An overview of Machine Learning

Alessandro Rudi, Francis Bach – Inria, ENS Paris

Thanks to Pierre Gaillard for the slides!

February 4, 2021

Overview

Introduction

General information

Teachers: Alessandro Rudi and Francis Bach.
Practical sessions: Raphal Berthier.

Website: `https://www.di.ens.fr/appstat/`

The class will last 52 hours (30 hours of class + 22 hours of practical sessions) and can be validated for 9 ECTS. Final grade: 50% final exam, 50% homework.

General information

Teachers: Alessandro Rudi and Francis Bach.
Practical sessions: Raphal Berthier.

Website: `https://www.di.ens.fr/appstat/`

The class will last 52 hours (30 hours of class + 22 hours of practical sessions) and can be validated for 9 ECTS. Final grade: 50% final exam, 50% homework.

Special online format: inverted classroom!

- Read lecture notes before
- Short review of material
- $\approx$ One mandatory question per student per session
- Have video feeds open

Prerequesites

Linear algebra (matrix operations, linear systems)

Probability (e.g. notion of random variables, conditional expectation)

Basic coding skills in Python: Jupyter notebooks, Anaconda

- If you do not know the language Python, please read (and code the examples of) this 10-minutes introduction to Python: `https://www.stavros.io/tutorials/python/`.
- For next week: run

`https://www.di.ens.fr/appstat/spring-2020/TP/TD0-prerequisites/crash_test.ipynb`

Rule

questions if something is unclear (we like them especially if you think they are stupid).

What is ML?

artificial intelligence which can learn and model some phenomena without being explicitly programmed

Examples of “success stories”:

- Spam classification
- Machine translation
- Speech recognition
- Self-driving cars

What is ML? Examples

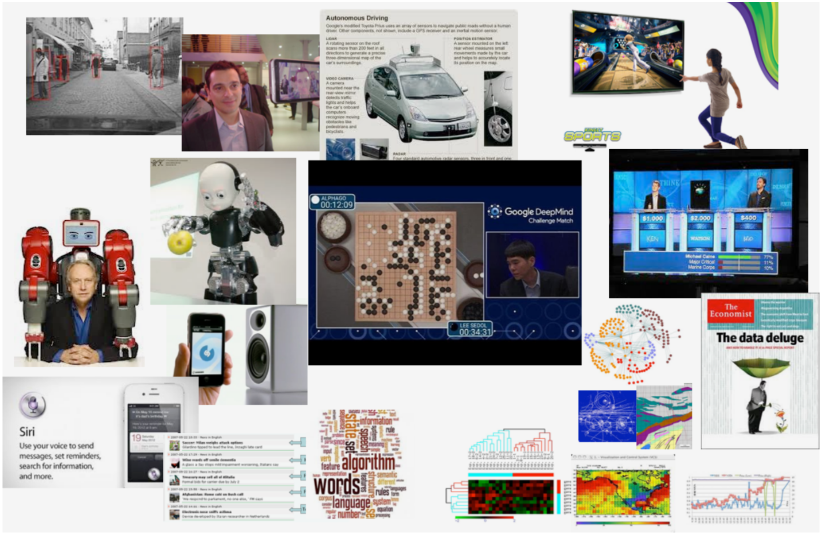
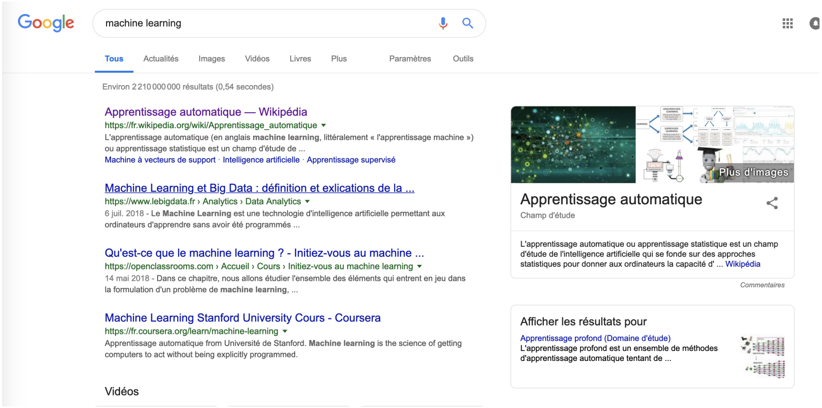


Image from Lorenzo Rosasco lecture

Search engines



machine learning

Tous · Actualités · Images · Vidéos · Livres · Plus · Paramètres · Outils

Environ 2 210 000 000 résultats (0,54 secondes)

Apprentissage automatique — Wikipédia
https://fr.wikipedia.org/wiki/Apprentissage_automatique
L'apprentissage automatique (en anglais **machine learning**, littéralement « l'apprentissage machine ») ou apprentissage statistique est un champ d'étude de ...
Machine à vecteurs de support · Intelligence artificielle · Apprentissage supervisé

Machine Learning et Big Data : définition et exlications de la ...
https://www.lebigdata.fr › Analytics › Data Analytics
6 juil. 2018 - Le **Machine Learning** est une technologie d'intelligence artificielle permettant aux ordinateurs d'apprendre sans avoir été programmés ...

Qu'est-ce que le machine learning ? - Initiez-vous au machine ...
https://openclassrooms.com › Accueil › Cours › Initiez-vous au machine learning
14 mai 2018 - Dans ce chapitre, nous allons étudier l'ensemble des éléments qui entrent en jeu dans la formulation d'un problème de **machine learning**, ...

Machine Learning Stanford University Cours - Coursera
https://fr.coursera.org/learn/machine-learning
Apprentissage automatique from Université de Stanford. **Machine learning** is the science of getting computers to act without being explicitly programmed.

Vidéos

Plus d'images

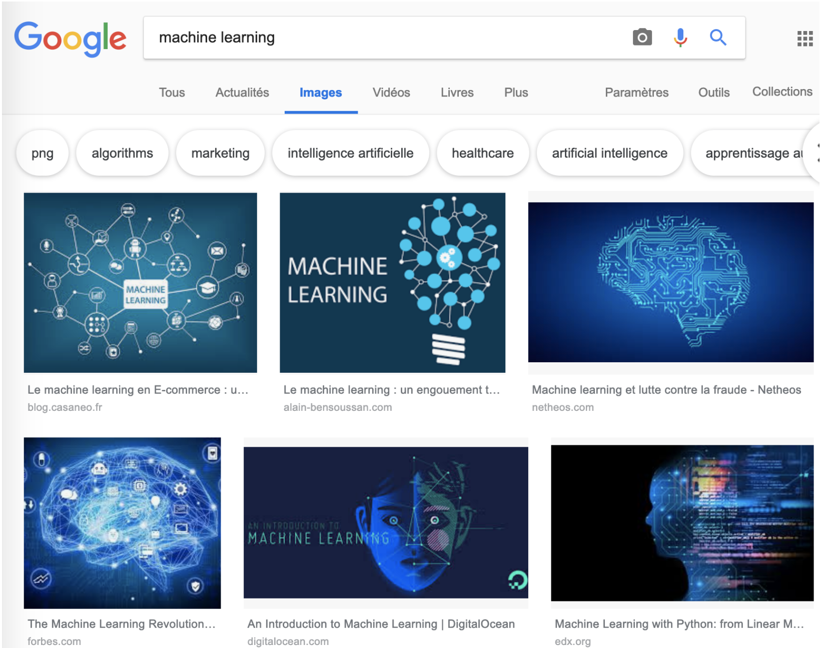
Apprentissage automatique
Champ d'étude

L'apprentissage automatique ou apprentissage statistique est un champ d'étude de l'intelligence artificielle qui se fonde sur des approches statistiques pour donner aux ordinateurs la capacité d' ... Wikipédia

Commentaires

Afficher les résultats pour
Apprentissage profond (Domaine d'étude)
L'apprentissage profond est un ensemble de méthodes d'apprentissage automatique tentant de ...

Search engines



machine learning

Tous Actualités Images Vidéos Livres Plus Paramètres Outils Collections

png algorithms marketing intelligence artificielle healthcare artificial intelligence apprentissage au

Le machine learning en E-commerce : u…
blog.casaneo.fr

Le machine learning : un engouement t…
alain-bensoussan.com

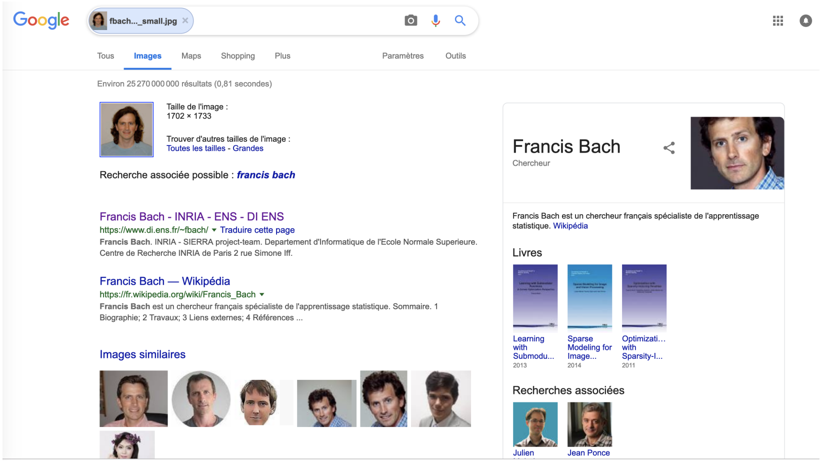
Machine learning et lutte contre la fraude - Netheos
netheos.com

The Machine Learning Revolution…
forbes.com

An Introduction to Machine Learning | DigitalOcean
digitalocean.com

Machine Learning with Python: from Linear M…
edx.org

Search engines



fbach..._small.jpg ×

Environ 25 270 000 000 résultats (0,81 secondes)

Taille de l'image :
1702 × 1733

Trouver d'autres tailles de l'image :
Toutes les tailles - Grandes

Recherche associée possible : ***francis bach***

Francis Bach - INRIA - ENS - DI ENS
https://www.di.ens.fr/~fbach/ ▾ Traduire cette page
Francis Bach. INRIA - SIERRA project-team. Departement d'Informatique de l'Ecole Normale Superieure. Centre de Recherche INRIA de Paris 2 rue Simone Iff.

Francis Bach — Wikipédia
https://fr.wikipedia.org/wiki/Francis_Bach ▾
Francis Bach est un chercheur français spécialiste de l'apprentissage statistique. Sommaire. 1 Biographie; 2 Travaux; 3 Liens externes; 4 Références ...

Images similaires

Francis Bach
Chercheur

Francis Bach est un chercheur français spécialiste de l'apprentissage statistique. Wikipédia

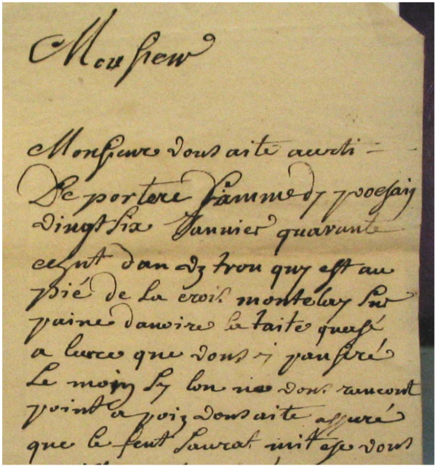
Livres

Learning with Submodu... 2013
Sparse Modeling for Image... 2014
Optimizati... with Sparsity-I... 2011

Recherches associées

Julien
Jean Ponce

Text recognition



Monſieu

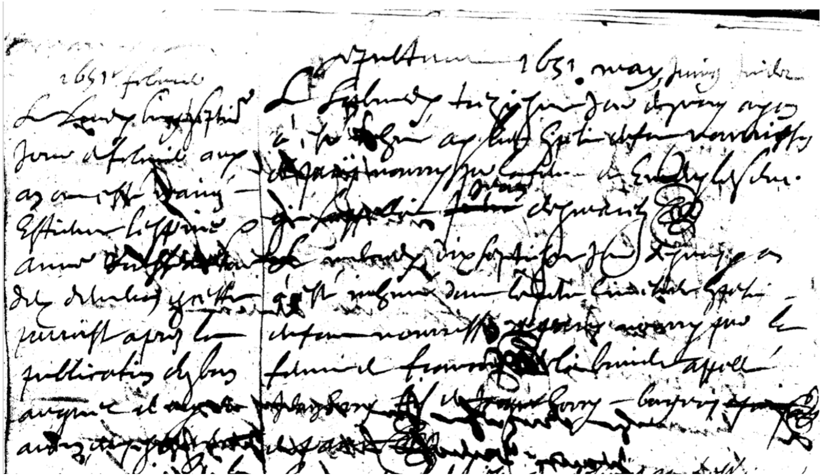
Monſieure vous aite averti –
De portere ſammedy procſai
vingt ſix Janvier quarante
ecunt dan dz trou quy eſt au
pié de la croix montelay ſur
paine davoire la taite caſſé
a lurce que vous y panſeré
Le moins ſy lon ne vous rancont
point a voiz vous aite aſſuré
que le feut ſaurat mit eſe vous

Monsieur,
Vous êtes averti de porter samedi prochain 26 janvier quarante écus dans un trou qui est au pied de la croix Montelay sous peine d'avoir la tête cassée à l'heure que vous y penserez le moins. Si l'on ne vous rencontre point vous êtes assuré que le feu sera mis chez vous. S'il en est parlé à qui que ce soit la tête cassée vous aurez.

Archives du Val d'Oise - 1737

Image from Francis Bach lecture

Text recognition


1651 [illegible] [illegible] 1651. may [illegible]

[illegible]

Image from Francis Bach lecture

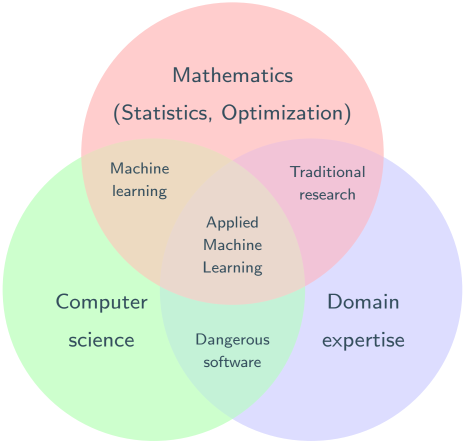
Bioinformatics

Large data – Complex data

What is ML?

Machine Learning : artificial intelligence which can learn and model some phenomena without being explicitly programmed

Machine Learning $\subset$ Statistics + Computer Sciences



What is ML?

Machine Learning : artificial intelligence which can learn and model some phenomena without being explicitly programmed

Machine Learning $\subset$ Statistics + Computer Sciences

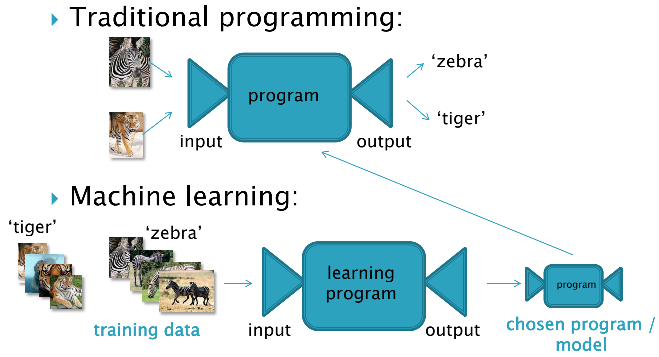


Image from Francis Bach lecture

Why is ML successful?

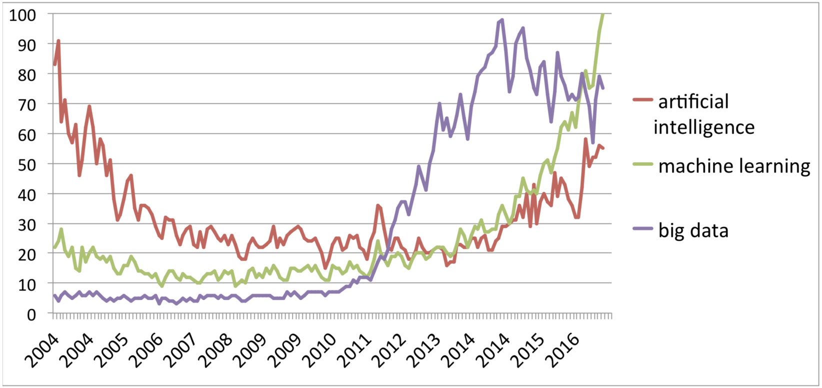
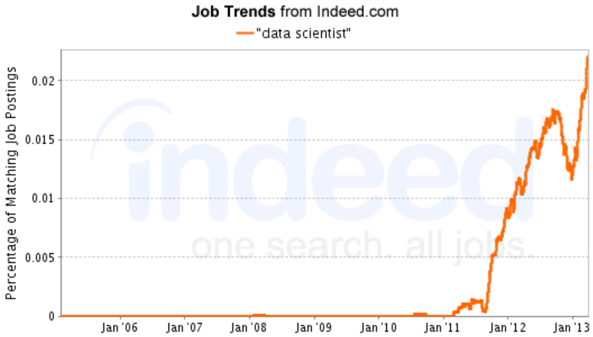


Image from Francis Bach lecture

Sexiest job of the century

"Data scientist is the sexiest job of the 21st century."

Harvard Business Review

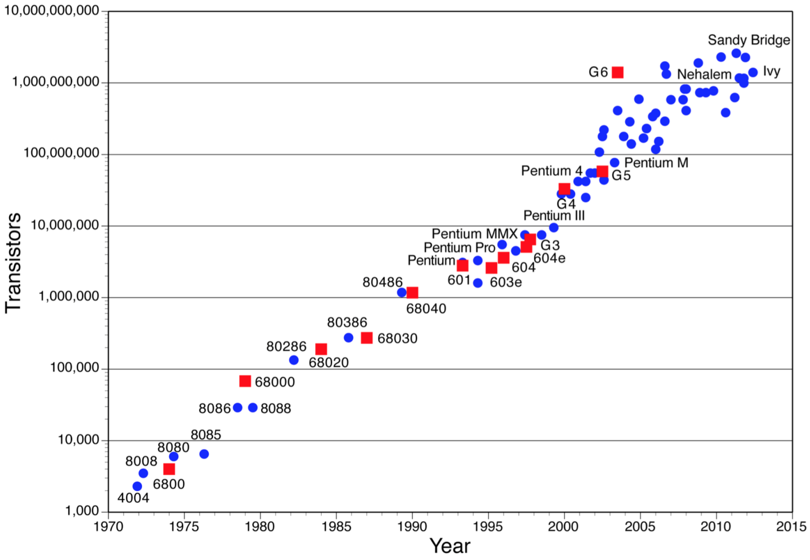


The machine learning revolution

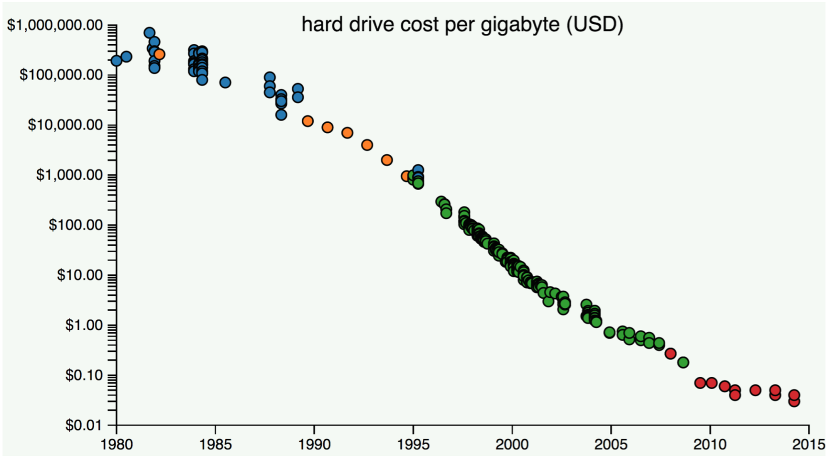
Big data / machine learning / data science / artificial intelligence / deep learning, a revolution?

- **Technical progress:** increase in computing power and storage capacity, lower costs

Moore's law: more computing power



Moore's Law: reduced costs



Limits : – debits do not follow

– miniaturization → reach the limits of classical physics → quantum mechanics

The machine learning revolution

Big data / machine learning / data science / artificial intelligence / deep learning, a revolution?

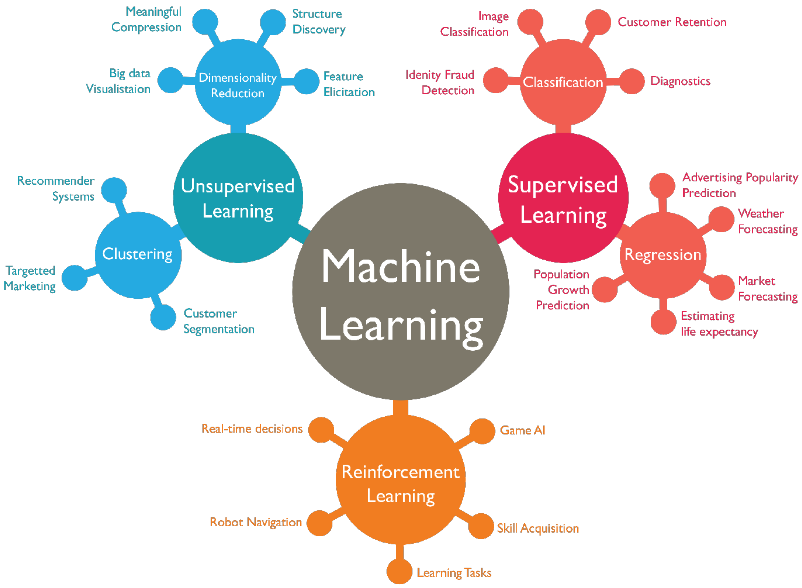
- **Technical progress:** increase in computing power and storage capacity, lower costs
- **Exponential increase in amount of data:** Volume, Variability, Velocity, Veracity
  - IBM: 10^{18} bytes created each day — 90% of the data $\leqslant 2$ years
  - In all area: sciences, industries, personal life
  - In all forms: video, text, clicks, numbers

The machine learning revolution

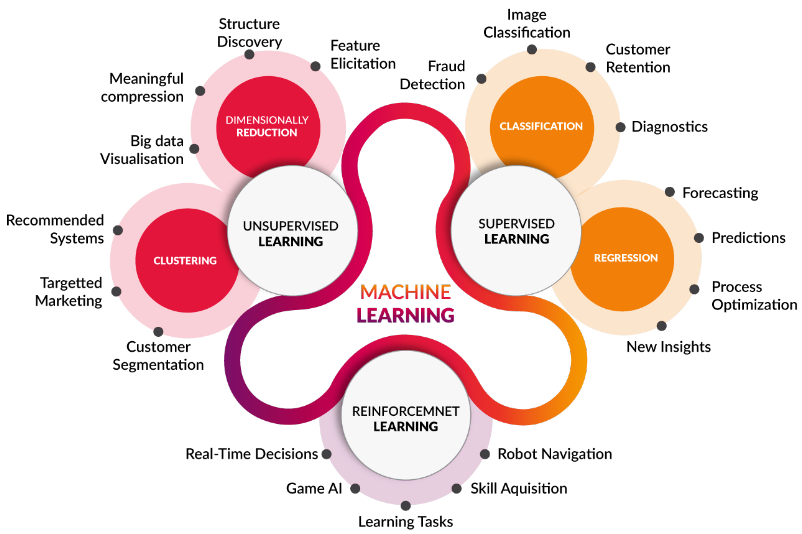
Big data / machine learning / data science / artificial intelligence / deep learning, a revolution?

- **Technical progress:** increase in computing power and storage capacity, lower costs
- **Exponential increase in amount of data:** Volume, Variability, Velocity, Veracity
  - IBM: 10^{18} bytes created each day — 90% of the data $\leqslant$ 2 years
  - In all area: sciences, industries, personal life
  - In all forms: video, text, clicks, numbers
- **Methodological advancement** to analyze complex datasets: high dimensional statistics, deep learning, reinforcement learning,. . .

Overview of Machine Learning



Overview of Machine Learning

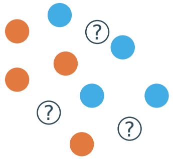


Overview of most popular machine learning methods

Two main categories of machine learning algorithms:

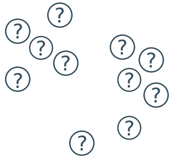
- **Supervised learning:** predict output Y from some input data X. The training data has a known label Y.

  Examples:
  - X is a picture, and Y is a cat or a dog
  - X is a picture, and $Y \in \{0, \dots, 9\}$ is a digit
  - X is are videos captured by a robot playing table tennis, and Y are the parameters of the robots to return the ball correctly
  - X is a music track and Y are the audio signals of each instrument



- **Unsupervised learning:** training data is not labeled and does not have a known result

  Examples:
  - detect change points in a non-stationary time-series
  - detect outliers
  - cluster data in homogeneous groups
  - compress data without loosing much information
  - density estimation



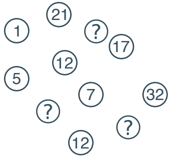
- **Others:** reinforcement learning, semi-supervised learning, online learning,. . .

Overview of most popular machine learning methods

Two main categories of machine learning algorithms:

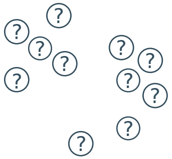
- **Supervised learning:** predict output Y from some input data X. The training data has a known label Y.

  Examples:
  - X is a picture, and Y is a cat or a dog
  - X is a picture, and $Y \in \{0, \dots, 9\}$ is a digit
  - X is are videos captured by a robot playing table tennis, and Y are the parameters of the robots to return the ball correctly
  - X is a music track and Y are the audio signals of each instrument



- **Unsupervised learning:** training data is not labeled and does not have a known result

  Examples:
  - detect change points in a non-stationary time-series
  - detect outliers
  - cluster data in homogeneous groups
  - compress data without loosing much information
  - density estimation



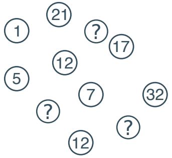
- **Others:** reinforcement learning, semi-supervised learning, online learning,. . .

Overview of most popular machine learning methods

Two main categories of machine learning algorithms:

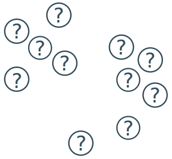
- **Supervised learning:** predict output Y from some input data X. The training data has a known label Y.

Classification	Regression
SVM Logistic regression Random Forest	Lasso, Ridge Nearest Neighbors Neural Networks



- **Unsupervised learning:** training data is not labeled and does not have a known result

Clustering	Dimensionality reduction
K-means, the Apriori algorithm, Birch, Ward, Spectral Cluster	PCA, ICA word embedding



- **Others:** reinforcement learning, semi-supervised learning, online learning,. . .

Supervised learning

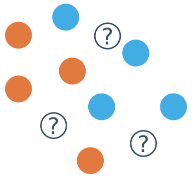
Supervised learning

Goal: from training data, we want to predict an output Y (or the best action) from the observation of some input X.

Difficulties: Y is not a deterministic function of X. There can be some noise:

$$Y = f(X) + \varepsilon$$

The function f is unknown and can be sophisticated.
$\rightarrow$ hard to perform well systematically



Possible theoretical approaches: perform well

- in the worst-case: minimax theory, game theory
- in average, or with high probability

Algorithmic approaches:

- local averages: K-nearest neighbors, decision trees
- empirical risk minimization: linear regression, lasso, spline regression, SVM, logistic regression
- online learning
- deep learning
- probabilistic models: graphical models, Bayesian methods

Supervised learning: theory

Some data $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ is distributed according to a probability distribution P.

We observe training data $D_n := \big\{(X_1, Y_1), \dots, (X_n, Y_n)\big\}$.

We must form prediction into a decision set $\mathcal{A}$ by choosing a prediction function

$$f : \underbrace{\mathcal{X}}_{\text{observation}} \to \underbrace{\mathcal{A}}_{\text{decision}}$$

Our performance is measured by a loss function $\ell : \mathcal{A} \times \mathcal{Y} \to \mathbb{R}$. We define the risk

$$R(f) := \mathbb{E}\big[\ell\big(f(X), Y\big)\big] \qquad = \qquad \text{expected loss of } f$$

Goal: minimize $R(f)$ by approaching the performance of the oracle $f^* = \arg\min_{f\in\mathcal{F}} R(f)$

	Least square regression	**Classification**
$\mathcal{A} = \mathcal{Y}$	$\mathbb{R}$	$\{0, 1, \dots, K-1\}$
$\ell(a, y)$	$(a-y)^2$	$\mathbb{1}_{a \neq y}$
$R(f)$	$\mathbb{E}\big[(f(X)-Y)^2\big]$	$\mathbb{P}(f(X) \neq Y)$
f^*	$\mathbb{E}[Y\|X]$	$\arg\max_k \mathbb{P}(Y = k\|X)$

Supervised learning: theory

Outline

Empirical risk minimization

Idea: estimate $R(f)$ thanks to the training data with the empirical risk

$$\underbrace{\hat{R}_n(f) := \frac{1}{n}\sum_{i=1}^{n} \ell(f(X_i), Y_i)}_{\text{average error on training data}} \approx \underbrace{R(f) = \mathbb{E}[\ell(f(X), Y)]}_{\text{expected error}}$$

We estimate $\hat{f}_n$ by minimizing the empirical risk

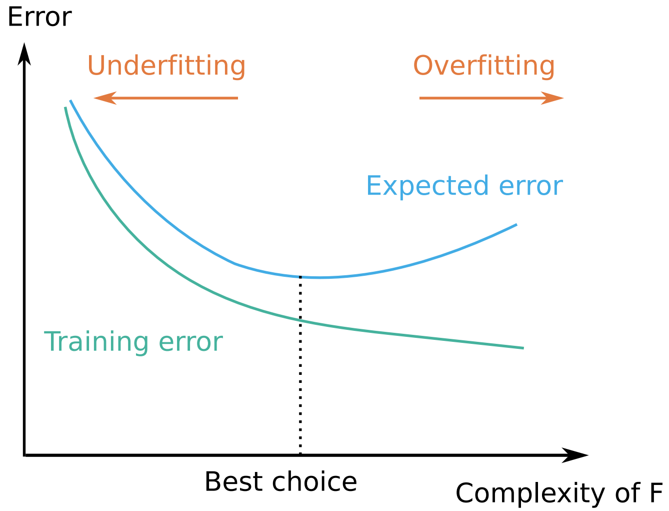
$$\hat{f}_n \in \arg\min_{f \in \mathcal{F}} \hat{R}_n(f) \,.$$

Many methods are based on empirical risk minimization: ordinary least square, logistic regression, Ridge, Lasso,. . .

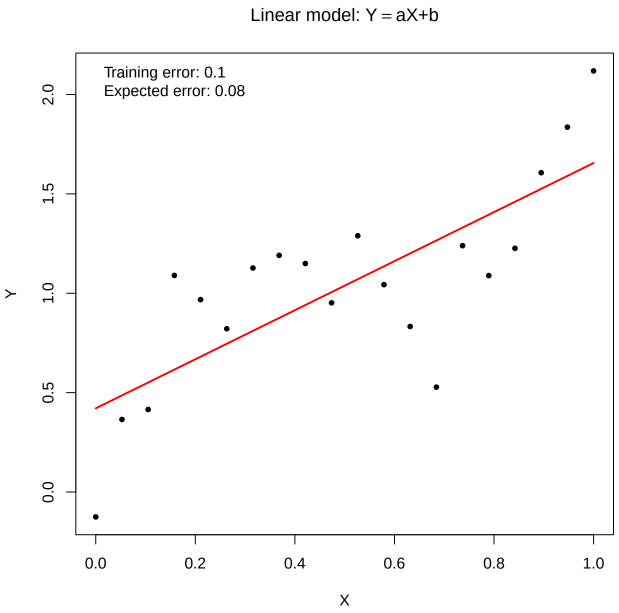
Choosing the right model: $\mathcal{F}$ is a set of models which needs to be properly chosen:

$$R(\hat{f}_n) = \underbrace{\min_{f \in \mathcal{F}} R(f)}_{\text{Approximation error}} + \underbrace{R(\hat{f}_n) - \min_{f \in \mathcal{F}} R(f)}_{\text{Estimation error}}$$

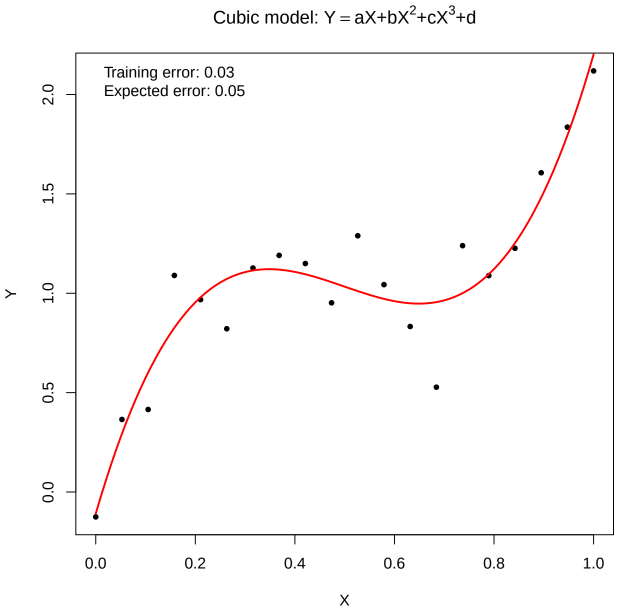
Overfitting



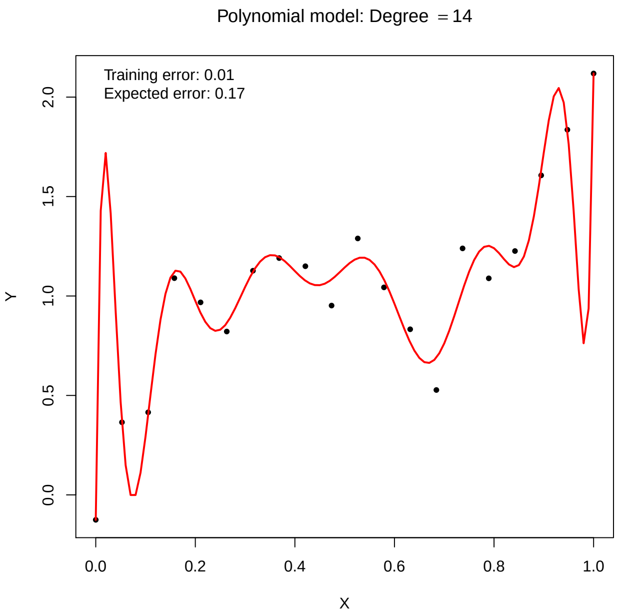
Overfitting: example in regression



Overfitting: example in regression



Overfitting: example in regression

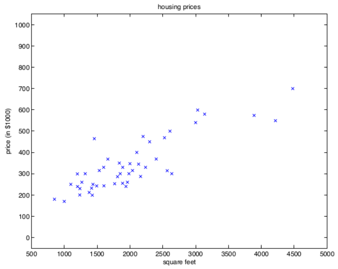


Least square linear regression

DATA

Living area (feet^2)	Price (1000\$s)
2104	400
1600	330
2400	369
1416	232
3000	540
$\vdots$	$\vdots$

example taken from Coursera



$$(x_1, y_1), \ldots, (x_n, y_n)$$

Living area (feet^2)	#bedrooms	Price (1000\$s)
2104	3	400
1600	3	330
2400	3	369
1416	2	232
3000	4	540
$\vdots$	$\vdots$	$\vdots$

Least square Linear regression

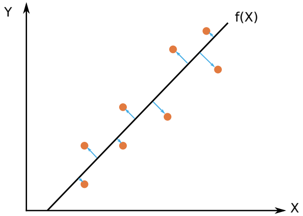
Given training data (X_i, Y_i) for $i = 1, \ldots, n$, with $X_i \in \mathbb{R}^d$ and $Y_i \in \{0, 1\}$ learn a predictor f such that our expected square loss

$$\mathbb{E}\big[(f(X) - Y)^2\big]$$

is small.

We assume here that f is a linear combination of the input $x = (x_1, \ldots, x_d)$

$$f_w(x) = \sum_{i=1}^{d} w_i x_i = w^\top x$$



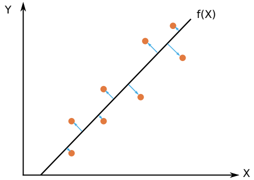
Ordinary Least Square

Input $X \in \mathbb{R}^d$, output $Y \in \mathbb{R}$, and ℓ is the square loss: $\ell(a, y) = (a - y)^2$.

The Ordinary Least Square regression (OLS) minimizes the empirical risk

$$\hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - w^\top X_i)^2$$

This is minimized in $w \in \mathbb{R}^d$ when $\boldsymbol{X}^\top \boldsymbol{X} w - \boldsymbol{X}^\top \boldsymbol{Y} = 0$, where $\boldsymbol{X} = [X_1, \ldots, X_n]^\top \in \mathbb{R}^{n \times d}$ and $\boldsymbol{Y} = [Y_1, \ldots, Y_n]^\top \in \mathbb{R}^n$.

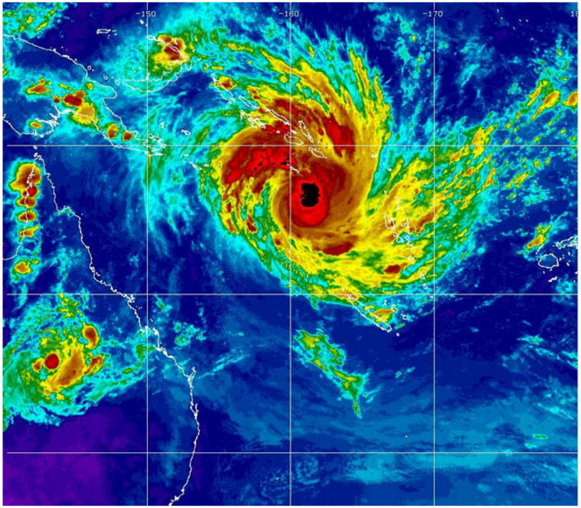


Assuming X is injective (i.e., $X^\top X$ is invertible) and there is an exact solution

$$\hat{w} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{Y} \, .$$

What happens if $d \gg n$?

Geo-science



Ordinary Least Square: how to compute $\hat{w}_n$?

If the design matrix $X^\top X$ is invertible, the OLS has the closed form:

$$\hat{w}_n \in \arg\min_w \hat{R}_n(w) = \left(\boldsymbol{X}^\top \boldsymbol{X}\right)^{-1} \boldsymbol{X}^\top \boldsymbol{Y}\,.$$

Question: how to compute it?

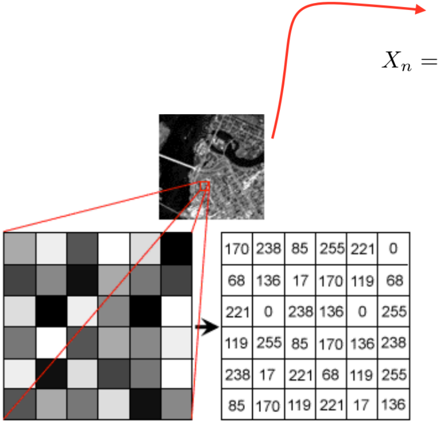
- inversion of $(\boldsymbol{X}^\top \boldsymbol{X})$ can be prohibitive (the cost is $\mathcal{O}(d^3)$!)
- QR-decomposition: we write $\boldsymbol{X} = QR$, with Q an orthogonal matrix and R an upper-triangular matrix. One needs to solve the linear system:

$$R\hat{w} = Q^\top \boldsymbol{Y}, \qquad \text{with} \qquad R = \begin{pmatrix} x & x & \cdots & x \\ & \ddots & & \vdots \\ & & \ddots & \\ 0 & & & x \end{pmatrix}$$

- iterative approximation with convex optimization algorithms [Bottou, Curtis, and Nocedal 2016]: (stochastic)-gradient descent, Newton,...

$$w_{i+1} = w_i - \eta \nabla \hat{R}_n(w_i)$$

Beyond vectors?

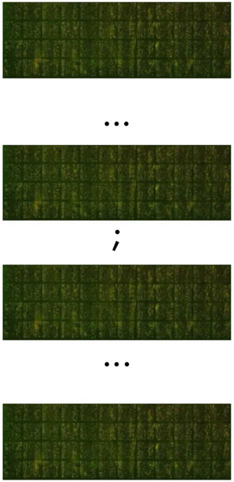


$$X_n = \begin{pmatrix} x_1^1 & \dots & \dots & \dots & x_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^1 & \dots & \dots & \dots & x_n^p \end{pmatrix}$$

170	238	85	255	221	0
68	136	17	170	119	68
221	0	238	136	0	255
119	255	85	170	136	238
238	17	221	68	119	255
85	170	119	221	17	136

Image from Lorenzo Rosasco lecture

Beyond vectors?

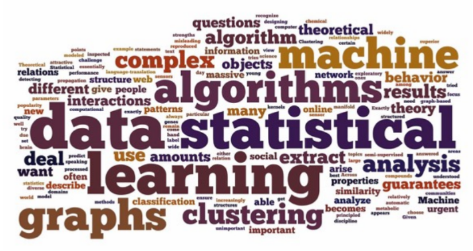


***n* patients *p* gene expression measurements**

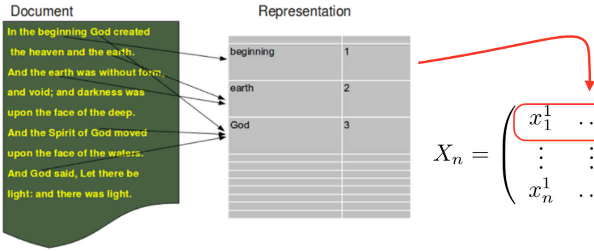
$$X_n = \begin{pmatrix} x_1^1 & \dots & \dots & \dots & x_1^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^1 & \dots & \dots & \dots & x_n^p \end{pmatrix}; Y_n = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Image from Lorenzo Rosasco lecture

Beyond vectors?



$$Y_n = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$



Document
Representation
In the beginning God created
the heaven and the earth.
And the earth was without form,
and void; and darkness was
upon the face of the deep.
And the Spirit of God moved
upon the face of the waters.
And God said, Let there be
light: and there was light.
beginning
1
earth
2
God
3

$$X_n = \begin{pmatrix} \boxed{x_1^1 \quad \dots \quad \dots \quad \dots \quad x_1^p} \\ \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ x_n^1 \quad \dots \quad \dots \quad \dots \quad x_n^p \end{pmatrix}$$

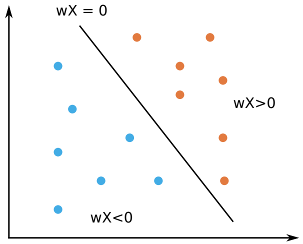
Image from Lorenzo Rosasco lecture

Classification

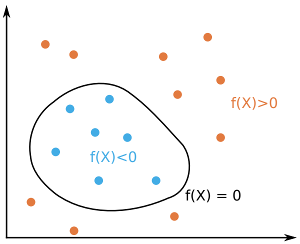
Given training data (X_i, Y_i) for $i = 1, \ldots, n$, with $X_i \in \mathbb{R}^d$ and $Y_i \in \{0, 1\}$ learn a classifier $f(x)$ such that

$$f(X_i) \begin{cases} \geqslant 0 & \Rightarrow \quad Y_i = +1 \\ < 0 & \Rightarrow \quad Y_i = 0 \end{cases}$$

Linearly separable



Non Linearly separable



Linear classification

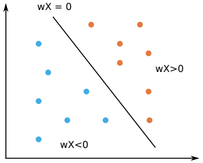
We would like to find the best linear classifier such that

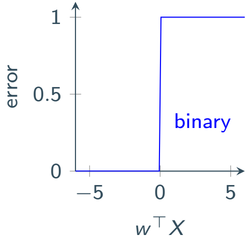
$$f_w(X) = w^\top X \begin{cases} \geqslant 0 & \Rightarrow \quad Y = +1 \\ < 0 & \Rightarrow \quad Y = 0 \end{cases}$$

Empirical risk minimization with the binary loss?

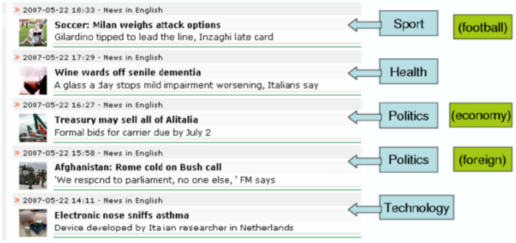
$$\hat{w}_n = \underset{w \in \mathbb{R}^d}{\arg\min} \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{Y_i \neq \mathbb{1}_{w^\top X_i \geqslant 0}} .$$

This is not convex in w. Very hard to compute!





Spam filters



» 2007-05-22 18:33 · News in English
Soccer: Milan weighs attack options
Gilardino tipped to lead the line, Inzaghi late card

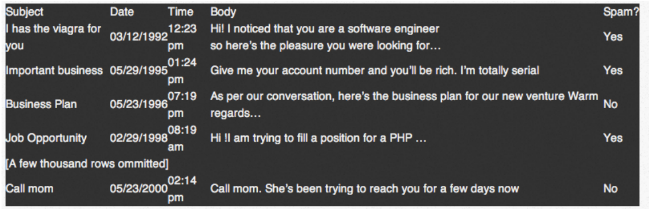
» 2007-05-22 17:29 · News in English
Wine wards off senile dementia
A glass a day stops mild impairment worsening, Italians say

» 2007-05-22 16:27 · News in English
Treasury may sell all of Alitalia
Formal bids for carrier due by July 2

» 2007-05-22 15:58 · News in English
Afghanistan: Rome cold on Bush call
'We respond to parliament, no one else,' FM says

» 2007-05-22 14:11 · News in English
Electronic nose sniffs asthma
Device developed by Italian researcher in Netherlands

Sport
(football)
Health
Politics
(economy)
Politics
(foreign)
Technology



Subject	Date	Time	Body	Spam?
I has the viagra for you	03/12/1992	12:23 pm	Hi! I noticed that you are a software engineer so here's the pleasure you were looking for...	Yes
Important business	05/29/1995	01:24 pm	Give me your account number and you'll be rich. I'm totally serial	Yes
Business Plan	05/23/1996	07:19 pm	As per our conversation, here's the business plan for our new venture Warm regards...	No
Job Opportunity	02/29/1998	08:19 am	Hi !I am trying to fill a position for a PHP ...	Yes
[A few thousand rows ommitted]				
Call mom	05/23/2000	02:14 pm	Call mom. She's been trying to reach you for a few days now	No

Image from Lorenzo Rosasco lecture

Image recognition

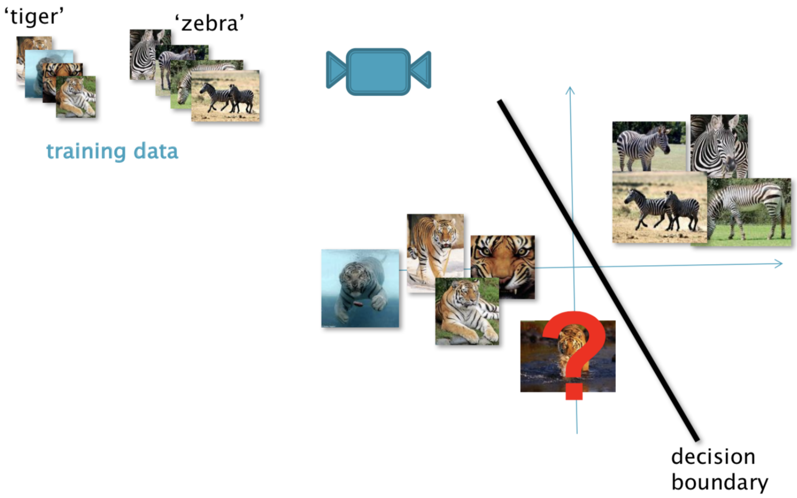
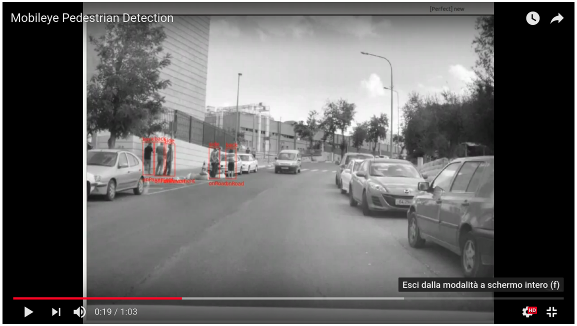


Image from Francis Bach lecture

Pedestrian recognition

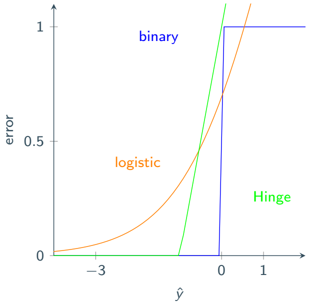

Mobileye Pedestrian Detection
Esci dalla modalità a schermo intero (f)
0:19 / 1:03

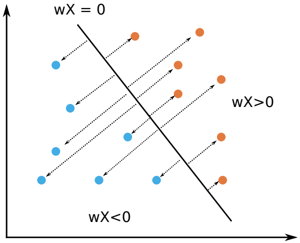
https://youtu.be/lU4w0o-caFM

Logistic regression

Idea: replace the loss with a convex loss

$$\ell(w^\top X, y) = y \log\left(1 + e^{-w^\top X}\right) + (1-y)\log\left(1 + e^{w^\top X}\right)$$





Probabilistic interpretation: based on likelihood maximization of the model:

$$\mathbb{P}(Y=1|X) = \frac{1}{1+e^{-w^\top X}} \in [0,1]$$

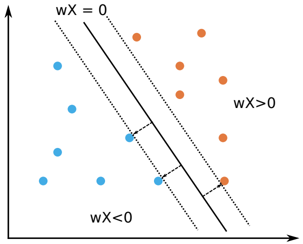
Satisfied for many distributions of $X|Y$: Bernoulli, Gaussian, Exponential, ...

Computation of the minimizer of the empirical risk (No closed form of the solution)

- Use a convex optimization algorithm (Newton, gradient descent,...)

Support Vector Machine (SVM)

In SVM, the linear separator (hyperplane) is chosen by maximizing the margin. Not by minimizing the empirical risk.



👍 Sparsity: it only depends on a few training points, called the support vectors

In practice, we use soft margins because no perfect linear separation is possible.

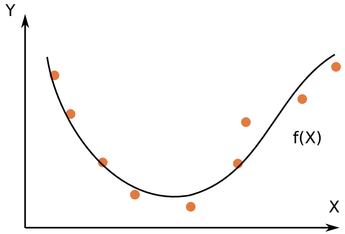
Non-linear regression/classification

Until now, we have only considered linear predictions of $x = (x_1, \dots, x_d)$

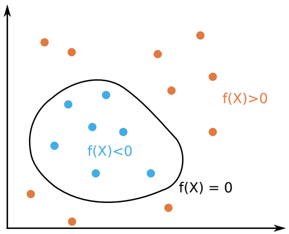
$$f_w(x) = \sum_{i=1}^{d} w_i x_i \, .$$

But this can perform pretty bad... How to perform non-linear regression?

Non linear regression



Non Linearly separable



Non-linear regression/classification

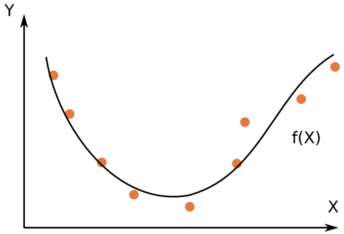
Idea: map the input X into a higher dimensional space where the problem is linear.

Example: given an input $x = (x_1, x_2, x_3)$ perform a linear method on a transformation of the input like

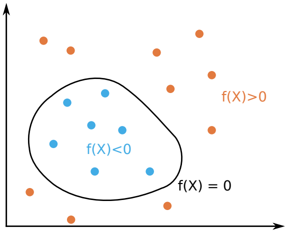
$$\Phi(x) = \left(x_1x_1, x_1x_2, \dots, x_3x_2, x_3x_3\right) \in \mathbb{R}^9$$

Linear transformations of $\Phi(x)$ are polynomials of x! The previous methods works by replacing x with $\Phi(x)$.

Non linear regression



Non Linearly separable

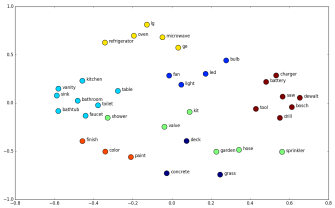


Example: Word Embedding (Word2Vect)

`http://wordrepresentation.appspot.com`

Words

$\downarrow$



```
>>> print_analogy('Paris', 'France', 'Rome', words)
Paris-France is like Rome-Italy

>>> print_analogy('man', 'king', 'woman', words)
man-king is like woman-queen

>>> print_analogy('walk', 'walked' , 'go', words)
walk-walked is like go-went

>>> print_analogy('quick', 'quickest' , 'far', words)
quick-quickest is like far-furthest
```

Spline regression

A spline of degree p is a function formed by connecting polynomial segments of degree p so that:

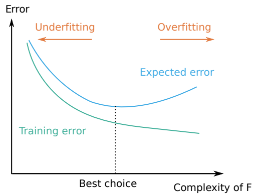
- the function is continuous
- the function has D 1 continuous derivatives
- the pth-derivative is constant between knots

This can be done by choosing the good transformation $\Phi_p(x)$ and the right regularization $\|\Phi_p(x)\|$.

Difficulties: choose the number of knots and the degree

Regularization

How to avoid over-fitting if there is not enough data?



Control the complexity of the solution

- explicitly by choosing $\mathcal{F}$ small enough: choose the degree of the polynomials,...
- implicitly by adding a regularization term

$$\min_{f \in \mathcal{F}} \hat{R}_n(f) + \lambda \|f\|^2$$

The higher the norm $\|f\|$ is, the more complex the function is.

👍 We do not need to know the best complexity $\mathcal{F}$ in advance

👎 Complexity controlled by λ, which need to be calibrated.

Ridge regression

The most classic regularization in statistics for linear regression:

$$\widehat{w}_n = \arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (Y_i - w^\top X_i)^2 \; + \; \lambda \sum_{i=1}^{d} w_i^2$$

The exact solution is unique because the problem is now strongly convex:

$$\hat{w}_n = \left(\boldsymbol{X}^\top \boldsymbol{X} + n\lambda \boldsymbol{I} \right)^{-1} \boldsymbol{X}^\top \boldsymbol{Y}$$

The regularization parameter λ controls the matrix conditioning:

- if $\lambda = 0$: ordinary linear regression
- if $\lambda \to \infty$: $\hat{w}_n \to 0$

The Lasso: how to choose among a large set of variables with few observations

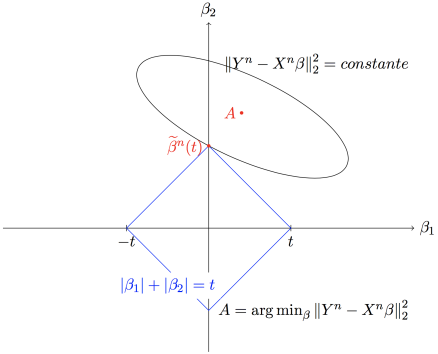
The Lasso corresponds to L_1 regularization:

$$\hat{w}_n = \arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (Y_i - w^\top X_i)^2 \; + \; \lambda \sum_{i=1}^{d} |w_i|$$

- Powerful if $d \gg n$: many potential variables, few observations
- $\hat{w}_n$ is sparse: most of its values will be 0 $\rightarrow$ can be used to choose variables

Other formulation of the Lasso:
$\exists \beta > 0$ such that

$$\hat{w}_n \in \arg\min_{\|w\|_1 \leqslant \beta} \frac{1}{n} \sum_{i=1}^{n} (Y_i - w^\top X_i)^2$$



β_2
$\|Y^n - X^n\beta\|_2^2 = constante$
A
$\widetilde{\beta}^n(t)$
β_1
$-t$
t
$|\beta_1| + |\beta_2| = t$
$A = \arg\min_\beta \|Y^n - X^n\beta\|_2^2$

The Lasso: how to choose among a large set of variables with few observations

The Lasso corresponds to L_1 regularization:

$$\hat{w}_n = \arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (Y_i - w^\top X_i)^2 \ + \ \lambda \sum_{i=1}^{d} |w_i|$$

👍 Powerful if $d \gg n$: many potential variables, few observations
👍 $\hat{w}_n$ is sparse: most of its values will be 0 $\rightarrow$ can be used to choose variables

👎 **The Lasso is biased**: $\hat{w}_n^\top X \neq \mathbb{E}[Y|X]$. Hence, it is better to:

$$\begin{tikzcd} \text{Perform Lasso} \arrow[d] \\ \text{Choose variables with } \hat{w}_i > 0 \arrow[d] \\ \text{Perform Ridge on this sub-model only} \end{tikzcd}$$

Another solution is Elastic Net:

$$\hat{w}_n = \arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (Y_i - w^\top X_i)^2 \ + \ \lambda_1 \sum_{i=1}^{d} |w_i| + \lambda_2 \sum_{i=1}^{d} w_i^2$$

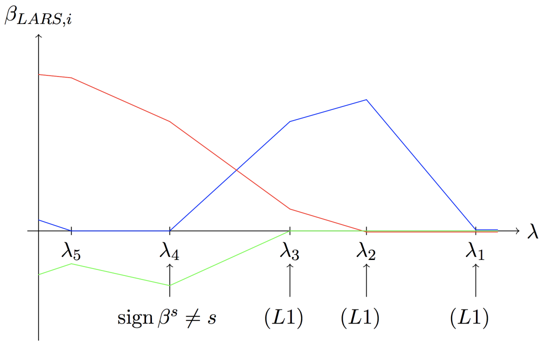
Many extensions of the Lasso exist: Group Lasso,...

Lasso: the regularization path

The Lasso corresponds to L_1 regularization:

$$\hat{w}_n = \arg\min_{w \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} (Y_i - w^\top X_i)^2 \; + \; \lambda \sum_{i=1}^{d} |w_i|$$

Plot of the evolution of the coefficients of $\hat{w}_n$ as a function of λ:

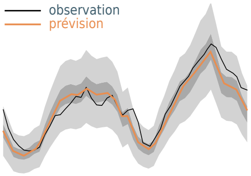


Probabilistic prediction

In some situation, we are not interested by prediction the average case $\mathbb{E}[Y|X]$ only, but by the distribution of $Y|X$. $\rightarrow$ give a measure of uncertainty of our prediction

Solution: modify the loss function:

- square loss $\ell(a, y) = (a - y)^2$: prediction of the expected value
- absolute loss $\ell(a, y) = |a - y|$: prediction of the median (50% to be above Y, and 50% chance to be below)
- pinball loss $\ell(a, y) = (a - y)(\tau - \mathbb{1}_{a<y})$: prediction of the τ-quantile ($(1 - \tau)$ chance to be above Y and τ chance to be below)

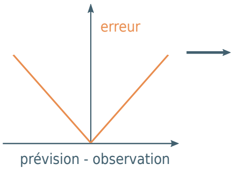


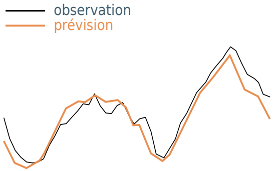
Probabilistic prediction

In some situation, we are not interested by prediction the average case $\mathbb{E}[Y|X]$ only, but by the distribution of $Y|X$. $\rightarrow$ give a measure of uncertainty of our prediction

Solution: modify the loss function:

- square loss $\ell(a, y) = (a - y)^2$: prediction of the expected value
- absolute loss $\ell(a, y) = |a - y|$: prediction of the median
  (50% to be above Y, and 50% chance to be below)
- pinball loss $\ell(a, y) = (a - y)(\tau - \mathbb{1}_{a<y})$: prediction of the τ-quantile
  ($(1 - \tau)$ chance to be above Y and τ chance to be below)



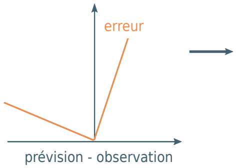


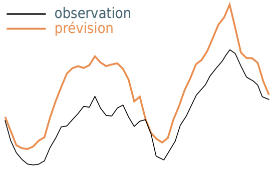
Probabilistic prediction

In some situation, we are not interested by prediction the average case $\mathbb{E}[Y|X]$ only, but by the distribution of $Y|X$. $\rightarrow$ give a measure of uncertainty of our prediction

Solution: modify the loss function:

- square loss $\ell(a, y) = (a - y)^2$: prediction of the expected value
- absolute loss $\ell(a, y) = |a - y|$: prediction of the median
  (50% to be above Y, and 50% chance to be below)
- pinball loss $\ell(a, y) = (a - y)(\tau - \mathbb{1}_{a<y})$: prediction of the τ-quantile
  ($(1 - \tau)$ chance to be above Y and τ chance to be below)



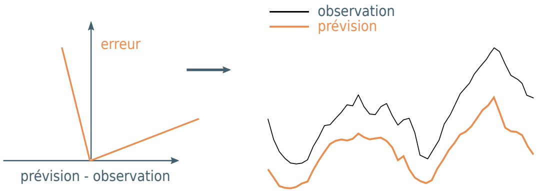


Probabilistic prediction

In some situation, we are not interested by prediction the average case $\mathbb{E}[Y|X]$ only, but by the distribution of $Y|X$. $\rightarrow$ give a measure of uncertainty of our prediction

Solution: modify the loss function:

- square loss $\ell(a,y) = (a-y)^2$: prediction of the expected value
- absolute loss $\ell(a,y) = |a-y|$: prediction of the median
  (50% to be above Y, and 50% chance to be below)
- pinball loss $\ell(a,y) = (a-y)(\tau - \mathbb{1}_{a<y})$: prediction of the τ-quantile
  ($(1-\tau)$ chance to be above Y and τ chance to be below)

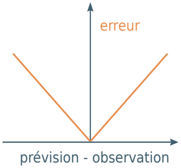


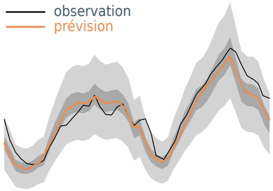
Probabilistic prediction

In some situation, we are not interested by prediction the average case $\mathbb{E}[Y|X]$ only, but by the distribution of $Y|X$. $\rightarrow$ give a measure of uncertainty of our prediction

Solution: modify the loss function:

- square loss $\ell(a, y) = (a - y)^2$: prediction of the expected value
- absolute loss $\ell(a, y) = |a - y|$: prediction of the median
  (50% to be above Y, and 50% chance to be below)
- pinball loss $\ell(a, y) = (a - y)(\tau - \mathbb{1}_{a<y})$: prediction of the τ-quantile
  ($(1 - \tau)$ chance to be above Y and τ chance to be below)





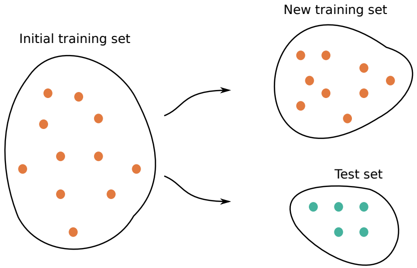
Outline

How to choose the parameters? Test set

All the methods in machine learning depend on learning parameters.

How to choose them? First solution: use a test set.

- randomly choose 70% of the data to be in the training set
- the remainder is a test set



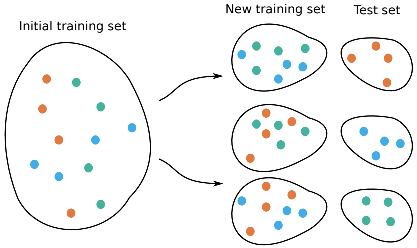
We choose the parameter with the smallest error on the test set.

👍 very simple
👎 waste data: the best method is fitted only with 70% of the data
👎 with bad luck the test set might be lucky or unlucky

How to choose the parameters? Cross-validation

Cross-validation:

- randomly break data into K groups
- for each group, use it as a test set and train the data on the $(K-1)$ other groups



We choose the parameter with the smallest average error on the test sets.

👍 only $1/K$ of the data lost for training

👎 K times more expensive

In practice: choose $K \approx 10$.

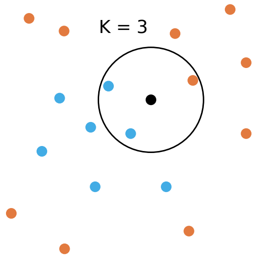
Outline

K-Nearest Neighbors

Classify data based on similarity with neighbors.

When observing a new input x, find the *k*-closest training data points to x and for

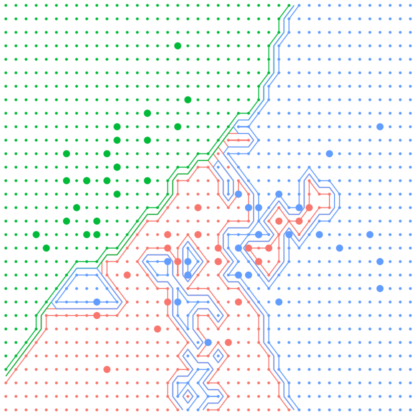
- classification: predict the most frequently occuring class
- regression: predict the average value



K-Nearest Neighbors

Classify data based on similarity with neighbors.

When observing a new input x, find the *k-closest training data* points to x and for
- classification: predict the most frequently occuring class
- regression: predict the average value

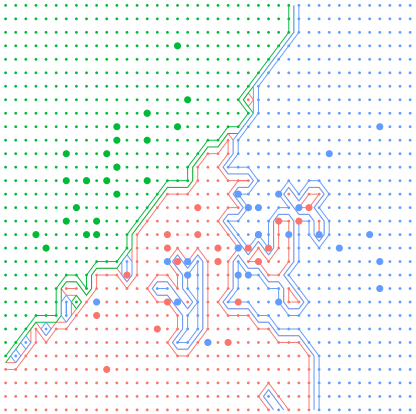


$K = 1$

K-Nearest Neighbors

Classify data based on similarity with neighbors.

When observing a new input x, find the k-closest training data points to x and for
- classification: predict the most frequently occuring class
- regression: predict the average value



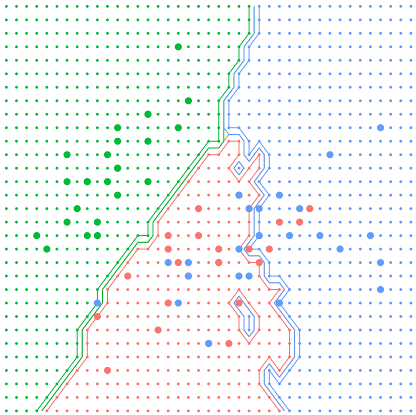
$K = 3$

K-Nearest Neighbors

Classify data based on similarity with neighbors.

When observing a new input x, find the *k*-closest training data points to x and for

- classification: predict the most frequently occuring class
- regression: predict the average value



$K = 20$

K-Nearest Neighbors

Advantages:

- No optimization or training
- Easy to implement
- Can get very good performance

Drawbacks:

- Slow at query time: must pass through all training data at each
- Easily fooled by irrelevant inputs
- Bad for high-dimensional data ($d > 20$)

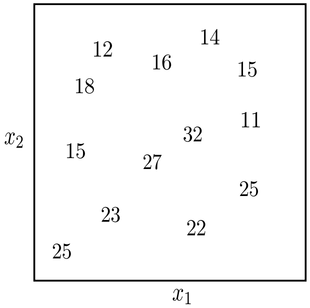
Difficulties:

- choice of K
- what distance for complex data?

Decision Tree

Introduced by Breiman et al. 1984

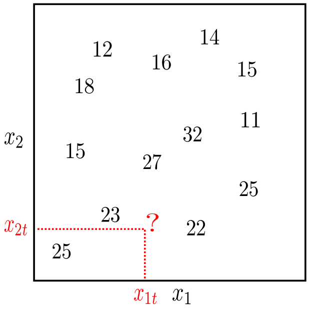
Idea: partitioned the input space in an inductive and diadic fashion.



Decision Tree

Introduced by Breiman et al. 1984

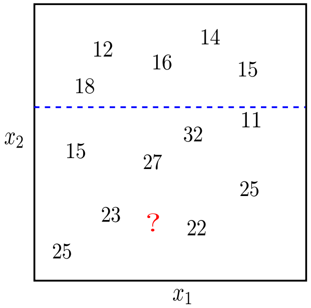
Idea: partitioned the input space in an inductive and diadic fashion.



Decision Tree

Introduced by Breiman et al. 1984

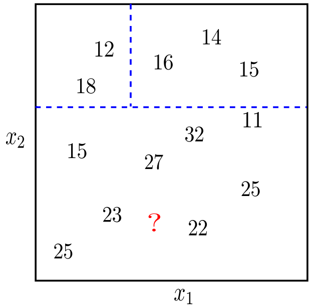
Idea: partitioned the input space in an inductive and diadic fashion.



Decision Tree

Introduced by Breiman et al. 1984

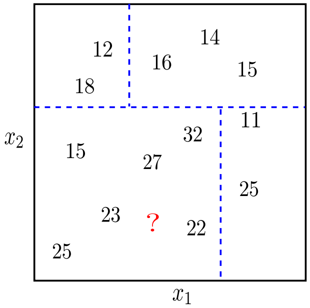
Idea: partitioned the input space in an inductive and diadic fashion.



Decision Tree

Introduced by Breiman et al. 1984

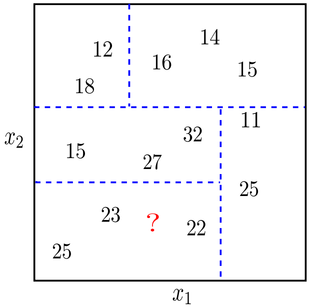
Idea: partitioned the input space in an inductive and diadic fashion.



Decision Tree

Introduced by Breiman et al. 1984

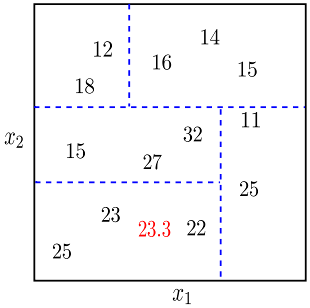
Idea: partitioned the input space in an inductive and diadic fashion.



Decision Tree

Introduced by Breiman et al. 1984

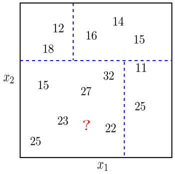
Idea: partitioned the input space in an inductive and diadic fashion.



Decision Tree

Introduced by Breiman et al. 1984

Idea: partitioned the input space in an inductive and diadic fashion.



To construct the tree, we need to answer two questions:

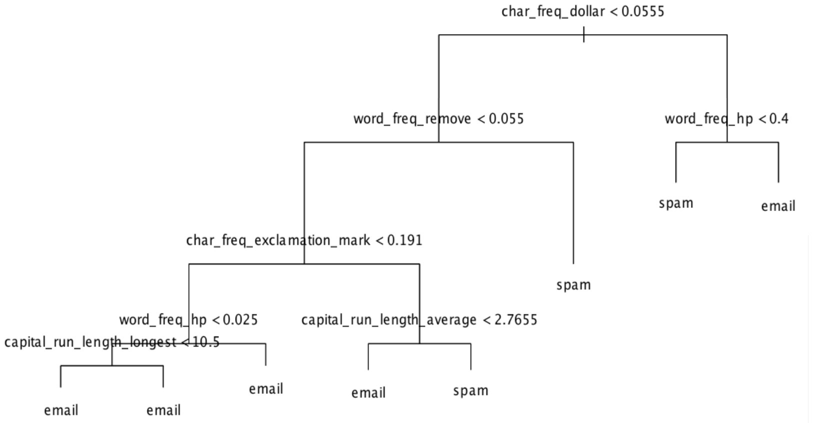
- **Location of the cuts:** which variable, what threshold?
  $\rightarrow$ minimize the inter-groups variance
- **Depth of the tree:** when do we stop? Over-fitting risk!
  - continue while variance decreases enough
  - pruning: build a large tree and prune it by minimizing a penalized error:

$$\text{Test error}(T) + \lambda \text{size}(T)$$

👍 **Advantage:** interpretable

👎 **Drawbacks:** instable (butterfly effect), computational cost

Decision trees: example for spam detection


char_freq_dollar < 0.0555
word_freq_remove < 0.055
word_freq_hp < 0.4
spam
email
char_freq_exclamation_mark < 0.191
spam
word_freq_hp < 0.025
capital_run_length_average < 2.7655
capital_run_length_longest < 10.5
email
email
spam
email
email

Ensemble algorithms

Ensemble algorithms are based on the following idea: averaging adds stability.

Example: Assume that $Y \in \{0,1\}$ and that you have K independent classification methods $f_k, k = 1, \dots, K$ such that $\mathbb{P}(f_k(X) \neq Y) \leqslant \varepsilon$. Then from Hoeffding's inequality:

$$\mathbb{P}\big(\text{majority voting of } f_k(X) \neq Y\big) \lesssim e^{-K\varepsilon^2}$$

$\rightarrow$ exponential decrease to 0!

Idea: build base methods as independent as possible and average them.

1. split the training set into K subsets of size n/K
2. train a different "base learner" on each subset

Issue: n may be too small $\rightarrow$ not enough data per "base learner" $\rightarrow$ Bagging

Bagging (Boostrap) AGGregatING

Introduced by Breiman 1996

To fit a new "base learner"

1. sample n data with replacement from the training set
2. train the "base learner" on this subset of observations

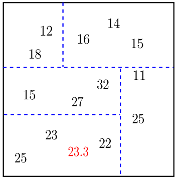
Each base learner gets $\approx$ 36.8% of the data. Remaining points are called "out-of-bag".

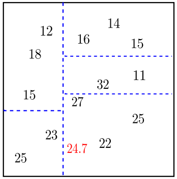
We can estimate the performance of each base learner with the out-of-bag error

Random Forests

Introduced by Breiman 2001

Idea: build many (≈ 400) random decisions trees and average their predictions.





predict $\frac{24.7+23.3}{2} = 24$

How to build uncorrelated trees?

- bagging: each tree is built over sample of training points
- random choice of the covariate to cut

Advantages:

- No over-fitting (the more trees we build, the better)
- Easy computation of an error estimate: "out-of-bag": no-need of cross validation
- efficient for small data sets n

Drawbacks: computational cost, black box

Variable selection with random forests

Random forests is a powerful tool to order explanatory variables by predictive importance.

First, we build the forest and compute E its “out-of-bag” error.

For each variable X_i, we compute its importance as follows

- randomly permute the values of X_i among training data
- update the “out-of-bag” error E_i
- get the importance of X_i given by $E_i - E$

Outline

Deep learning

Successful application domains: Image (object recognition), Audio (speech recognition), Text (parsing)

What is it used for?

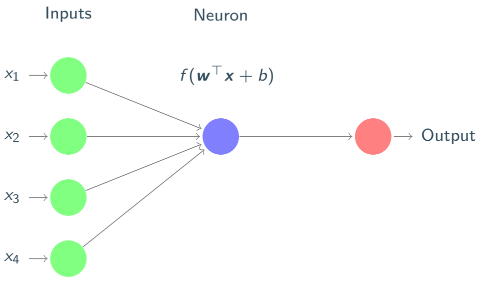
- Prediction: regression, classification,
- Generation: denoising, reconstruction of partial/missing data, generation of new data

What is it?

- Models with graphs structure (networks) with multiple layers (deep)
- Typically non-linear models

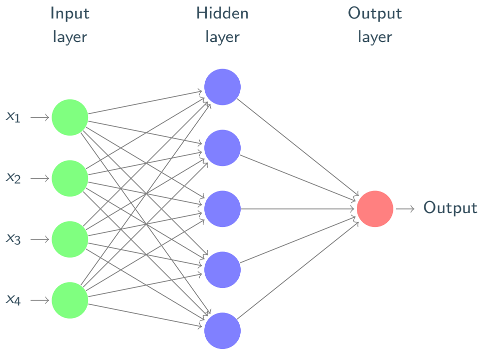
Deep neural network

- A **neuron** is a non-linear transformation of a linear combination of inputs.
- A **column of neurons** taking the same input $\boldsymbol{x}$ forms a new layer



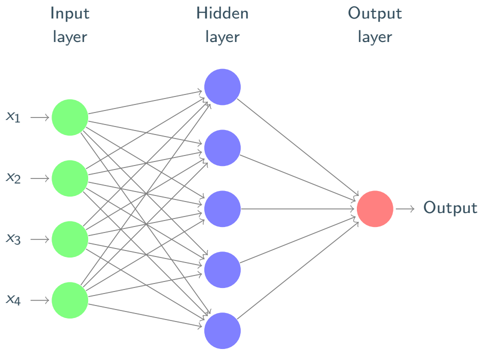
Deep neural network

- A **neuron** is a non-linear transformation of a linear combination of inputs.
- A **column of neurons** taking the same input ***x*** forms a new layer



Deep neural network

- A **neuron** is a non-linear transformation of a linear combination of inputs.
- A **column of neurons** taking the same input $\boldsymbol{x}$ forms a new layer

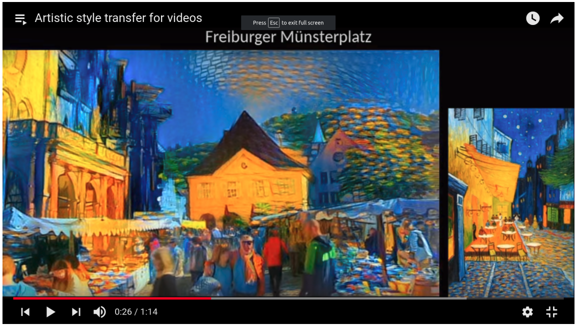


Training a neural networks: backpropagation (gradient descent using $\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$).

Avoid over-fitting: dropout [Hinton et al. 2012]

Build data-specific models: convolutional neural networks [LeCun et al. 1998]

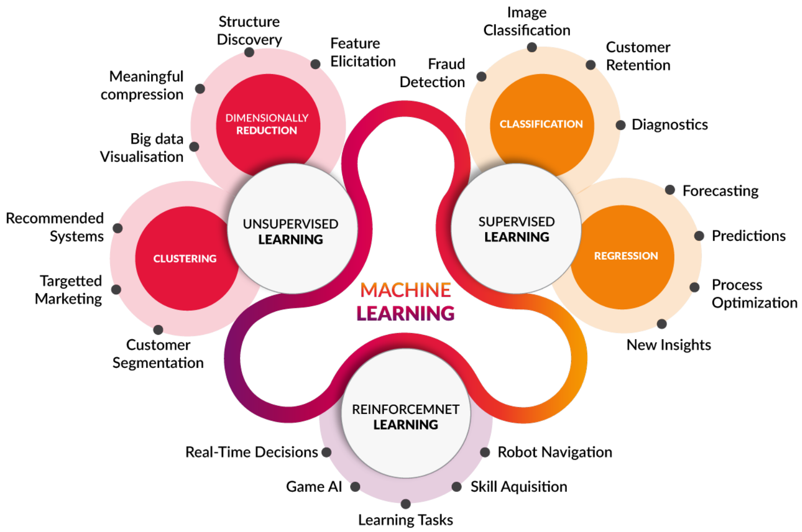
What can you do with DNN?


Artistic style transfer for videos
Press Esc to exit full screen
Freiburger Münsterplatz
0:26 / 1:14

`https://youtu.be/Khuj4ASldmU`

Unsupervised learning

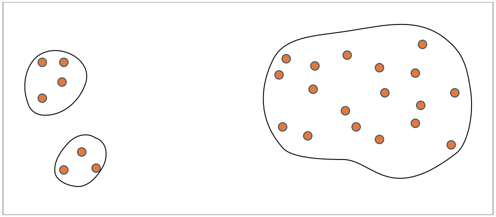
Overview of Machine Learning



Outline

Clustering

- **Idea:** group together similar instances
- Requires data but no labels
- Useful when you don't know what you are looking for



The similarity is measured by a metric (ex: $\|x - y\|_2^2$).

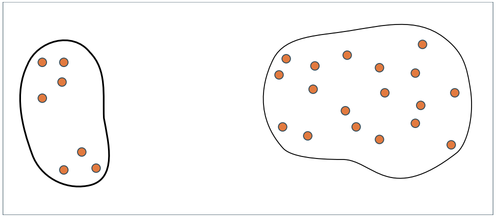
The results crucially depends on the metric choice: depends on data.

Types of clustering algorithms:

- model based clustering (mixture of Gaussian)
- hierarchical clustering: a hierarchy of nested clusters is build using divisive or agglomerative approach
- Flat clustering: no hierarchy (k-means, spectral clustering)

Clustering

- **Idea:** group together similar instances
- Requires data but no labels
- Useful when you don't know what you are looking for



The similarity is measured by a metric (ex: $\|x - y\|_2^2$).

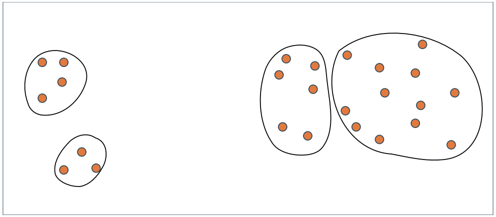
The results crucially depends on the metric choice: depends on data.

Types of clustering algorithms:

- model based clustering (mixture of Gaussian)
- hierarchical clustering: a hierarchy of nested clusters is build using divisive or agglomerative approach
- Flat clustering: no hierarchy (k-means, spectral clustering)

Clustering

- **Idea:** group together similar instances
- Requires data but no labels
- Useful when you don't know what you are looking for



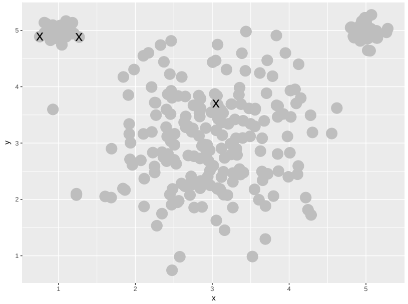
The similarity is measured by a metric (ex: $\|x - y\|_2^2$).

The results crucially depends on the metric choice: depends on data.

Types of clustering algorithms:

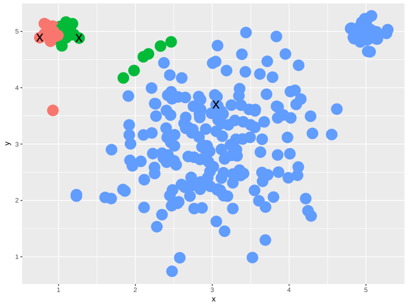
- model based clustering (mixture of Gaussian)
- hierarchical clustering: a hierarchy of nested clusters is build using divisive or agglomerative approach
- Flat clustering: no hierarchy (k-means, spectral clustering)

K-means



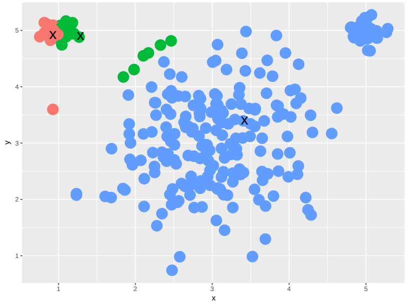
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



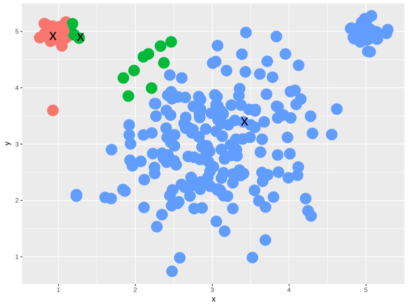
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



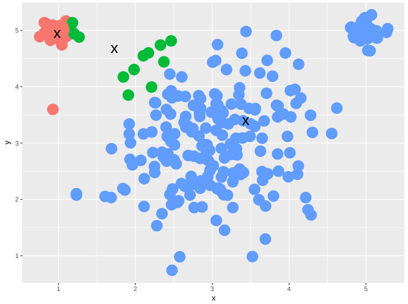
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



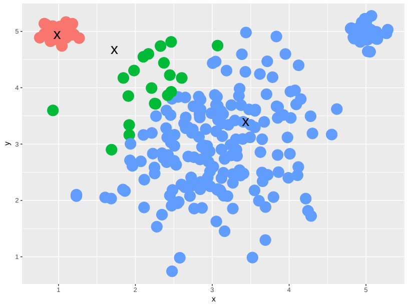
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



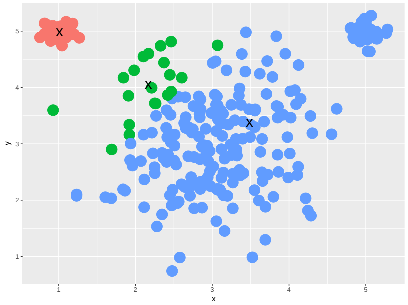
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



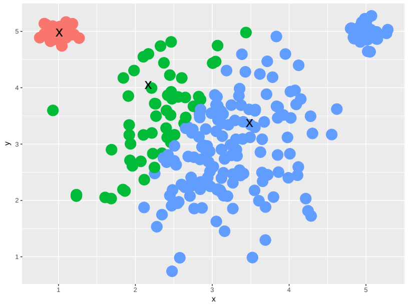
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



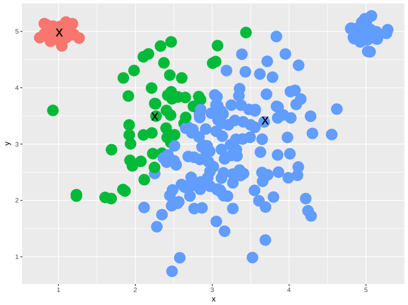
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



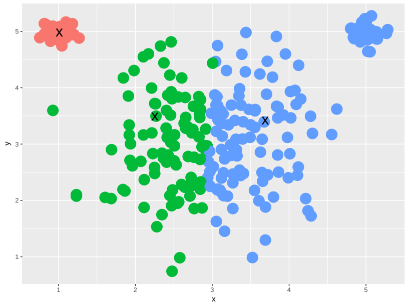
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



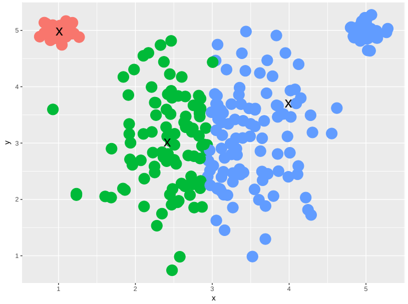
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



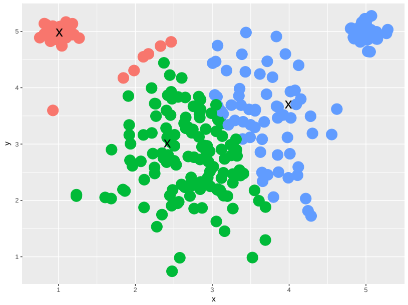
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



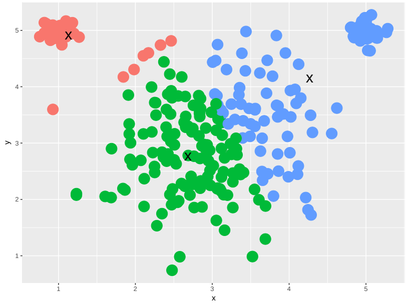
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



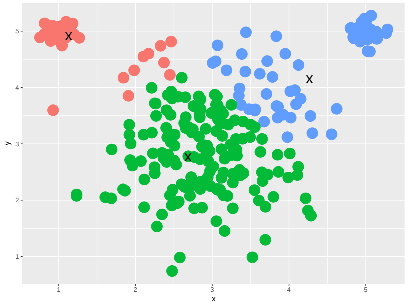
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



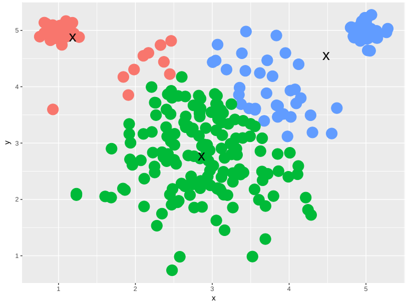
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



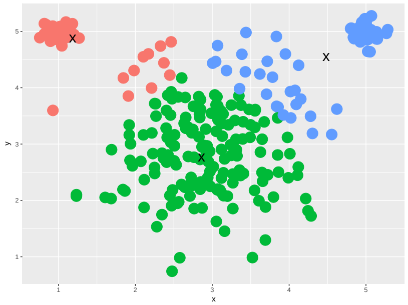
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



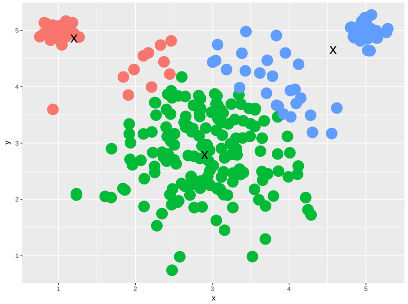
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



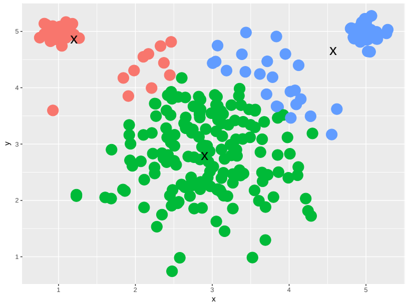
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



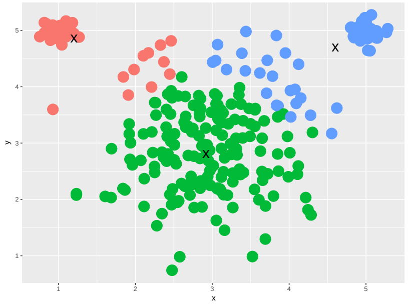
- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



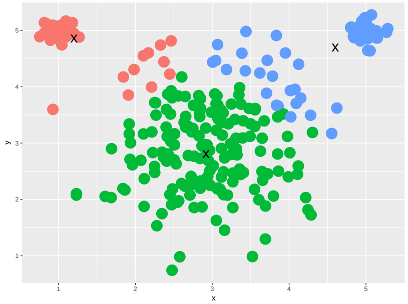
- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means



- **Initialization**: sample K points as cluster centers
- **Alternate:**
  1. Assign points to closest center
  2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

K-means

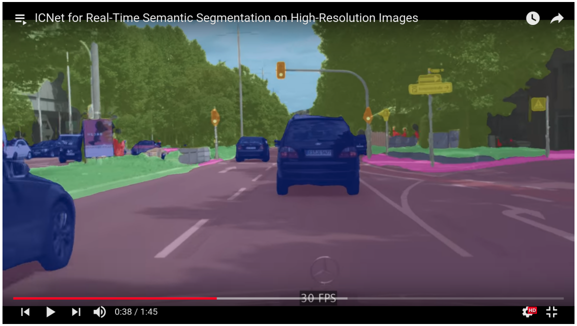


- **Initialization**: sample K points as cluster centers
- **Alternate:**
    1. Assign points to closest center
    2. Update cluster to the averaged of its assigned points
- **Stop** when no point's assignment change.

Guaranteed to converge in a finite number of iterations.

Initialization is crucial.

Example: Segmentation


ICNet for Real-Time Semantic Segmentation on High-Resolution Images
30 FPS
0:38 / 1:45

`https://youtu.be/qWl9idsCuLQ`

Outline

Principal component analysis

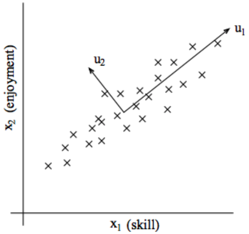
Assume that you have a data matrix (with column-wise zero empirical mean)

$$X := \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,p} \\ \vdots & \vdots & \dots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,p} \end{bmatrix}$$

If p is large, some columns (i.e., explanatory variables) may be linearly correlated.

- **bad statistical property**: risk minimization not identifiable, the covariance matrix $(X^\top X)$ is not invertible $\rightarrow$ unstable estimators
- **bad computational property**: we need to store $p \gg 1$ columns with redundant information

PCA reduces the p dimensions of the data set X down to k principal components.



Principal component analysis

Assume that you have a data matrix (with column-wise zero empirical mean)

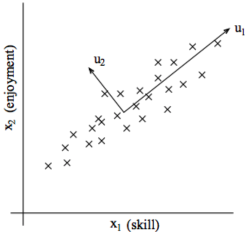
$$X := \begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,p} \\ \vdots & \vdots & \dots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,p} \end{bmatrix}$$

How does it work?

1. Find the vector u_1 such that the projection of the data on u has the greatest variance.

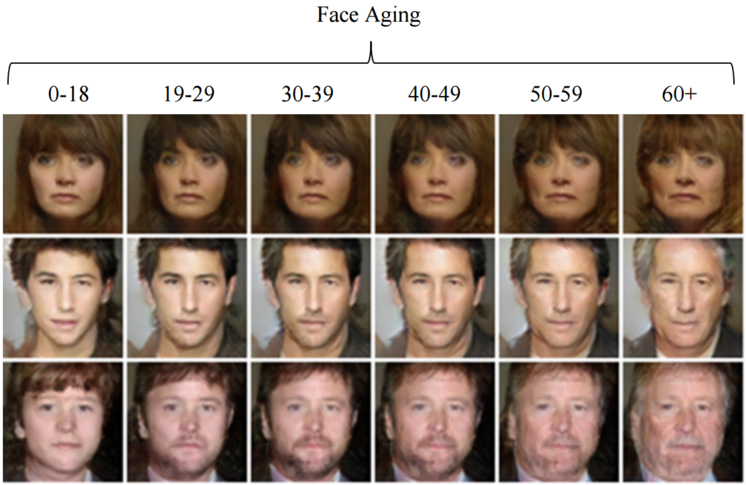
$$u_1 := \underset{\|\boldsymbol{u}\|=1}{\arg\max} \|X^\top \boldsymbol{u}\|^2 = u^\top X^\top X u$$

   $\Rightarrow$ this is the principal eigenvector of $X^\top X$.
2. More generally, if we wish a k-dimensional subspace we choose $u_1, \dots, u_k$ the top k eigenvectors of $X^\top X$.
3. The u_i form a new orthogonal basis of the data

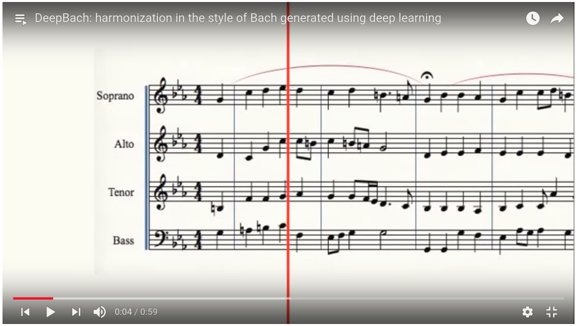


Generative Models

Generative Models

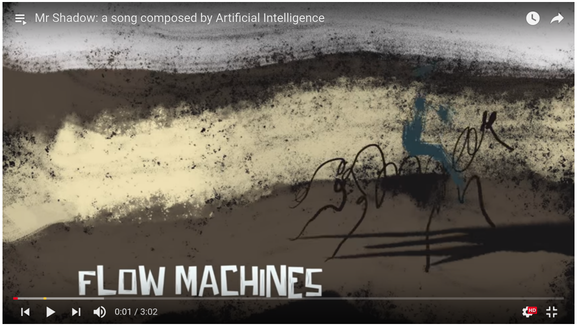


Generative Models


DeepBach: harmonization in the style of Bach generated using deep learning
Soprano
Alto
Tenor
Bass
0:04 / 0:59

https://youtu.be/QiBM7-5hA6o

Generative Models



`https://youtu.be/lcGYEXJqun8`

Planning of the class

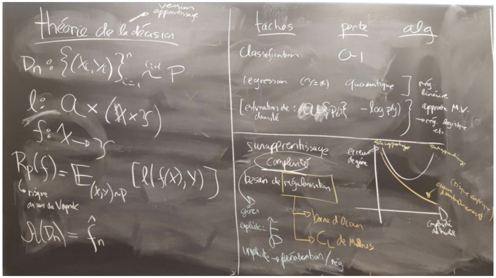
Objective of the class

The goal of the class is to introduce the basics of machine learning. We will mix:

- **theory:** some theorems will be proved!
- **practice**: some algorithms will be implemented on real data

Disclaimer: at the end of the class, you will most likely not be able to reproduce all examples seen in this introduction!

Typical session will be a lecture from 8h30 to 10h20, followed by a 20min break and the practical work (PW) from 10h40 to 12h30. 2021: Online inverted classroom



Planning

Voir `https://www.di.ens.fr/appstat/spring-2021/`

For next week...

Prepare your personal laptops in practical sessions with python (jupyter, anaconda) working on it.

Check the crash-test Jupyter notebook:

`https://www.di.ens.fr/appstat/spring-2020/TP/TD0-prerequisites/crash_test.ipynb`

References

L. Bottou, F. E. Curtis, and J. Nocedal. “Optimization methods for large-scale machine learning”. In: *arXiv preprint arXiv:1606.04838* (2016).

L. Breiman. “Bagging predictor”. In: *Machine Learning* 24.2 (1996), pp. 123–140.

L. Breiman. “Random Forests”. In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32.

L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and Regression Trees*. Belmont, CA: Wadsworth International Group, 1984.

N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

T. J. Hastie, R. J. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer New York Inc., 2001.

G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. “Improving neural networks by preventing co-adaptation of feature detectors”. In: *arXiv preprint arXiv:1207.0580* (2012).

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

Wikipedia. The free encyclopedia.