Computational Mathematics for Learning and Data Analysis: introduction

Antonio Frangioni

Department of Computer Science
University of Pisa
www.di.unipi.it/~frangio
frangio@di.unipi.it

Computational Mathematics for Learning and Data Analysis
Master in Computer Science – University of Pisa

A.Y. 2020/21

Outline

- 1 course (9 CFU/ECTS)
- 1 program
- 1 exam
- 2 related but $\neq$ areas of mathematics $=$ 2 lecturers

Federico Poloni (Numerical methods)

Dipartimento di Informatica, room 343
Tel. 050 2213143, e-mail: federico.poloni@unipi.it
Office hours: Friday 11:00 – 13:00

Antonio Frangioni (Optimization)

Dipartimento di Informatica, room 381
Tel. 050 2212789, e-mail: frangio@di.unipi.it
Office hours: Tuesday 9:00 – 11:00

- Course Schedule
  - Tue 14:15 – 16:00
  - ~~Wed 11:00 – 12:45~~ (!!)
  - Thu 14:15 – 16:00
  - Fri 9:00 – 10:45

- Web page: `https://elearning.di.unipi.it/course/view.php?id=198`

- Team for lectures: `https://teams.microsoft.com/l/team/19%3ad5f424681cc0412ea634f8884d29041d%40thread.tacv2/conversations?groupId=2750a4f8-7916-4b55-bf8e-621b912e7e31&tenantId=c7456b31-a220-47f5-be52-473828670aa1`

- Exam: project (groups of 2) + oral exam

  Projects either "ML" or "no-ML", but no difference in work and grading

Outline

- Huge amounts of data is generated and collected, but one has to make sense of it in order to use it: learn it
- Take something big and unwieldy and produce something small and nimble that can be used in its stead ("actionable")
- That's a (mathematical) model
- Word comes from "modulus", diminutive from "modus" = "measure": "small measure", "measure in the small" (small is nice)
- First known uses in architecture: proving in the small that the real building won't collapse (particularly famous the models of Filippo Brunelleschi for the Cupola of the Cathedral of Florence)
- Countless many physical models afterwards (planes, cars, ...), but mathematics is cheaper than bricks / wood / iron ...
- Yet, mathematical problems can be difficult, too, for various reasons (and, of course, only truly viable after computers)

Choosing a mathematical model

- How a mathematical model should be:
  1. accurate (describes well the process at hand)
  2. computationally inexpensive (gives answers rapidly)
  3. general (can be applied to many different processes)

  Typically impossible to have all three!
- Developing general models is convenient (work once, apply many)
- The shape of the model controls the computational cost
- But how to get accuracy for a given application?
  model is parametric, learn the right values of the parameters
- In other words: within the family of (usually, infinitely many) models with the given shape, find the one that better represent your phenomenon
- This is fitting, and it is clearly some sort of optimization problem
- Solving the fitting problem is typically the computational bottleneck
- However, ML $\gg$ fitting: fitting minimizes training error $\equiv$ empirical risk, but ML aims at minimizing test error $\equiv$ risk $\equiv$ generalization error!

Example 1: Linear Estimation

- A phenomenon measured by one number y is believed to depend on a vector $x = [\, x_1 \,, \ldots \,, x_n \,]$ of other numbers
- Available set of observations $(y^1, x^1), \ldots, (y^m, x^m)$
- Horribly optimistic assumption: the dependence is linear, i.e.,

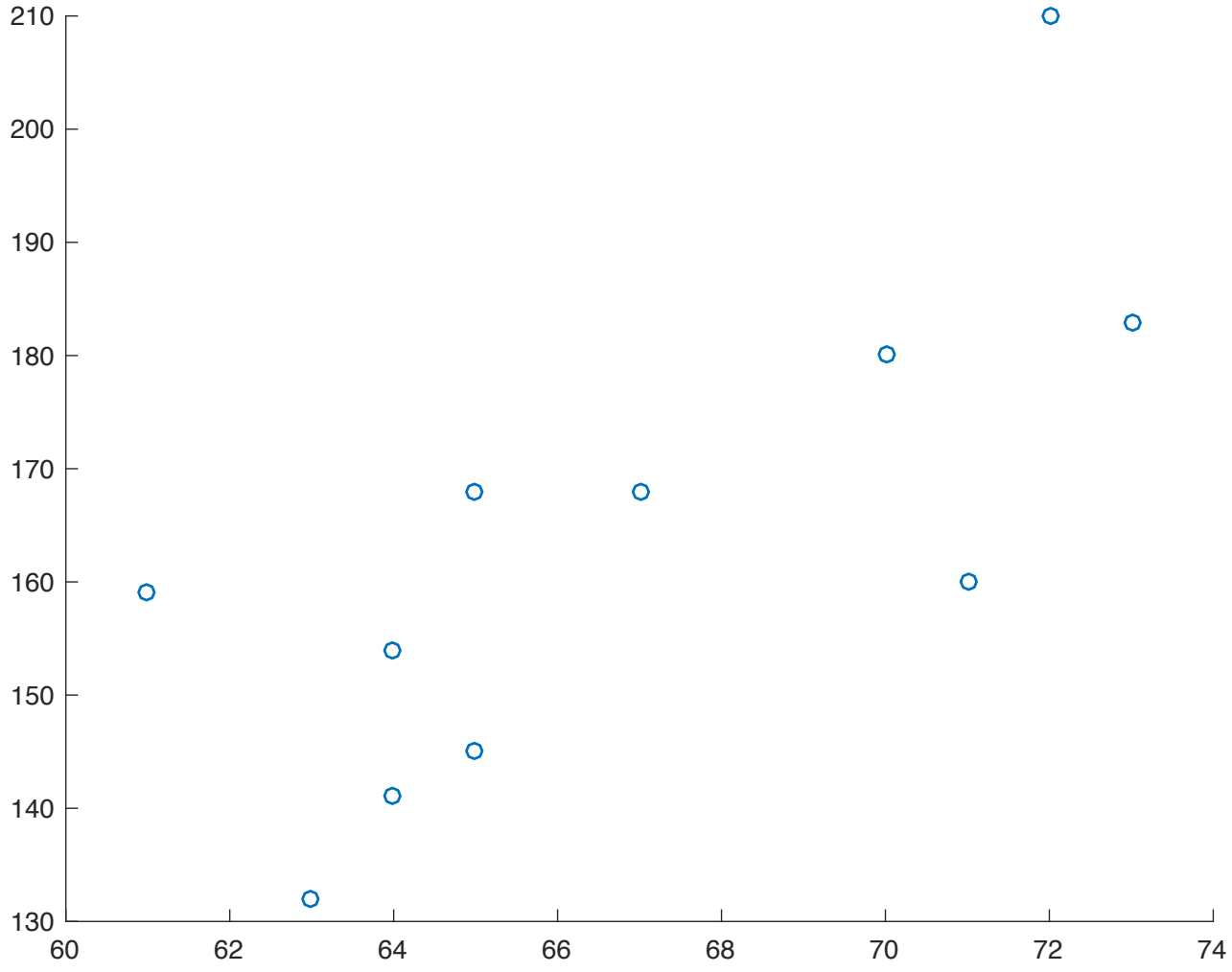
$$y = \sum_{i=1}^{n} w_i x_i + w_0 = wx + w_0$$

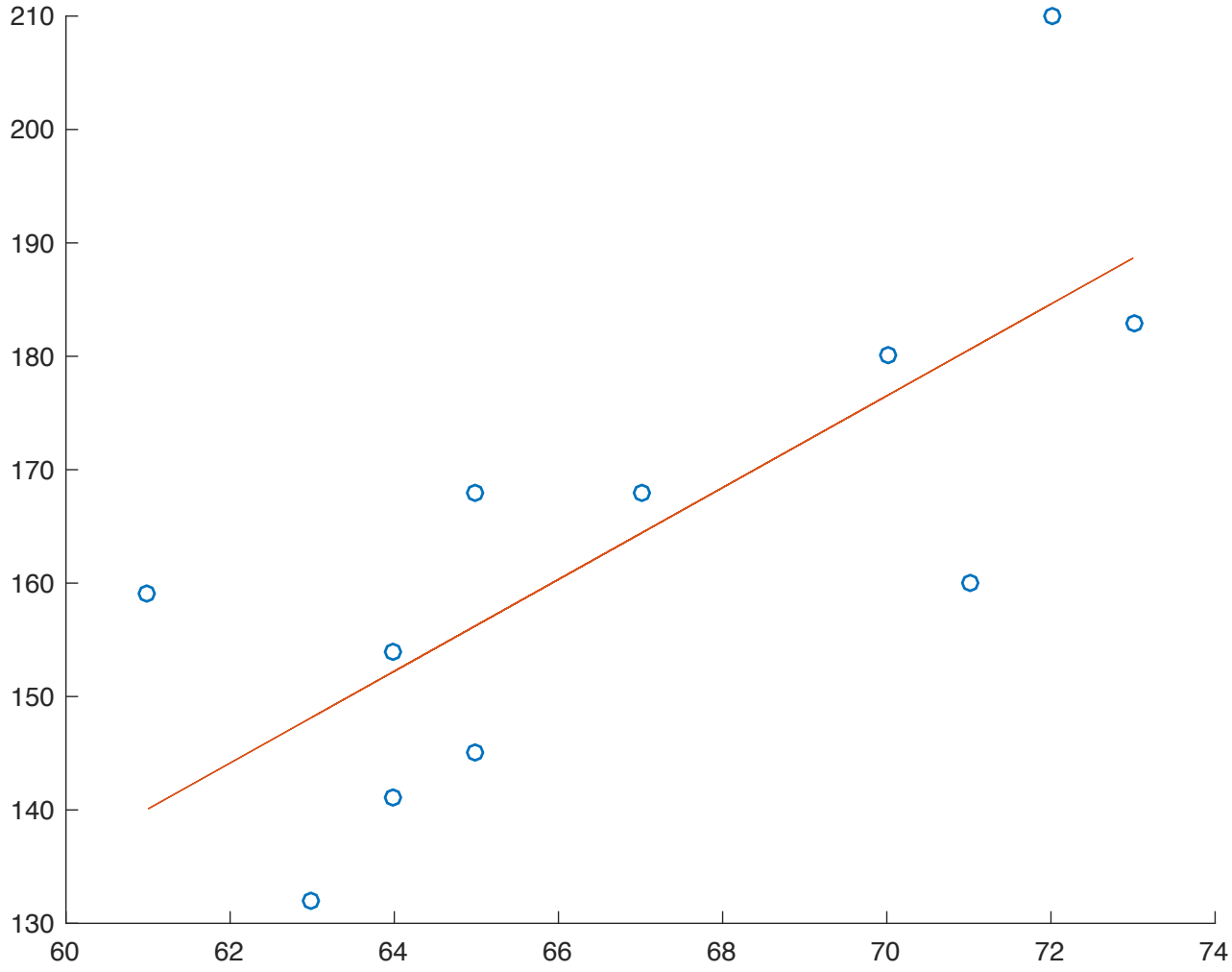
  for fixed $n+1$ real parameters $w = [\, w_0 \,, w_+ = [w_1 \,, \ldots \,, w_n \,] \,]$
- This would imply that $y^i = w_+ x^i + w_0$ for all $i = 1, \ldots, m$, which is not really true for any w and w_0
- Find the w for which it is less untrue (Linear Least Squares):

$$y = \begin{bmatrix} y^1 \\ \vdots \\ y^m \end{bmatrix} \;, \quad X = \begin{bmatrix} 1 & x^1 \\ \vdots & \vdots \\ 1 & x^m \end{bmatrix} \;, \quad \min_w \left|\left| y - Xw \right|\right|$$

- Simple closed formula: $XX^T w = X^T y \;\; \leftarrow w = (XX^T)^{-1} X^T y$

Linear Estimation (cont.d)





- In `Matlab`, this is just `w = X \ y`
- Trade-off: very simple fitting for exceedingly crude model $\Longrightarrow$ high risk
- Then, of course Nonlinear Estimation ...

Example 2: Low-rank approximation

- A (large, sparse) matrix $M \in \mathbb{R}^{n \times m}$ describes a phenomenon depending on pairs (e.g., objects chosen from customers)
- Describe $M \approx AB$ with "tall and thin" $A \in \mathbb{R}^{n \times k}$ and "fat and large" $B \in \mathbb{R}^{k \times m}$ $(k \ll n, m)$

$$\boxed{M} \approx \boxed{A} \cdot \boxed{B} \quad , \quad \min_{A,B} \left|\left| M - AB \right|\right|$$

  $\equiv$ find a few features that describe most of users' choices
- Many applications (neural networks, community analysis, ...)
- A, B can be obtained from eigenvectors of $M^T M$ and MM^T ...

Example 2: Low-rank approximation

- A (large, sparse) matrix $M \in \mathbb{R}^{n \times m}$ describes a phenomenon depending on pairs (e.g., objects chosen from customers)
- Describe $M \approx AB$ with "tall and thin" $A \in \mathbb{R}^{n \times k}$ and "fat and large" $B \in \mathbb{R}^{k \times m}$ $(k \ll n, m)$

$$\boxed{M} \approx \boxed{A} \cdot \boxed{B} \quad , \quad \min_{A,B} \left|\left| M - AB \right|\right|$$

  $\equiv$ find a few features that describe most of users' choices
- Many applications (neural networks, community analysis, . . .)
- A, B can be obtained from eigenvectors of $M^T M$ and MM^T . . .
  . . . but that's a huge, possibly dense matrix
- Efficiently solving this problem requires:
  1. low-complexity computation (of course)
  2. avoiding ever explicitly forming $M^T M$ and MM^T (too much memory)
  3. exploiting structure of M (sparsity, similar columns, . . .)
  4. ensuring the solution is numerically stable

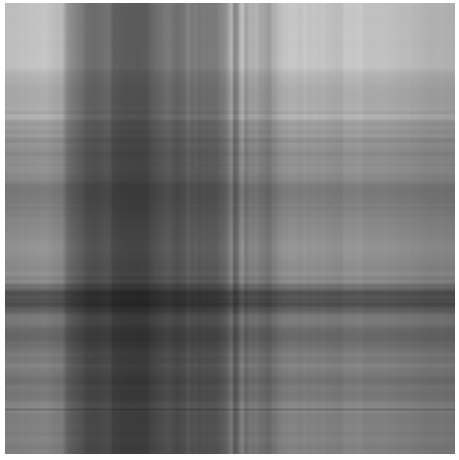
Black/white image $\equiv M$ with color intensities $\in [0, 1]$

Original (512×512) $k = 1$ $k = 10$

$k = 25$ $k = 50$ $k = 100$

Black/white image $\equiv M$ with color intensities $\in [0, 1]$

Original (512×512)



$k = 1$

$k = 10$

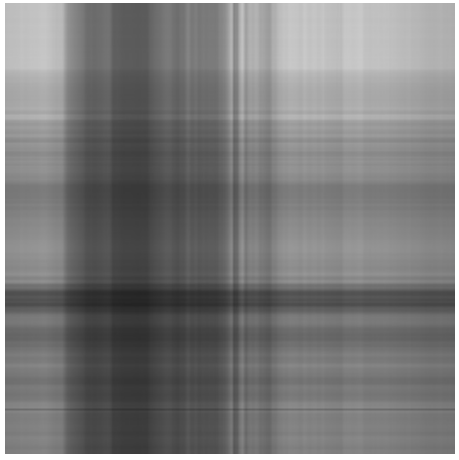
$k = 25$

$k = 50$

$k = 100$

Black/white image $\equiv M$ with color intensities $\in [0, 1]$

Original (512×512)



$k = 1$

$k = 10$

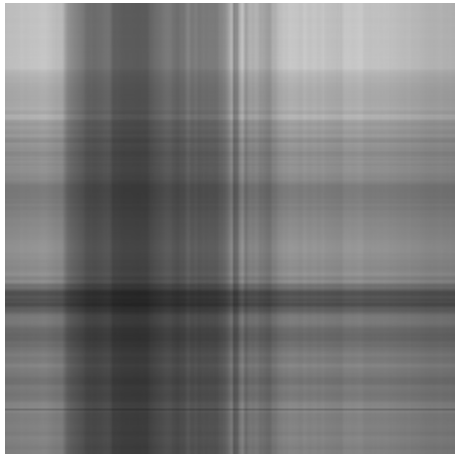
$k = 25$

$k = 50$

$k = 100$

Black/white image $\equiv M$ with color intensities $\in [0, 1]$

Original (512×512)



$k = 1$

$k = 10$

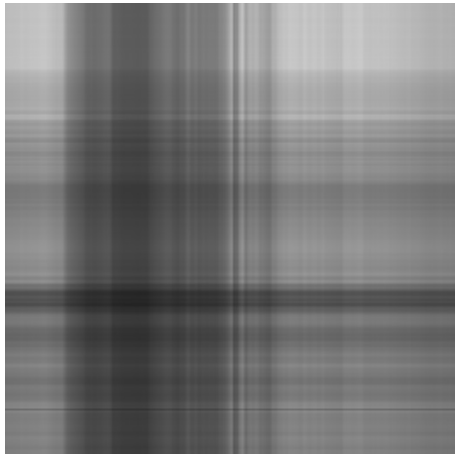
$k = 25$

$k = 50$

$k = 100$

Black/white image $\equiv M$ with color intensities $\in [0, 1]$

Original (512×512)



$k = 1$

$k = 10$

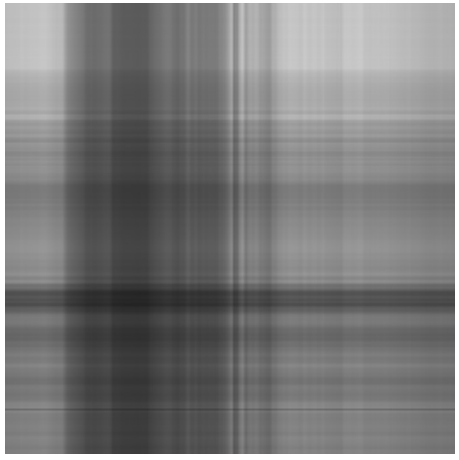
$k = 25$

$k = 50$

$k = 100$

Black/white image $\equiv M$ with color intensities $\in [0, 1]$

Original (512×512)



$k = 1$

$k = 10$

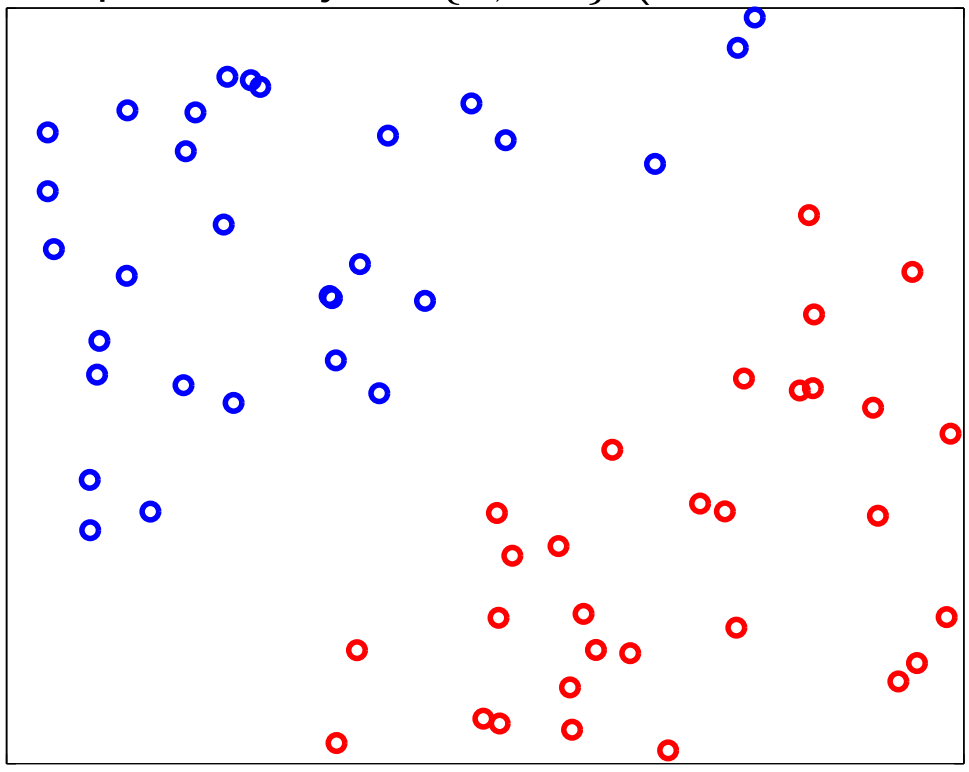
$k = 25$

$k = 50$

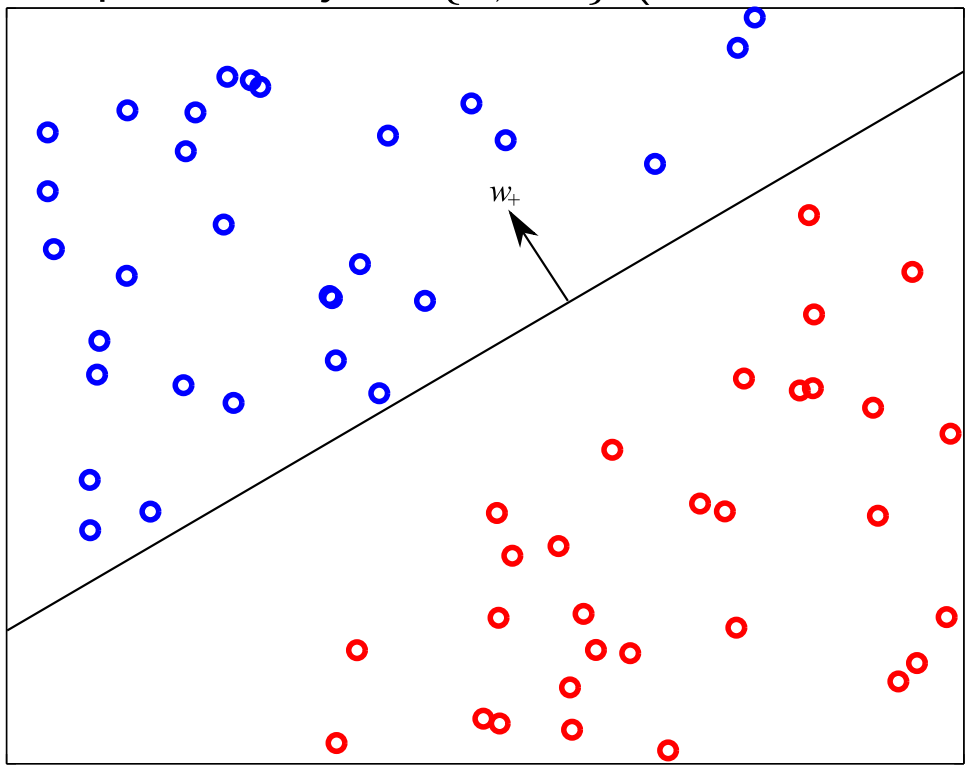
$k = 100$

Example 3: Support Vector Machines

- Same setting as Example 1 but $y^i \in \{1, -1\}$ (have cancer or not)

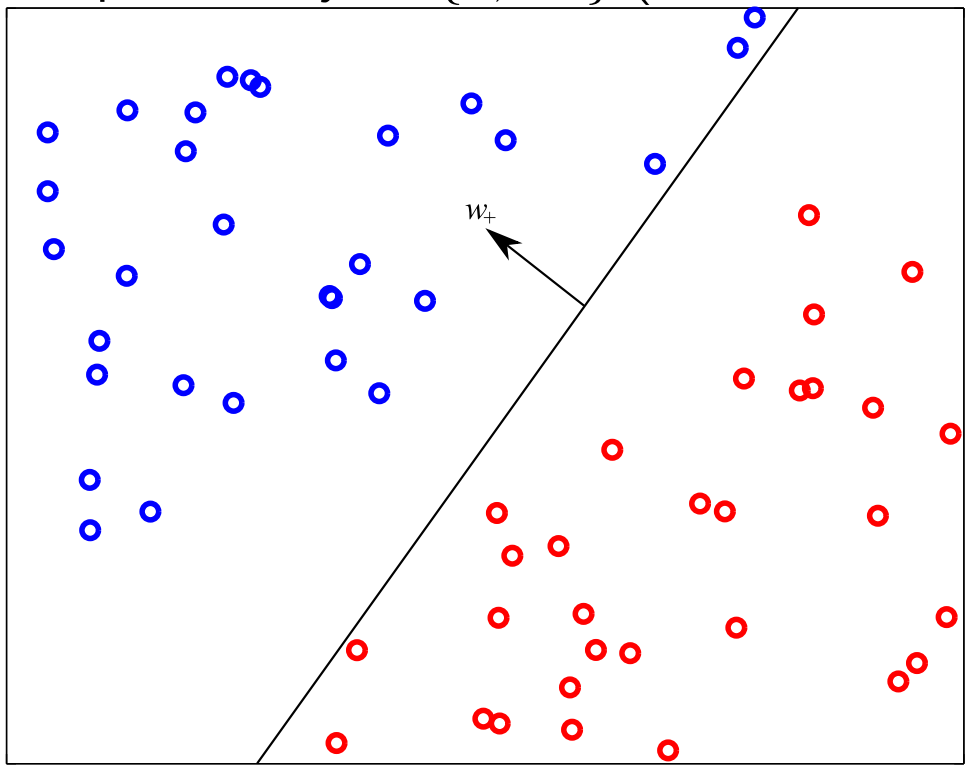


- Same setting as Example 1 but $y^i \in \{1, -1\}$ (have cancer or not)



- Want to linearly separate the two sets (diagnose the next patient)
- Countless many applications (medical diagnosis, OCR, spam filtering, fraud detection, marketing, image processing . . .)

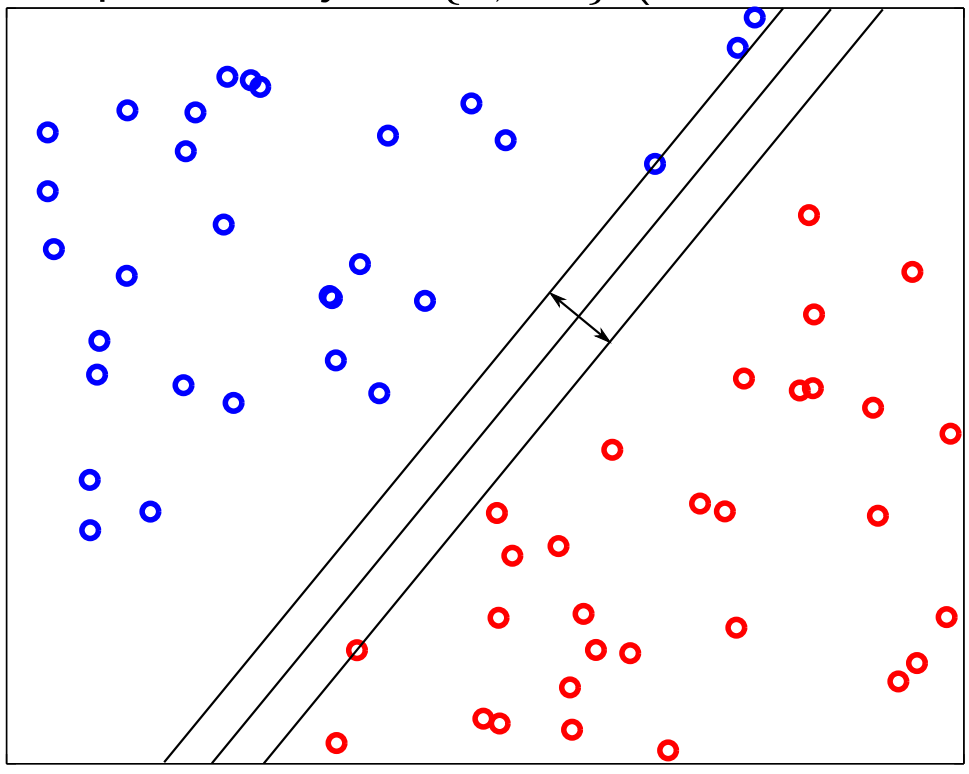
- Same setting as Example 1 but $y^i \in \{1, -1\}$ (have cancer or not)



- Want to linearly separate the two sets (diagnose the next patient)
- Countless many applications (medical diagnosis, OCR, spam filtering, fraud detection, marketing, image processing . . .)
- But which hyperplane do we choose?

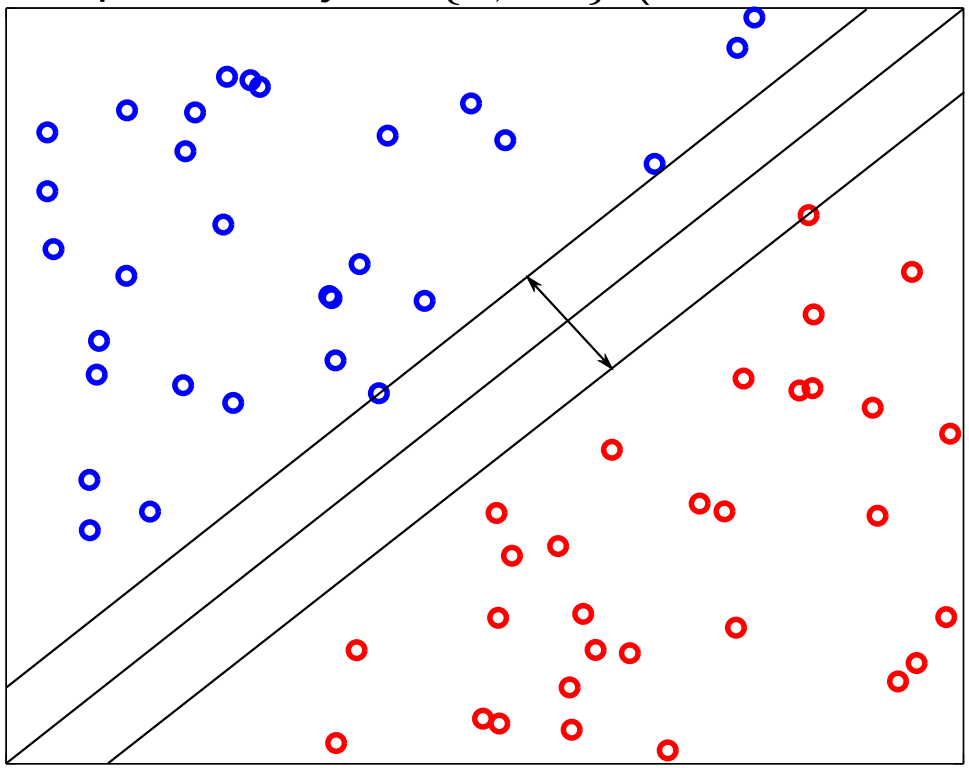
Example 3: Support Vector Machines

- Same setting as Example 1 but $y^i \in \{1, -1\}$ (have cancer or not)



- Want to linearly separate the two sets (diagnose the next patient)
- Countless many applications (medical diagnosis, OCR, spam filtering, fraud detection, marketing, image processing . . .)
- But which hyperplane do we choose?
- Intuitively, the margin is important

- Same setting as Example 1 but $y^i \in \{1, -1\}$ (have cancer or not)



- Want to linearly separate the two sets (diagnose the next patient)
- Countless many applications (medical diagnosis, OCR, spam filtering, fraud detection, marketing, image processing . . .)
- But which hyperplane do we choose?
- Intuitively, the margin is important
- More margin $\Longrightarrow$ more "robust" classification

- Distance of // hyperplanes (w_+, w_0) and (w_+, w_0') is $|\, w_0 - w_0' \,| \,/\, ||\, w_+ \,||$
- We can always take the hyperplane in "the middle" + scale w
  $\Longrightarrow w_+ x^i + w_0 \geq 1$ if $y^i = 1$, $w_+ x^i + w_0 \leq -1$ if $y^i = -1$
- The maximum margin separating hyperplane is the solution of

$$\min_w \left\{ ||\, w_+ \,||^2 \;:\; y^i (w_+ x^i + w_0) \geq 1 \quad i = 1, \ldots, m \right\}$$

  (margin $= 2/||w_+||$, "2" because I say so), assuming any exists
- What if it does not? Support Vector Machine

$$\text{(SVM-P)} \quad \min_{w,\xi} ||\, w_+ \,||^2 + C \sum_{i=1}^m \xi_i$$
$$y^i (w_+ x^i + w_0) \geq 1 - \xi_i \;\;,\;\; \xi_i \geq 0 \qquad i = 1, \ldots, m$$

  C weighs violation of separation against margin (how?)
- A convex constrained problem with "complex constraints"

- Equivalently, one can solve the dual problem (??? what ???)

$$\text{(SVM-D)} \quad \max_{\alpha} \ \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \langle x^i , x^j \rangle \alpha_j$$

$$\sum_{i=1}^{m} y^i \alpha_i = 0$$

$$0 \leq \alpha_i \leq C \qquad i = 1, \ldots, m$$

  a convex constrained quadratic program, but with "simple constraints"

- Solve one problem by solving an apparently different one:
  α^* optimal for (SVM-D) $\Longrightarrow$ $w_+^* = \sum_{i=1}^{m} \alpha_i^* y^i x^i$ optimal for (SVM-P)

- Dual formulation $\Longrightarrow$ kernel trick: input space $\rightsquigarrow$ (larger) feature space

$$\langle x^i , x^j \rangle \rightsquigarrow \langle \phi(x^i) , \phi(x^k) \rangle$$

  where points are hopefully "more linearly separable"

- Feature space can be infinite-dimensional, provided that
  scalar product can be (efficiently) computed

- Efficient algorithms: (SVM-P) or (SVM-D) (or both), complexity, ...

Outline

- . . . of Computational Mathematics for Learning and Data Analysis

- . . . of Computational Magic for Learning and Data Analysis

Who is this course for?

Who is this course for?

- Mostly, the ordinary Witcher who needs a good hands-on knowledge of magic without being a full baton-wielding mage

Who is this course for?

- Mostly, the ordinary AI expert who needs a good hands-on knowledge of mathematics without being a mathematician

Who is this course for?

- Occasionally, the odd full baton-wielding mage

Who is this course for?

- Occasionally, the odd full baton-wielding mage

  . . . because it feels good to be a mage

Who is this course for?

- Occasionally, the odd expert of mathematics of AI

  . . . because it feels good to be a mathematician

Main quests and side quests

- There are two main quests in the course
  1. Get a general understanding of several different classes of numerical algorithms and their underlying mathematical principles
  2. Be able to actually implement, debug, and tune a few of them

Main quests and side quests

- There are two main quests in the course
  1. Get a general understanding of several different classes of numerical algorithms and their underlying mathematical principles
  2. Be able to actually implement, debug, and tune a few of them
- However, there are also a number of side quests
- These mostly involve proving theorems
- Just can't do without: reasoning about mathematical objects is proving theorems, and algorithms are mathematical objects
- No-one will not pass the exam just for not knowing a proof by heart
- But you will have a lot more fun if you face side quests seriously
- Exercises are there for the same reason

Syllabus

- Linear algebra and calculus background
- Unconstrained optimization and systems of equations
- Direct and iterative methods for linear systems and least-squares
- Numerical methods for unconstrained optimization
- Iterative methods for computing eigenvalues
- Constrained optimization and systems of equations
- Duality (Lagrangian, linear, quadratic, conic, Fenchel's)
- Numerical methods for constrained optimization
- Software tools for numerical computations (Matlab, Octave, ...)
- Sparse hints to AI/ML applications

- Slides prepared by the lecturers
- `Matlab` programs + data
- Recording of lectures
- L. N. Trefethen, D. Bau, Numerical Linear Algebra, SIAM, 1997
- J. Demmel, Applied Numerical Linear Algebra, SIAM, 1996
- S. Boyd, L. Vandenberghe, Convex optimization, 2004 (`http://web.stanford.edu/~boyd/cvxbook/`)
- M.S. Bazaraa, H.D. Sherali, C.M. Shetty, Nonlinear programming: theory and algorithms, Wiley & Sons, 2006
- J. Nocedal, S. Wright, Numerical Optimization, Springer Series in Operations Research and Financial Engineering, 2006
- Students-produced notes $\exists$, hoping to make them more textbook

Outline

- Learning as a computational, hence mathematical, process
- Mathematical foundations of many important learning processes
  ≡ nonlinear optimization and numerical analysis techniques
- Easy problems (linear, quadratic, conic, convex) or local optima,
  because size is huge (hard because large, not hard because hard)
- Besides, in ML the global optimal solution can be bad!
- Emphasis on what can be done by linear algebra
- Focus on methods and software tools
- Applications to be seen in "Machine Learning" and/or "Data Mining"
  (in parallel, you can do it, we talk to each other)


S
Mikrul Gauldurson