacity / plasticity of these systems

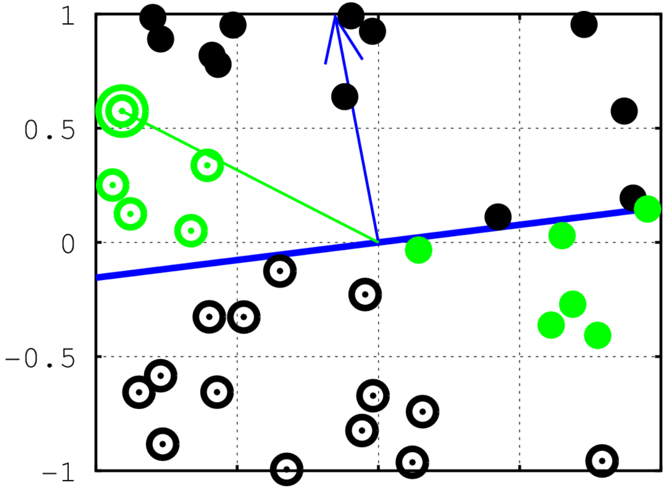
Algorithm: $\mathcal{D} = \{(X_n, Y_n)\}_{n=1}^N$

$\mathbf{w} \leftarrow \mathbf{0}$
while there exists (X_n, Y_n): $Y_n \langle \mathbf{w}, X_n \rangle \leq 0$ **do**
  $\mathbf{w} \leftarrow \mathbf{w} + Y_n X_n$
end while

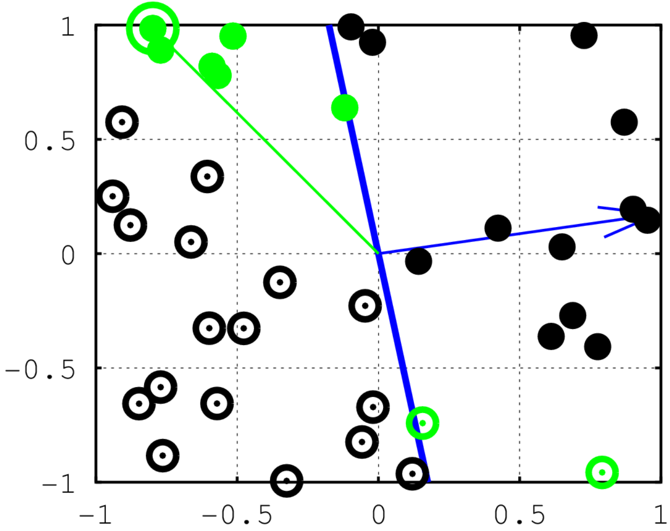
Perceptron learning in action



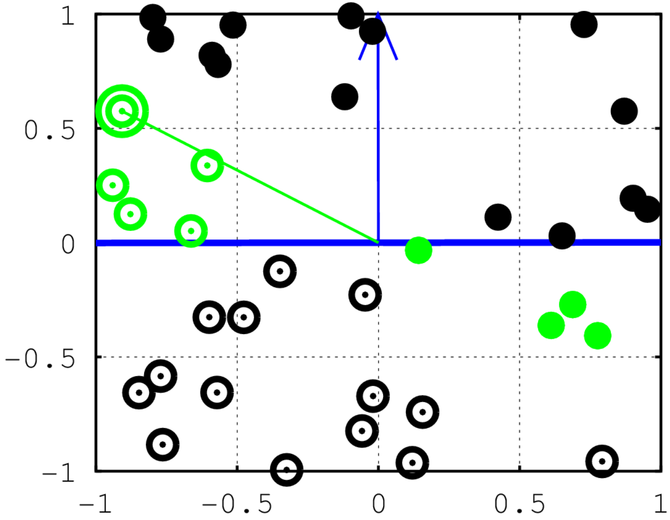
Perceptron learning in action



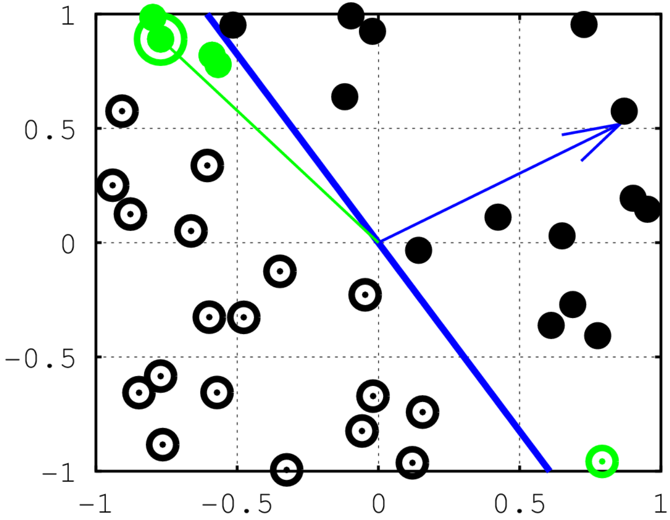
Perceptron learning in action



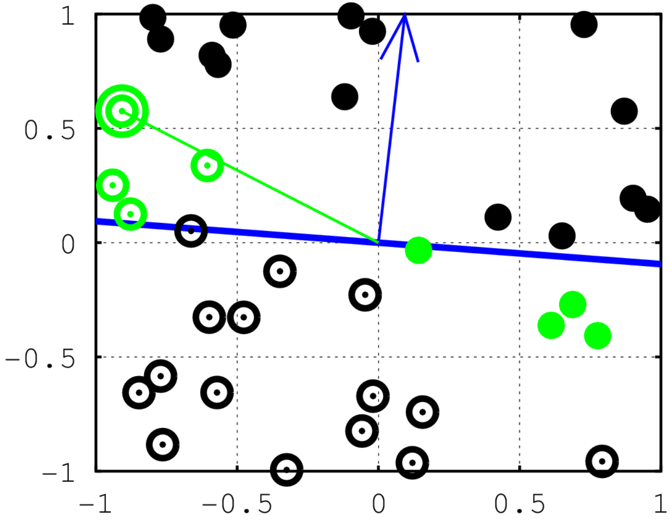
Perceptron learning in action



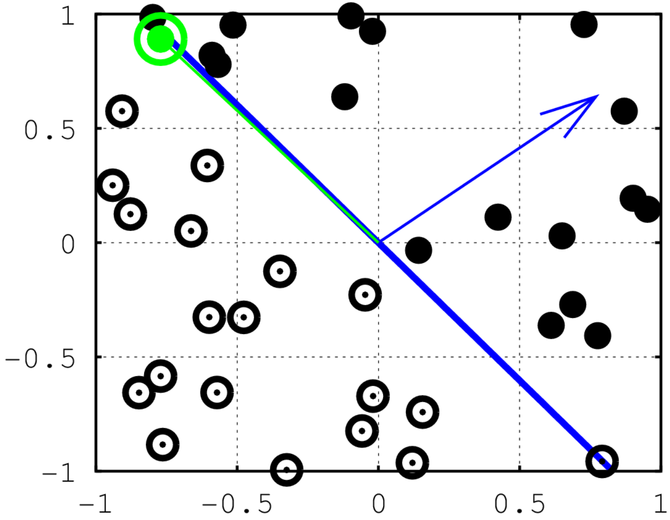
Perceptron learning in action



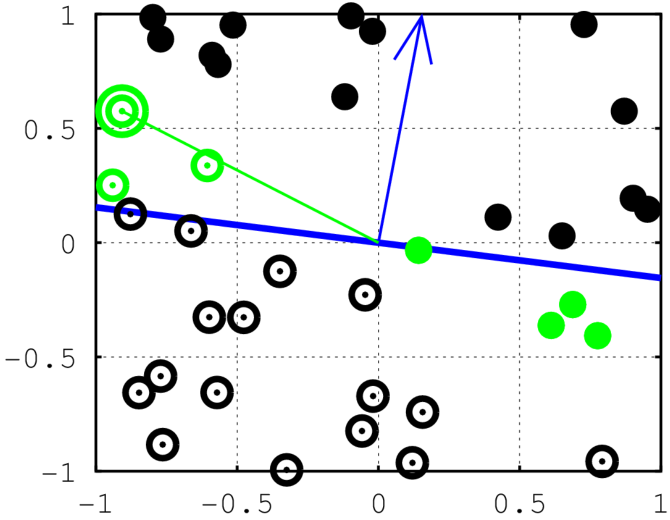
Perceptron learning in action



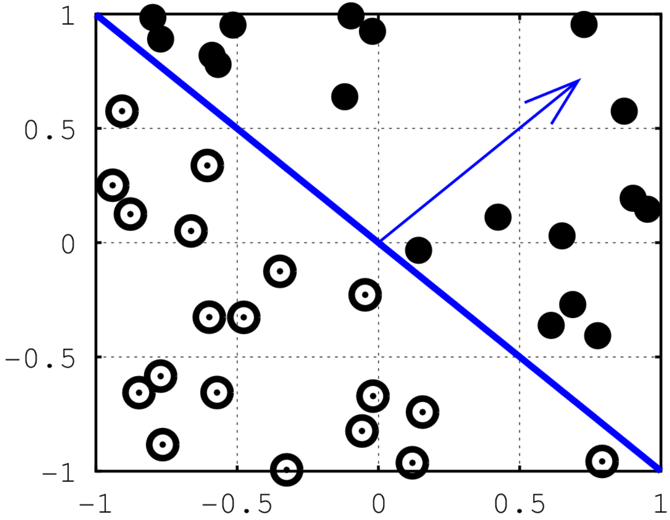
Perceptron learning in action



Perceptron learning in action



Perceptron learning in action



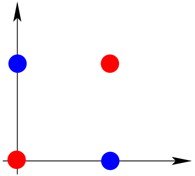
Perceptron: a few results

Theorem (Bound on the number of updates, Novikoff, 1962)

If there exist $\gamma > 0$, $\mathbf{w}^*$, $\|\mathbf{w}^*\| = 1$, $\|X_n\| \leq R, \forall n = 1, \ldots, N$, *et* $Y_n \langle \mathbf{w}^*, X_n \rangle \geq \gamma$ *then the Perceptron algorithm converges in less than* R^2/γ^2 *updates*

Theorem (XOR, Minsky, Papert, 1969)

The Perceptron (algorithm) cannot solve the XOR problem

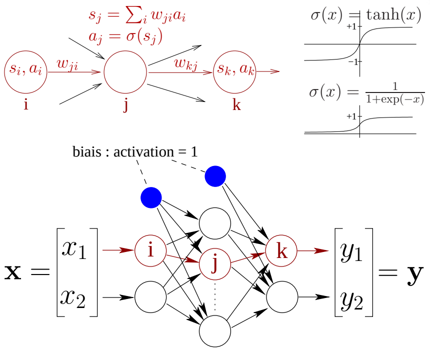


Theorem (Generalization error, Vapnik et Chevonenkis, 1979)

$\forall \mathbf{w} \in \mathbb{R}^d$: *with high probability*

$$R(w) \leq \hat{R}(\mathbf{w}, \mathcal{D}) + \tilde{O}\left(\sqrt{\frac{d}{n}}\right)$$

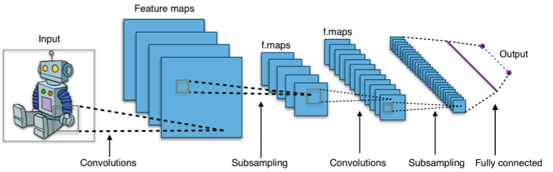
Multilayer Perceptron, Convolutional Networks



Up until the 90's

- Feedforward networks
- Gradient backpropagation (Rumelhart et al. 86)
- Preferred task: multiclass classification

Multilayer Perceptron, Convolutional Networks



(By Aphex34 - Own work, CC BY-SA 4.0, Wikimedia CNN)

Since 2005

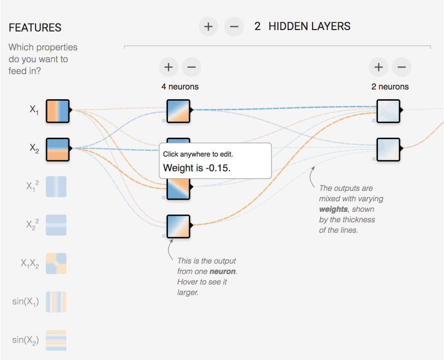
- Feedforward networks, recurrent networks
- Backpropagation (and autodiff), layerwise learning, computational power
- Tasks: almost everything (provided there is data)

But, more importantly

- Libraries: Tensorflow, Theano, Keras, Torch, Caffe (see là)
- Hardware: GPU, TPU (Tensor Processing Units)
- Data...

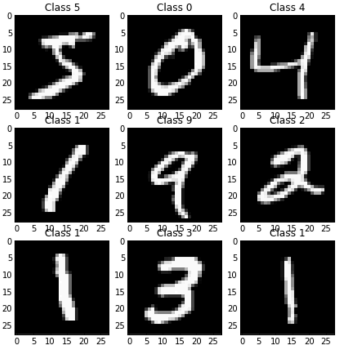
Deep Learning: Hands-on

Visualization



https://tinyurl.com/ydclvgas

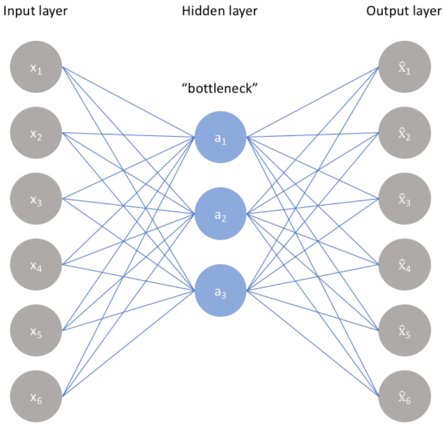
Keras Mnist Tutorial



https://tinyurl.com/ydzypus4

Dozens of examples can be found on Keras code examples page

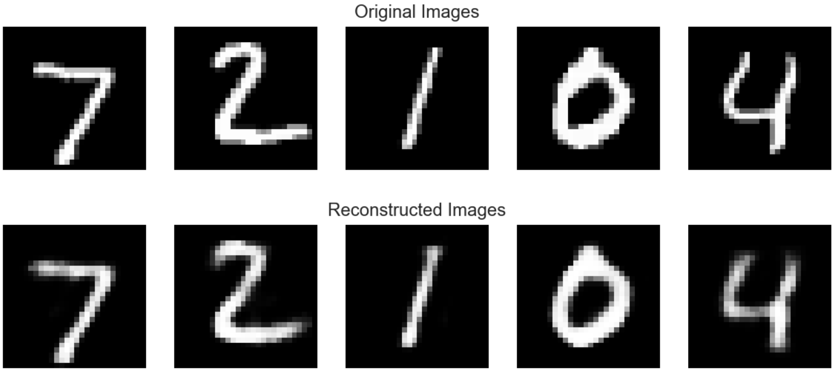
Unsupervised Deep Learning: auto-encoders



(From An introduction to Autoencoders)

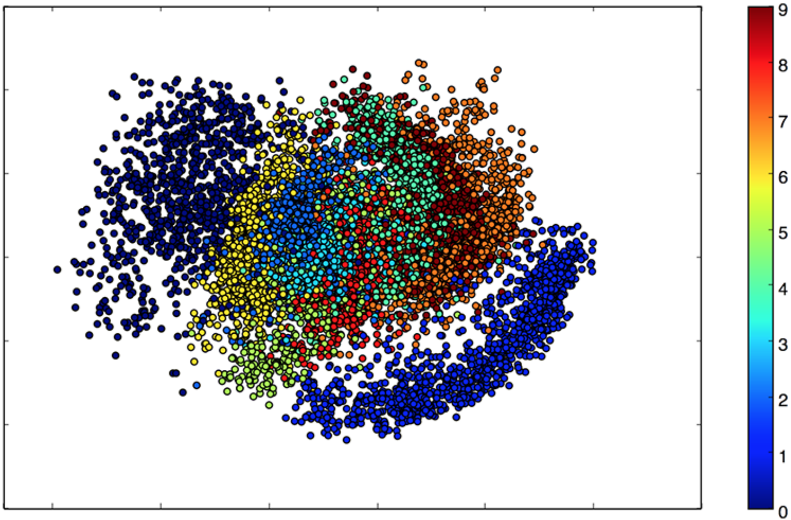
Code: `https://www.tensorflow.org/tutorials/generative/autoencoder`

Unsupervised Deep Learning: auto-encoders



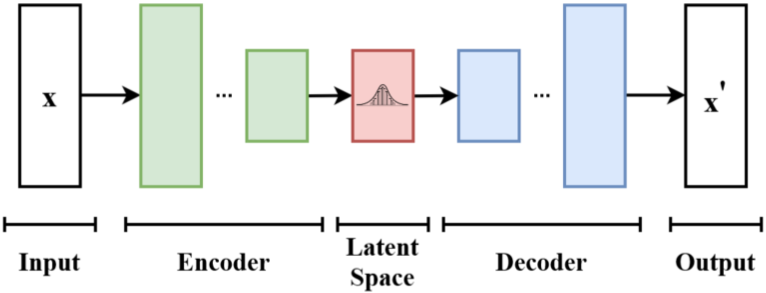
(From Applied Deep Learning - Part 3: Autoencoders)

Unsupervised Deep Learning: auto-encoders



(From Building Autoencoders in Keras)

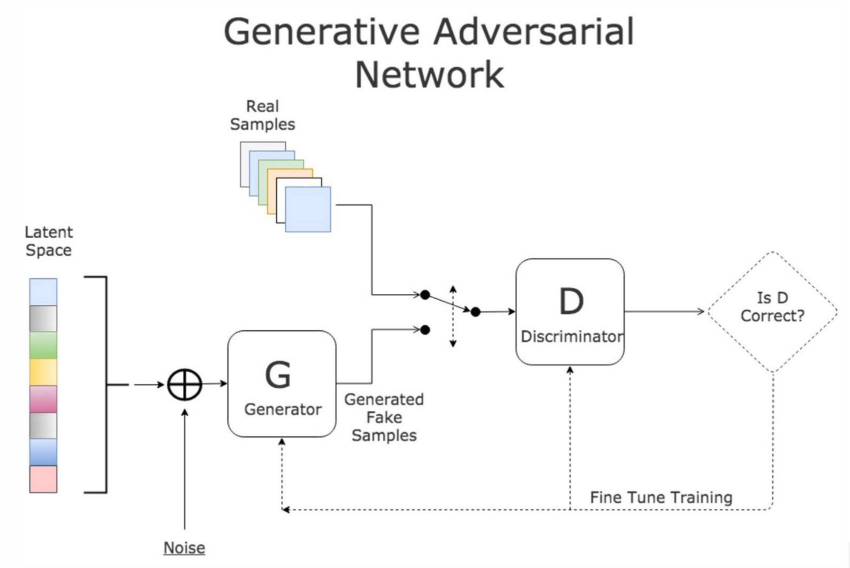
Unsupervised/Generative Deep Learning: Variational Auto-Encoders (Kingma and Welling, 2014)



(From Wikipedia VAE page)

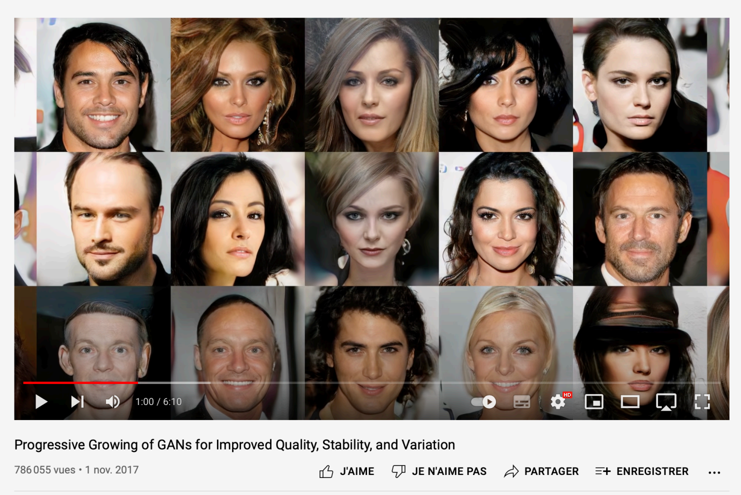
Code: `https://deeplearning.neuromatch.io/tutorials/W2D5_GenerativeModels/student/W2D5_Tutorial1.html`

Generative Deep Learning: GANs, (Goodfellow and al, 2014



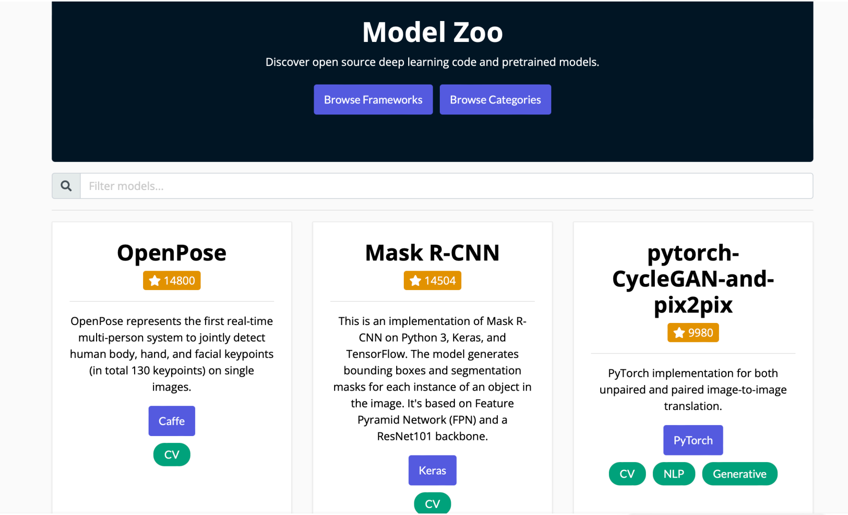
(From GANs from Scracth)

Generative Deep Learning: GANs, (Goodfellow and al, 2014



(From NVidia Video)

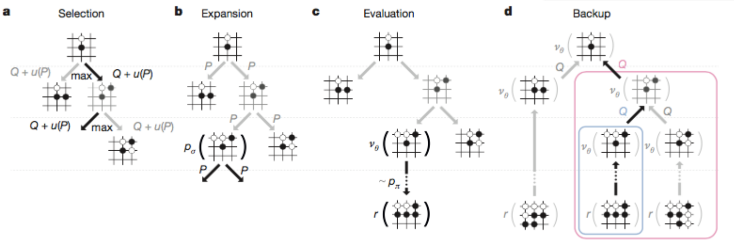
Models Zoo

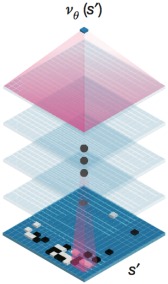


https://modelzoo.co

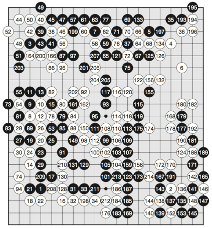
Outline

AlphaGo (Silver et al. 2016)





Game 5
Fan Hui (Black), AlphaGo (White)
AlphaGo wins by resignation

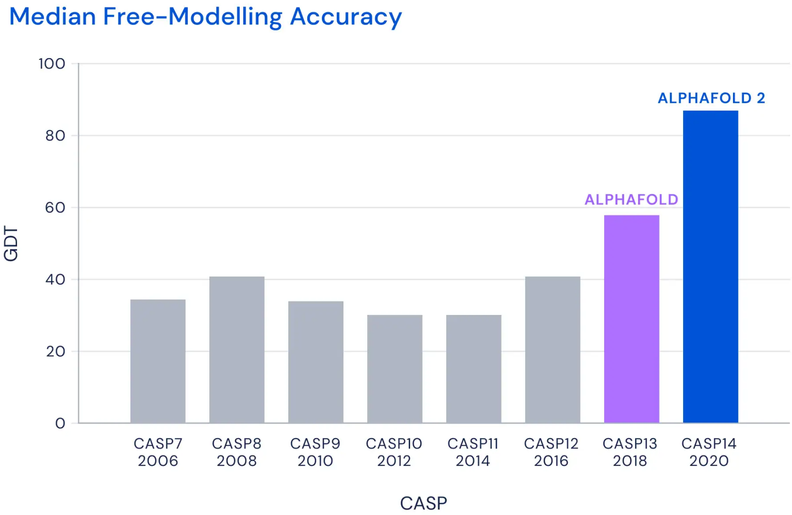


https://deepmind.com/blog/alphago-zero-learning-scratch/

AlphaGo (Silver et al. 2016)

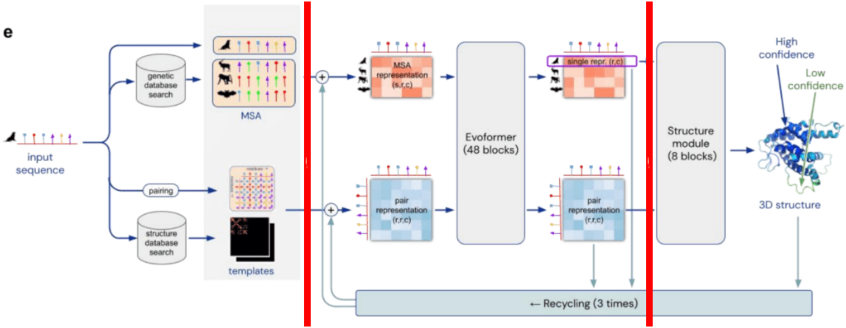
(From AlphaGo Netflix (Deepmind youtube))

AlphaFold (Jumper et al, Nature 2021)



(From AlphaFold: a solution to a 50-year-old grand challenge in biology)

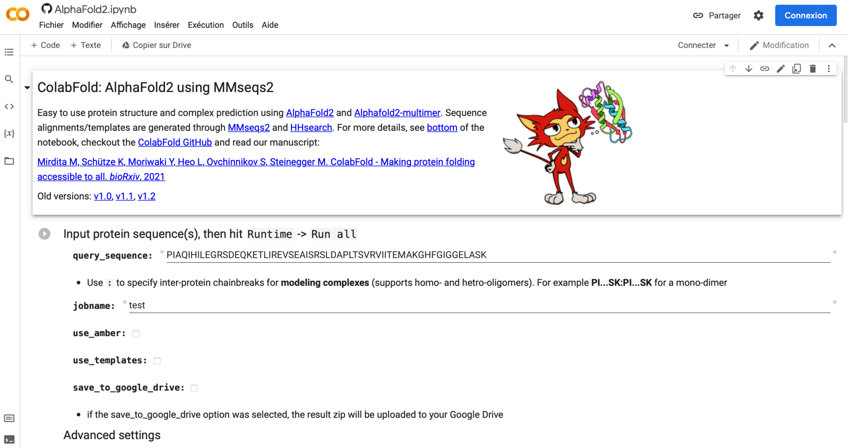
AlphaFold (Jumper et al, Nature 2021)



(From Jumper et al, Nature, 2021)

AlphaFold (Jumper et al, Nature 2021)

A notebook to play around



(From AlphaFold Notebook)

Outline

Machine Learning: a Variety of Problems/Mixes

Many application fields

- Computer vision
- NLP
- Robotics
- Advertising, recommendation systems
- Games (Go, chess, poker)
- Biology
- ...

Many problems

- Algorithmics
- Statistics
- Modelling
- ... and beyond

Conclusion

Machine Learning: a field in itselft

- A vivid branch of AI
- At the crossroads of computer science and mathematics
- Ever-growing community (from applied research to more fundamental one)

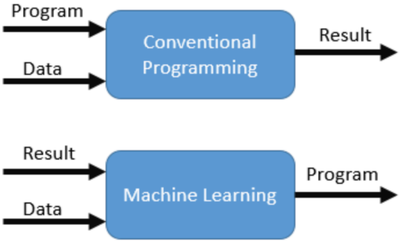
Machine Learning is ubiquituous

- At the heart of data science
- In many real-world applications
- ML at the time of revisiting other well-established fields of research

Example of future problems

- ML and small datasets: prior knowledge, active learning, feature selection
- ML & other fields: game theory, cryptography, biology, physics, law...

Hot AI topics (personal take)



Revisit classical fields from the Machine Learning perspective

- Privacy-Preserving ML: MLize encryption mechanisms, distributed computing
- Repeated Mechanism Design: MLize game theory, deal with coopetitive and competitve agents
- Green ML: hardware-aware methods, communication-sensitive methods...