



Apprentissage Statistique

Statistical and Machine Learning

Benjamin Guedj, Ph.D.

Équipe-projet Modal
Inria Lille - Nord Europe

2016–2017

- ▶ benjamin.guedj@inria.fr
- ▶ <https://bguedj.github.io>

14h cours + 6h TD

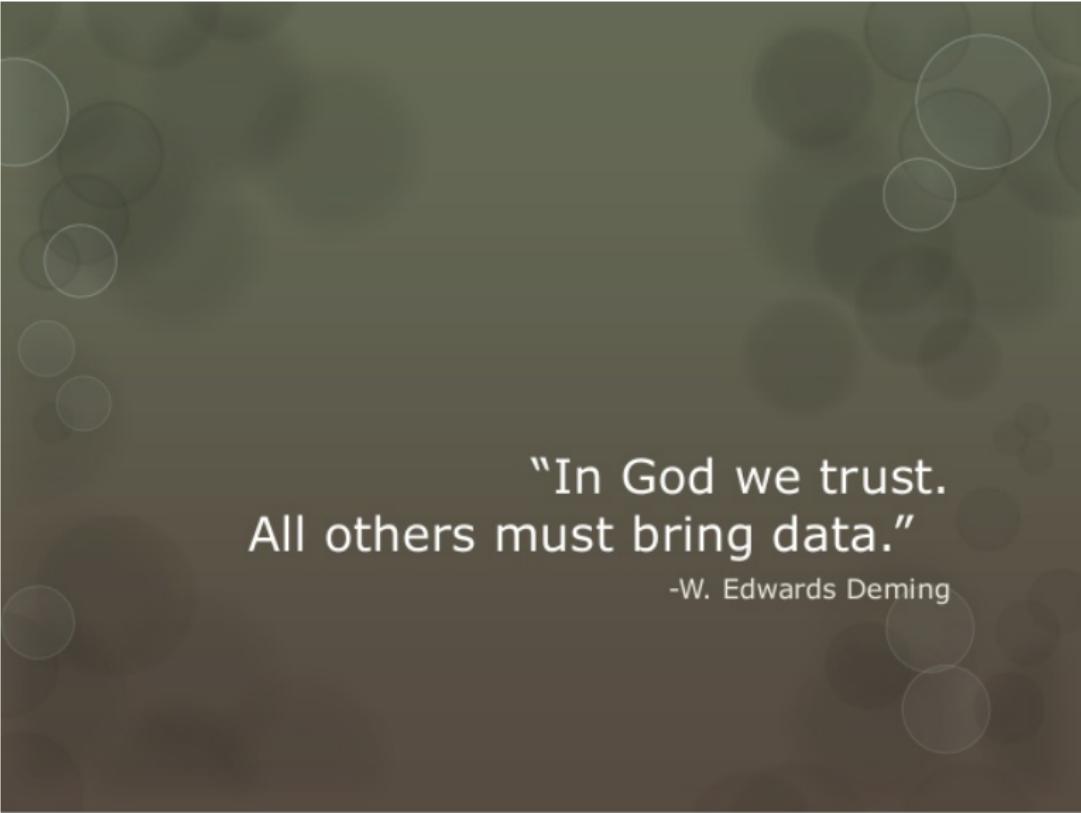
2h contrôle (C) ◇ 2h examen (E)

Note finale

$$N = \max \left(E, \frac{E + C}{2} \right).$$

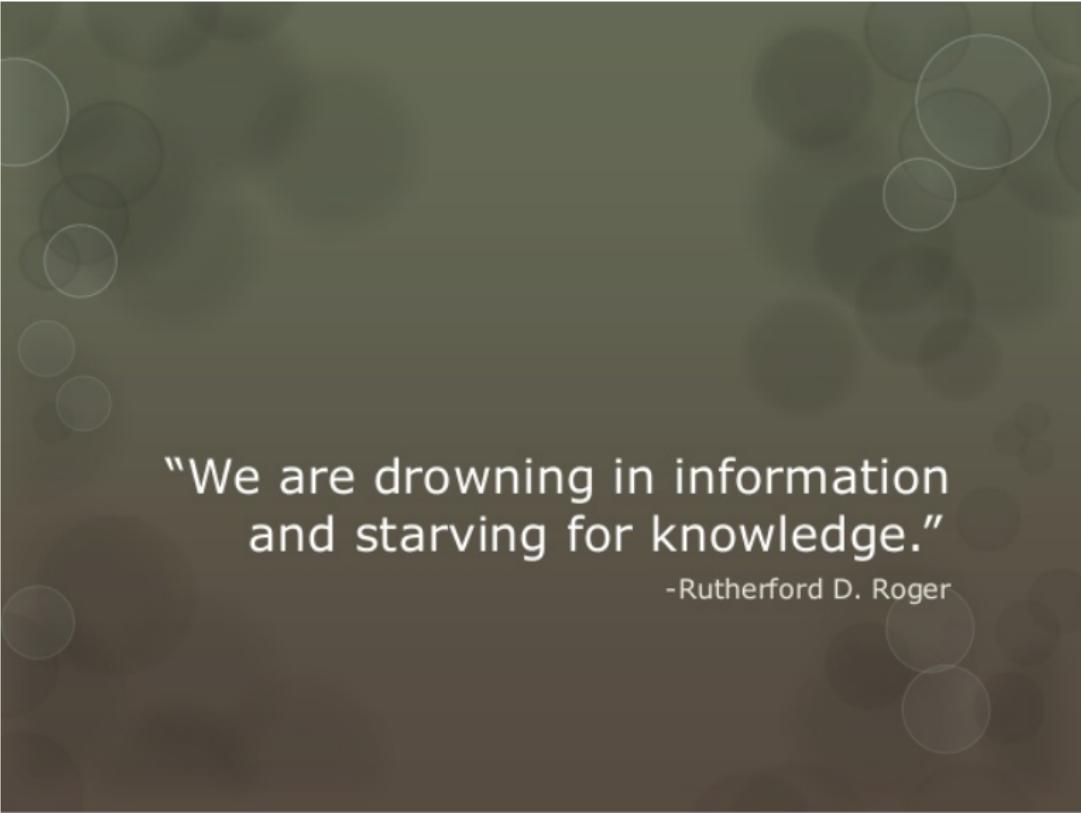
Une intelligence artificielle ?

Introduction



“In God we trust.
All others must bring data.”

-W. Edwards Deming



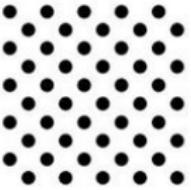
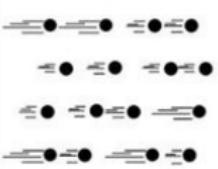
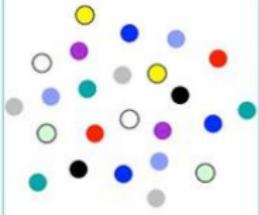
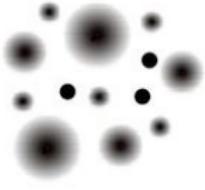
“We are drowning in information
and starving for knowledge.”

-Rutherford D. Roger

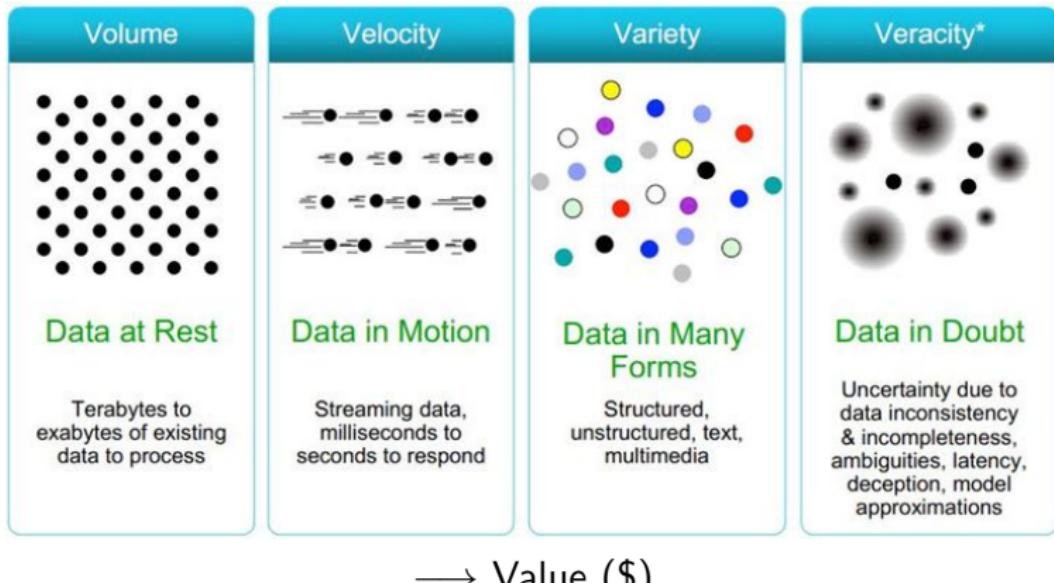
It is vital to remember
that information - in the
sense of raw data - is not
knowledge, that
knowledge is not wisdom,
and that wisdom is not
foresight. But information
is the first essential step
to all of these.

Arthur C Clarke

Big Data 4 V's

| Volume | Velocity | Variety | Veracity* |
|---|---|---|---|
|  |  |  |  |
| Data at Rest Terabytes to exabytes of existing data to process | Data in Motion Streaming data, milliseconds to seconds to respond | Data in Many Forms Structured, unstructured, text, multimedia | Data in Doubt Uncertainty due to data inconsistency & incompleteness, ambiguities, latency, deception, model approximations |

Big Data 4 V's



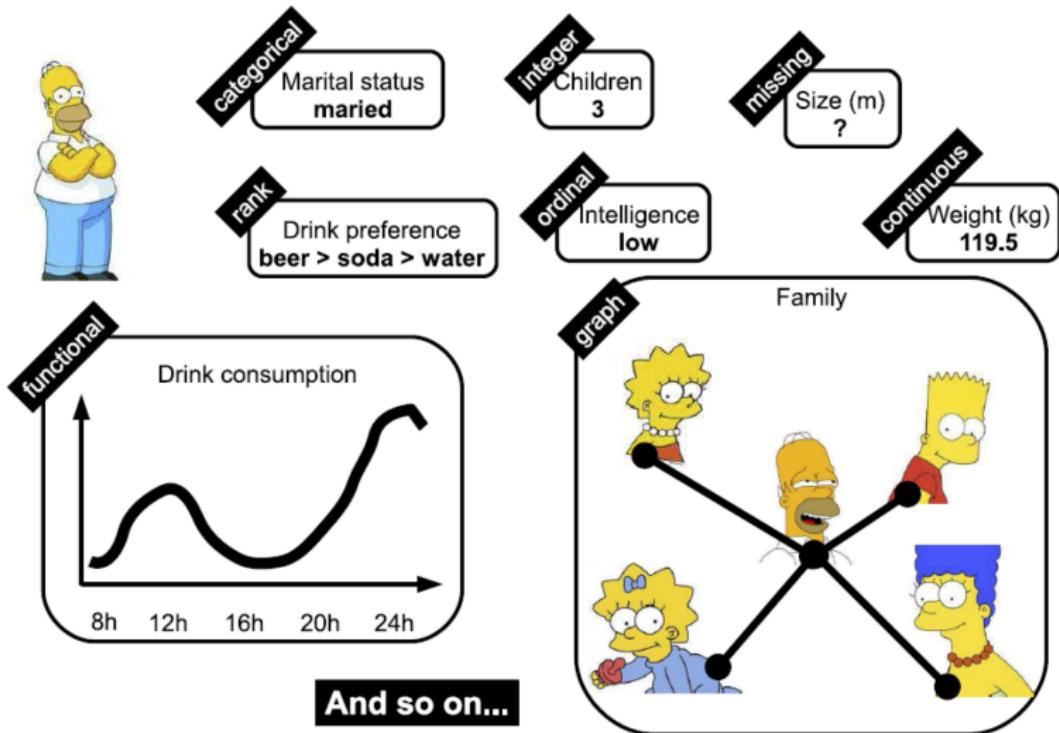
Data Scientists: 100,000 jobs by 2020. Demand is expected to exceed supply by 50 to 60% (McKinsey, 2015)

Volume / Velocity

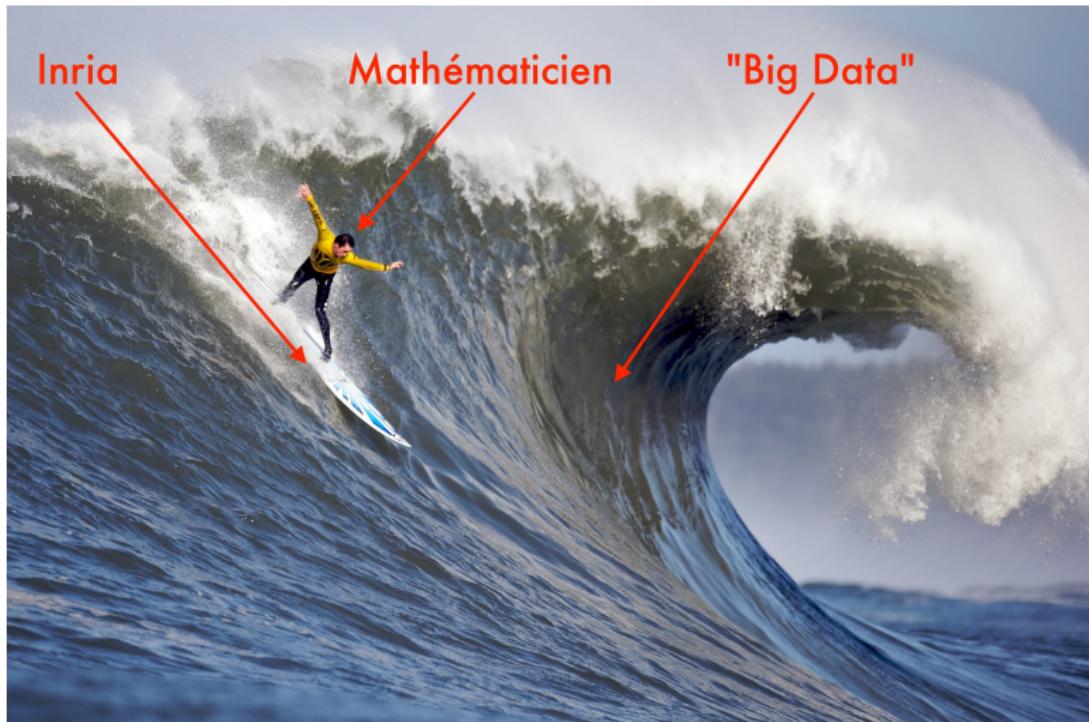
| i | 3 | 6 | 9 | 12 | 15 | 18 | 21 | 24 | |
|--------|------|------|------|------|------|-----|------|-------|-------|
| 10^i | kilo | mega | giga | tera | peta | exa | zeta | yotta | bytes |

- ▶ Google: 24 petabytes/day
- ▶ Facebook: 10 terabytes/day
- ▶ Twitter: 7 terabytes/day
- ▶ Large Hadron Collider: 40 terabytes/second
- ▶ Google Street View: 20 petabytes
- ▶ AT&T network: 30 petabytes/day
- ▶ Human brain may store about 2.5 petabytes of binary data
- ▶ ...

Variety / Veracity



My job (allegory)



Data sources

<https://www.kaggle.com>

<https://archive.ics.uci.edu/ml/datasets.html>

<https://www.data.gov>

<https://data.gov.uk>

<https://www.data.gouv.fr/fr/>

<http://www.census.gov/data.html>

<http://data.europa.eu/euodp/en/data/>

<http://www.healthdata.gov>

<https://aws.amazon.com/fr/datasets/>

<https://www.gapminder.org/data/>

<https://www.google.com/trends/explore>

<https://www.google.com/finance>

...

Boston Housing



Attribute Information:

1. CRIM: per capita crime rate by town
2. ZN: proportion of residential land zoned for lots over 25,000 sq.ft.
3. INDUS: proportion of non-retail business acres per town
4. CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
5. NOX: nitric oxides concentration (parts per 10 million)
6. RM: average number of rooms per dwelling
7. AGE: proportion of owner-occupied units built prior to 1940
8. DIS: weighted distances to five Boston employment centres
9. RAD: index of accessibility to radial highways
10. TAX: full-value property-tax rate per \$10,000
11. PTRATIO: pupil-teacher ratio by town
12. B: $1000(Bk - 0.63)^2$ where Bk is the proportion of blacks by town
13. LSTAT: % lower status of the population
14. MEDV: Median value of owner-occupied homes in \$1000's

Wine Quality



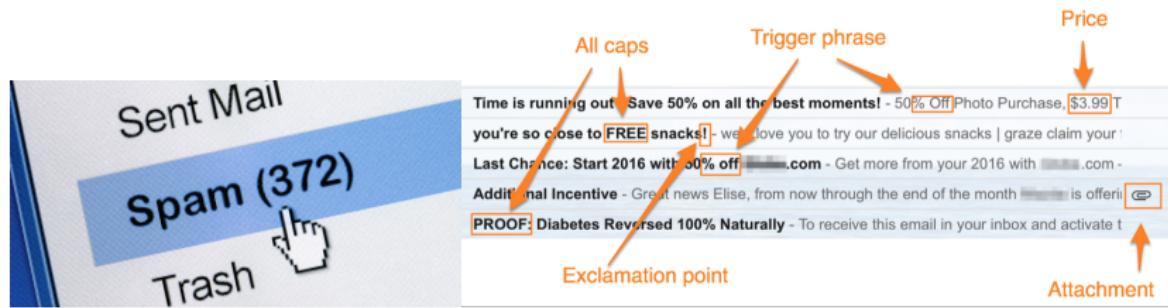
Attribute Information:

For more information, read [Cortez et al., 2009].
Input variables (based on physicochemical tests):

- 1 - fixed acidity
- 2 - volatile acidity
- 3 - citric acid
- 4 - residual sugar
- 5 - chlorides
- 6 - free sulfur dioxide
- 7 - total sulfur dioxide
- 8 - density
- 9 - pH
- 10 - sulphates
- 11 - alcohol

Output variable (based on sensory data):
12 - quality (score between 0 and 10)

Email Spam Detection



Used Cars' Prices

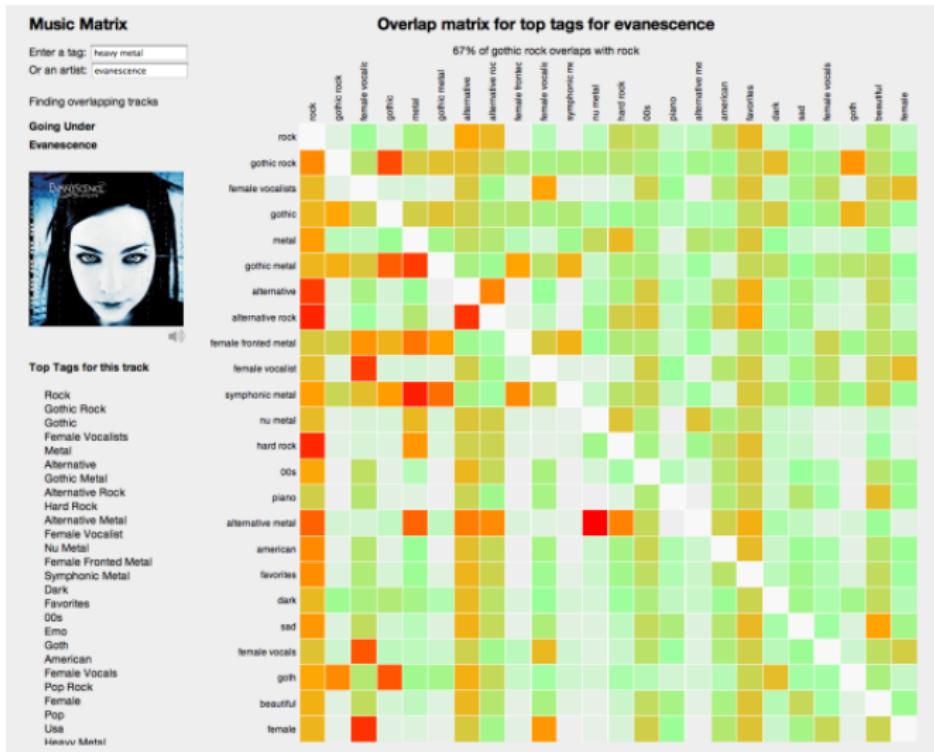


Attribute Information:

Attribute: Attribute Range

1. symboling: -3, -2, -1, 0, 1, 2, 3.
2. normalized-losses: continuous from 65 to 256.
3. make:
alfa-romero, audi, bmw, chevrolet, dodge, honda, isuzu, jaguar, mazda, mercedes-benz, mercury, mitsubishi, nissan, peugeot, plymouth, porsche, renault, saab, subaru, toyota, volkswagen, volvo
4. fuel-type: diesel, gas.
5. aspiration: std, turbo.
6. num-of-doors: four, two.
7. body-style: hardtop, wagon, sedan, hatchback, convertible.
8. drive-wheels: 4wd, fwd, rwd.
9. engine-location: front, rear.
10. wheel-base: continuous from 86.6 120.9.
11. length: continuous from 141.1 to 208.1.
12. width: continuous from 60.3 to 72.3.
13. height: continuous from 47.8 to 59.8.
14. curb-weight: continuous from 1488 to 4066.
15. engine-type: dohc, dohcvt, i, ohc, ohcf, ohcv, rotor.
16. num-of-cylinders: eight, five, four, six, three, twelve, two.
17. engine-size: continuous from 61 to 326.
18. fuel-system: 1bbl, 2bbl, 4bbl, idl, mfi, mpfi, spdi, spfi.
19. bore: continuous from 2.54 to 3.94.
20. stroke: continuous from 2.07 to 4.17.
21. compression-ratio: continuous from 7 to 23.
22. horsepower: continuous from 48 to 288.
23. peak-rpm: continuous from 4150 to 6600.
24. city-mpg: continuous from 13 to 49.
25. highway-mpg: continuous from 16 to 54.
26. price: continuous from 5118 to 45400.

Million Song Dataset



Chess (1997)



LOOKS LIKE COMPUTERS
WILL BEAT HUMANS AT
GO PRETTY SOON.

WOW.
THAT'S THE LAST
OF THE BIG ONES.

YEAH.



WELL, AT LEAST HUMANS
ARE STILL BETTER AT, UH,
COMING UP WITH REASSURING
PARABLES ABOUT THINGS
HUMANS ARE BETTER AT?

HMM.



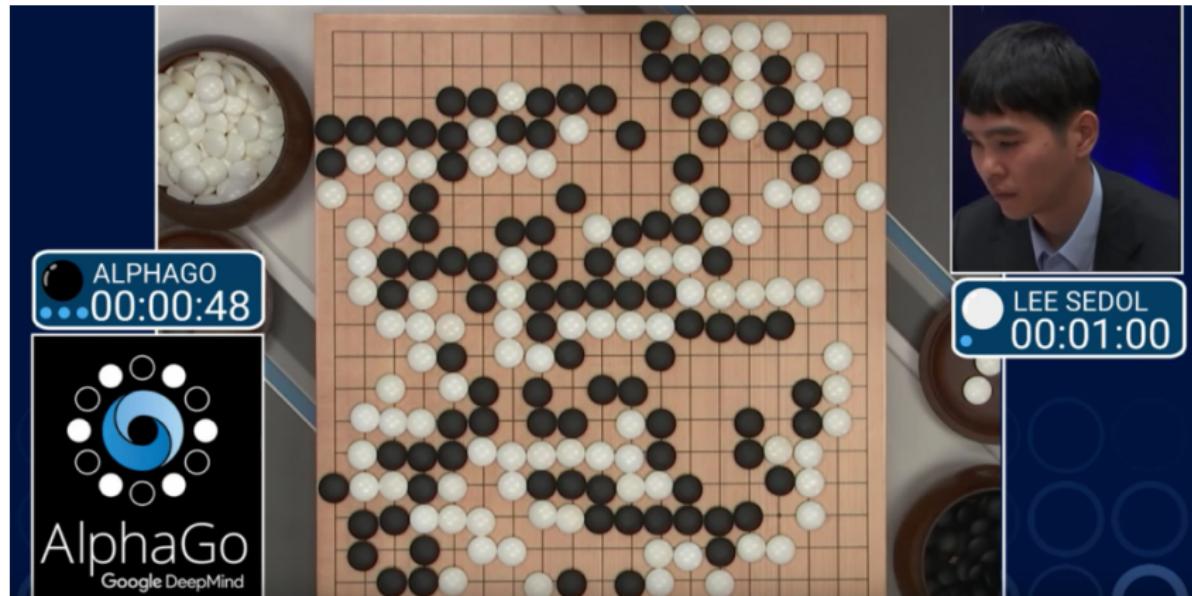
I MADE A PYTHON SCRIPT
THAT GENERATES THOUSANDS
OF REASSURING PARABLES
PER SECOND.

DAMMIT.

COMPUTERS WILL NEVER
UNDERSTAND A SONNET
COMPUTERS WILL NEVER
ENJOY A SALAD COMP-



Go (2016)

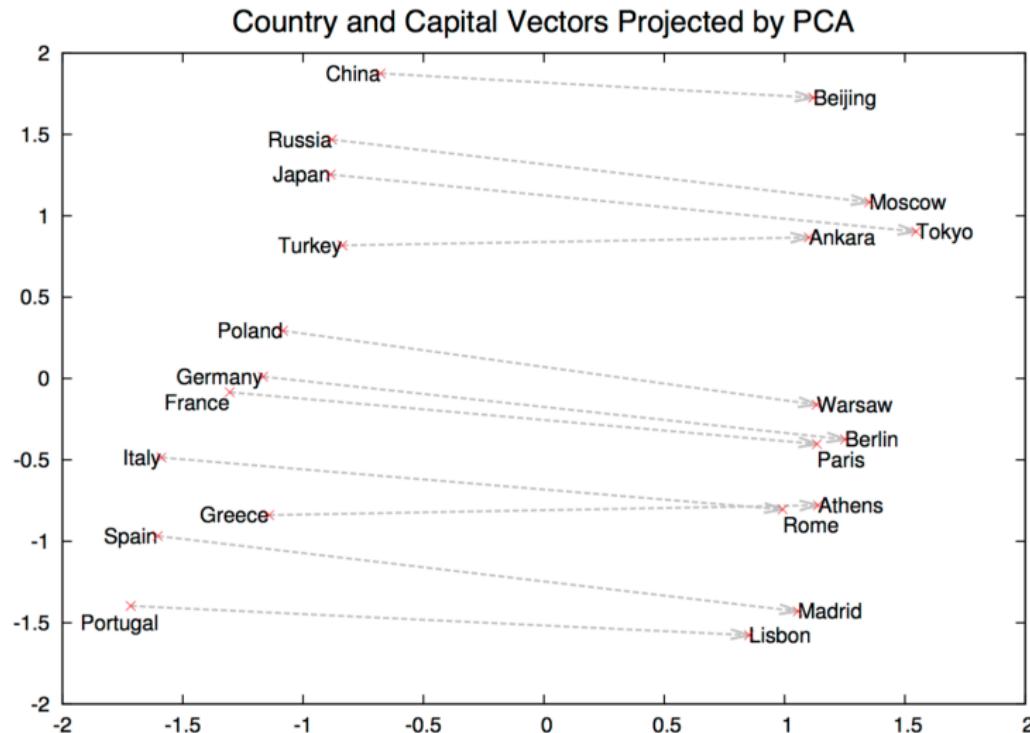


Digits recognition

[Demo]



Learning words representations



Learning words representations

| Relationship | Example 1 | Example 2 | Example 3 |
|----------------------|---------------------|-------------------|----------------------|
| France - Paris | Italy: Rome | Japan: Tokyo | Florida: Tallahassee |
| big - bigger | small: larger | cold: colder | quick: quicker |
| Miami - Florida | Baltimore: Maryland | Dallas: Texas | Kona: Hawaii |
| Einstein - scientist | Messi: midfielder | Mozart: violinist | Picasso: painter |
| Sarkozy - France | Berlusconi: Italy | Merkel: Germany | Koizumi: Japan |
| copper - Cu | zinc: Zn | gold: Au | uranium: plutonium |
| Berlusconi - Silvio | Sarkozy: Nicolas | Putin: Medvedev | Obama: Barack |
| Microsoft - Windows | Google: Android | IBM: Linux | Apple: iPhone |
| Microsoft - Ballmer | Google: Yahoo | IBM: McNealy | Apple: Jobs |
| Japan - sushi | Germany: bratwurst | France: tapas | USA: pizza |

$$\text{Paris} - \text{France} + \text{Italy} = \text{Rome}$$

[Demo]

[Demo 2]

A person riding a motorcycle on a dirt road.



Two dogs play in the grass.



A skateboarder does a trick on a ramp.



A dog is jumping to catch a frisbee.



A group of young people playing a game of frisbee.



Two hockey players are fighting over the puck.



A little girl in a pink hat is blowing bubbles.



A refrigerator filled with lots of food and drinks.



A herd of elephants walking across a dry grass field.



A close up of a cat laying on a couch.



A red motorcycle parked on the side of the road.



A yellow school bus parked in a parking lot.



Describes without errors

Describes with minor errors

Somewhat related to the image

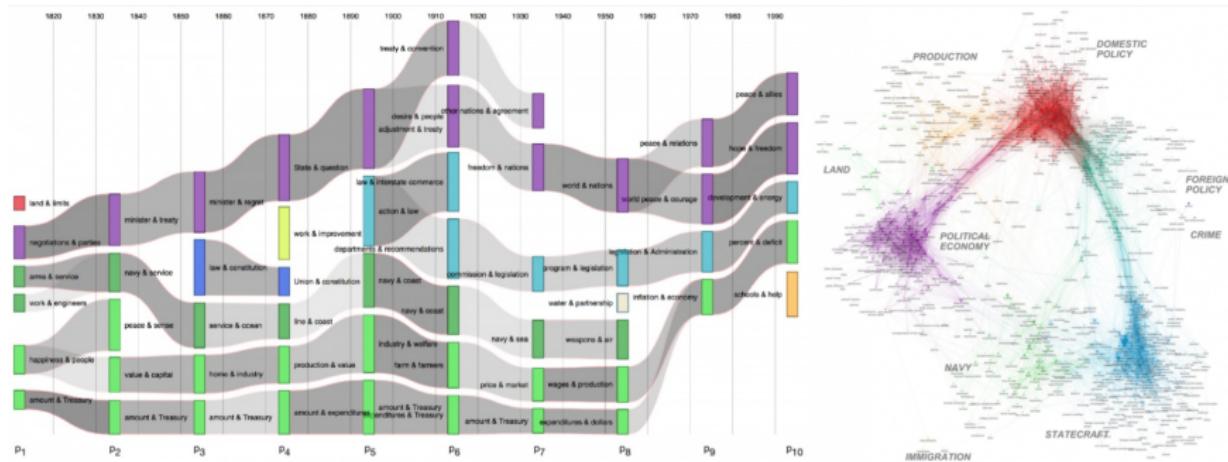
Unrelated to the image

Captcha

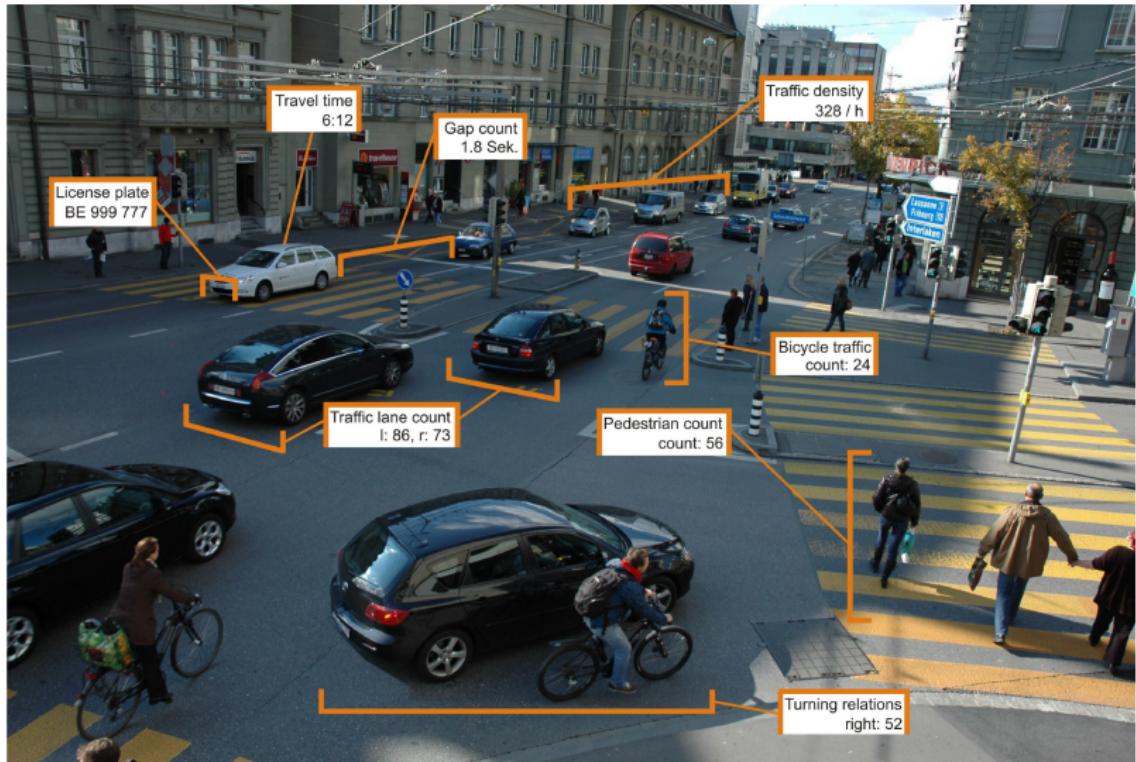
| Characters under typical distortions | Recognition rate |
|---|------------------|
|  | ~100% |
|  | 96+% |
|  | 100% |
|  | 98% |
|  | ~100% |
|  | 95+% |

Text

- ▶ Translation → [Video]
- ▶ Plagiarism detection, automatic summary
- ▶ Topics detection
- ▶ Sentiment analysis → [Demo]

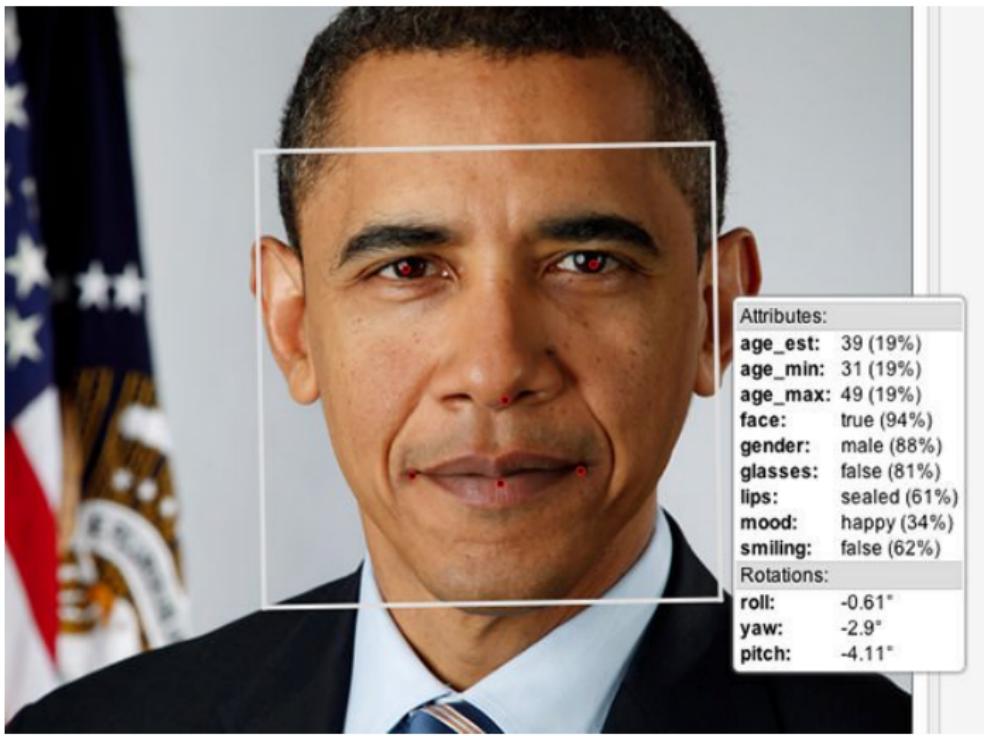


[Video]



[Demo]

[Demo 2]



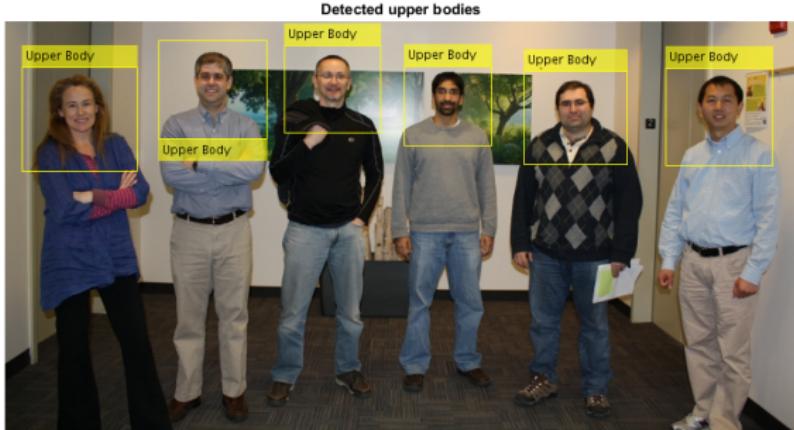
Attributes:

age_est: 39 (19%)
age_min: 31 (19%)
age_max: 49 (19%)
face: true (94%)
gender: male (88%)
glasses: false (81%)
lips: sealed (61%)
mood: happy (34%)
smiling: false (62%)

Rotations:

roll: -0.61°
yaw: -2.9°
pitch: -4.11°

Detection and tracking

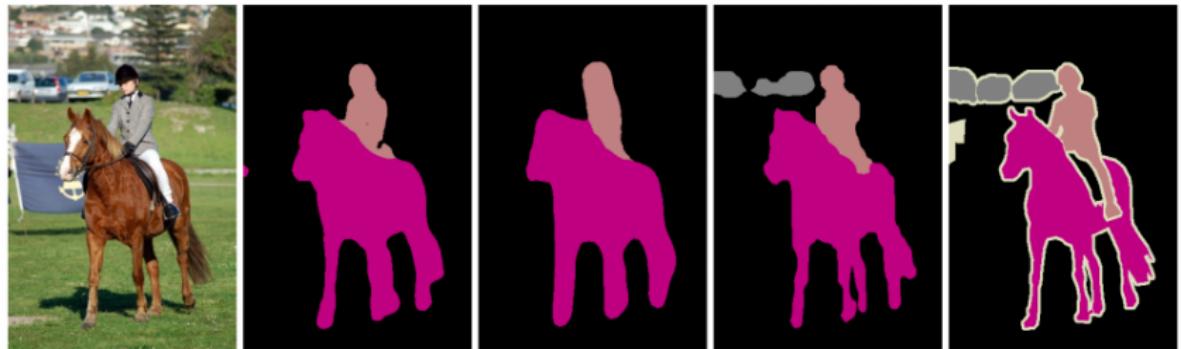


[Video]

[Video 2]

[Video 3]

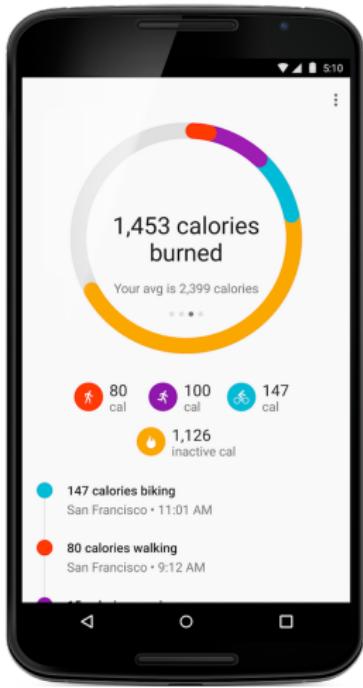
Image segmentation



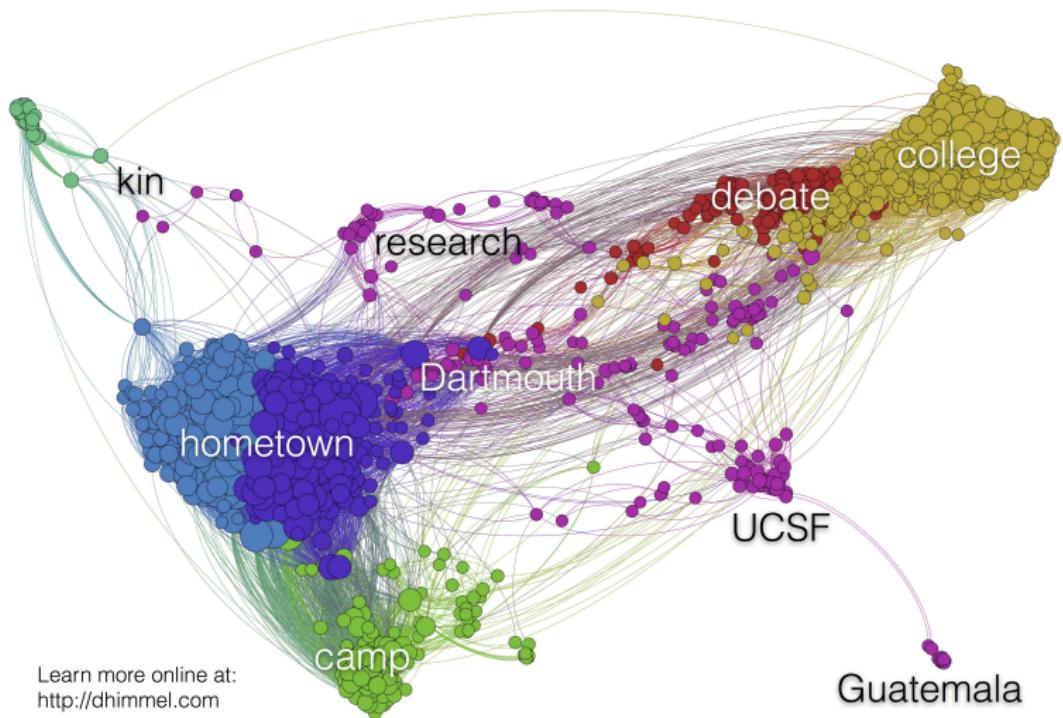
[Video]
[Video 2]

Sound

- ▶ Source separation —> [Video] [Video 2]
- ▶ Denoising/restoration
- ▶ Speaker recognition
- ▶ Music classification
- ▶ ...



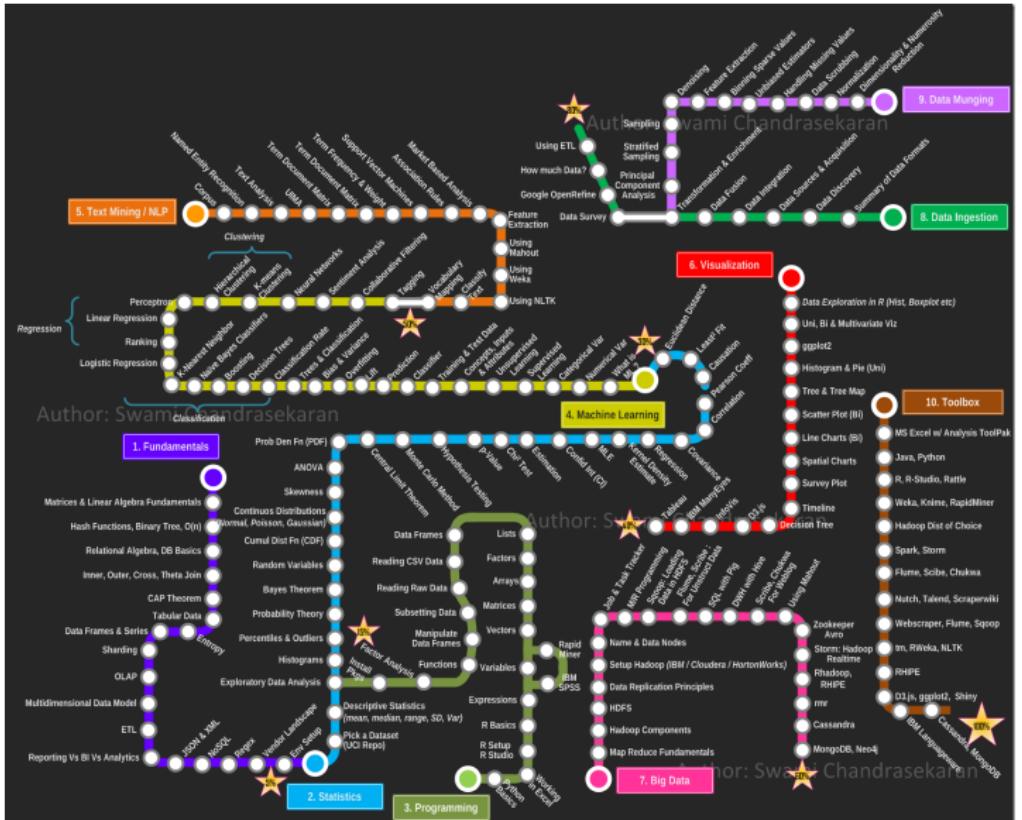
The Friendship Network of Daniel Himmelstein

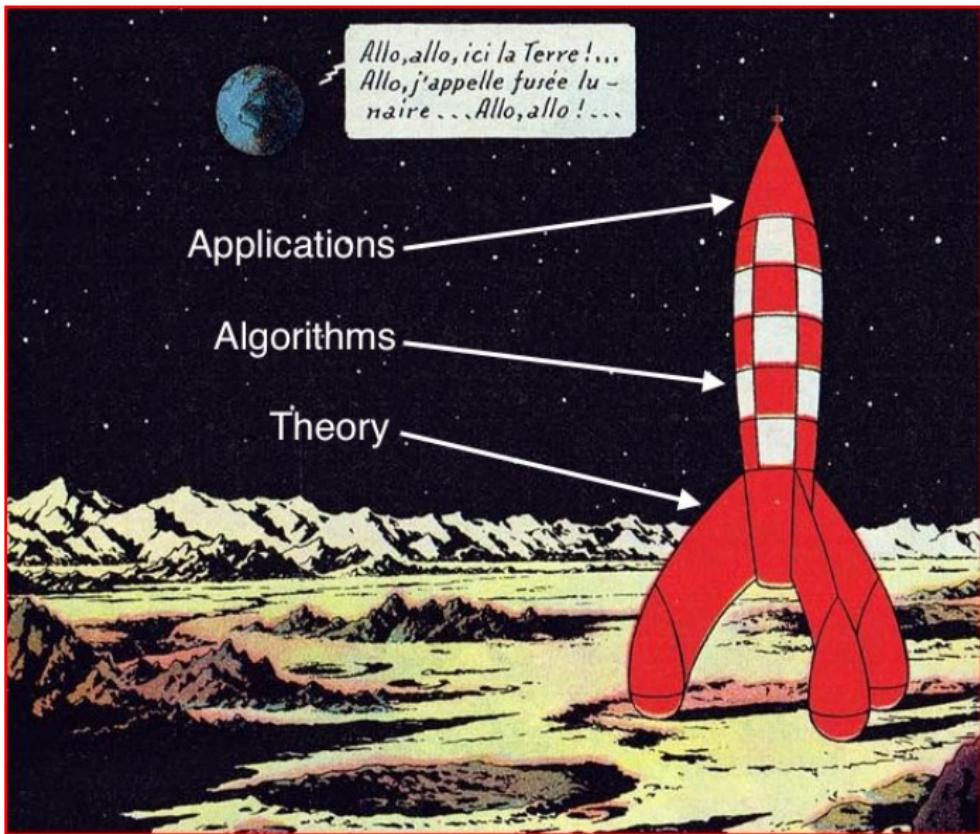




Spotify







1 Théorie

Apprentissage statistique

{Statistical, Machine} Learning: building automatic procedures to infer general rules from examples.

{Statistical, Machine} Learning: building automatic procedures to infer general rules from examples.

In the (rather) long term: mimic the inductive functioning of the humain brain, to develop an artificial intelligence.

{Statistical, Machine} Learning: building automatic procedures to infer general rules from examples.

In the (rather) long term: mimic the inductive functioning of the humain brain, to develop an artificial intelligence.

In the Big Data Era, very dynamic field at the crossroads of Computer Science and Statistics.

Probabilistic framework: n -sample $\mathcal{D}_n = (\mathbf{X}_i, \mathbf{Y}_i)_{i=1}^n$ of i.i.d. replications of some random variable

$$(\mathbf{X}, \mathbf{Y}) \in \mathcal{X} \times \mathcal{Y},$$

$$\dim(\mathcal{X}) = d, \quad \dim(\mathcal{Y}) = m.$$

Probabilistic framework: n -sample $\mathcal{D}_n = (\mathbf{X}_i, \mathbf{Y}_i)_{i=1}^n$ of i.i.d. replications of some random variable

$$(\mathbf{X}, \mathbf{Y}) \in \mathcal{X} \times \mathcal{Y},$$

$$\dim(\mathcal{X}) = d, \quad \dim(\mathcal{Y}) = m.$$

We want to infer the link between the explanatory variable \mathbf{X} and the response variable \mathbf{Y} , *i.e.*, use \mathcal{D}_n to build up $\hat{\phi}$ such that $\hat{\phi}(\mathbf{X})$ is a "good" approximation of \mathbf{Y} . Denote by \mathbb{P} the distribution of (\mathbf{X}, \mathbf{Y}) .

Probabilistic framework: n -sample $\mathcal{D}_n = (\mathbf{X}_i, \mathbf{Y}_i)_{i=1}^n$ of i.i.d. replications of some random variable

$$(\mathbf{X}, \mathbf{Y}) \in \mathcal{X} \times \mathcal{Y},$$

$$\dim(\mathcal{X}) = d, \quad \dim(\mathcal{Y}) = m.$$

We want to infer the link between the explanatory variable \mathbf{X} and the response variable \mathbf{Y} , *i.e.*, use \mathcal{D}_n to build up $\hat{\phi}$ such that $\hat{\phi}(\mathbf{X})$ is a "good" approximation of \mathbf{Y} . Denote by \mathbb{P} the distribution of (\mathbf{X}, \mathbf{Y}) .

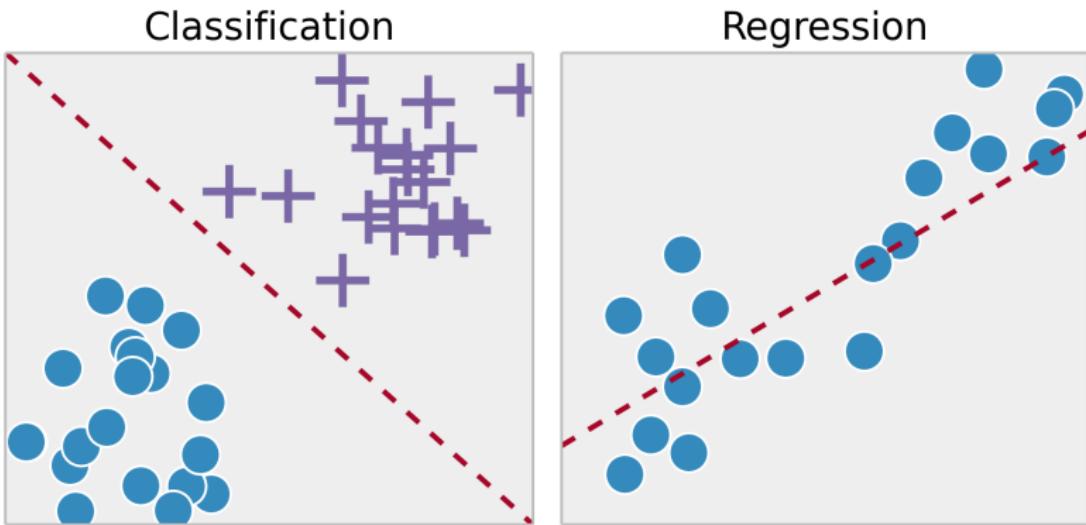
- ▶ Classification: \mathcal{Y} is discrete.
- ▶ Regression: \mathcal{Y} is a continuum.

- ▶ Online learning: observations are revealed over time.
- ▶ Batch learning: all observations are revealed at once.

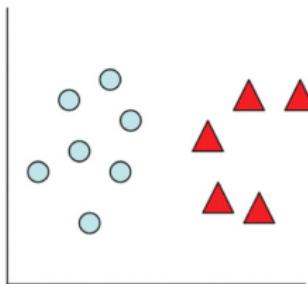
- ▶ Online learning: observations are revealed over time.
- ▶ Batch learning: all observations are revealed at once.

Big Data Era: easy/cheap to collect massive amounts of data,
hence typically $\mathcal{X} = \mathbb{R}^d$ where d may be (extremely) large.
In the sequel, $\dim(\mathcal{Y}) = 1$ and $\mathbf{X} = (X_1, \dots, X_d)$.

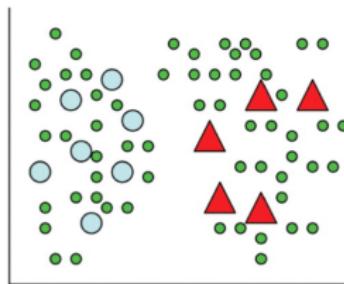
- ▶ Supervised learning: all of the \mathbf{Y}_i 's are observed.



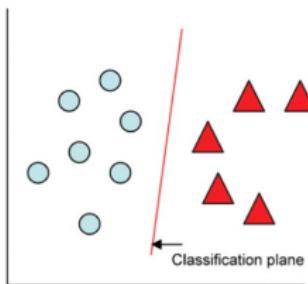
- ▶ Semi-supervised learning: some of the \mathbf{Y}_i s are observed (labeling is expensive or difficult).



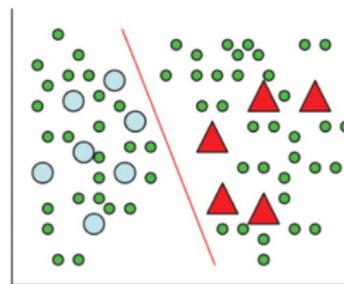
Labeled Data
(a)



Labeled and Unlabeled Data
(b)

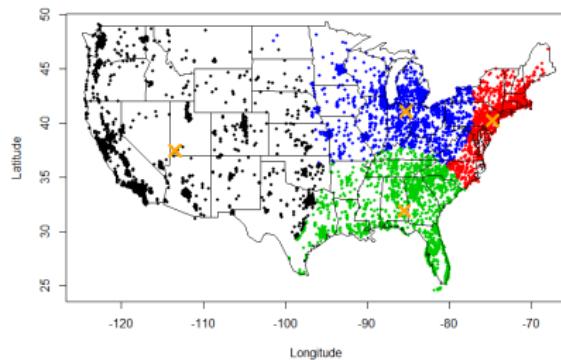
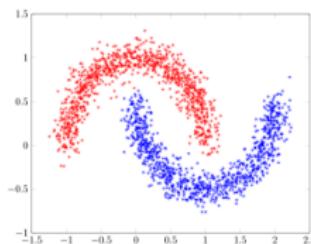
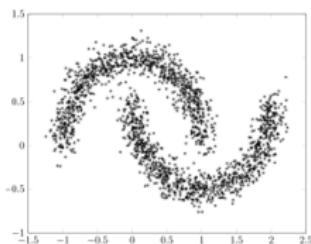


Supervised Learning
(c)



Semi-Supervised Learning
(d)

- ▶ Unsupervised learning: none of the \mathbf{Y}_i 's are observed (detect patterns).



- ▶ Unsupervised learning: none of the \mathbf{Y}_i 's are observed (detect patterns).



- ▶ Reinforcement learning: possible feedback from the environment (robotics, adversarial environments, training...).



Loss function:

$$\ell: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+.$$

Loss function:

$$\ell: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+.$$

$\ell(\hat{\phi}(\mathbf{X}), Y)$ (random) quantifies how a predictor $\hat{\phi}(\mathbf{X})$ is a "good" approximation of \mathbf{Y} .

Loss function:

$$\ell: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+.$$

$\ell(\hat{\phi}(\mathbf{X}), Y)$ (random) quantifies how a predictor $\hat{\phi}(\mathbf{X})$ is a "good" approximation of \mathbf{Y} .

A predictor is any mapping

$$\hat{\phi}: (\mathcal{X} \times \mathcal{Y})^n \rightarrow \mathcal{Y}^{\mathcal{X}}.$$

Loss function:

$$\ell: \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}_+.$$

$\ell(\hat{\phi}(\mathbf{X}), Y)$ (random) quantifies how a predictor $\hat{\phi}(\mathbf{X})$ is a "good" approximation of \mathbf{Y} .

A predictor is any mapping

$$\hat{\phi}: (\mathcal{X} \times \mathcal{Y})^n \rightarrow \mathcal{Y}^{\mathcal{X}}.$$

Risk:

$$R(\hat{\phi}) = \mathbb{E} \left[\ell \left(\hat{\phi}(\mathbf{X}), Y \right) \right].$$

- ▶ Quadratic loss (regression): $\ell(a, b) = (a - b)^2$.

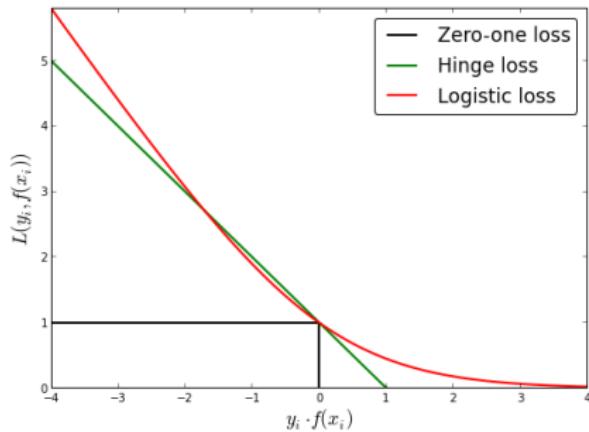
- ▶ Quadratic loss (regression): $\ell(a, b) = (a - b)^2$.
- ▶ Absolute loss (regression): $\ell(a, b) = |a - b|$.

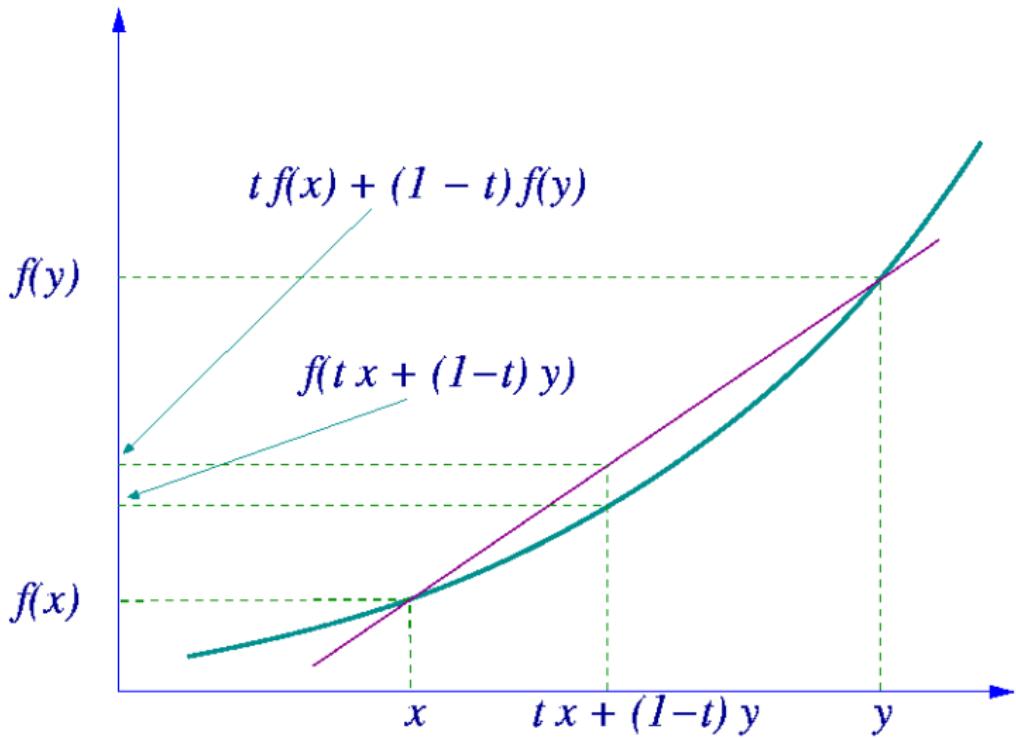
- ▶ Quadratic loss (regression): $\ell(a, b) = (a - b)^2$.
- ▶ Absolute loss (regression): $\ell(a, b) = |a - b|$.
- ▶ 0-1 loss (classification): $\ell(a, b) = \mathbb{1}_{\{a \neq b\}}$.

- ▶ Quadratic loss (regression): $\ell(a, b) = (a - b)^2$.
- ▶ Absolute loss (regression): $\ell(a, b) = |a - b|$.
- ▶ 0-1 loss (classification): $\ell(a, b) = \mathbb{1}_{\{a \neq b\}}$.
- ▶ Hinge loss (classification): $\ell(a, b) = \max(0, 1 - ab)$.

- ▶ Quadratic loss (regression): $\ell(a, b) = (a - b)^2$.
- ▶ Absolute loss (regression): $\ell(a, b) = |a - b|$.
- ▶ 0-1 loss (classification): $\ell(a, b) = \mathbb{1}_{\{a \neq b\}}$.
- ▶ Hinge loss (classification): $\ell(a, b) = \max(0, 1 - ab)$.
- ▶ Logistic loss (classification): $\ell(a, b) = \log[1 + \exp(-ab)]$.

- ▶ Quadratic loss (regression): $\ell(a, b) = (a - b)^2$.
- ▶ Absolute loss (regression): $\ell(a, b) = |a - b|$.
- ▶ 0-1 loss (classification): $\ell(a, b) = \mathbb{1}_{\{a \neq b\}}$.
- ▶ Hinge loss (classification): $\ell(a, b) = \max(0, 1 - ab)$.
- ▶ Logistic loss (classification): $\ell(a, b) = \log[1 + \exp(-ab)]$.





Statistical Learning vs. Machine Learning

Different approaches:

- ▶ In machine learning, given some dataset (\mathbf{x}_i, y_i) , solve

$$\hat{\phi}(\cdot) = \arg \min_m \left\{ \sum_{i=1}^n \ell(y_i, m(\mathbf{x}_i)) \right\}.$$

- ▶ In statistical modeling, assume that the Y_i s are realisations of some random variable Y (given \mathbf{X}) with distribution P . Solve

$$\hat{\phi}(\cdot) = \arg \max_m \left\{ \sum_{i=1}^n \log dP(y_i, m(\mathbf{x}_i)) \right\}.$$

Bayes predictor:

$$\phi^* \in \arg \min_{\phi \in \mathcal{Y}^X} R(\phi).$$

Ultimate goal: do almost as well as the Bayes predictor.

Bayes predictor:

$$\phi^* \in \arg \min_{\phi \in \mathcal{Y}^X} R(\phi).$$

Ultimate goal: do almost as well as the Bayes predictor.

Excess risk:

$$\mathcal{E}(\cdot) = R(\cdot) - R^* \geq 0, \quad R^* = R(\phi^*).$$

Theorem 1

Suppose that $\forall x \in \mathcal{X}$, the infimum of $t \mapsto \mathbb{E}[\ell(Y, t) | \mathbf{X} = x]$ is reached. Then

$$\phi^*(x) \in \arg \min_{t \in \mathcal{Y}} \mathbb{E}[\ell(Y, t) | \mathbf{X} = x]$$

is a Bayes predictor.

Proof:



Theorem 2

Binary classification: $\mathcal{Y} = \{0, 1\}$ and $\ell(a, b) = \mathbb{1}_{\{a \neq b\}}$.

$$\phi^*(x) = \mathbb{1}_{\{\eta^*(x) > 1/2\}}, \quad \eta^*(x) = \mathbb{P}[Y = 1 | \mathbf{X} = x].$$

Further,

$$\mathcal{E}(\phi) = \mathbb{E}[(\phi(\mathbf{X}) - \phi^*(\mathbf{X}))(1 - 2\eta^*(\mathbf{X}))].$$

Proof:



Theorem 3

Quadratic regression: $\mathcal{Y} = \mathbb{R}$ and $\ell(a, b) = (a - b)^2$.

$$\phi^*(x) = \eta^*(x), \quad \eta^*(x) = \mathbb{E}[Y | \mathbf{X} = x].$$

Further,

$$\mathcal{E}(\phi) = \mathbb{E} [(\phi(\mathbf{X}) - \phi^*(\mathbf{X}))^2].$$

Proof:



Link between Binary Classification and Regression

1. Estimate the regression function $\eta^*(x)$ by $\hat{\eta}(x)$,
2. Define $\hat{\phi}(x) = \mathbb{1}_{\{\hat{\eta}(x)>1/2\}}$.

How good is the plug-in rule $\hat{\phi}_n$?

Theorem 4

$$\mathcal{E}^{\text{class}}(\hat{\phi}) \leq 2\sqrt{\mathcal{E}^{\text{reg}}(\hat{\eta})}.$$

Proof:



Instead of targeting the Bayes predictor, we relax the problem with the notion of oracle:

$$\arg \inf_{f \in \mathcal{F}} R(f),$$

where \mathcal{F} is a family of possible procedures (e.g., linear functions).

Instead of targeting the Bayes predictor, we relax the problem with the notion of oracle:

$$\arg \inf_{f \in \mathcal{F}} R(f),$$

where \mathcal{F} is a family of possible procedures (e.g., linear functions).

$$R_n(\hat{\phi}) = \frac{1}{n} \sum_{i=1}^n \ell\left(\hat{\phi}(\mathbf{X}_i), Y_i\right).$$

Instead of targeting the Bayes predictor, we relax the problem with the notion of oracle:

$$\arg \inf_{f \in \mathcal{F}} R(f),$$

where \mathcal{F} is a family of possible procedures (e.g., linear functions).

$$R_n(\hat{\phi}) = \frac{1}{n} \sum_{i=1}^n \ell\left(\hat{\phi}(\mathbf{X}_i), Y_i\right).$$

Empirical Risk Minimization (ERM):

$$\hat{\phi}^{\text{ERM}} \in \arg \inf_{f \in \mathcal{F}} R_n(f).$$

This might be (computationally) intractable (e.g., if the loss ℓ is not convex), or lead to poor performance.

Some Models

Linear parametric model

$$Y = \mathbf{X}\beta^* + \varepsilon.$$

Semiparametric model

$$Y = f^*(\mathbf{X}\beta^*) + \varepsilon.$$

Nonparametric (additive) model

$$Y = f^*(\mathbf{X}) + \varepsilon, \quad Y = \sum_{j=1}^d f_j(X_j) + \varepsilon.$$

Taking $\mathcal{F} = \mathcal{Y}^{\mathcal{X}}$ is a bad idea: in all generality, for any learning problem, the number of minimizer of the empirical risk is not finite. Further, we are exposed to the overfitting phenomenon.



Taking $\mathcal{F} = \mathcal{Y}^{\mathcal{X}}$ is a bad idea: in all generality, for any learning problem, the number of minimizer of the empirical risk is not finite. Further, we are exposed to the overfitting phenomenon.



Hint: take \mathcal{F} big enough to approximate any function reasonably well, not too big to avoid overfitting and computational dead-end.

Taking $\mathcal{F} = \mathcal{Y}^{\mathcal{X}}$ is a bad idea: in all generality, for any learning problem, the number of minimizer of the empirical risk is not finite. Further, we are exposed to the overfitting phenomenon.



Hint: take \mathcal{F} big enough to approximate any function reasonably well, not too big to avoid overfitting and computational dead-end.

"Size" of \mathcal{F} : complexity. Other approach: add a penalization term to $R_n(\cdot)$ to account for the complexity/possible irregularity of g .

Notation:

$$g^* \in \arg \min_{\phi \in \mathcal{Y}^{\mathcal{X}}} R(\phi),$$

$$g_{\mathcal{F}}^* \in \arg \min_{\phi \in \mathcal{F}} R(\phi).$$

Notation:

$$g^* \in \arg \min_{\phi \in \mathcal{Y}^{\mathcal{X}}} R(\phi),$$

$$g_{\mathcal{F}}^* \in \arg \min_{\phi \in \mathcal{F}} R(\phi).$$

Since for any predictor g ,

$$R(g) \geq R(g_{\mathcal{F}}^*) \geq R(g^*) = R^*,$$

Notation:

$$g^* \in \arg \min_{\phi \in \mathcal{Y}^{\mathcal{X}}} R(\phi),$$

$$g_{\mathcal{F}}^* \in \arg \min_{\phi \in \mathcal{F}} R(\phi).$$

Since for any predictor g ,

$$R(g) \geq R(g_{\mathcal{F}}^*) \geq R(g^*) = R^*,$$

$$\mathcal{E}(\hat{g}^{\text{ERM}}) = \underbrace{R(\hat{g}^{\text{ERM}}) - R(g_{\mathcal{F}}^*)}_{\text{estimation error}} + \underbrace{R(g_{\mathcal{F}}^*) - R^*}_{\text{approximation error}}$$

Lemma 1

$$R(\hat{g}^{\text{ERM}}) - R(g_{\mathcal{F}}^*) \leq 2 \sup_{g \in \mathcal{F}} |R(g) - R_n(g)|.$$

Proof:



Assume that for any g , $\mathbb{E}\ell^2(Y, g(\mathbf{X})) < +\infty$, then from the LLN and CLT,

Assume that for any g , $\mathbb{E}\ell^2(Y, g(\mathbf{X})) < +\infty$, then from the LLN and CLT,

$$R_n(g) \xrightarrow[n \rightarrow +\infty]{a.s.} R(g),$$

Assume that for any g , $\mathbb{E}\ell^2(Y, g(\mathbf{X})) < +\infty$, then from the LLN and CLT,

$$R_n(g) \xrightarrow[n \rightarrow +\infty]{a.s.} R(g),$$

$$\sqrt{n}(R_n(g) - R(g)) \xrightarrow[n \rightarrow +\infty]{\mathbb{P}} \mathcal{N}(0, \mathbb{V}(\ell(Y, g(\mathbf{X}))).$$

Message: for any predictor g , the random variable $R_n(g)$ deviates from its mean $R(g)$ with terms $\mathcal{O}(1/\sqrt{n})$.

PAC Oracle Inequalities

For some procedure $\hat{\phi}$, establish oracle inequalities of the flavor:

PAC Oracle Inequalities

For some procedure $\hat{\phi}$, establish oracle inequalities of the flavor:

$$\mathbb{P} \left[R(\hat{\phi}) \leq C \inf_{f \in \mathcal{F}} \left\{ R(f) + \Delta_{n,d,\varepsilon}(f) \right\} \right] \geq 1 - \varepsilon, \quad \forall \varepsilon > 0,$$

PAC Oracle Inequalities

For some procedure $\hat{\phi}$, establish oracle inequalities of the flavor:

$$\mathbb{P} \left[R(\hat{\phi}) \leq C \inf_{f \in \mathcal{F}} \left\{ R(f) + \Delta_{n,d,\varepsilon}(f) \right\} \right] \geq 1 - \varepsilon, \quad \forall \varepsilon > 0,$$

or

$$\mathbb{E} R(\hat{\phi}) \leq C \inf_{f \in \mathcal{F}} \left\{ \mathbb{E} R(f) + \Delta_{n,d}(f) \right\}.$$

PAC Oracle Inequalities

For some procedure $\hat{\phi}$, establish oracle inequalities of the flavor:

$$\mathbb{P} \left[R(\hat{\phi}) \leq C \inf_{f \in \mathcal{F}} \left\{ R(f) + \Delta_{n,d,\varepsilon}(f) \right\} \right] \geq 1 - \varepsilon, \quad \forall \varepsilon > 0,$$

or

$$\mathbb{E} R(\hat{\phi}) \leq C \inf_{f \in \mathcal{F}} \left\{ \mathbb{E} R(f) + \Delta_{n,d}(f) \right\}.$$

- ▶ $C \geq 1$. If $C = 1$, the inequality is *exact* or *sharp*.
- ▶ The remainder term Δ grows with d and the size of \mathcal{F} . It decreases with n .

Hoeffding inequality

Let V_1, \dots, V_n be independent real-valued random variables such that $a_i \leq V_i \leq b_i$ a.s. Let $\bar{V}_n = \frac{1}{n} \sum_{i=1}^n V_i$.

Hoeffding inequality

Let V_1, \dots, V_n be independent real-valued random variables such that $a_i \leq V_i \leq b_i$ a.s. Let $\bar{V}_n = \frac{1}{n} \sum_{i=1}^n V_i$. Then

$$\mathbb{P}(\bar{V}_n - \mathbb{E}\bar{V}_n > t) \leq \exp\left(-\frac{2n^2t^2}{\sum_{i=1}^n (b_i - a_i)^2}\right), \forall t > 0.$$

Proof:



Theorem 5

Assume that there exist $a < b$ such that $a < \ell(y, y') < b$ for any $y, y' \in \mathcal{Y}$.

Theorem 5

Assume that there exist $a < b$ such that $a < \ell(y, y') < b$ for any $y, y' \in \mathcal{Y}$. Then for any $\varepsilon \in (0, 1)$, with probability at least $1 - \varepsilon$,

$$R(\hat{g}^{\text{ERM}}) - R(g_{\mathcal{F}}^*) \leq (b - a) \sqrt{\frac{2 \log(2|\mathcal{F}|\varepsilon^{-1})}{n}}.$$

Proof:



Comments

- ▶ Complexity of \mathcal{F} and consistency.
- ▶ Practical computation of the risk.

Alexey Chervonenkis (1938–2014)

Vladimir Vapnik (1935–)



Goal: obtain tighter upper bounds, even in the case where \mathcal{F} contains an infinity of functions.

Goal: obtain tighter upper bounds, even in the case where \mathcal{F} contains an infinity of functions.

Idea: what is relevant is not the cardinality of \mathcal{F} , but the number of functions in \mathcal{F} leading to different predictions on \mathcal{D}_n .

$\mathcal{Y} = \{0, 1\}$. Any predictor g may be identified as the set $G = \{x \in \mathcal{X}: g(x) = 1\}$, such that $g = \mathbb{1}_G$.

$\mathcal{Y} = \{0, 1\}$. Any predictor g may be identified as the set $G = \{x \in \mathcal{X}: g(x) = 1\}$, such that $g = \mathbb{1}_G$.

For any set $S = \{x_1, \dots, x_k\}$, let 2^S denote the power set of S . Let

$$T_{\mathcal{F}}(S) = \{S \cap G: G \in \mathcal{F}\} \subset 2^S$$

be the trace of \mathcal{F} on S .

$\mathcal{Y} = \{0, 1\}$. Any predictor g may be identified as the set $G = \{x \in \mathcal{X}: g(x) = 1\}$, such that $g = \mathbb{1}_G$.

For any set $S = \{x_1, \dots, x_k\}$, let 2^S denote the power set of S . Let

$$T_{\mathcal{F}}(S) = \{S \cap G: G \in \mathcal{F}\} \subset 2^S$$

be the trace of \mathcal{F} on S .

When $T_{\mathcal{F}}(S) = 2^S$, we say that S is shattered by \mathcal{F} .

VC dimension

$V_{\mathcal{F}} = \max \{J \in \mathbb{N}: \exists S \subset \mathcal{X} \text{ s.t. } |S| = J \text{ and } S \text{ is shattered by } \mathcal{F}\}.$

VC dimension

$V_{\mathcal{F}} = \max \{ J \in \mathbb{N} : \exists S \subset \mathcal{X} \text{ s.t. } |S| = J \text{ and } S \text{ is shattered by } \mathcal{F} \}.$

$$V_{\mathcal{F}}(\mathcal{D}_n) = \max \left\{ J \in \mathbb{N} : \max_{S \subset \{X_1, \dots, X_n\} : |S|=J} |\mathcal{T}_{\mathcal{F}}(S)| = 2^J \right\}.$$

VC dimension

$V_{\mathcal{F}} = \max \{ J \in \mathbb{N} : \exists S \subset \mathcal{X} \text{ s.t. } |S| = J \text{ and } S \text{ is shattered by } \mathcal{F} \}.$

$$V_{\mathcal{F}}(\mathcal{D}_n) = \max \left\{ J \in \mathbb{N} : \max_{S \subset \{X_1, \dots, X_n\} : |S|=J} |\mathcal{T}_{\mathcal{F}}(S)| = 2^J \right\}.$$

- ▶ $V_{\mathcal{F}} \geq 0,$
- ▶ $V_{\mathcal{F}}(\mathcal{D}_n) \leq n,$
- ▶ $V_{\mathcal{F}} = 0 \iff |\mathcal{F}| = 1.$

Sauer's Lemma

For any $S = \{X_1, \dots, X_n\} \in \mathcal{X}^n$,

$$|T_{\mathcal{F}}(S)| \leq \sum_{k=0}^{V_{\mathcal{F}}(n)} \binom{n}{k} \leq (n+1)^{V_{\mathcal{F}}(\mathcal{D}_n)}.$$

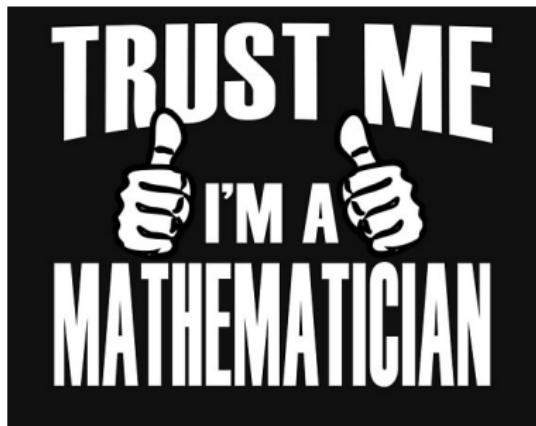
Proof:

Sauer's Lemma

For any $S = \{X_1, \dots, X_n\} \in \mathcal{X}^n$,

$$|T_{\mathcal{F}}(S)| \leq \sum_{k=0}^{V_{\mathcal{F}}(n)} \binom{n}{k} \leq (n+1)^{V_{\mathcal{F}}(\mathcal{D}_n)}.$$

Proof: omitted.



Theorem 6

$\mathcal{Y} = \{0, 1\}$ and $\ell: \mathcal{Y} \times \mathcal{Y} \rightarrow \{0, 1\}$ denotes the 0-1 loss. For any $\varepsilon > 0$, with probability at least $1 - \varepsilon$,

Theorem 6

$\mathcal{Y} = \{0, 1\}$ and $\ell: \mathcal{Y} \times \mathcal{Y} \rightarrow \{0, 1\}$ denotes the 0-1 loss. For any $\varepsilon > 0$, with probability at least $1 - \varepsilon$,

$$R(\hat{g}^{\text{ERM}}) - R(g_{\mathcal{F}}^*) \leq 2 \sqrt{\frac{2V_{\mathcal{F}}(\mathcal{D}_{2n}) \log [4(2n+1)\varepsilon^{-1}]}{n}}.$$

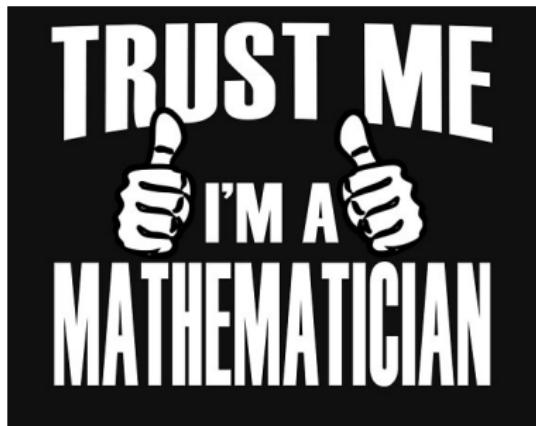
Proof:

Theorem 6

$\mathcal{Y} = \{0, 1\}$ and $\ell: \mathcal{Y} \times \mathcal{Y} \rightarrow \{0, 1\}$ denotes the 0-1 loss. For any $\varepsilon > 0$, with probability at least $1 - \varepsilon$,

$$R(\hat{g}^{\text{ERM}}) - R(g_{\mathcal{F}}^*) \leq 2 \sqrt{\frac{2V_{\mathcal{F}}(\mathcal{D}_{2n}) \log [4(2n+1)\varepsilon^{-1}]}{n}}.$$

Proof: omitted.



2 Algorithmes

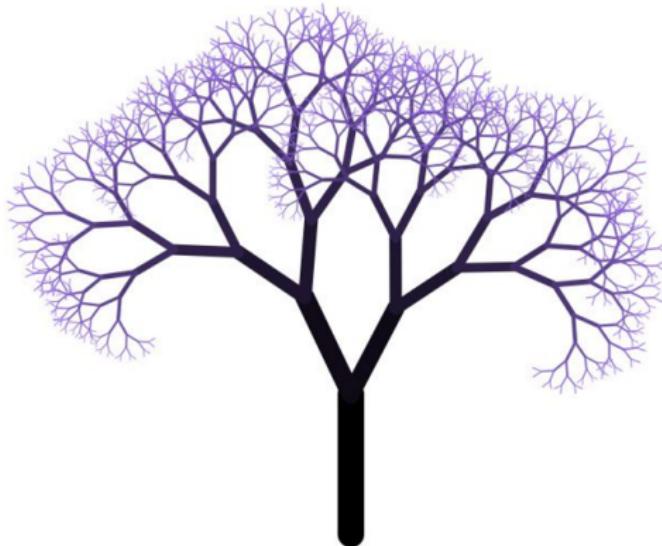
2.1 Arbres et forêts aléatoires

Leo Breiman (1928–2005)



Binary Tree

- ▶ Build recursively a binary decision tree until a stop criterion is met.
- ▶ No model! This is a purely data-driven method.



In a nutshell

- ▶ Principle: Build a partition of \mathbb{R}^d using (simple) hyperplans.
The cells of the partition are the leaves of the tree.

In a nutshell

- ▶ Principle: Build a partition of \mathbb{R}^d using (simple) hyperplans. The cells of the partition are the leaves of the tree.
- ▶ How the tree is grown: At each node, select a covariate X_j ($j \in \{1, \dots, d\}$) and a cut $\{X_j \geq k\} \cup \{X_j < k\}$ for some $k \in \mathbb{R}$, using some criterion.

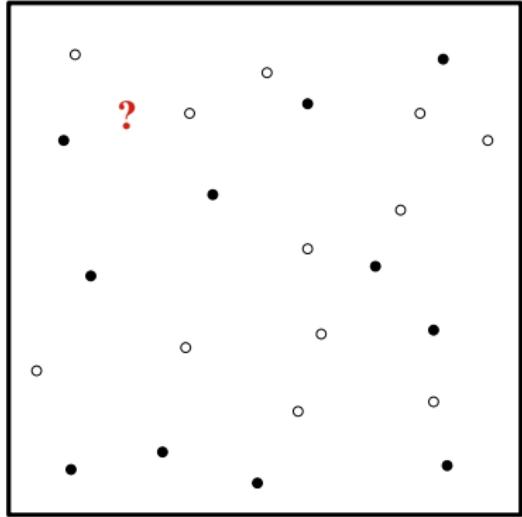
In a nutshell

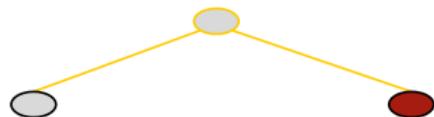
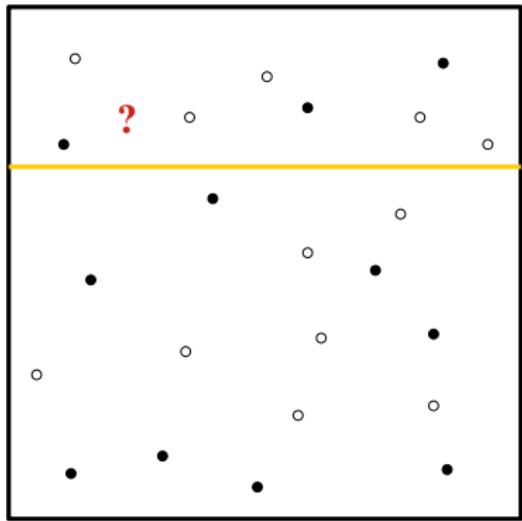
- ▶ Principle: Build a partition of \mathbb{R}^d using (simple) hyperplans. The cells of the partition are the leaves of the tree.
- ▶ How the tree is grown: At each node, select a covariate X_j ($j \in \{1, \dots, d\}$) and a cut $\{X_j \geq k\} \cup \{X_j < k\}$ for some $k \in \mathbb{R}$, using some criterion.
- ▶ Classification: The label of a new data point is obtained through a majority vote within its cell / leaf.

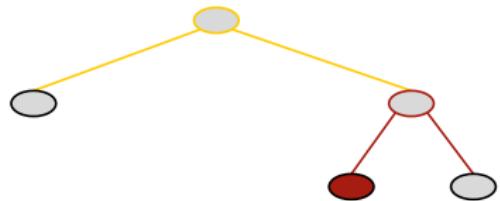
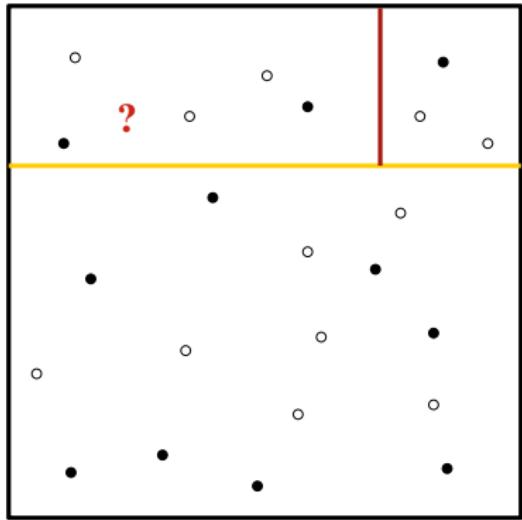
In a nutshell

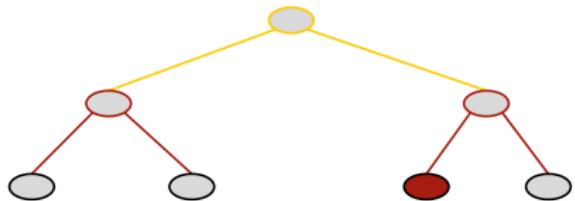
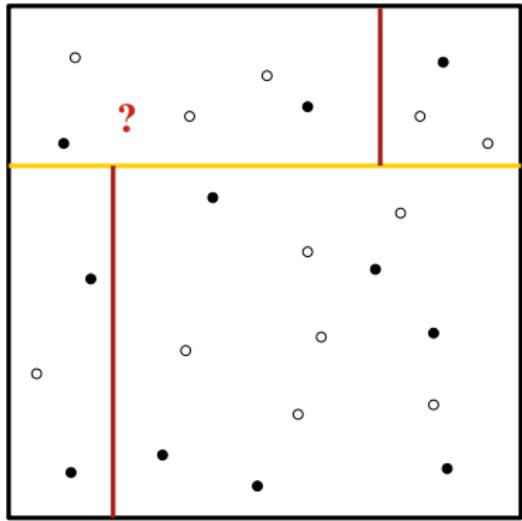
- ▶ Principle: Build a partition of \mathbb{R}^d using (simple) hyperplans. The cells of the partition are the leaves of the tree.
- ▶ How the tree is grown: At each node, select a covariate X_j ($j \in \{1, \dots, d\}$) and a cut $\{X_j \geq k\} \cup \{X_j < k\}$ for some $k \in \mathbb{R}$, using some criterion.

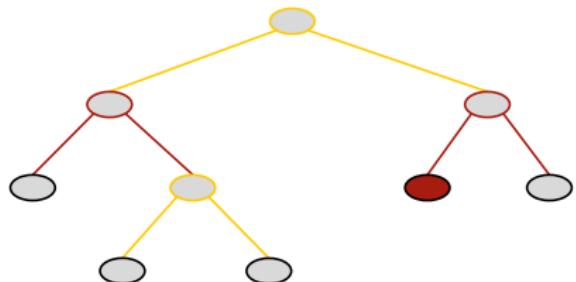
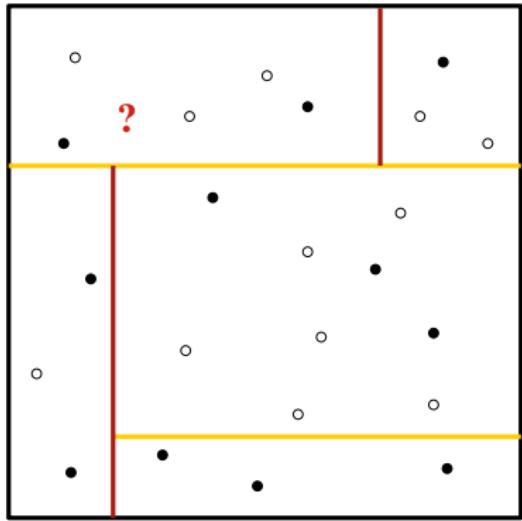
- ▶ Classification: The label of a new data point is obtained through a majority vote within its cell / leaf.
- ▶ Regression: The response of a new data point is the mean of its cell / leaf.

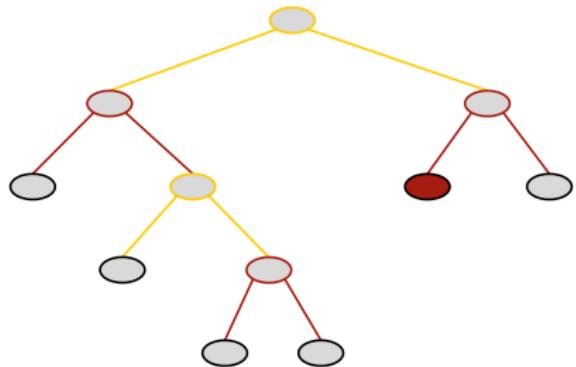
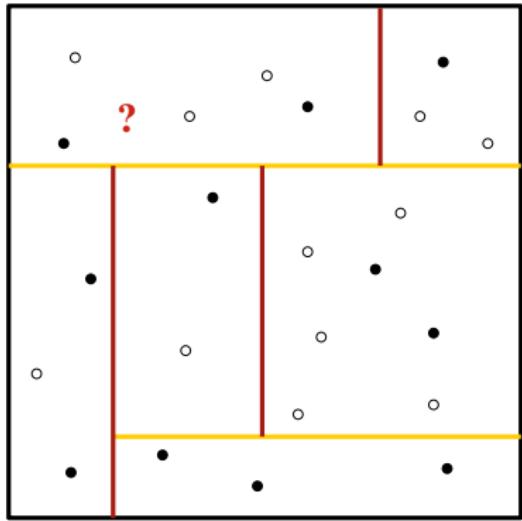


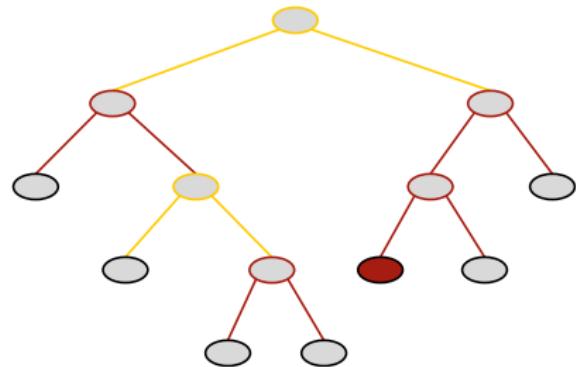
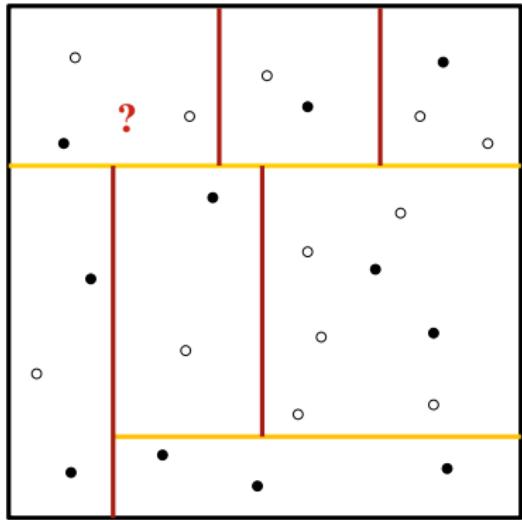












Pseudocode

Initialization. TREE=root

Expansion. For each node N of TREE

 If N does not meet the stop criterion

 Create subnodes

 TREE = TREE + (subnodes)

 End If

Pruning. For each node N of TREE

 If N meets the pruning condition

 TREE = TREE - (node N and its subnodes)

 End If

Cutting

- ▶ Classification: Our goal is to obtain homogeneous groups. Achieved by minimizing the Gini Index \mathcal{I}_G at each node f :

$$\mathcal{I}_G(f) = 1 - \sum_{i=1}^2 f_i^2,$$

where f_i is the fraction of observations with label i in the node f .

- ▶ Regression: Our goal is to have the smaller possible variance within each node. Achieved by minimizing the empirical variance

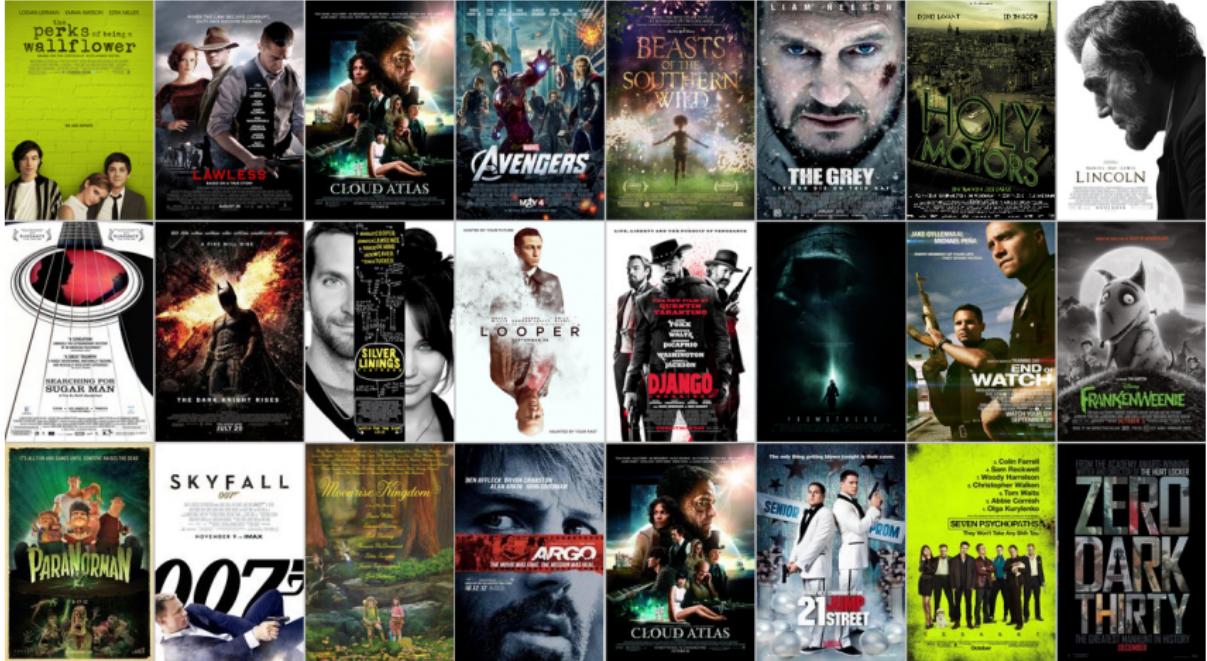
$$\frac{1}{n_f} \sum_{i \in f} (Y_i - \bar{Y}_f)^2$$

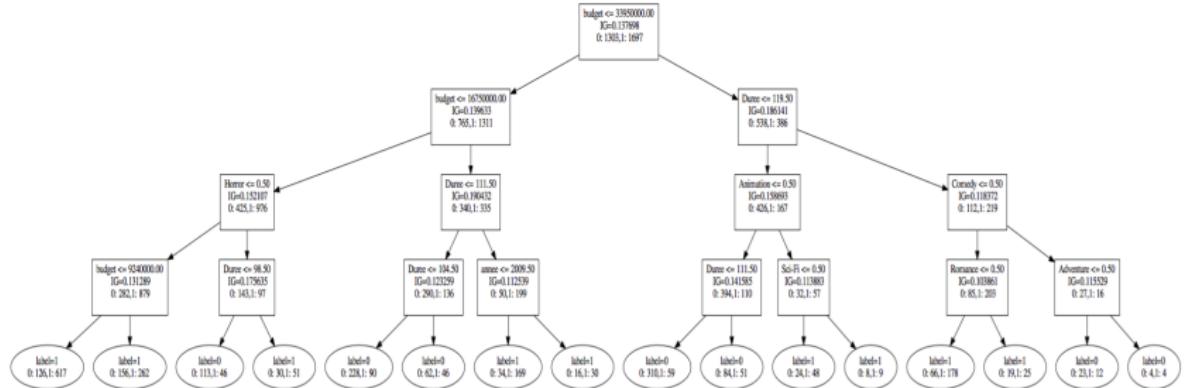
at each node f .

Stop criterion, pruning & importance

- ▶ Typically, the algorithm stops when the tree reaches a fixed depth, when the number of leaves exceeds a fixed bound, or when the number of individuals / points in each nodes falls below a fixed threshold.
- ▶ The maximal tree might not be the best choice: it has a large variance and is computationally more demanding. Hence the idea of pruning (finding the "best" sub-tree in the maximal tree). We may remove a branch if it does not degrade "too much" the predictive performance \Rightarrow compromise between the performance and the number of leaves.
- ▶ Variable selection: notion of importance.

NETFLIX





About the instability and lack of robustness: averaging M trees still delivers a predictor $\mathcal{X} \rightarrow \mathcal{Y}$, and massively improves its stability. We call a set of $M \geq 2$ trees a... forest.

Simple (unoptimized) and randomized (different) trees:

- ▶ No pruning
- ▶ At each node, select only m_{try} variables among $\{1, \dots, d\}$, and propose a cut with the CART criterion, or even a random cut

A cut may be represented by a vector $\theta = (\theta_1, \dots, \theta_N)$ where N is the number of nodes (same for all trees). Let m_θ denote the tree formed with the cut θ .

Denote $(\theta^1, \dots, \theta^M)$ a sequence of M cuts, a random forest for regression is

$$m: \mathbf{x} \mapsto \frac{1}{M} \sum_{j=1}^M m_{\theta^j}(\mathbf{x}).$$

R package `randomForest`.

Trees and forests

- ▶ A great tool!
 - ▶ Nonparametric & no model specification
 - ▶ Performs classification & regression
 - ▶ Qualitative and quantitative input variables
 - ▶ Easy interpretation (trees)
 - ▶ Deals with complex and massive data
- ▶ Sadly not perfect...
 - ▶ Very few theoretical results (no oracle inequalities)
 - ▶ Unstable: performance highly depends on the cut (trees)
 - ▶ No robustness to outliers (trees)
 - ▶ No interpretation (forests)

2 Algorithmes

2.2 Méthodes régularisées

Simple linear regression model

$$Y_i = \mathbf{X}_i\beta^* + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma_i^2), \quad i = 1, \dots, n.$$

In the Gauss-Markov model (the errors are homoscedastic, *i.e.*, $\sigma_i^2 = \sigma^2 \forall i$, and the $(X_{ij})_j$ are independent), least-squares fit

$$\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} = \arg \min_{\beta \in \mathbb{R}^d} \|\mathbf{Y} - \mathbf{X}\beta\|^2.$$

Curse of dimensionality

Assuming that $\mathbf{X}'\mathbf{X}$ is invertible (restrictive), its computational cost is of magnitude $\mathcal{O}(d^3)$.

Curse of dimensionality

Assuming that $\mathbf{X}'\mathbf{X}$ is invertible (restrictive), its computational cost is of magnitude $\mathcal{O}(d^3)$.

Solution: assume a sparse representation of β^* , i.e., with at most $d_0 \ll \min(n, d)$ nonzero components.

Curse of dimensionality

Assuming that $\mathbf{X}'\mathbf{X}$ is invertible (restrictive), its computational cost is of magnitude $\mathcal{O}(d^3)$.

Solution: assume a sparse representation of β^* , i.e., with at most $d_0 \ll \min(n, d)$ nonzero components.

If those components were known, we could build the least squares estimator $\hat{\beta}_0$ and obtain oracle inequalities of the flavor

$$\mathbb{E}[R(\hat{\beta}_0) - R(\beta^*)] \leq \text{cste} \times \frac{\sigma^2 d_0}{n}.$$

Curse of dimensionality

Assuming that $\mathbf{X}'\mathbf{X}$ is invertible (restrictive), its computational cost is of magnitude $\mathcal{O}(d^3)$.

Solution: assume a sparse representation of β^* , i.e., with at most $d_0 \ll \min(n, d)$ nonzero components.

If those components were known, we could build the least squares estimator $\hat{\beta}_0$ and obtain oracle inequalities of the flavor

$$\mathbb{E}[R(\hat{\beta}_0) - R(\beta^*)] \leq \text{cste} \times \frac{\sigma^2 d_0}{n}.$$

Obviously, neither those components nor d_0 are known!

ℓ^0 penalty

- ▶ Natural idea: extend least squares minimization by penalizing the number of nonzero coordinates: for some $\lambda > 0$,

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \{ \| \mathbf{Y} - \mathbf{X}\beta \|^2 + \lambda \|\beta\|_0 \}.$$

ℓ^0 penalty

- ▶ Natural idea: extend least squares minimization by penalizing the number of nonzero coordinates: for some $\lambda > 0$,

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \{ \| \mathbf{Y} - \mathbf{X}\beta \|^2 + \lambda \|\beta\|_0 \}.$$

- ▶ This approach— ℓ^0 -penalization—is known to deliver solid theoretical guarantee: for $\lambda \asymp \log(d)/n$,

$$\mathbb{E}[R(\hat{\beta}_\lambda) - R(\beta^*)] \leq \text{cst} \times \frac{\sigma^2 d_0 \log(d)}{n}.$$

ℓ^0 penalty

- ▶ Natural idea: extend least squares minimization by penalizing the number of nonzero coordinates: for some $\lambda > 0$,

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \{ \| \mathbf{Y} - \mathbf{X}\beta \|^2 + \lambda \|\beta\|_0 \}.$$

- ▶ This approach— ℓ^0 -penalization—is known to deliver solid theoretical guarantee: for $\lambda \asymp \log(d)/n$,

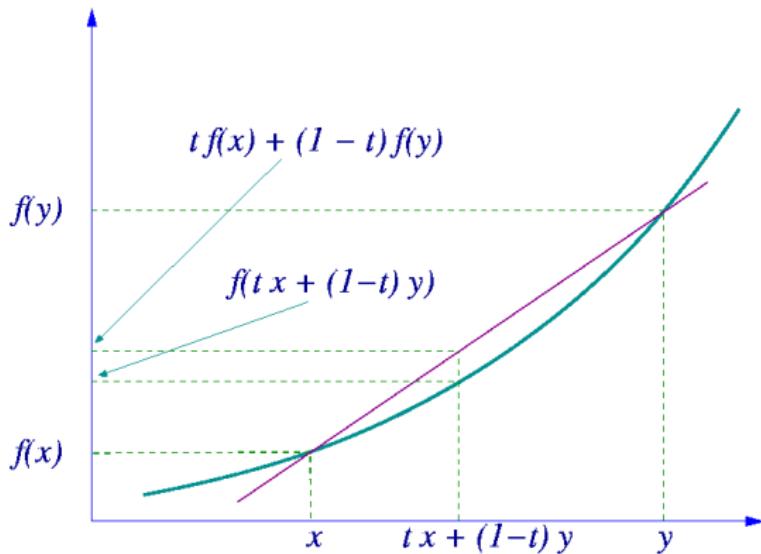
$$\mathbb{E}[R(\hat{\beta}_\lambda) - R(\beta^*)] \leq \text{cst} \times \frac{\sigma^2 d_0 \log(d)}{n}.$$

- ▶ Unfortunately, computing such estimators is out of reach as soon as d is a few tens.

Convexity (the return)

Convex function f : for any $t \in (0, 1)$,

$$f(tx + (1 - t)y) \leq tf(x) + (1 - t)f(y).$$



Convex surrogates for the ℓ^0 norm

$||\theta||_0 = \#\{i | \theta_i \neq 0\}$. ℓ^p norm for $p \in \{1, 2, \dots\}$:

$$||\theta||_p = \left(\sum_{j=1}^d |\theta_j|^p \right)^{\frac{1}{p}}.$$

Convex surrogates for the ℓ^0 norm

$||\theta||_0 = \#\{i | \theta_i \neq 0\}$. ℓ^p norm for $p \in \{1, 2, \dots\}$:

$$||\theta||_p = \left(\sum_{j=1}^d |\theta_j|^p \right)^{\frac{1}{p}}.$$

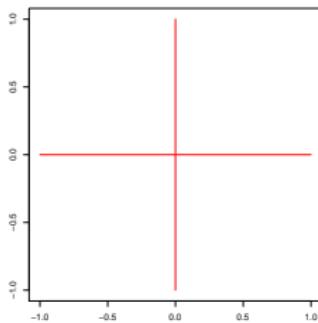
Unit ball in ℓ^0 , ℓ^1 and ℓ^2 norms.

Convex surrogates for the ℓ^0 norm

$||\theta||_0 = \#\{i|\theta_i \neq 0\}$. ℓ^p norm for $p \in \{1, 2, \dots\}$:

$$||\theta||_p = \left(\sum_{j=1}^d |\theta_j|^p \right)^{\frac{1}{p}}.$$

Unit ball in ℓ^0 , ℓ^1 and ℓ^2 norms.

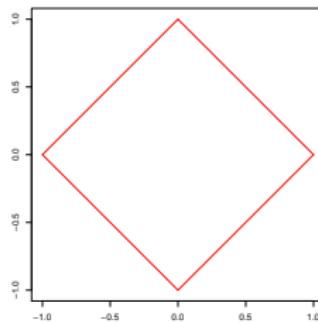
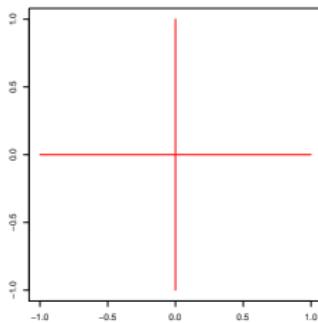


Convex surrogates for the ℓ^0 norm

$||\theta||_0 = \#\{i|\theta_i \neq 0\}$. ℓ^p norm for $p \in \{1, 2, \dots\}$:

$$||\theta||_p = \left(\sum_{j=1}^d |\theta_j|^p \right)^{\frac{1}{p}}.$$

Unit ball in ℓ^0 , ℓ^1 and ℓ^2 norms.

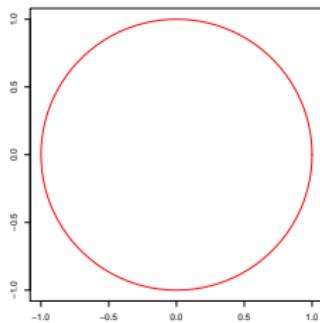
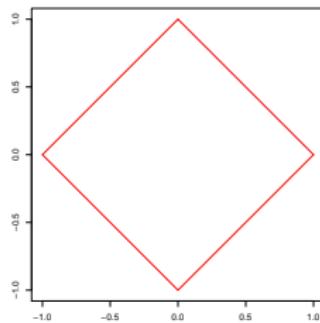
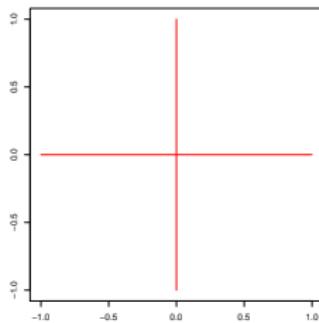


Convex surrogates for the ℓ^0 norm

$||\theta||_0 = \#\{i|\theta_i \neq 0\}$. ℓ^p norm for $p \in \{1, 2, \dots\}$:

$$||\theta||_p = \left(\sum_{j=1}^d |\theta_j|^p \right)^{\frac{1}{p}}.$$

Unit ball in ℓ^0 , ℓ^1 and ℓ^2 norms.



ℓ^1 penalty

Goal: Circumvent the computational dead-end of the ℓ^0 penalty while preserving its good statistical properties.

ℓ^1 penalty

Goal: Circumvent the computational dead-end of the ℓ^0 penalty while preserving its good statistical properties.

Convexified problem

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \{ \| \mathbf{Y} - \mathbf{X}\beta \|^2 + \lambda \|\beta\|_1 \}.$$

Oracle inequality

For $\lambda \asymp \log(d)/n$,

$$\mathbb{E} \|\hat{\beta}_\lambda - \beta^*\|^2 \leq \text{cst} \times d_0 \sqrt{\frac{\log(d)}{n}}.$$

Gradient Descent

Goal: minimize a differentiable function f (compute $\arg \min_x f(x)$).

Gradient Descent

Goal: minimize a differentiable function f (compute $\arg \min_x f(x)$).

Input: tolerance ϵ , initialization x_0 , step size α .

Gradient Descent

Goal: minimize a differentiable function f (compute $\arg \min_x f(x)$).

Input: tolerance ϵ , initialization x_0 , step size α .

While $f'(x_k) \geq \epsilon$ $x_{k+1} = x_k - \alpha f'(x_k)$

A Variety of Penalties

Ridge regression (a.k.a. Tikhonov regularization):

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \{ ||\mathbf{Y} - \mathbf{X}\beta||^2 + \lambda ||\beta||_2^2 \}.$$

A Variety of Penalties

Ridge regression (a.k.a. Tikhonov regularization):

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \{ ||\mathbf{Y} - \mathbf{X}\beta||^2 + \lambda ||\beta||_2^2 \}.$$

Fused Lasso:

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \left\{ ||\mathbf{Y} - \mathbf{X}\beta||^2 + \lambda ||\beta||_1 + \lambda \sum_{j=2}^d |\beta_j - \beta_{j-1}| \right\}.$$

A Variety of Penalties

Smoothed Lasso:

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \left\{ \|\mathbf{Y} - \mathbf{X}\beta\|^2 + \lambda \|\beta\|_1 + \lambda \sum_{j=2}^d (\beta_j - \beta_{j-1})^2 \right\}.$$

A Variety of Penalties

Smoothed Lasso:

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \left\{ \|\mathbf{Y} - \mathbf{X}\beta\|^2 + \lambda \|\beta\|_1 + \lambda \sum_{j=2}^d (\beta_j - \beta_{j-1})^2 \right\}.$$

Elastic Net:

$$\hat{\beta}_\lambda \in \arg \min_{\beta \in \mathbb{R}^d} \left\{ \|\mathbf{Y} - \mathbf{X}\beta\|^2 + \lambda \|\beta\|_1 + \lambda \|\beta\|_2^2 \right\}.$$

2 Algorithmes

2.3 kNN

k Nearest Neighbors



2 Algorithmes

2.4 Adaboost

Adaboost

Binary classification $Y \in \{0, 1\}$.

Parameters: M , weak learners h_1, \dots, h_M .

Initialisation: $D_1(i) = 1/n$ for any $i = 1, \dots, n$.

For $k = 1, \dots, M - 1$

1. Take h_k and compute

$$\epsilon_k = \sum_{i: h_k(\mathbf{X}_i) \neq Y_i} D_k(i).$$

2. Compute $\alpha_k = \frac{1}{2} \log \frac{1-\epsilon_k}{\epsilon_k}$.

- 3.

$$D_{k+1}(i) = \begin{cases} \frac{D_k(i)}{W_{k+1}} \exp(-\alpha_k) & \text{if } h_k(\mathbf{X}_i) = Y_i, \\ \frac{D_k(i)}{W_{k+1}} \exp(\alpha_k) & \text{if } h_k(\mathbf{X}_i) \neq Y_i. \end{cases}$$

Final decision: $\hat{f}(\cdot) = \sum_{k=1}^M \alpha_k h_k(\cdot)$.

2 Algorithmes

2.5 NMF

NMF



2 Algorithms

2.6 Agrégation

*All models are wrong
but some are useful*



George E.P. Box



If the only tool you have is a hammer, you tend to see every problem as a nail.

(Abraham Maslow)



Principle

Assume that over some preliminary sample $\mathcal{D}_k = (\mathbf{X}_i, Y_i)_{i=1}^k$, you have generated M different predictors $\hat{r}_{k,1}, \dots, \hat{r}_{k,M}$, i.e.,

$$\hat{r}_{k,j}: \mathcal{X} \rightarrow \mathcal{Y}, \quad \forall j = 1, \dots, M.$$

Principle

Assume that over some preliminary sample $\mathcal{D}_k = (\mathbf{X}_i, Y_i)_{i=1}^k$, you have generated M different predictors $\hat{r}_{k,1}, \dots, \hat{r}_{k,M}$, i.e.,

$$\hat{r}_{k,j}: \mathcal{X} \rightarrow \mathcal{Y}, \quad \forall j = 1, \dots, M.$$

Let

$$\mathbb{D} = \{\hat{r}_{k,1}, \dots, \hat{r}_{k,M}\}.$$

Principle

Assume that over some preliminary sample $\mathcal{D}_k = (\mathbf{X}_i, Y_i)_{i=1}^k$, you have generated M different predictors $\hat{r}_{k,1}, \dots, \hat{r}_{k,M}$, i.e.,

$$\hat{r}_{k,j}: \mathcal{X} \rightarrow \mathcal{Y}, \quad \forall j = 1, \dots, M.$$

Let

$$\mathbb{D} = \{\hat{r}_{k,1}, \dots, \hat{r}_{k,M}\}.$$

Let $\mathcal{D}_\ell = (\mathbf{X}_i, Y_i)_{i=1}^\ell$ be another sample.

Principle

Assume that over some preliminary sample $\mathcal{D}_k = (\mathbf{X}_i, Y_i)_{i=1}^k$, you have generated M different predictors $\hat{r}_{k,1}, \dots, \hat{r}_{k,M}$, i.e.,

$$\hat{r}_{k,j}: \mathcal{X} \rightarrow \mathcal{Y}, \quad \forall j = 1, \dots, M.$$

Let

$$\mathbb{D} = \{\hat{r}_{k,1}, \dots, \hat{r}_{k,M}\}.$$

Let $\mathcal{D}_\ell = (\mathbf{X}_i, Y_i)_{i=1}^\ell$ be another sample.

An *aggregation procedure* is a functional Γ_ℓ of the elements of the dictionary \mathbb{D} such that Γ_ℓ is still a predictor.

Selection

$$\Gamma_\ell(\hat{r}_{k,1}, \dots, \hat{r}_{k,M}) = \hat{r}_{k,j} \quad \text{for some } j \in \{1, \dots, M\}.$$

Selection

$$\Gamma_\ell(\hat{r}_{k,1}, \dots, \hat{r}_{k,M}) = \hat{r}_{k,j} \quad \text{for some } j \in \{1, \dots, M\}.$$

Example: for some $\lambda > 0$,

$$\Gamma_\ell(\hat{r}_{k,1}, \dots, \hat{r}_{k,M}) = \arg \inf_{f \in \mathbb{D}} \{R_\ell(f) + \text{pen}_\lambda(f)\}.$$

Convex aggregation

$$\Gamma_\ell(\hat{r}_{k,1}, \dots, \hat{r}_{k,M}) = \sum_{j=1}^M \omega_{\ell,j} \hat{r}_{k,j},$$

such that $\omega_{\ell,1}, \dots, \omega_{\ell,M} \geq 0$ and $\sum_{j=1}^M \omega_{\ell,j} = 1$.

Convex aggregation

$$\Gamma_\ell(\hat{r}_{k,1}, \dots, \hat{r}_{k,M}) = \sum_{j=1}^M \omega_{\ell,j} \hat{r}_{k,j},$$

such that $\omega_{\ell,1}, \dots, \omega_{\ell,M} \geq 0$ and $\sum_{j=1}^M \omega_{\ell,j} = 1$.

Examples: Uniform weights ($\omega_{\ell,j} = 1/M$), Exponentially Weighted Aggregation (EWA).



EWA

For some (inverse temperature parameter) $\lambda > 0$,

$$\Gamma_{\ell,\lambda}^{\text{ewa}} = \sum_{j=1}^M \frac{\exp[-\lambda R_\ell(\hat{r}_{k,j})]}{\sum_{i=1}^M \exp[-\lambda R_\ell(\hat{r}_{k,i})]} \hat{r}_{k,j}.$$

EWA

For some (inverse temperature parameter) $\lambda > 0$,

$$\Gamma_{\ell,\lambda}^{\text{ewa}} = \sum_{j=1}^M \frac{\exp[-\lambda R_\ell(\hat{r}_{k,j})]}{\sum_{i=1}^M \exp[-\lambda R_\ell(\hat{r}_{k,i})]} \hat{r}_{k,j}.$$

$\lambda \rightarrow 0$:

EWA

For some (inverse temperature parameter) $\lambda > 0$,

$$\Gamma_{\ell,\lambda}^{\text{ewa}} = \sum_{j=1}^M \frac{\exp[-\lambda R_\ell(\hat{r}_{k,j})]}{\sum_{i=1}^M \exp[-\lambda R_\ell(\hat{r}_{k,i})]} \hat{r}_{k,j}.$$

$\lambda \rightarrow 0$: uniform weights.

EWA

For some (inverse temperature parameter) $\lambda > 0$,

$$\Gamma_{\ell,\lambda}^{\text{ewa}} = \sum_{j=1}^M \frac{\exp[-\lambda R_\ell(\hat{r}_{k,j})]}{\sum_{i=1}^M \exp[-\lambda R_\ell(\hat{r}_{k,i})]} \hat{r}_{k,j}.$$

$\lambda \rightarrow 0$: uniform weights.

$\lambda \rightarrow \infty$:

EWA

For some (inverse temperature parameter) $\lambda > 0$,

$$\Gamma_{\ell,\lambda}^{\text{ewa}} = \sum_{j=1}^M \frac{\exp[-\lambda R_\ell(\hat{r}_{k,j})]}{\sum_{i=1}^M \exp[-\lambda R_\ell(\hat{r}_{k,i})]} \hat{r}_{k,j}.$$

$\lambda \rightarrow 0$: uniform weights.

$\lambda \rightarrow \infty$: ERM.

EWA is supported by sharp oracle inequalities of the flavor

$$\mathbb{E}[\Gamma_\ell^{\text{ewa}}(\mathbf{X}) - Y]^2 \leq \min_{j=1,\dots,M} \mathbb{E}[\hat{r}_{k,j}(\mathbf{X}) - Y]^2 + \frac{\log(M)}{\lambda n}.$$

Linear aggregation

$$\Gamma_\ell(\hat{r}_{k,1}, \dots, \hat{r}_{k,M}) = \sum_{j=1}^M \omega_{\ell,j} \hat{r}_{k,j}.$$

Linear aggregation

$$\Gamma_\ell(\hat{r}_{k,1}, \dots, \hat{r}_{k,M}) = \sum_{j=1}^M \omega_{\ell,j} \hat{r}_{k,j}.$$

Example: PAC-Bayesian Aggregation.



2 Algorithmes

2.7 Théorie PAC-Bayésienne

PAC-Bayesian Theory



2 Algorithmes

2.8 Sampling algorithms: Gibbs sampler, MCMC

Metropolis-Hastings Algorithm



Conclusion

Take-home messages

- ▶ Sound mathematical foundations for statistical learning

Take-home messages

- ▶ Sound mathematical foundations for statistical learning
- ▶ Abundance of methods to deal with prediction: parametric, semiparametric, nonparametric...

Take-home messages

- ▶ Sound mathematical foundations for statistical learning
- ▶ Abundance of methods to deal with prediction: parametric, semiparametric, nonparametric... and meta-methods: aggregation
- ▶ Theoretical goal: tight risk upper bounds (oracle inequalities)

Take-home messages

- ▶ Sound mathematical foundations for statistical learning
- ▶ Abundance of methods to deal with prediction: parametric, semiparametric, nonparametric... and meta-methods: aggregation
- ▶ Theoretical goal: tight risk upper bounds (oracle inequalities)
- ▶ Algorithmical goal: tractable methods which scale up to modern (massive and complex) data

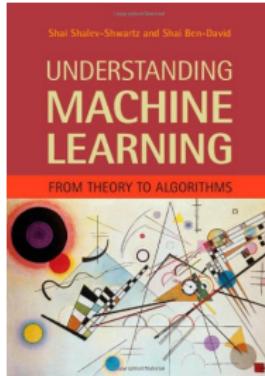
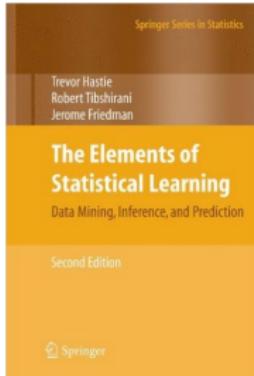
Take-home messages

- ▶ Sound mathematical foundations for statistical learning
- ▶ Abundance of methods to deal with prediction: parametric, semiparametric, nonparametric... and meta-methods: aggregation
- ▶ Theoretical goal: tight risk upper bounds (oracle inequalities)
- ▶ Algorithmical goal: tractable methods which scale up to modern (massive and complex) data
- ▶ One of the most exciting fields to work in!



References

- T. Hastie, R. Tibshirani and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, Springer, 2009.
- S. Shalev-Shwartz and S. Ben-David. *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, 2014.





Benjamin Guedj, Ph.D.

Équipe-projet Modal
Inria Lille - Nord Europe