

Machine Learning

Introduction to Machine Learning
Notes of Lect. 1-3

Alessio Micheli

micheli@di.unipi.it



Dipartimento di Informatica
Università di Pisa - Italy

**Computational Intelligence &
Machine Learning Group**

Sept, 2020

About ML

- (EN) **Machine Learning**
- *Master Programme in Computer Science*
Master Programme in Data Science and Business Informatics
Master Programme in Digital Humanities
- **Lecturer: Alessio Micheli:** micheli@di.unipi.it
- **Code: 654AA Credits (ECTS): 9 Semester: 1**
- **Material: Moodle** <https://elearning.di.unipi.it/>
- **Record of lectures autumn 2020: Teams platform**
 - Link through "esami.unipi.it" (search "[MACHINE LEARNING](#)")
 - For in-class questions: please use the teams chat
 - Volunteers for camera*

This Lecture

- A. What is ML: a motivational introduction**
- B. Interlude: ML in the curricula and other master degrees**
- C. ML Course info: prerequisites, course structure, Q&A, bibliography, exam, math background**

What is ML

A first look



Prologue: Learning

Learning : universal principles for living beings, society,
andmachines

*The problem of **learning** is arguably
at the very core of the problem of
intelligence, both biological and
artificial*

Poggio, Shelton, *AI Magazine* 1999



What is ML? First view

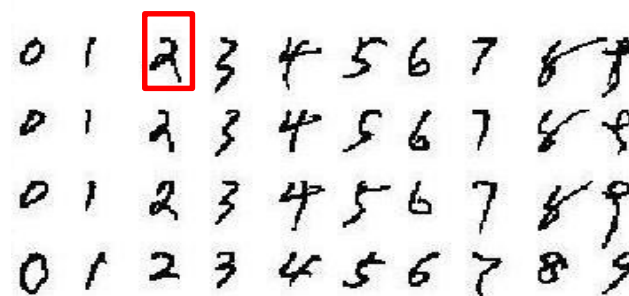
- In Computer Science, theoretical and applicative field called:
 - Apprendimento Automatico *in italiano (it)*
 - Machine Learning (ML) *English and literature* (aka learning systems)
- **Machine Learning** has emerged as an area of research combining the aims of creating *computers that could learn* (AI) and new *powerful adaptive/statistical tools* with rigorous foundation in computational science
- Machines that *learn* by itself. Why? Luxury or necessity?
 - Growing availability and need for analysis of empirical data
Central/methodological role due to changing of paradigm in science: **data-driven**
 - Difficult to provide adaptivity/intelligence by programming (Turing)
→ *learning* as the only choice to provide *intelligence* into the systems ...
- **Pave the way to a new AI era** (Growing data+ HPC+ ML flexibility)

What is ML? Simple examples

- Automatized learning by the system of the experience (set of examples) to address a computational task



Email spam classification



Character/face/speech recognition

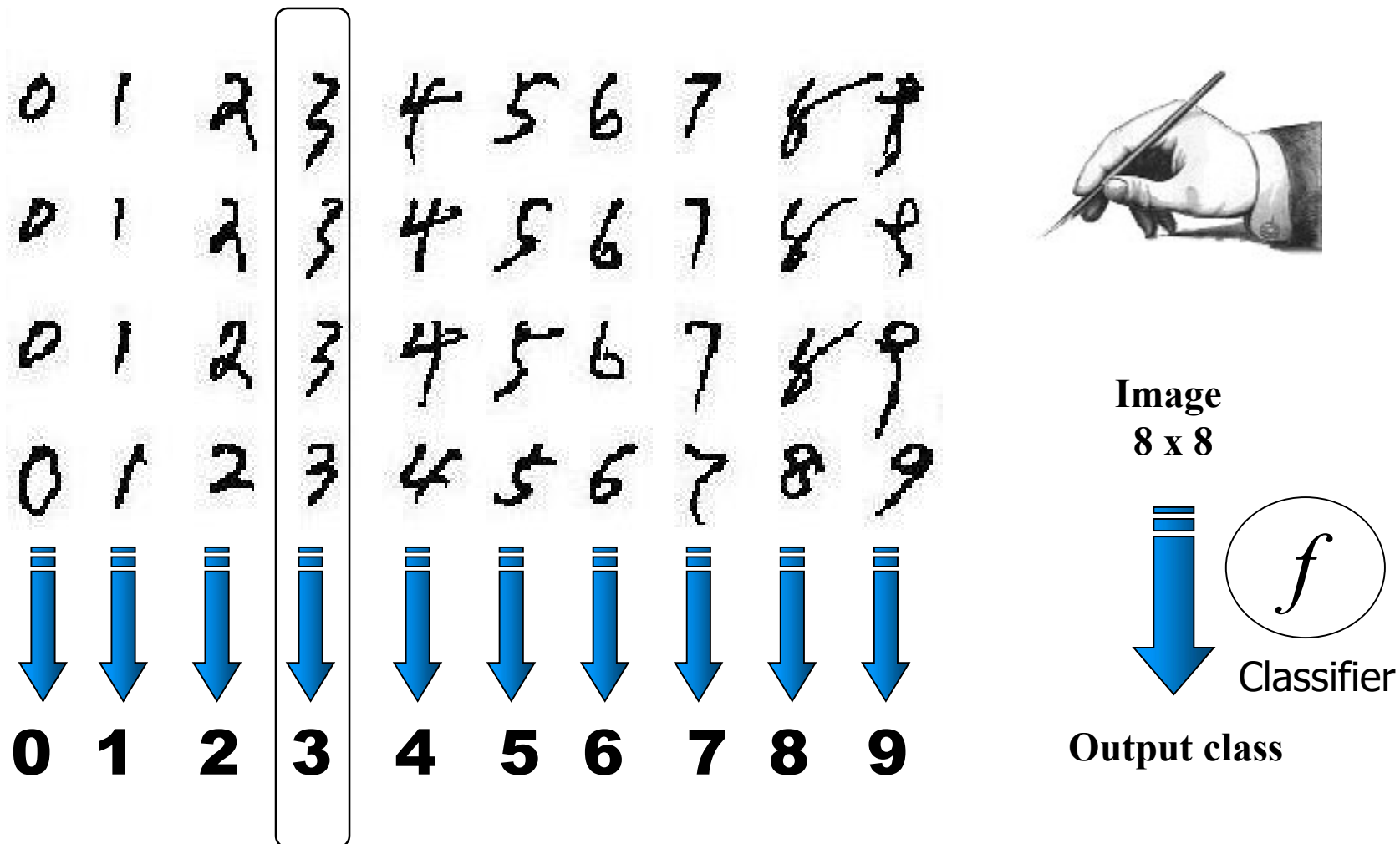
- ... no (or poor) prior knowledge/rules for the solution
but it is (more) easy to have a source of
training experience (data with known results)

Zoom: Handwritten Digits

Recognition: Learn a classifier



Dip. Informatica
University of Pisa



Applicative areas

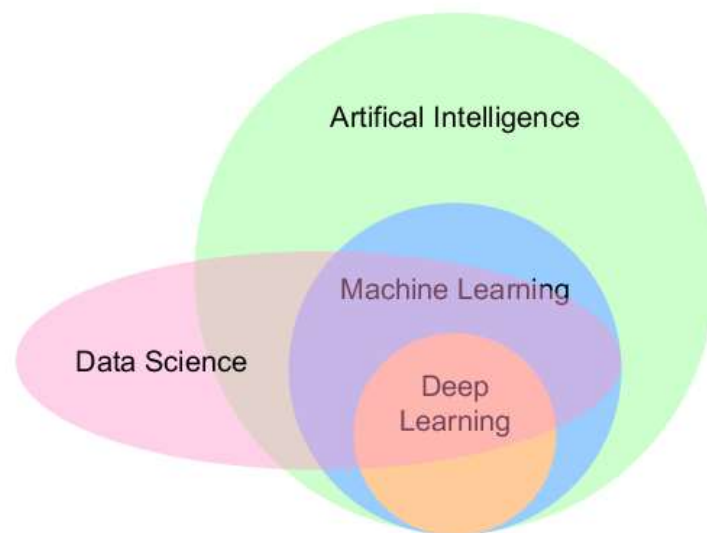
Applicative and related areas:

1. Real-World systems (pervasive, from OCR, to human computer interface , to search engine)
2. New interdisciplinary area, encompassing (since the 80s):
 - **Pattern Recognition** (e.g. face and speech recognition), **Computer Vision, Natural Language Processing** (from text, e.g. email spam classification, to speech processing),
 - **Robotics**, Adaptive Systems and Filters, **Intelligent Sensor Networks**, Personalized components, ...
 - **Knowledge Discovery and Data Mining, Information Retrieval**, Analysis of complex data (Med, **Bio**, Chem, Web, Marketing), Financial forecasting, ...
 - The impact is revolutionary for some fields, e.g. for NLP, Pattern Recognition and Computer Vision, ...

And related areas

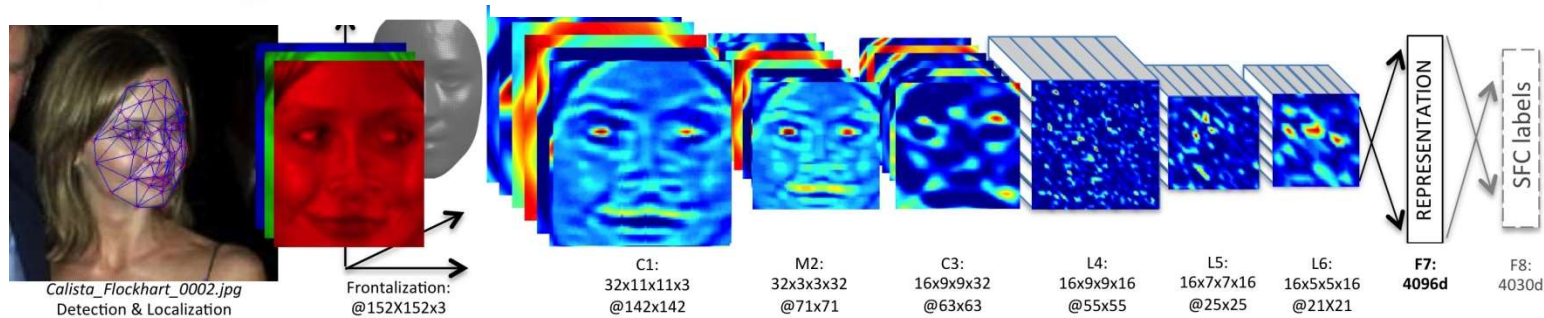
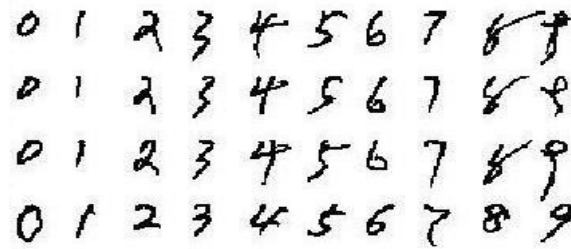
Basis (theories and methods) for new AI approaches ([AI revolution](#))

- [Open huge application areas and opportunities](#)
 - Large effort and investments in Industry
 - Latest GPT (General-purpose technologies: potential to drastically affect pre-existing economic and social structures)
 - Opportunity to be part of a big challenge for AI/CS/Society
 - “ML scientist” (top ranked in job growing demand)



*Not all the story
but nice view at
a glance*

A look by pictures to current successful applications (SoA)



Face recognition (Facebook)



Go winner
(DeepMind
- Big G)



march 2016



Self-driving cars



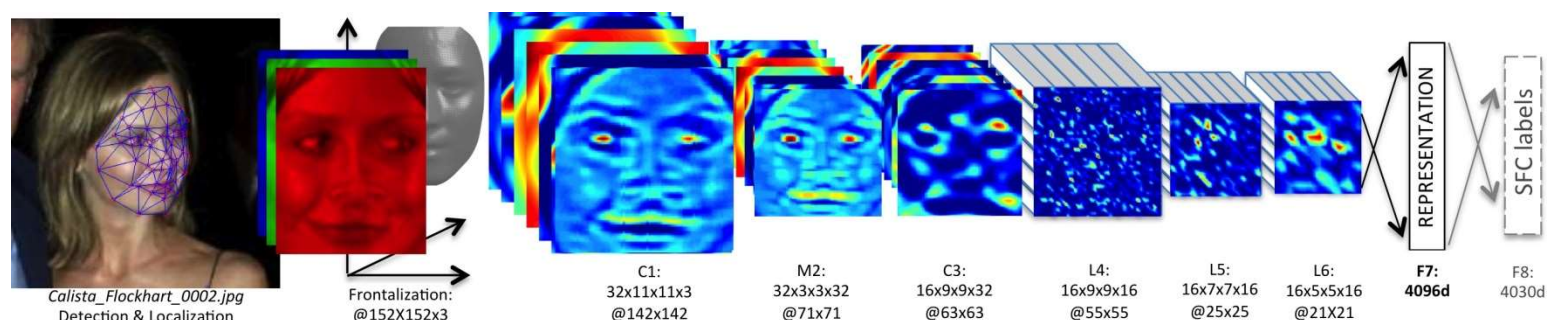
Sensor data
Smart *
IoT

An instance on a “recent” result (CVPR, 2014)



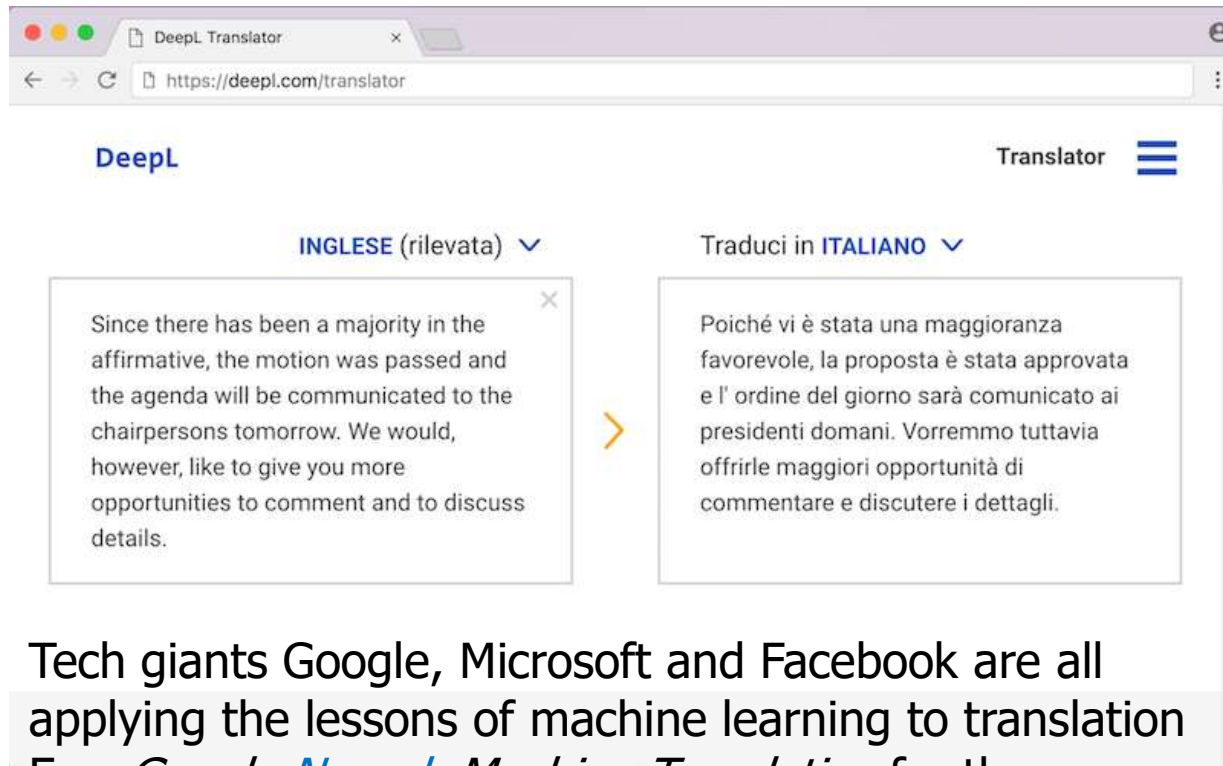
Dip. Informatica
University of Pisa


- **Face recognition** combining (deep) **Neural Networks** and other ML approaches
- Starting from four million facial images belonging to more than 4,000 identities



- Asked whether two photos show the same person, DeepFace answers correctly 97.25% of the time ... just a shade behind humans (97.53%).
- From a news on newspaper LaRepubblica April14 ;-)

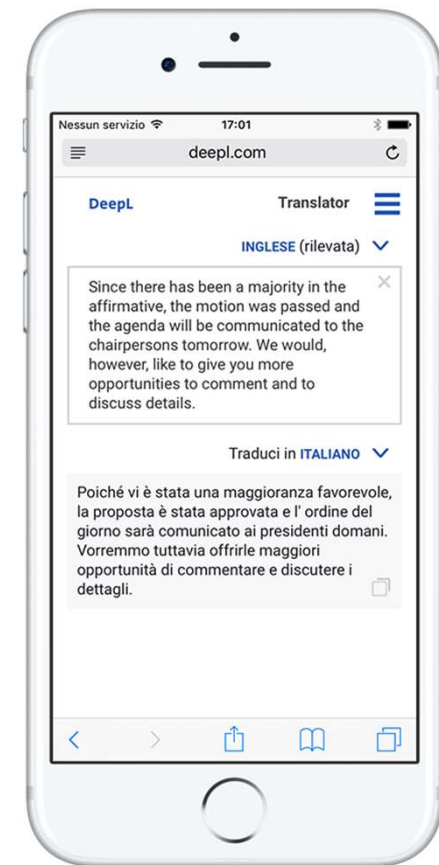
(Automatic) Machine Translation



Tech giants Google, Microsoft and Facebook are all applying the lessons of machine learning to translation
E.g. *Google Neural Machine Translation* for the <Google Translate> tool since 2016, 
but also small companies are making big progress:
An Example by <DeepL>

August 2017

A fully *Neural Network* based system.



Ultimate Aim

- Anything that is powerful may be dangerous (by misuses) but the ultimate aim of AI/ML is
 - To bring benefits to people by solving big and small problems
 - To accelerate human progress
 - To add intelligence for any other science field
- AI/ML to *augment our abilities enhancing our humanness in unprecedented ways* (P. Domingo, Scientific American sept. 2018)

HELLO
BENEFITS

Turing award 2018

- On 27-th March 2019, the Association for Computing Machinery, the world's largest society of computing professionals, announced that Drs. Hinton, LeCun and Bengio had won this year's Turing Award for their work on **neural networks**. The Turing Award, which was introduced in 1966, is often called the *Nobel Prize of computing*, and it includes a \$1 million prize, which the three scientists will share.
- "For conceptual and engineering breakthroughs that have made deep neural networks a critical component of computing"



Y. Bengio



G. Hinton



Y. LeCun

News ML/AI?

For upgrade, follow any media channel, e.g.

- Industrial:
 - Google AI: <https://ai.google/>
- Academic:
 - MIT: <http://news.mit.edu/topic/machine-learning>
 - Stanford: <https://news.stanford.edu/topic/artificial-intelligence/>
 - IEEE: <https://spectrum.ieee.org/robotics/artificial-intelligence>
- ...
- **Our Blog** *(for general and Unipi activity for AI):*
<http://ai.di.unipi.it/>

Why study ML?



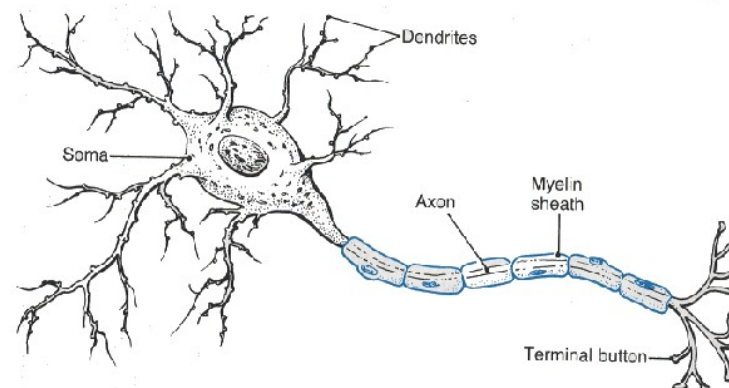
Machine learning in Master Degree



Dip. Informatica
University of Pisa

Why study *Machine Learning* ?

- To **know the basic principles** of learning processes (computational aspects)
- To **know new computing paradigms**
e.g. natural inspired models: **Neural Networks**
 - Studied as computing paradigms since the 40 `
 - Neurobiological inspiration
 - Nowadays: set of powerful computing models for *function approximation* and with predictive capabilities supported by a rigorous theoretical ground (*learning theory*)
 - Models for **Deep Learning**
- To be able to **apply** this model with a rigorous approach



Modern ML

- From a collection of heuristic methods (often collections of tricks originating from the contributing fields of biology, statistics, neural networks, fuzzy logics, ...) to the construction of general conceptual frameworks
- *Fundamental concepts and principles* that govern the learning processes: a new understanding has emerged:
 - Expression of different models in different “languages” **does not change the principles**
 - Different applications fields **do not change the principles**
 - New technological advancements **do not change the principles**
- The core of the *principles* and the *methods* to build flexible/powerful computational models from data is “*the Machine Learning*”

ML aims: points of view

- As AI methodology → [Build Adaptive/Intelligent Systems](#)
 - As statistical learning (inference of hypothesis within math. principles)
→ [Build powerful predictive system for Intelligent Data Analysis](#)
 - As computer science method for innovative application areas
→ [Using models as a tool for complex \(interdisciplinary\) problems](#)
-
- ❖ All these points of view will be considered in our course,
 - ❖ following a critical approach to discuss the different ML methods with evaluation of their *limitations* and appropriateness of use!
 - ❖ Attention to the principles underlying the methodologies.

Practical aims

Two concrete use of the course:

- Learn how to *develop new models* in the ML field
- Learn to *apply* current state-of-the-art methodologies for problems in other (interdisciplinary) fields



Great effort: why?

- Many topics and continuously evolving field: we have to cover them
- ML course to provide solid ground for Data Science and modern AI (and all the following courses)
- For your professional activity, it is expected from **you** to be **the ML expert** !!!
 - Keep in mind (to motivate your effort)! But also:
 - Interesting: *ML: the driving force of AI* (N. Cesa-Bianchi 2019).
 - A lot of fun: working with learning machine is intriguing



Intrerlude: ML in the curricula

Since 2017 organization by curricula of the CS master programme

Interlude of this lecture:

- Welcome to the new students in CS
- The curricula
- ML in the curricula
- Other master degree students
- AA1 and ML: from 6 to 9 credits or from 9 to 6 credits

In class: statistics*

How to study this course



Prerequisites

- No other course is strictly needed (we start from scratch)
- Best: IIA (with 3 credits introduction to ML): **WHO?**
- General prerequisites (typically from a First Cycle «Laurea» Degree Programme in CS/Math/...):
 - mathematical analysis (functions, differential calculus)
 - elements of matrix notation and calculus
 - elements of probability and statistics
 - algorithms

See few slides at the end of this lecture for notation references!

Objectives



Dip. Informatica
University of Pisa

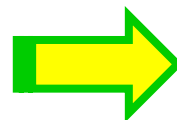
- We introduce the principles and the critical analysis of the main paradigms (models and algorithms, with a focus on neural networks) for learning from data.
- Methods:
 - The concepts are progressively introduced *starting from simpler approaches up to the state-of-the-art models* in the general conceptual *framework* of modern machine learning (learning of functions from examples).

The course focuses on the

- critical analysis of the characteristics for the design and use of ML
- rigorous experimental evaluation (**use of ML tools *versus* correct/good use of ML**)
- computational aspects of learning systems (CS!)

Course structure - preview

- There is a strong structure in the course:
 - *starting from simpler approaches up to the state-of-the-art models*
 - *the fundamental concepts will be introduced by the means of different models*
 - If you follow the *file rouge* (if you recognize that the main concepts do not change) you gain an easier approach to study the ML course!!!
Practical: 1) Study the singular methods
2) Consider the connections/relationships/comparison among them
- We will use different “language” building in parallel different approaches to learn (to infer hypothesis) from data
 - From simpler to more flexible models, along with the support of theoretical results
 - Toward the understanding of the relationship between *flexibility* and *complexity control* as a fundamental concept for the model generalization capability
- A graphical overview will help us to collocate the different lectures in the course structure
 - where we are, done, where we are going to, and why



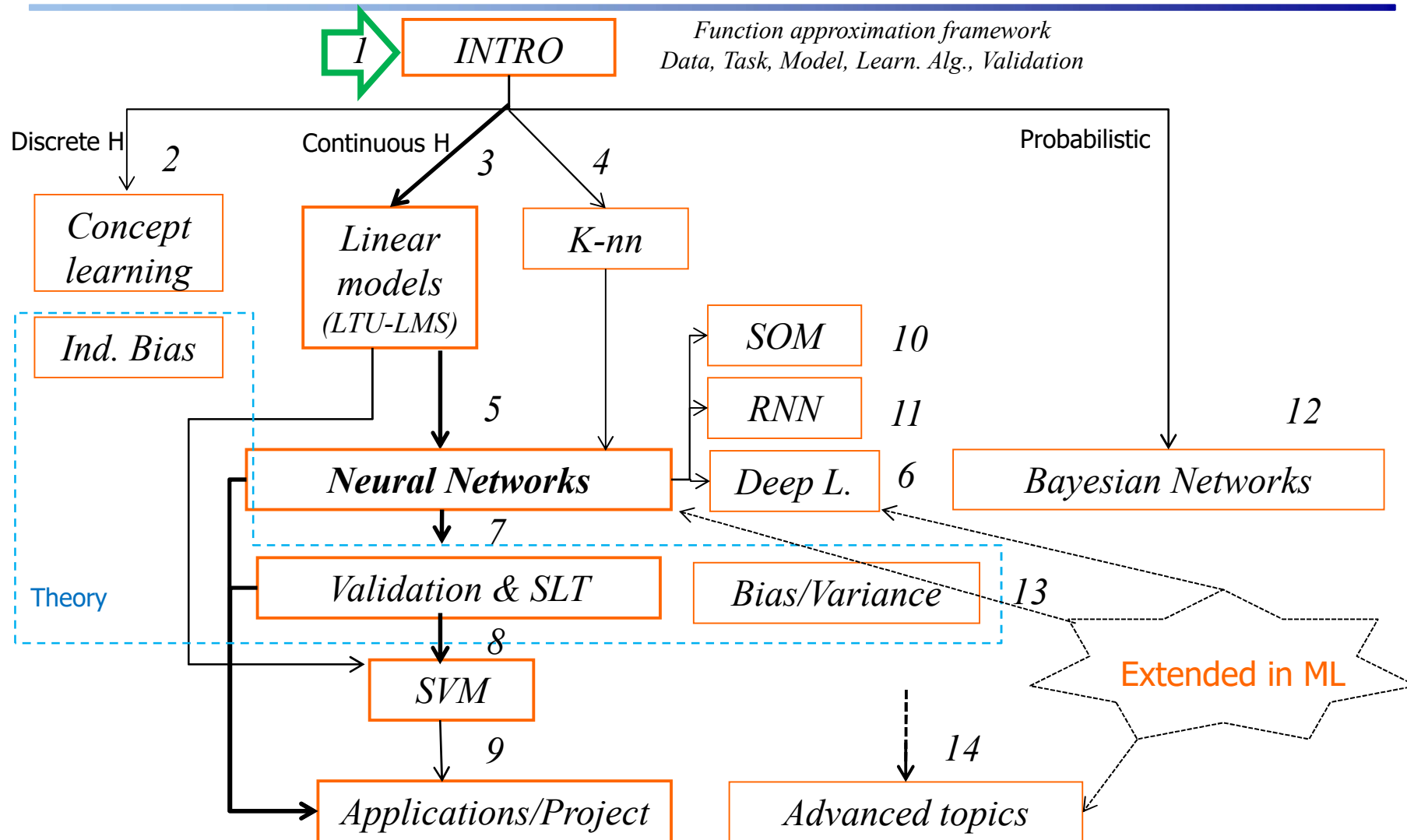
(speech)

ML Course structure

- preview



Dip. Informatica
University of Pisa



Again emphasis on the course structure



Dip. Informatica
University of Pisa

- This is NOT just a diagram on the order of the lectures
- It describes our path and the approach: we introduce some «bricks» and then will use the bricks to build ML models (like NN from linear models or a ML systems from validation more in general, etc.).
- In this way things will be simple at the end (by reusing what we did before)
- So keep in mind this: the intermediate models/concepts are often a piece of a puzzle/building that we will compose later: you have to look back at the end
 - Always follow the structure and please ask to clarify if needed



Course contents (skip now) (Indicative synthetic programme)



Dip. Informatica
University of Pisa

WARNING: See the final program provided at the end of the course

- **Introduction:** Computational learning tasks, prediction, generalization.
- **Basic concepts and models:** structure of the hypothesis space, discrete and continuous spaces, linear models (models and properties), nearest neighbor (models and properties), propositional and rule based models, inductive bias.
- **Neural Networks (NN):** Perceptron and computational properties. Introduction to multilayer feedforward NN: architectures and learning algorithms*. Regularization. Recurrent NN*. *Intro to recent NN paradigms**: Deep Learning, Randomized NN.
- **Principles of learning processes and general practical aspects:** Validation: model selection and model assessment, Bias-Variance analysis. Elements of Statistical Learning Theory, VC-dimension.
- **Support Vector Machines:** linear case, kernel-based models.
- **Bayesian and Probabilistic/Graphical models.**
- **Unsupervised learning:** vector quantization, self-organizing map.
- **Introduction to Benchmarks and Applications. ***
- **Advanced approaches:** Learning for Structured data (intro)*

*** These parts have been extended for the ML (9 CFU) course**

Assessment methods



Dip. Informatica
University of Pisa

Exam:

- **Project** (Written exam)
 - Students have the **opportunity** to develop a project realizing/applying a learning system simulator (typically a simple neural network) and to validate it through benchmarks. A written report will show the results.
 - Great *opportunity* to apply the concepts by yourself
 - Great *opportunity* to show your concrete understanding and effort for the exam
 - **Deadline:** ~(around) 10 days before the oral exam session (see the Moodle folder of your session for the exact deadline)
 - **See details in the lecture for project presentation** →
 - Last years: **competition** with *blind-test*
 - which is part of the benchmark results in the prj
 - also some joint proposals with CM course
- **Oral exam** (dates according to the exams sessions)



Exam (II): Some details

- **Project:** we will discuss in a specific lecture all the details but from now it useful to know that:
 - It is made by a couple of students (rare exceptions will be described later and MUST be justified and authorized in advance)
 - It is made just one time, i.e. with 1 delivery
 - It is corrected/discussed jointly with the oral exam (at the exam session)
 - It is an implementation of a NN or an application of ML SW (type A or B)
 - It includes code (type A) + results (also for the competition) + report (type A and B)
 - Type A: Programming language is free (we will discuss later)
- **Oral:** Prj discussion + questions on *all the course content*
 - Including written questions for all at the date of the official exam session
 - And then we distribute the dates of the prj discussion and oral refinement in the following days (for each group)
 - Style: I'll ask answer **first** by math language (e.g. in the form of *equations* due to clarity, synthesis, ...; **then** we can discuss on them)

Exam: schematic synthesis

In the following order:

- 1. Course lectures** (stay on-line with them!)
- 2. Project work** (you can start around in the middle of the semester)
- 3. Project package delivery:** results, report etc. at the date specified in the Moodle (around 10 days before the official exam session)
- 4. Oral:** at the date of the official exam session (for the questions in written form) (it is in the same session of the prj delivery).
- 5. Project discussion and finalization** according to a specific calendar for each group, i.e. as soon as possible in one of the following days (both the partner students must be present).

The main hints

- Follow the lectures and slides as a guide, studying *progressively* during the course
 - Special interactive classes will be used to assess activities and make a ML discussion forum (Q&A classes).



- A major hint *from past students*:
 1. **FIRST** study the course content
 2. **THEN** apply for the project

- For the PRJ: Develop (implement) a self made NN simulator if you are self-motivated and with good programming skills (type A), else apply existing tools (type B)
 - There is no difference in the grade, but a different effort for implementing (type A), or to make experiments and comparisons (type B)
 - Neither A nor B: ask to me (very very rare case, e.g. PhD students)

Good news

- This year we will have a ML course assistant
- to help for the project development
- Take the opportunity to interact with him during the course



Other info



Dip. Informatica
University of Pisa

- **Lectures schedule:**

- Tuesday(it: Mar) 16-18 Aula C1,
- Thursday (it: Gio) 16-18 Aula C1
- Friday (it: Ven) 16-18 Aula C1

SPECIAL autumn 2020:
We will have telematic lectures
We use the Teams platform

- **Q&A (with the teacher):** Tuesday **after lecture** *+ special Q&A classes.

- Or write to me an email for very personal questions.
- NOTE: the time after the lecture IS NOT mandatory, it is not part of the lecture, it is for singular/collective student Q&A reception

- **Make up for lost lectures:**

- Monday afternoon (they will be announced later, on the need)

- **ML mailing list:** I'll use Moodle news

- (take care to have on it the email address that you really use).

Introductory reading on ML



Dip. Informatica
University of Pisa

- Introductory, short, enjoyable reading on ML and modern AI are in the section "***What is Machine Learning***" of the site:
- <http://www.di.unipi.it/~micheli/DID/>
- E.g. For the paper di T. Poggio, S. Smale see just the first 2 pages.

Textbook and material (1)

- **Course notes (slides copy)**

with material from time to time indicated in the classroom:
bibliographic references at the end of each specific topic via
books chapters and/or online material. Hints:

1. *Course notes are a very useful guide to the selected topics!*
2. *Have a look in advance (I typically provide them in advance in the Moodle)*
3. *Take your notes [past student suggestion]: make connections among slide phrases, and slide themselves can be not sufficient to reconstruct the lecture line*
4. *Learn to take advantages reading different sources (**master degree level**)*

- **General Bibliography** (library or on-line):

- S. Haykin: ***Neural Networks: a comprehensive foundation***, IEEE Press; (2nd. Edition, 1998) → OR better

- S. Haykin: ***Neural Networks and Learning Machines***, Prentice Hall; (3rd Edition, 2008)

- T. M. Mitchell: ***Machine learning***, McGraw-Hill, 1997

- I. Goodfellow, Y. Bengio, A. Courville: ***Deep Learning***, MIT Press, 2016

On-line: <http://www.deeplearningbook.org/>



How to get the slides?

The electronic version of the slides is intended only for authorized student, **for their personal use.**

- The link to the electronic slides is reserved (draft material). **It is not a public site!**



- **Material: Moodle:** <https://elearning.di.unipi.it/>
 - See **Machine Learning 2020**
 - See also the **FaQ** section therein
- Please do not use the link in any web site/social media
- Please do not repost slides in any form
- Take care of last version (sometime after the lecture!!!)

How to get the slides? (II)

- If you not have a UNIPI student account yet:
- Please ask to me by email for a TEMPORARY guest account

A large, red, rectangular stamp with a distressed, ink-like texture. The word "TEMPORARY" is written in bold, white, uppercase letters across the center of the stamp.

Textbook and material (2)

- **Other references:**

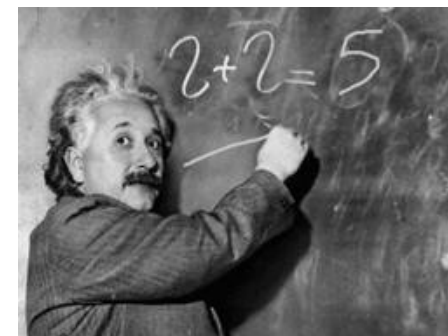
- (AIMA) Russell, Norvig: *Artificial Intelligence: A Modern Approach* Prentice Hall (3 edition, 2009)
- Hastie, Tibshirani, Friedman: ***The Elements of Statistical Learning***, Springer Verlag, 2001 (new eds. up to 2017) → 2017 [Free on-line copy](#)
- Cherkassky, Mulier: *Learning from data: concepts, theory, and methods*, Wiley, 1998 (new: 2nd Ed., 2007)
- C.M. Bishop: *Pattern Recognition and Machine Learning*, Springer 2006
- Bishop: *Neural Networks for Pattern Recognition*, 1995
- Duda, Hart, Stork: *Pattern Classification*, 2nd. ed. J. Wiley & Sons, 2001
- Shalev-Shwartz, Ben-David: ***Understanding Machine Learning: From Theory to Algorithms***, Cambridge University Press, 2014, [Free online copy](#)

Background



Background 1

- Very informal notation resume
- E.g., AIMA appendix A.2 and A.3 (mathematical background) free available at <http://aima.cs.berkeley.edu/newchapa.pdf>
- Or Deep Learning book chapters I.2, I.3, I.4 (i.e. part I: 2,3,4) <http://www.deeplearningbook.org/>
- And feel free to ask for any doubt!!!!
- Don't worry: IA/ML are multidisciplinary fields from the beginning!!!!



Background 2

NEW since 2017:

- In parallel we have the course of CM (Computational mathematics for learning and data analysis)
- The first two/three weeks are used to cover the mathematical background for learning. You can consider such lectures :
 - Time table: please, check by yourself

Notation

Generally, we will use consistent notation among the most important parts. However:

- We must **abstract** from /with respect to the notation
 - Anyway, the equivalent symbols for the same concept will be indicated
- Sometime **locally defined** notation will be used to *simplify* the concepts introduction and/or to be consistent with *different books*
- In any case, don't go crazy for notation issues: please ask to me if you have some issues, I will be happy to help (better immediately than later) and to correct if needed

Basic background (references)

- Multivariable/Multivariate calculus
Functions with multiple inputs: $z=f(x,y)$ or $f(\mathbf{x})$, where $\mathbf{x}=(x_1,x_2,\dots)$
 - *Partial derivative, gradient.*
- Probability:
 - *probability density function, mean and variance, Normal random variable*
 - *$p(x)$ or $P(X=x)$, conditional probability: $p(x|y)$, joint probabilities, ...*
- Matrix calculations and notations: $x, \mathbf{x}, \mathbf{X}$
 - *inner (dot, scalar) product, inverse, norms, ...*
- *Notions that can be studied in parallel(e.g. MNO - Numerical Methods and Optimization and the NEW **CM** course):*
 - *The linear least-squares problem: use of the Singular Value Decomposition*
 - *Optimization: Lagrange multipliers*

Inner (dot, scalar) product

$$\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n = \sum_{i=1}^n a_i b_i \quad \mathbf{a} \cdot \mathbf{b} = |\mathbf{a}| |\mathbf{b}| \cos \theta$$

- Other notations $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T \mathbf{b} = \mathbf{a}^t \mathbf{b} = \langle \mathbf{a}, \mathbf{b} \rangle$ and even $\mathbf{a} \mathbf{b}$ if the context is clear
- Magnitude/size (it '*Modulo*') or length or *Euclidean norm* of a vector \mathbf{x}

$$\sqrt{\mathbf{x}^T \mathbf{x}} = \sqrt{\mathbf{x}^T \cdot \mathbf{x}} = \sqrt{\sum_i x_i^2} = d(\mathbf{x}, 0) = \|\mathbf{x}\|_2 = \|\mathbf{x}\| = |\mathbf{x}|$$

Simplified

- Generalization: function that relate a couple of vectors to a number (a scalar):
(bilinear symmetric form, by the notation \langle, \rangle)

$$\langle v, w \rangle = \langle w, v \rangle$$

$$\langle v + w, u \rangle = \langle v, u \rangle + \langle w, u \rangle$$

$$\langle kv, w \rangle = k \langle v, w \rangle$$

- Cauchy-Schwarz inequality $|\langle x, y \rangle| \leq \|x\| \cdot \|y\| \quad \forall x, y \in V$

Tensors & other defs.

Tensor

- Simple view: an array of numbers arranged on a regular grid with a variable number of axes is known as a tensor.
- i.e. A **multi-dimensional array** of numerical values.
- E.g. $X_{i,j,k}$ (3 indices)
- Used trivially in ML just for notation advantage, but in general with more properties in math useful dealing with change of basis, basis independence etc. in linear algebra, geometry, physics.....
- In ML help focus when using GPU etc.

Other defs.

$$\|\mathbf{x}\|_1 = \sum_i |x_i| \quad L^1 \text{ norm}$$

$$\|\mathbf{x}\|_\infty = \max_i |x_i| \quad \text{Max norm}$$

Partial derivative: calculus

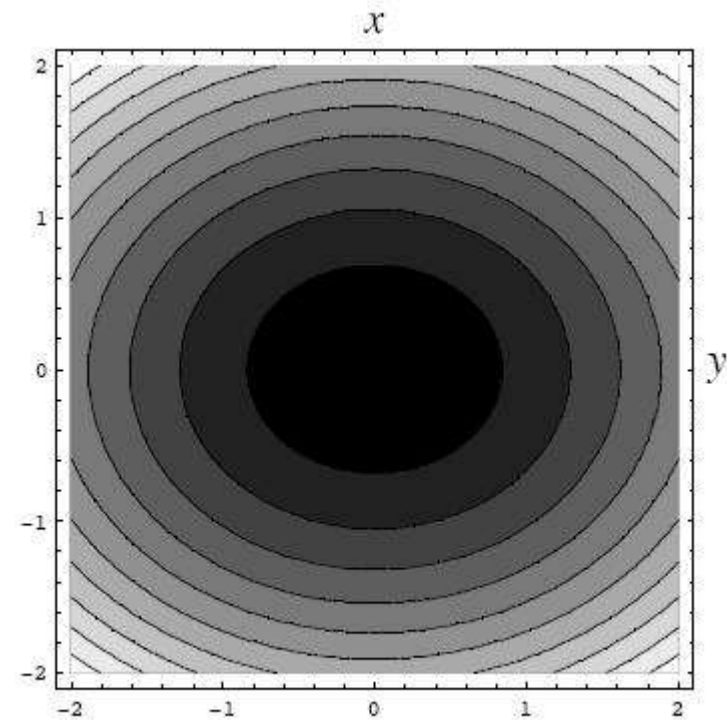
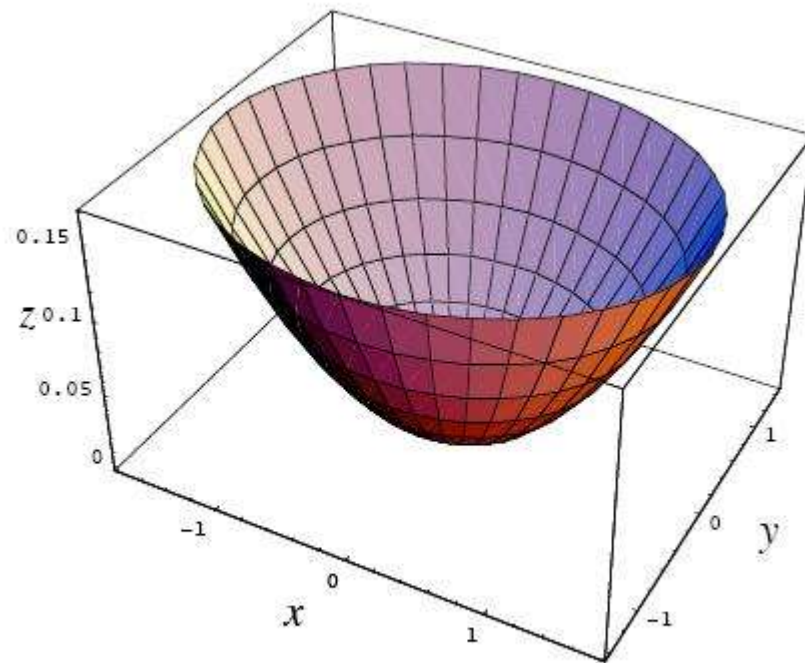
- The **partial derivative** generalizes the notion of the derivative to higher dimensions. A partial derivative of a multivariable function is a derivative with respect to one variable (e.g. x) with all other variables held constant.
- Hence, if $Df(x) = f'(x) = df/dx$
- For $f(x,y,z)$ we can compute: $df/dx, df/dy, df/dz$

$$\frac{\partial f}{\partial x} \quad \frac{\partial f}{\partial y} \quad \frac{\partial f}{\partial z}$$

Gradient

- When a function of two variables $f(x, y)$ have partial derivatives at each point (x, y) we can associate the vector of the two partial derivatives $\left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$ called the **gradient** of f , often denoted ∇f or *grad f*.
- *The gradient is the vector field whose components are the partial derivatives of f :*
$$\text{grad } f = \mathbf{e}_1 \frac{\partial f(x_1, \dots, x_n)}{\partial x_1} + \dots + \mathbf{e}_n \frac{\partial f(x_1, \dots, x_n)}{\partial x_n}$$
where the \mathbf{e}_i are the orthogonal unit vectors pointing in the coordinate directions (e.g. 001, 010, 100 in 3D).
- **The gradient at a point is a vector pointing in the direction of the steepest slope at that point.**
- The steepness of the slope at that point is given by the magnitude of the gradient vector.

Local Minimum

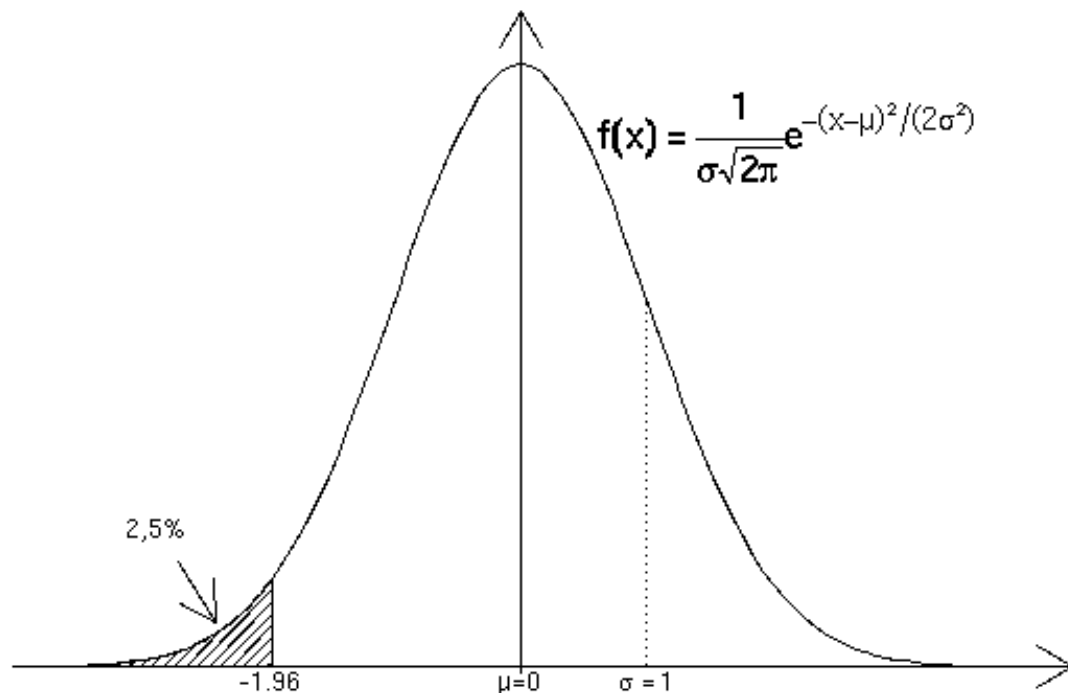


- Example of a stationary point: the gradient is null
(while are also stationary saddle points or local max)

Density (example)

- The density function of the normal random variable with mean 0 and variance 1 (called standard normal distribution, see the figure) and the analytical expression of the corresponding density in the generic case (*mean μ and variance σ^2*).

The probability density function (pdf) for a **normal distribution**: the “Gaussian function”



Next steps

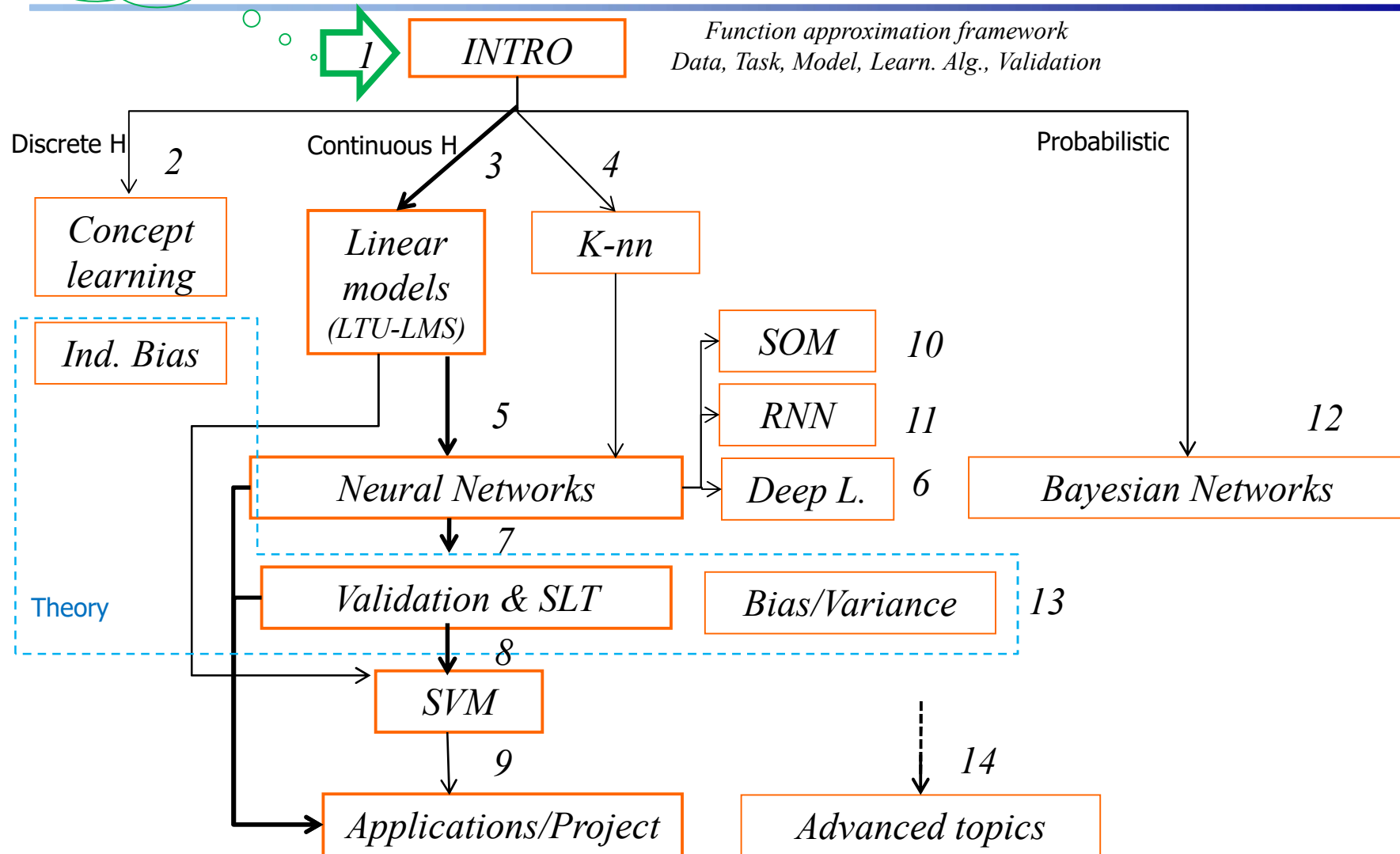


AA1 Course structure - preview



Dip. Informatica
University of Pisa

We are here



Summary of the Intro to ML

- Part I (now)
 - Motivations, contextualization in CS
 - Course info
- Part II (in Lect. 2)
 - Utility of ML
 - Learning as function approximation (pilot example)
 - Design components of a ML system, including
 - Learning tasks
 - Hypothesis space (and first overview)
 - Loss and learning tasks
 - Generalization (first part)
- Part III (in Lect.3 ???)
 - Generalization and Validation

Aim: *overview and terminology
before starting to study models and learning algorithms*

AA1 (320AA) \leftrightarrow ML

INFO for AA1 (old exam) \leftrightarrow ***ML students***

In summary:

- From AA1 to ML (you need 9 credits): I'll provide a list of topics for integrative exam, see also the symbol for the new parts:
- From ML to AA1 (you need 6 credits): **contact me as soon as possible. Anyway, I'll provide a program for the 6 credits exam.**
- Note: AA1 does not exist now, it is only for past students



PROLOGUE END



Alessio Micheli
micheli@di.unipi.it



**Computational Intelligence & Machine
Learning Group**



Dipartimento di Informatica
Università di Pisa - Italy

A Short Introduction to Machine Learning

Lect 2

Alessio Micheli

micheli@di.unipi.it



Dipartimento di Informatica
Università di Pisa - Italy



**Computational Intelligence &
Machine Learning Group**

2019

Introduction to ML: plan of the next lectures



Dip. Informatica
University of Pisa

- Introduction aims:
 - Critical contextualization of the ML in comp. science [lect 1 and 2]
 - **Overview and Terminologies** [lect 2, 3]
 - the relevant concepts will be developed later in the course (therefore Lect 2 is a bit longer than usual !)
 - First basic models and learning algorithms [lect 4, 5, 6]
 - Then, we will start with NN !

See the "Course structure" slide!

Learning

*The problem of **learning** is arguably at the very core of the problem of **intelligence**, both biological and artificial*

Poggio, Shelton, *AI Magazine* 1999

i.e. Learning as a major challenge and a strategic way to provide intelligence into the systems



Machine Learning (I)

We restrict to the *computational* framework:

- Principles, methods and algorithms for learning and prediction:
 - Learning by the system of the experience (known data) to approach a defined computational task
 - Build a model (or hypothesis) to be used for predictions
❖ (see examples on email-spam or face recognition)

Most common specific framework :

- Infer a model / *function* from a set of examples which allows the **generalization** (to provide accurate response on new data)

Machine Learning (II): When?

Opportunity (if useful) and *awareness* (needs and limits)

- Utility of predictive models: (in the following cases)
 - **no (or poor) theory** (or knowledge to explain the phenomenon)
 - **uncertain, noisy or incomplete data** (which hinder formalization of solutions)
- Requests:
 - source of training experience (representative data)
 - tolerance on the precision of results

Machine Learning (III): When?

- Models to solve real-world problems that are difficult to be treated with traditional techniques (*complementary* to analytical models based on previous knowledge, algorithms and imperative programming, classical AI, ...)
- Examples of appropriate applications versus standard programming:
 - Knowledge is too difficult (to be formalized by 'hand-made' algorithm)
 - e.g. face recognition: humans can do it but cannot describe how they do it
 - e.g. voice automatic telephone answering service
 - Not enough human knowledge
 - e.g., predicting binding strength of molecules to proteins)
 - Personalized behavior
 - scoring email messages or web pages according to user preferences)
 - individualized (intelligent) human-computer interfaces
- Due to this flexibility ML applicative area is very large: see lecture 1

General challenges



- Build autonomous Intelligent/learning systems:
 - Robotics, HRI, search engines, ...
- Build powerful tools for emerging challenges in intelligent data analysis
 - Tools for the “data scientist”
- Open new areas of applications in CS: innovative interdisciplinary open problems (more in general, “machine learning scientist”)
 - Fantasy is your limitation !
 - ML in the era of “changing of paradigm in science, in which scientific advances are becoming more and more *data-driven*”
 - *Growing data sources opens up a huge application area for ML and related areas (Web, Social Net., IoT, BioMed, ...)*

An useful framework:

Learning as an approximation of an unknown function from examples

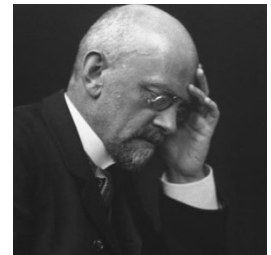
Specific vision but widespread in ML

For us:

- **Different tasks seen in uniform framework**
- **Enables a rigorous formulation**

=> Intro guided by intuitive examples

Please, note that the following example was already introduced in Lect 1



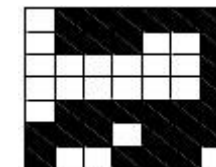
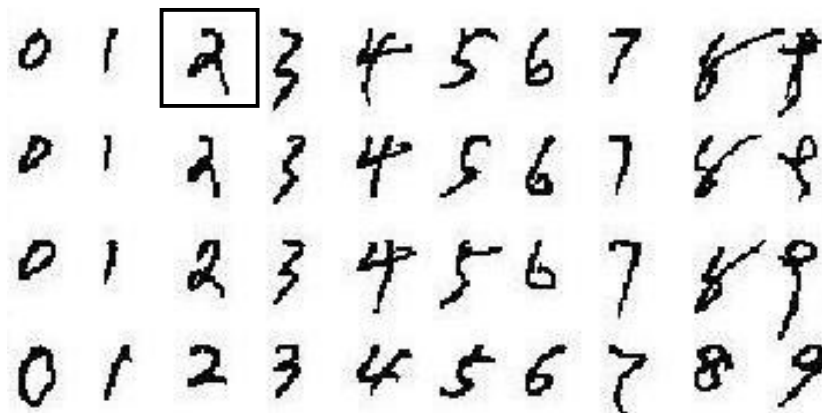
Hilbert spaces

An Example



Dip. Informatica
University of Pisa

- A pilot example: *recognition of handwritten digits*
- **Input:** collection of images of handwritten digits (arrays/matrix of values)
- **Problem:** build model that receives in input an image of handwritten digit and "predict" the digits



8 x 8

Build **a function** from examples



Dip. Informatica
University of Pisa

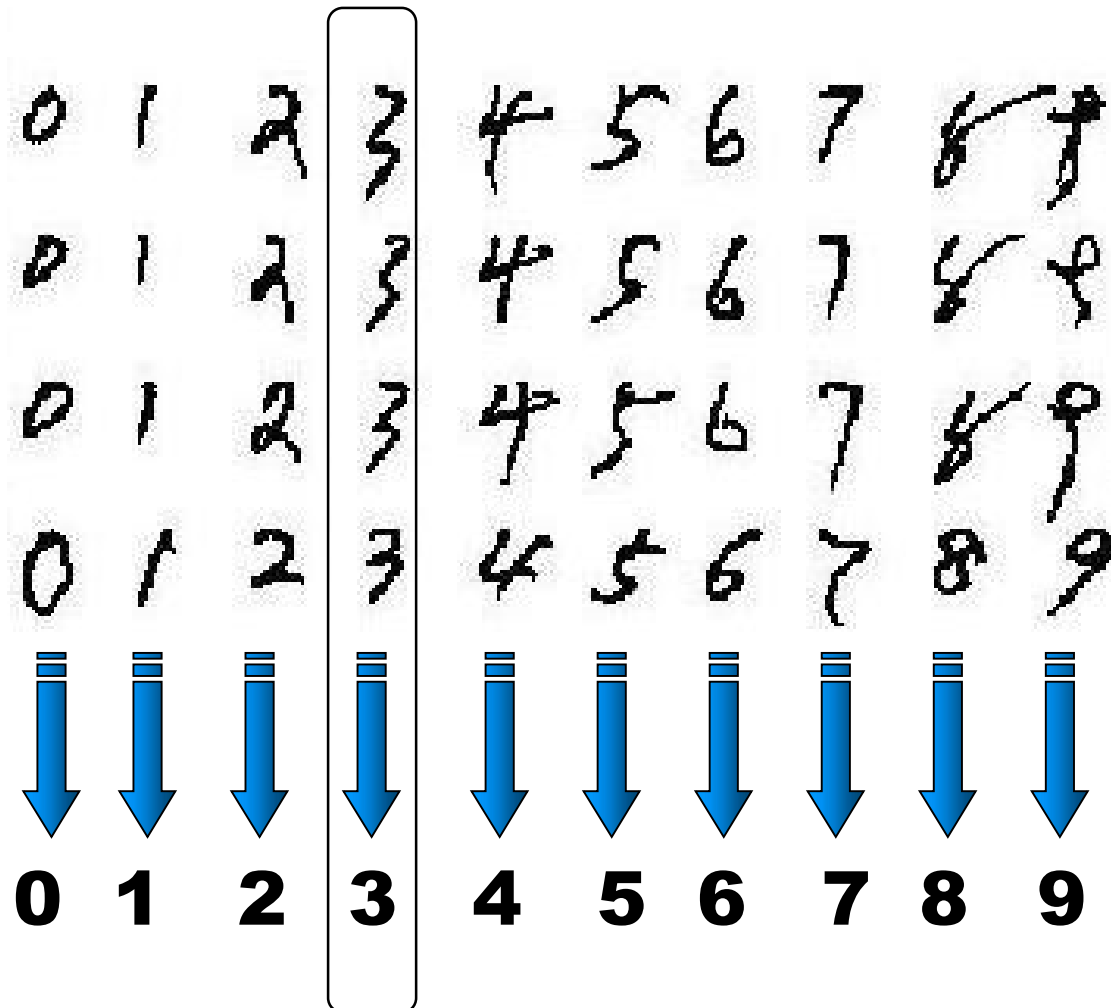
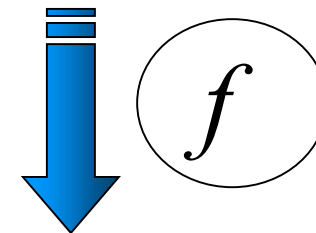


Image
8 x 8



Output class

Handwritten Digits Recognition



Dip. Informatica
University of Pisa

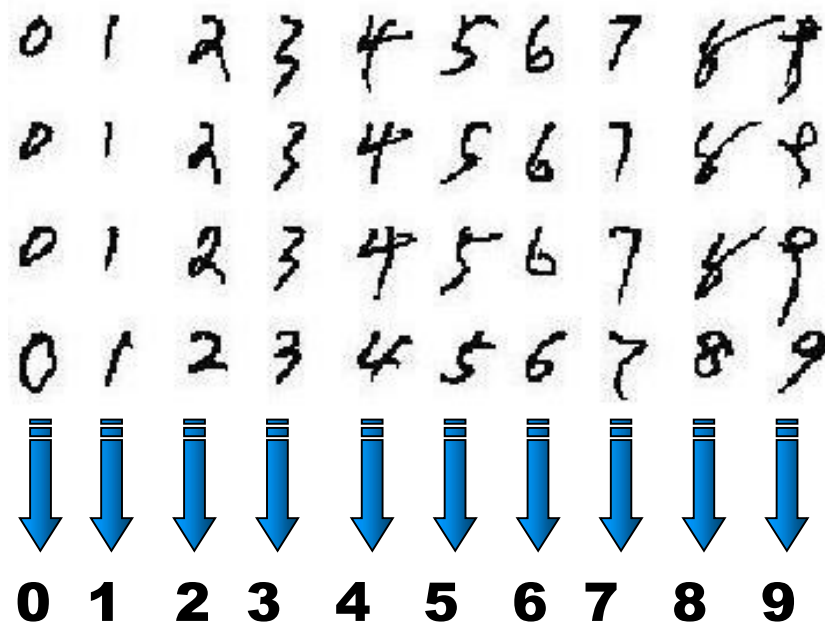
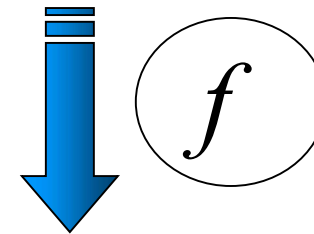


Image
8 x 8



Output class

Classification problem

- Difficult to **formalize** exactly the solution of the problem:
Possible presence of **noise** and **ambiguous** data;
- Relatively easy to collect collection of labeled examples

=> Example of successful application of the ML!

Machine Learning

A new extended definition (looking to the pilot example)

- The ML studies and proposes methods to build (infer) dependencies / **functions** / hypotheses from examples of observed data
 - that ***fits*** the know examples
 - able to ***generalize***, with reasonable accuracy for new data
 - According to verifiable results
 - Under statistical and computational conditions and criteria
 - Considering the expressiveness and algorithmic complexity of the models and learning algorithms

Examples of $x - f(x)$

Inferring general functions from know data:

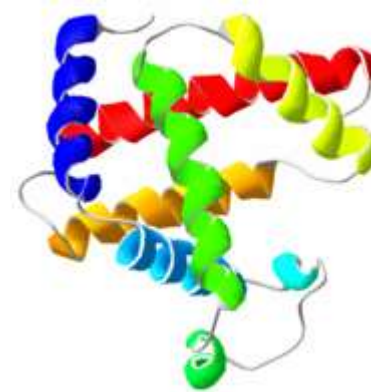
- Handwriting Recognition
 - x : Data from pen motion.
 - $f(x)$: Letter of the alphabet.
- Disease diagnosis (from database of past medical records)
 - x : Properties of patient (symptoms, lab tests)
 - $f(x)$: Disease (or maybe, recommended therapy)
 - $TR \langle x, f(x) \rangle$: database of past medical records
- Face recognition
 - x : Bitmap picture of person's face
 - $f(x)$: Name of the person.
- Spam Detection
 - x : Email message
 - $f(x)$: Spam or not spam.

Complex data

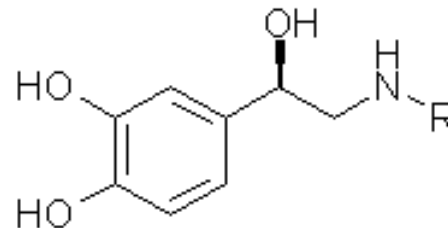


Dip. Informatica
University of Pisa

- Protein folding
 - x : sequence of amino acids
 - $f(x)$: sequence of atoms' 3D coordinates
 - TR $\langle x, f(x) \rangle$: known proteins
 - Type of x : string (variable length)
 - Type of $f(x)$: sequence of 3D vectors



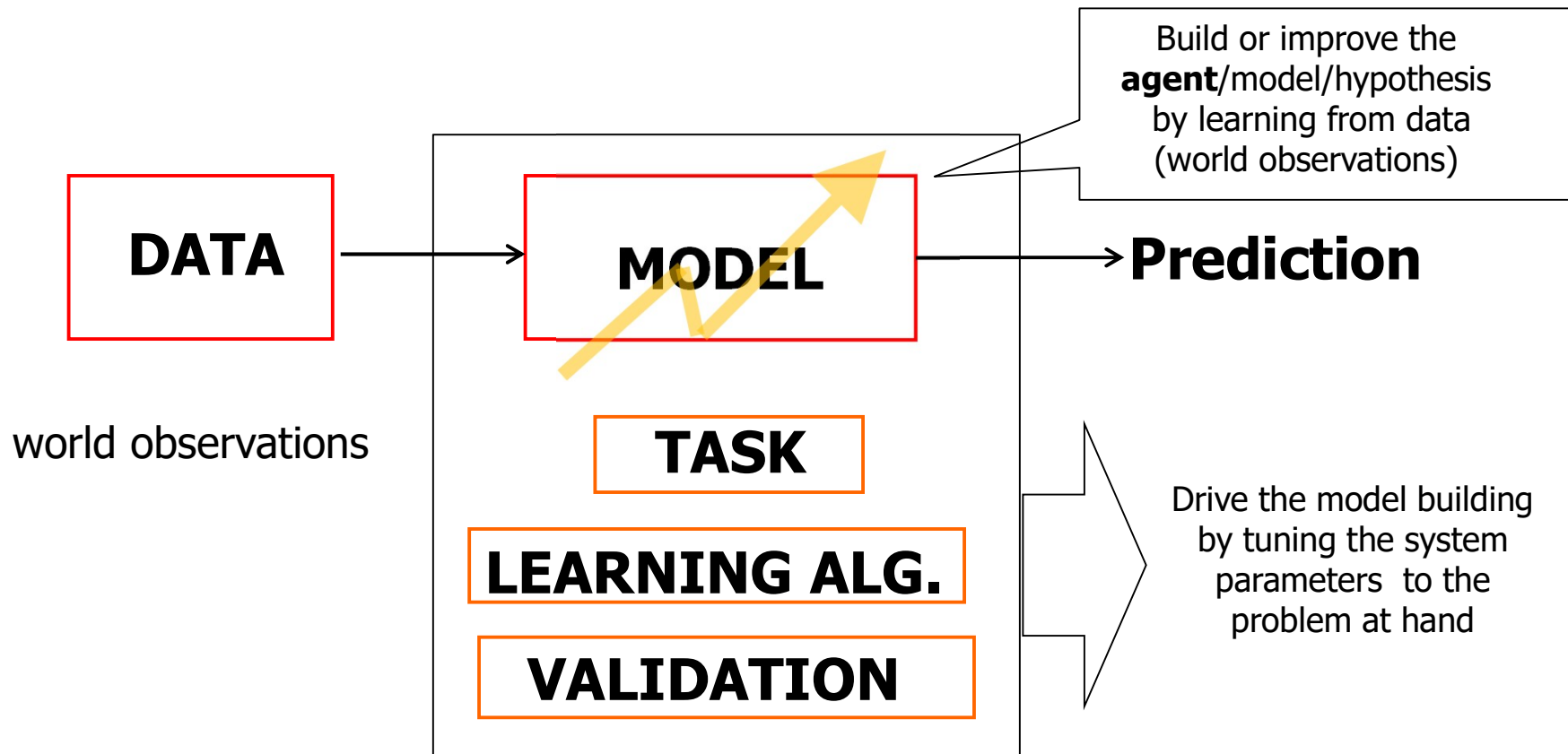
- Drug design
 - x : a molecule
 - $f(x)$: binding strength to HIV protease
 - TR $\langle x, f(x) \rangle$: molecules already tested
 - Type of x : a graph or a relational description of atoms/chemical bonds
 - Type of $f(x)$: a real number



Overview of a ML (predictive) System



Dip. Informatica
University of Pisa



Also as a guide to the **key design choices**
(ML system "**ingredients**")

DATA

- The data *represent* the available facts (*experience*).
 - Representation problem: to capture the structure of the analyzed objects

Types: Flat, Structured, ...

- **Flat** (*attribute-value language*):

fixed-size vectors of properties (*features*), single table of tuple (measurements of the objects)

Fruits	Weight	Cost \$	Color	Bio
Fruit 1 (lemon)	2.1	0.5	y	1
Fruit 2 (apple)	3.5	0.6	r	?

→ Attributes
(discrete/continuous)

- Categorical/continuous, missing data,...
- Preprocessing: e.g. Variable scaling, encoding*, selection...

DATA

Examples and terminologies



Dip. Informatica
University of Pisa

Medical records

	Age	Smoke	Sex	Lab Test
Pat 1	101	0.8	M	1
Pat 2	30	0.0	F	?

j

p

x_p

Attributes
(discrete/continuous)

- Each row (x , vector in bold): example, pattern, instance, sample,....
- Dimension of data set: number of examples l
- Dimension (of x): number of features n
- If we will index the features/inputs/variables by j : variable x_j is (typically) the j -th feature/property/attribute/element/component of x .
(but may be to simplify we need to use subscript index for other meanings)
- x_p or x_i is (typically) the p -th or i -th pattern/example/raw (vector)
- $x_{p,j}$ (for example) can be the attribute j of the pattern p

DATA Encoding

Flat case:

- Numerical encoding for categories: e.g.
 - 0/1 (or $-1/+1$) for 2 classes
 - 1,2,3... Warning: grade of similarity (1 vs 2 or 3): useful for “order categorical” variables (e.g small, medium, large)
 - *1-of-k* (or *1-hot*) encoding: useful for symbols

A	1	0	0
B	0	1	0
C	0	0	1

Useful both for input or output variables

**It will be useful
for the project !**

DATA : Structures

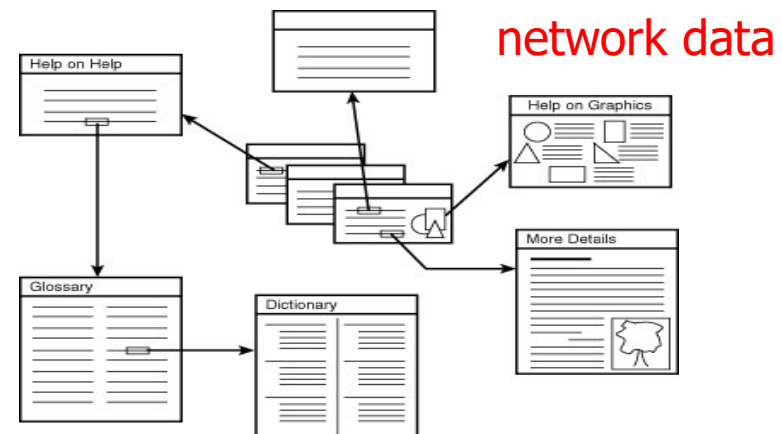
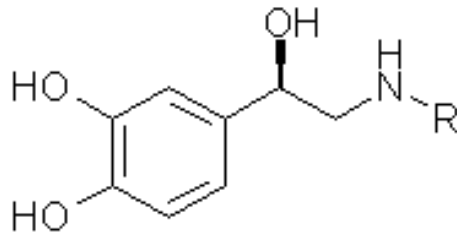
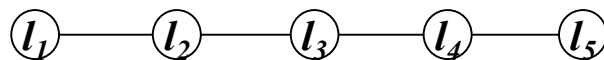


Dip. Informatica
University of Pisa

- **Structured:** Sequences (lists), trees, graphs, Multi-relational data (table) (in DB)

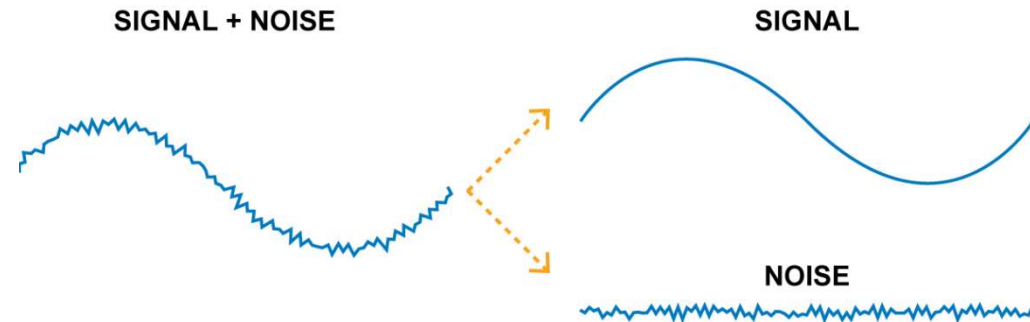
Examples: images, microarray, temporal data, strings of a language, DNA e proteins, hierarchical relationships, molecules, hyperlink connectivity in web pages, ...

Which natural representation?





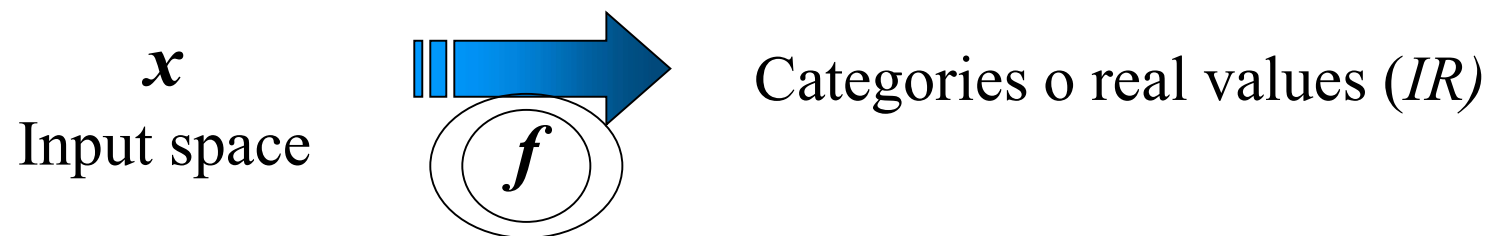
- **Noise:** addition of external factors to the stream of target information (*signal*); due to randomness in the measurements, not due to the underlying law: e.g. Gaussian noise



- **Outliers:** are unusual data values that are not consistent with most observations (e.g. due to abnormal measurements errors)
 - outlier detection – preprocessing: removal
 - Robust modeling methods
- **Feature selection:** selection of a small number of informative features: it can provide an optimal input representation for a learning problem

TASKS

- The task defines the purpose of the application:
 - Knowledge that we want to achieve? (e.g. pattern in DM or model in ML)
 - Which is the helpful nature of the result?
 - What information are available?
- **Predictive** (Classification, Regression): function approximation



E.g. recall the *"pilot" example on handwritten digits*: Build a *function* from examples

- **Descriptive** (Cluster Analysis, Association Rules): find subsets or groups of unclassified data

Tasks: Supervised Learning



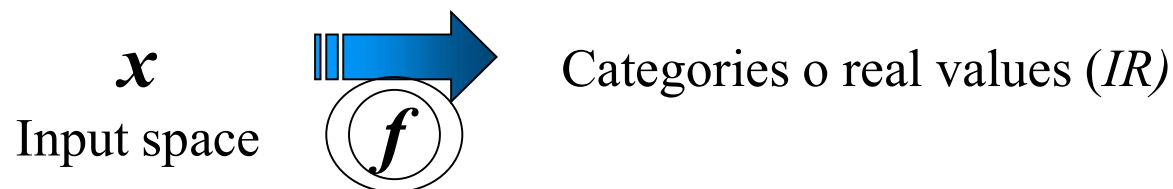
Dep. Informatica
University of Pisa

- **Given:** Training examples as $\langle input, output \rangle = \langle x, d \rangle$ (**labeled examples**)

Def

for an unknown function f (known only at the given points of example)

- Target value: desiderate value d or t or y ... is given by the teacher according to $f(x)$, to label the data
- **Find:** A *good* approximation to f (a hypothesis h that can be used for prediction on unseen data x')



- Target d (or t): a numerical/ categorical *label*
 - *Classification*: discrete value outputs:
 $f(x) \in \{1, 2, \dots, K\}$ *classes* (*discrete-valued function*)
 - *Regression*: real continuous output values (approximate a real-valued target function)

Unified vision thanks to the formalism of a
function approximation task

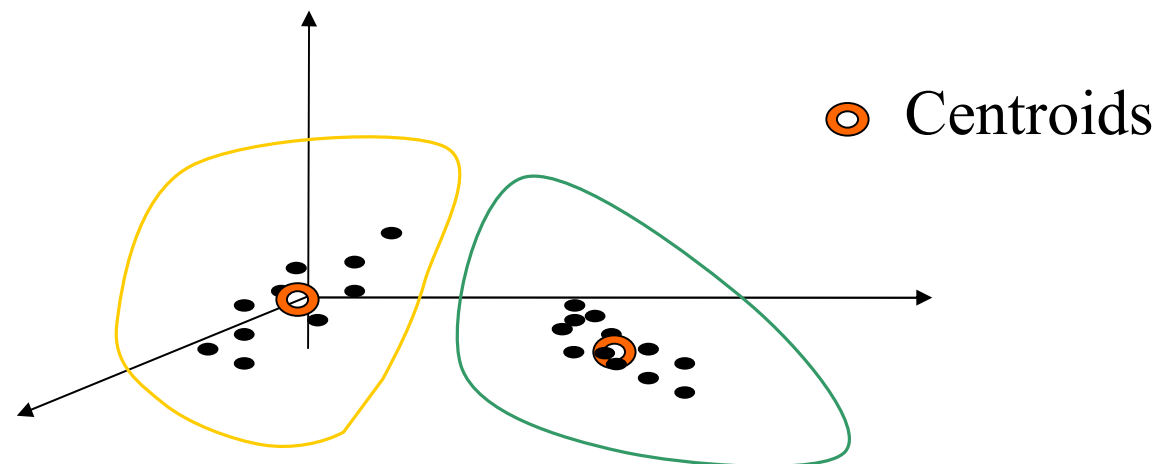
Tasks: Unsupervised Learning



Dip. Informatica
University of Pisa

Unsupervised Learning: No teacher!

- TR = set of unlabeled data $\langle x \rangle$
- E.g. to find *natural groupings* in a set of data
 - Clustering
 - Dimensionality reduction/ Visualization/Preprocessing
 - Modeling the data density



■ Clustering:

Partition of data into clusters (subsets of "similar" data)

Tasks: Classification

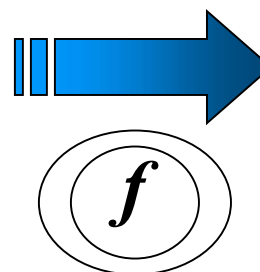
(Supervised) Classification: Patterns (features vectors) are seen as members of a class and the goal is to assign the patterns observed classes (label)

- *Classification*: $f(\mathbf{x})$ return the correct class for \mathbf{x}
- Number of classes:
 - **=2** : $f(\mathbf{x})$ is a Boolean function: binary classification, **concept learning** (T/F or 0/1 or $-1/+1$ or negative/positive),
 - **> 2**: multi-class problem ($C_1, C_2, C_3 \dots C_K$)

Example

From DATA to TASK (e.g. classification)

Patients	Age	Smoke	Sex	Lab Test
Pat 1	101	0.8	M	1
Pat 2	30	0.0	F	?



Target: diagnose
+
-

\mathbf{x} : Input space

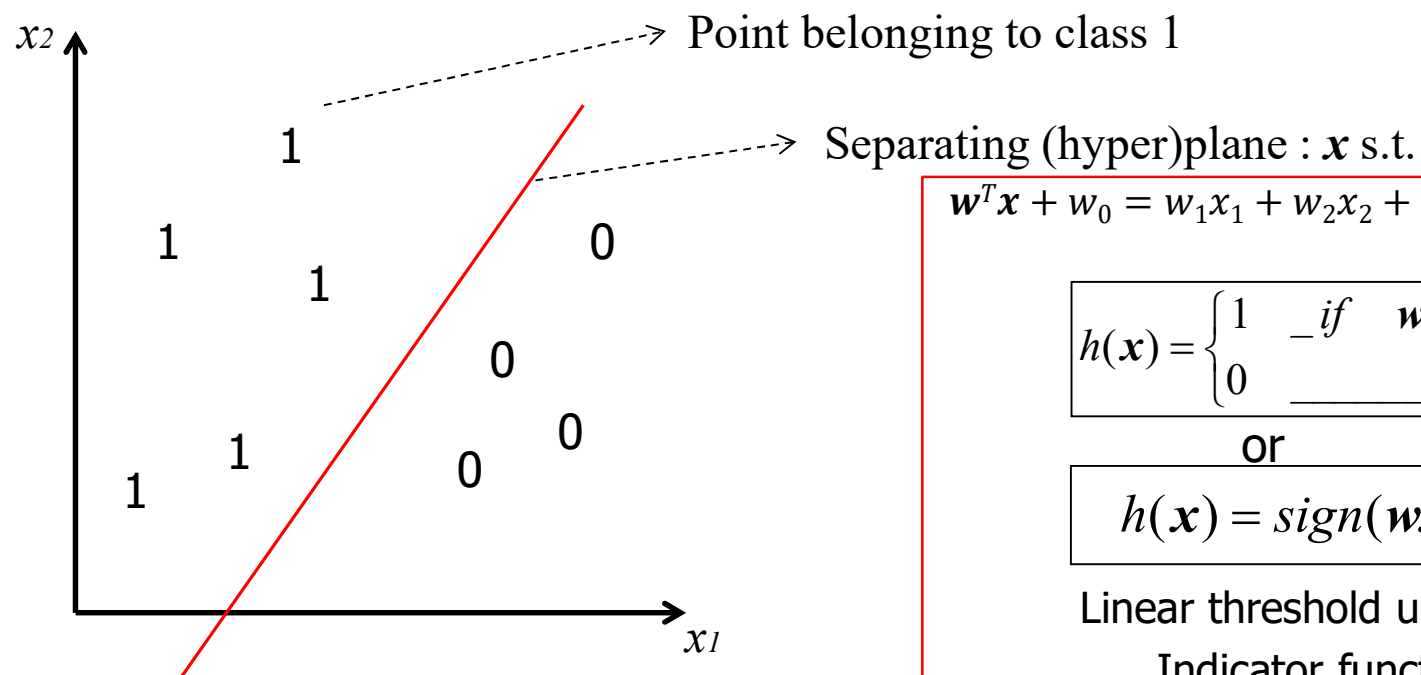
Terminology in statistics:

- Inputs are the “independent variables”
- Outputs are the “dependent variables” or “responses”

Tasks: Classification

The classification may be viewed as the allocation of the input space in decision regions (e.g. **0/1**)

Example: graphical illustration of a linear separator on a instance space $\mathbf{x}^T = (x_1, x_2)$ in \mathbb{R}^2 , $f(\mathbf{x}) = 0/1$ (or $-1/+1$)



PREVIEW

$$\mathbf{w}^T \mathbf{x} + w_0 = w_1 x_1 + w_2 x_2 + w_0 = 0$$

$$h(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w}^T \mathbf{x} + w_0 \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

or

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^T \mathbf{x} + w_0)$$

Linear threshold unit (LTU)

Indicator functions

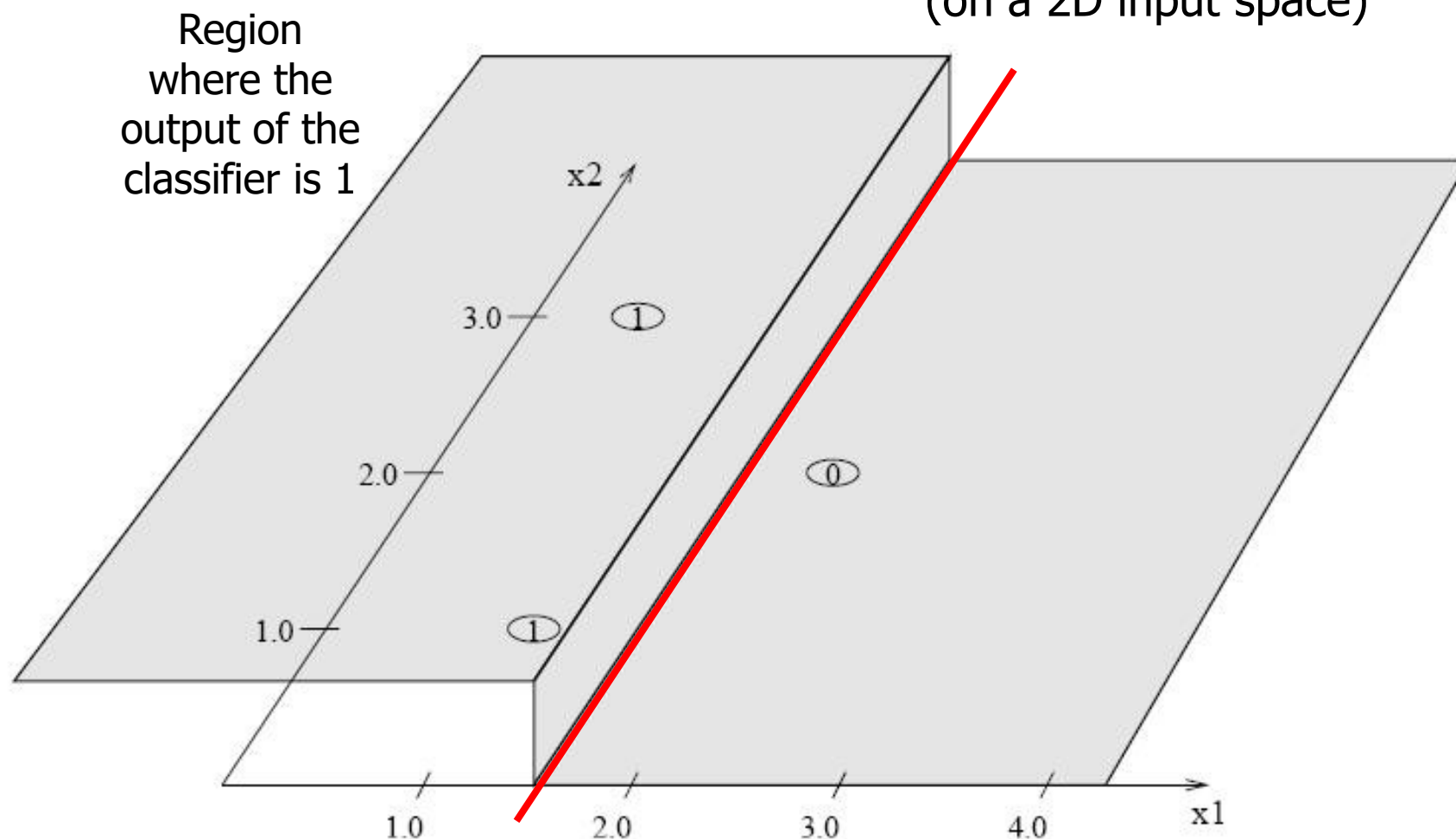


Geometrical 3D view: Classifier



Dip. Informatica
University of Pisa

The 0/1 classification function in 3D
(on a 2D input space)

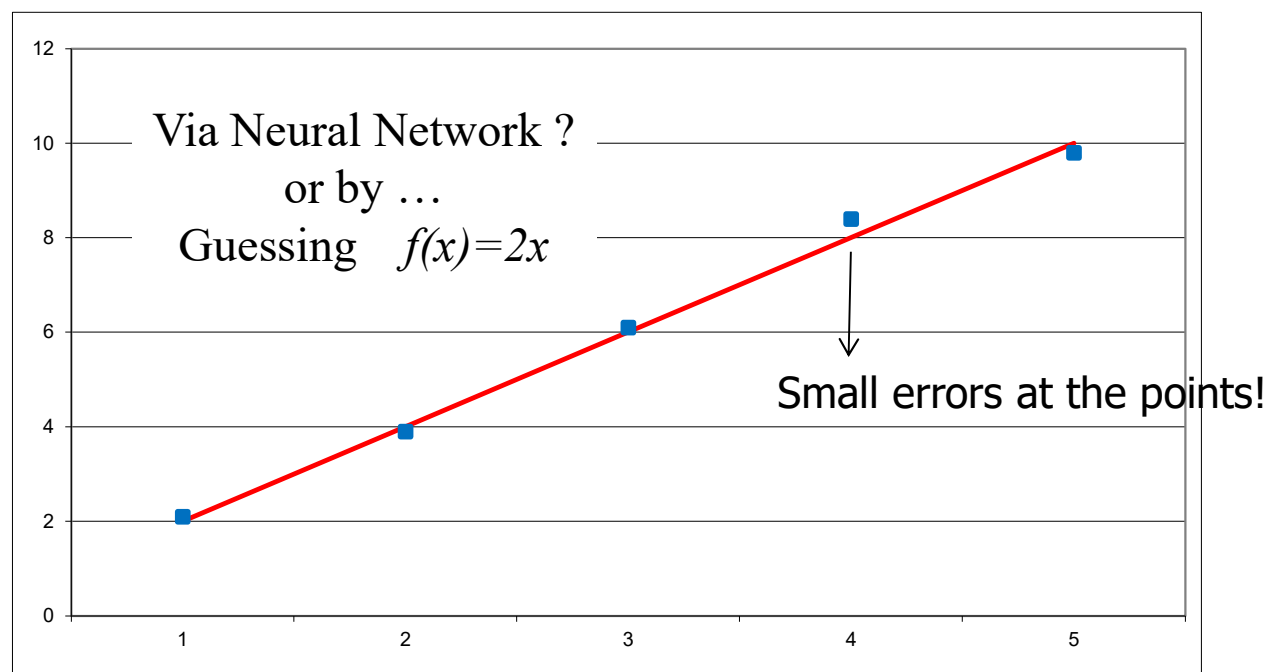


Tasks: Regression: example

- Process of estimating of a real-value function on the basis of finite set of noisy samples (supervised task)
 - known pairs $(x, f(x) + \text{random noise})$

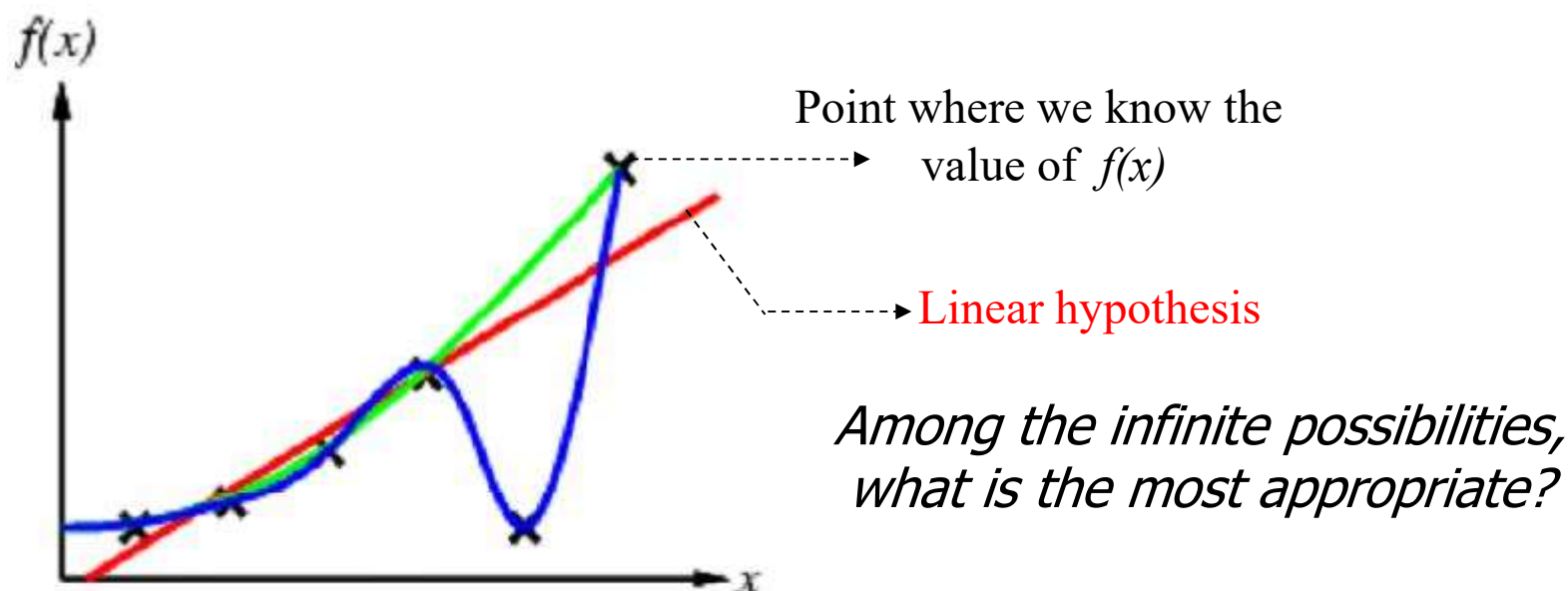
Task (exercise): find f for the data in the following table:

x	$target$
1	2.1
2	3.9
3	6.1
4	8.4
5	9.8
...	...



Tasks: regression

- *Regression*: x = variables (e.g. real values), $f(x)$ *real values*: curve fitting (x is 1-dim in the example but it becomes k -dim in general)
- Process of estimating of a real-value function on the basis of finite set of noisy samples
 - known pairs $(x, f(x) + \text{random noise})$

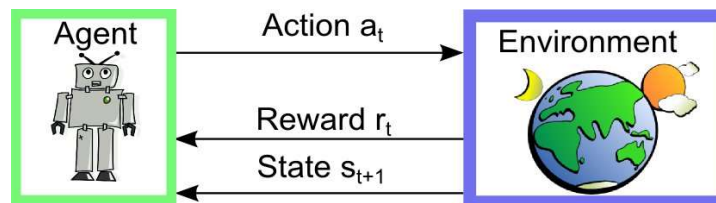


An example (**linear hypothesis**): $h_w(x) = w_1 x + w_0 = 0.2 x - 0.4$

Tasks: Other Topics ...



- **Semi-supervised learning**
 - combines both labeled and unlabeled examples to generate an appropriate function or classifier.
- **Reinforcement Learning** (learning with right/wrong critic).
 - Adaptation in *autonomous systems*
 - “the algorithm learns a *policy* of how to act given an observation of the world. Every action has some impact in the environment, and the environment provides feedback that guides the learning algorithm”.
 - Not step by step examples
 - Toward decision-making aims
 - Useful in modern AI



Reinforcement Learning Setup

Models

and survey of useful concepts



Dip. Informatica
University of Pisa

- **MODEL:**
 - Aim: to capture/describes the relationships among the data (on the basis of the task) by a “language”
 - The “language” is related to the *representation* used to get knowledge
 - It defines the class of functions that the learning machine can implement (*hypothesis space*)
 - E.g. set of functions $h(\mathbf{x}, \mathbf{w})$, where \mathbf{w} is the (abstract) parameter
- **Training example** (superv.): An example of the form $(\mathbf{x}, f(\mathbf{x}) + \text{noise})$
 \mathbf{x} is usually a vector of features, (d or y or) $t = f(\mathbf{x}) + \text{noise}$ is called the target value.
- **Target function:** The true function f
- **Hypothesis:** A proposed function h believed to be similar to f . An expression in a given language that describes the relationships among data.
- **Hypotheses space:** The space of all hypotheses (specific models) that can, in principle be output by the learning algorithm.

Models: few trivial examples....



Dip. Informatica
University of Pisa

Just to have a preview of different *representation* of hypothesis

(because you already know the language of equations, logic, probability):

- **Linear models** (representation of H defines a continuously parameterized space of potential hypothesis);

each assignment of w is a different hypothesis, e.g:

- $h(\mathbf{x}) = \text{sign}(\mathbf{w}\mathbf{x} + w_0)$

binary classifier

- $h_w(x) = w_1x + w_0$ E.g. $h_w(x) = 2x + 150$

simple linear regression

- **Symbolic Rules:** (hypothesis space is based on discrete representations); different rules are possible, e.g:

- if $(x_1=0)$ and $(x_2=1)$ then $h(\mathbf{x})=1$

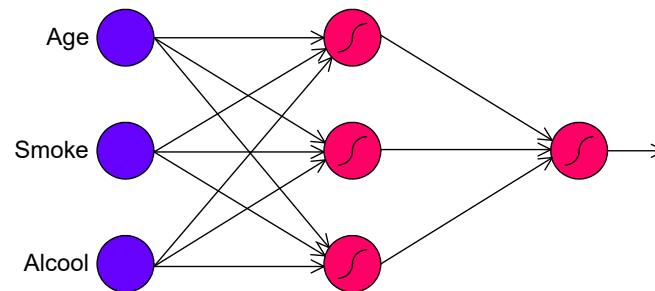
binary classifier

- else $h(\mathbf{x})=0$

- **Probabilistic models:** estimate $p(x,y)$
- K Nearest neighbor regression: Predict mean y value of nearest neighbors (memory-based)

Neural Networks (just a look)

An example: we will see a **neural networks**, beyond the *neurobiological inspiration*, as a computational model for the treatment of data, capable of approximating complex (*non-linear*) relationships between inputs and outputs



Again,
a class of functions !!!

Pardigms and methods (Languages for H)



Dip. Informatica
University of Pisa

- Symbolics and Rule-based (or *discrete* H)
 - Conjunction of literals^{*}, Decision trees (propositional rules)^{*}
 - Inductive grammars, Evolutionary algorithms, ...
 - Inductive Logic Programming (first order logic rules)
- Sub-symbolic (or *continuous* H)
 - Linear discriminant analysis, Multiple Linear Regression^{*}, LTU
 - Neural networks
 - Kernel methods (SVMs, gaussian kernels, spectral kernels, etc)
- Probabilistic/Generative
 - Traditional parametric models (density estimation, discriminant analysis, polynomial regression,...)
 - Graphical models: Bayesian networks, Naïve Bayes, PLSA, Markov models, Hidden Markov models, ...
- Instance-based
 - Nearest neighbor^{*}

Note: Underlined – >ML

1. Some models can be expressed by different languages
2. ^{*} Next lectures

How many models?

- Theory (*No Free Lunch Theorem*) : *there is no universal "best" learning method (without any knowledge, for any problems,...):*
if an algorithm achieves superior results on some problems, it must pay with inferiority on other problems. In this sense there is no free lunch.
- The course provide a set of models and the critical instruments to compare them
- However, not all the models are equivalent:
 - Important differences are for the **flexibility** of the approaches, toward models that can in principle approximate arbitrary functions (e.g. no just linear approximation seen in the examples)
 - Important differences are for the **control of the complexity** (we will see later)
 - Use of flexible models and principia for the control of the complexity are the core of ML

Learning Algorithms



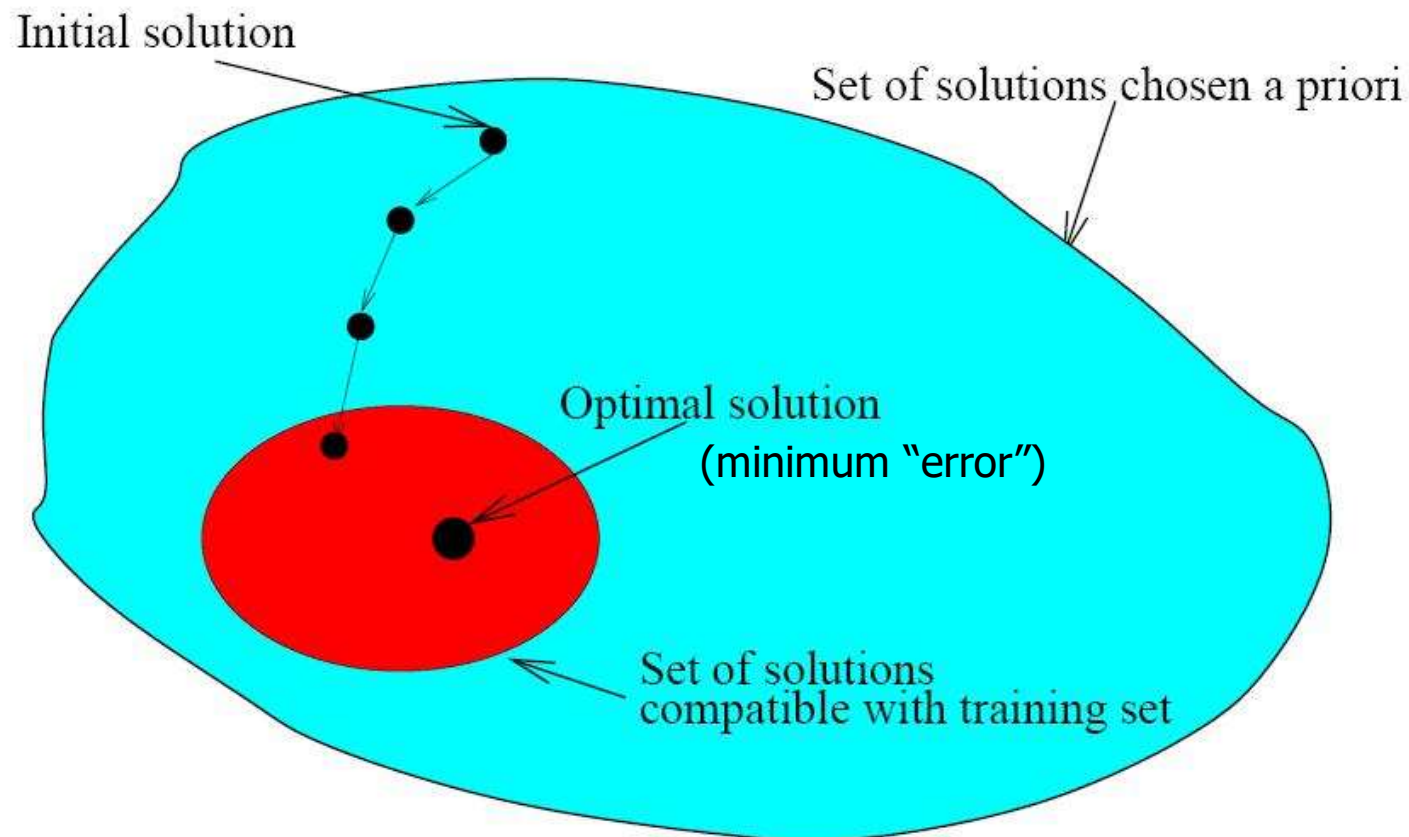
Dep. Informatica
University of Pisa

- **LEARNING ALG.** Basing on data, task and model
- (Heuristic) *search through the hypothesis space H of the **best hypothesis***
- i.e. the best approximation to the (unknown) target function
- Typically searching for the h with the minimum “error”
 - E.g. free parameters of the model are fitted to the task at hand:
 - Examples: best w in liner models, best rules for symbolic models,
 - Remember the regression example, we proposed $h(x)=2x$, for $h_w(x)=w_Ix+w_0$ assuming $w_I=2$ and $w_0=0$ as the best parameter value:
how?
- H may not coincide with the set of all possible functions and the search can not be exhaustive: we need to make assumptions →
(we will see the role of) *Inductive bias*

Learning Algorithms: search



Dip. Informatica
University of Pisa



Typically local search approaches

Learning (terminologies)



Dip. Informatica
University of Pisa

According to the different paradigms/contexts “learning” can be differently termed or have different acceptations:

- Inference (statistics)
- Inference: Abduction/Induction (logic)
- Adapting (biology, systems)
- Optimizing (mathematics)
- Training (e.g. Neural Networks)
- Function approximations (mathematics)

Can be more specifically found in other sub-fields:

- Regression analysis (statistics), curve fitting (math, CS), ...
- Or using other terminologies e.g. “*Fitting* a multivariate function”

Tasks + Models : Loss

A "**good**" approximation to f from examples.

How to measure the quality of the approximation?

- Recall that we produce $h(\mathbf{x})$ value (output of the model for input x)
- We want to measure the "distance" between $h(\mathbf{x})$ and d
(objective function for minimization of errors in training, check of errors in test)

We use a ("inner") *loss function*: $L(h(\mathbf{x}), d)$

e.g. high value \rightarrow poor approximation

The *Error* (or *Risk* or *Loss*) is an expected value of this L

e.g. a "sum" or mean of the inner loss L over the set of samples

$$E(\mathbf{w}) = \frac{1}{l} \sum_{p=1}^l L(h(\mathbf{x}_p), d_p)$$

Note:
index p is used for the
samples $p=1..l$

Which one? We will change L for different tasks

Note: at moment Error, Risk and Loss are considered equivalent, we will specify differences later through the course

Tasks: Common Tasks review

- A possible classifications of common learning tasks specifying the (changing of the) nature of the loss function (*in particular of L*), output and hypothesis space

Both:

- Survey of common learning tasks
- Nature of **models** (hypothesis spaces) for different class of tasks.
- *AKA Examples of loss functions: use it for future reference*

Regression

- Regression: *predicting a numerical value*
- **Output:** $d_i = f(x) + e$ (*real value function + random error*)
- **H:** a set of real-valued functions
- **Loss function** L : measures the approximation accuracy/error
- A *common* loss function for regression: the squared error

$$L(h(\mathbf{x}_i), d_i) = (d_i - h(\mathbf{x}_i))^2$$

- The mean over the data set provide the *Mean Square Error (MSE)*

MSE example

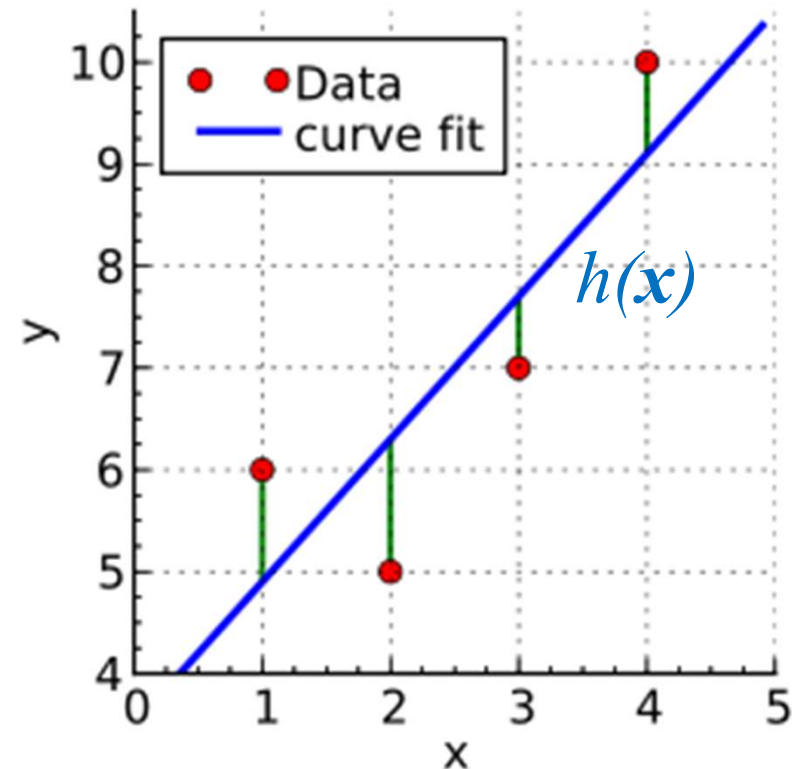
In the example we have

$h(x) = w_1 x + w_0$ as the blue line
and in green the errors at the **data points** (x_i, y_i) (in **red**), where the target d_i for x_i is denoted y_i in the example

The Mean Square Error (MSE)
is the mean of the square of the
green errors:

$$E(\mathbf{w}) = \frac{1}{l} \sum_{p=1}^l (y_p - h_{\mathbf{w}}(\mathbf{x}_p))^2$$

\mathbf{w} are the free parameters of
the linear model



Note: this plot is taken from internet, I used different colors before: here the line is in blue. Also the y are the desired (target d) values

Classification

- Classification of *data into discrete classes*
- **Output:** e.g. $\{0,1\}$
- **H:** a set of indicator functions
- **Loss function** L : measures the classification error

$$L(h(\mathbf{x}_i), d_i) = \begin{cases} 0 & \text{if } h(\mathbf{x}_i) = d_i \\ 1 & \text{otherwise} \end{cases}$$

Def

- The mean over the data set provide the *number/percentage of misclassified patterns*
- *E.g. 20 out of 100 are misclassified \rightarrow 20% errors, i.e. 80% of accuracy*

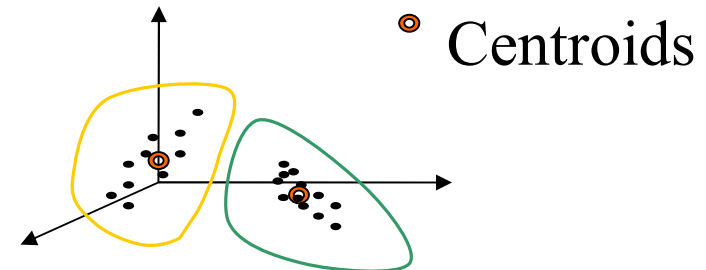
Clustering and Vector Quantization*



Dip. Informatica
University of Pisa

preview

- Goal: optimal partitioning of unknown distribution in \mathbf{x} -space into regions (clusters) approximated by a cluster center or *prototype*.



- **H**: a set of vector quantizers $x \rightarrow c(x)$
continuous space \rightarrow discrete space
- Loss function L : measures the vector quantizer optimality
- A *common* **loss function** would be the *squared error distortion*:

$$L(h(\mathbf{x}_i)) = (\mathbf{x}_i - h(\mathbf{x}_i)) \bullet (\mathbf{x}_i - h(\mathbf{x}_i))$$

$\bullet = \text{inner_product}$

\longrightarrow We'll see later

Proximity of the pattern to the centroid of its cluster

Density estimation* *preview*



Dip. Informatica
University of Pisa

- Density estimation (generative, “parametric methods”) from an assumed class of density
- **Output:** a density e.g. normal distribution with mean m and variance σ^2 : $p(x \mid m, \sigma^2)$
- **H:** a set of densities (e.g. m and σ^2 are the two unknown *parameters*)
- A *common* **loss function** L for density estimation:

$$L(h(\mathbf{x}_i)) = -\ln(h(\mathbf{x}_i))$$

—————> We’ll see later

- Related to “maximizing the (log) likelihood function”. [not hear]
- E.g. $P(x_1, x_2, x_3, \dots \mid m, \sigma^2)$

Machine Learning: generalization

This is a fundamental concept of the course

- *Learning*: search for a **good function** in a function space from known data (*typically minimizing an Error/Loss*)
- **Good** w.r.t. generalization error: it measures how accurately the model predicts over novel samples of data (*Error/Loss measured over new data*)

Generalization: crucial point of ML!!!

Easy to **use** ML tools *versus* **correct/good use** of ML

Generalization

- **Learning** phase (**training, fitting**): build the model from know data – *training data* (and bias)
- **Predictive** phase (deployment/ *Inference* use of the ML built model): apply the model to new examples.
For **test** we take the input x' , we compute the response by the model; comparing with the target value that the model has never seen we can evaluate our predictive hypothesis,
i.e. evaluation of the **generalization capability**

Note: *performance* in ML = generalization accuracy/ *predictive accuracy*
estimated by the error computed on the (hold out) **Test Set**

- **Theory**: E.g. Statistical Learning Theory [Vapnik] :
 - *under what (mathematical) conditions is a model able to generalize?* → see next lecture (just basic notions)

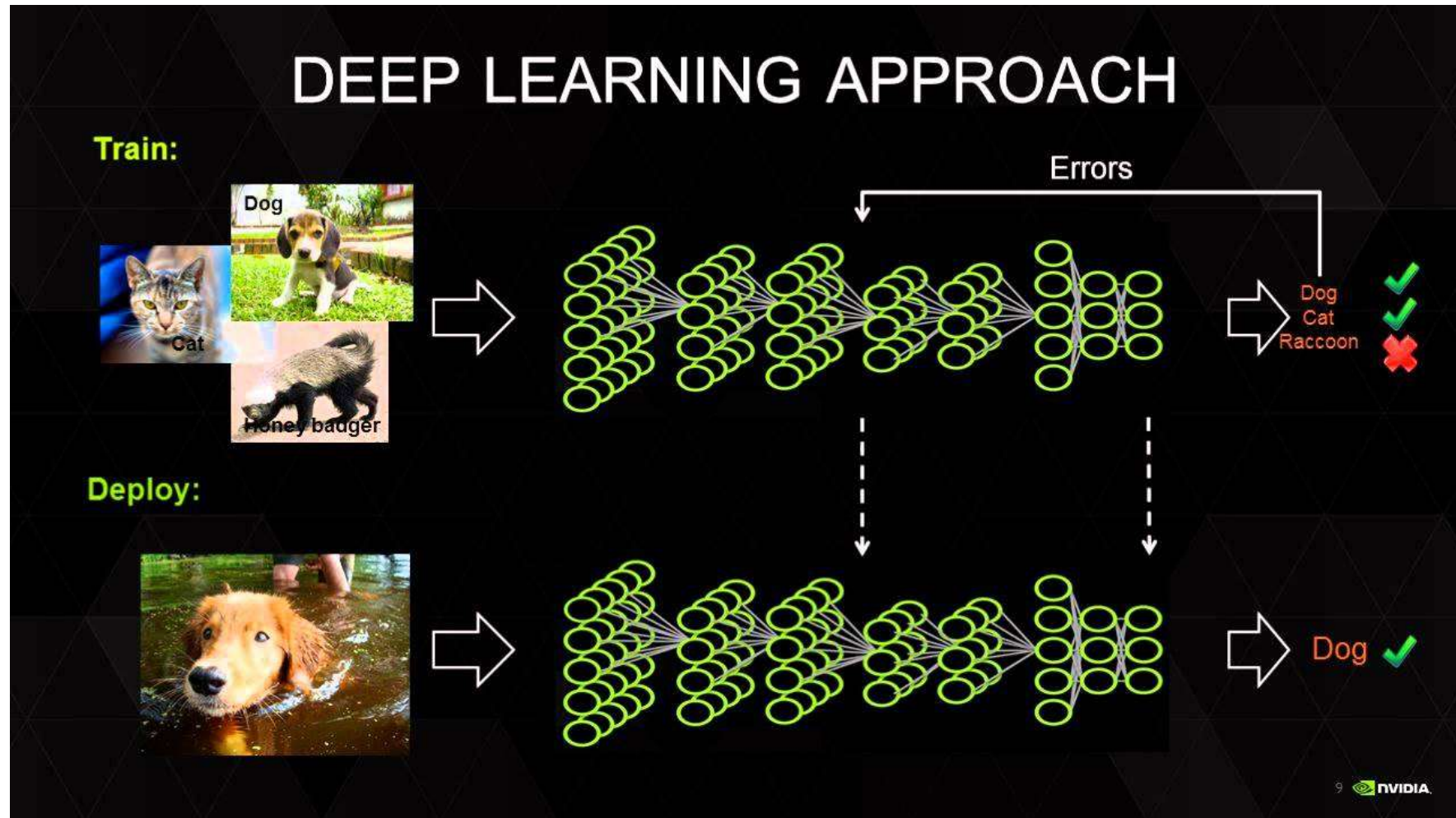
Validation

- Evaluation of performances for ML systems =
Predictive accuracy
- Validation !
- Validation !!
- Validation !!!
- *"The performance on training data provide an overoptimistic evaluation"*
- In the following: Some techniques

Exemplification of the Deployment/ *Inference* use



Dip. Informatica
University of Pisa



Exemplification of the Deployment/ *Inference* use



Dip. Informatica
University of Pisa



Even the inference part can be costly if you have millions of requests
(e.g. at google)

A Google server rack containing multiple **Tensor Processing Units**, a special-purpose chip designed specifically for machine learning
The original TPU was designed specifically to work best with Google's TensorFlow.

Just for inference (mapping) !!!!

Summary of the Intro to ML

- Part I (now)
 - Motivations, contextualization in CS
 - Course info
- Part II (in Lect. 2)
 - Utility of ML
 - Learning as function approximation (pilot example)
 - Design components of a ML system, including
 - Learning tasks
 - Hypothesis space (and first overview)
 - Loss and learning tasks
 - Generalization (first part)
- Part III (in Lect.3 ???)
 - Generalization and Validation
 - Aim:*** *overview and terminology*
 - before starting to study models and learning algorithms*

Per informazioni

Alessio Micheli
micheli@di.unipi.it

<http://ciml.di.unipi.it>



Dipartimento di Informatica
Università di Pisa - Italy



**Computational Intelligence &
Machine Learning Group**

Lect 3

Introduction to Machine Learning (continuation)

Introduction to Generalization in ML

Alessio Micheli

micheli@di.unipi.it

ML in a Nutshell

- DATA: available experience represented as vectors, structures,...
 - TASKS: supervised (classification, regression), unsupervised, ...
 - E.g. Given data as labeled examples, find good approximation of the unknown f .
 - MODELS
 - describes the relationships among the data / the knowledge
 - define the class of functions that the learning machine can implement (*hypothesis space*)
 - LEARNING ALGORITHM
 - (given data, task and model) the learning algorithm performs a (heuristic) *search* through a space of hypotheses that are valid in the given data
 - E.g. it adapts the free parameters of the model to the task at hand
- VALIDATION: evaluate generalization capabilities (of your hp)

ML issues

**Easy use of ML tools *versus* correct/good
use of ML**

ML issues

- Inferring general functions from known data: an *ill posed problem* (e.g. in principle the solution is not unique)
 - With finite data we cannot expect to find the exact solution
- Work with a restricted hypothesis space
- What can we represent ?
- (Secondary) What can we learn ?
(as if you cannot represent a function you cannot also learn it)

Generalization

- **Learning phase:** to build the model
- **Prediction phase:** evaluate the learned function over novel samples of data (generalization capability)
- Inductive learning hypothesis
 - Any h that approximates f well on training examples will also approximate f well on new (unseen) instances x (?)

Def

- **Overfitting:** A learner overfits the data if
 - It output a hypothesis $h(\cdot) \in H$ having true error (risk) R and empirical error E , but there is another $h'(\cdot) \in H$ having $E' > E$ and $R' < R$
- Critical aspect: accuracy / performance estimation
 - Theoretical
 - Empirical (training, test) and cross-validation techniques

Complexity on case of study

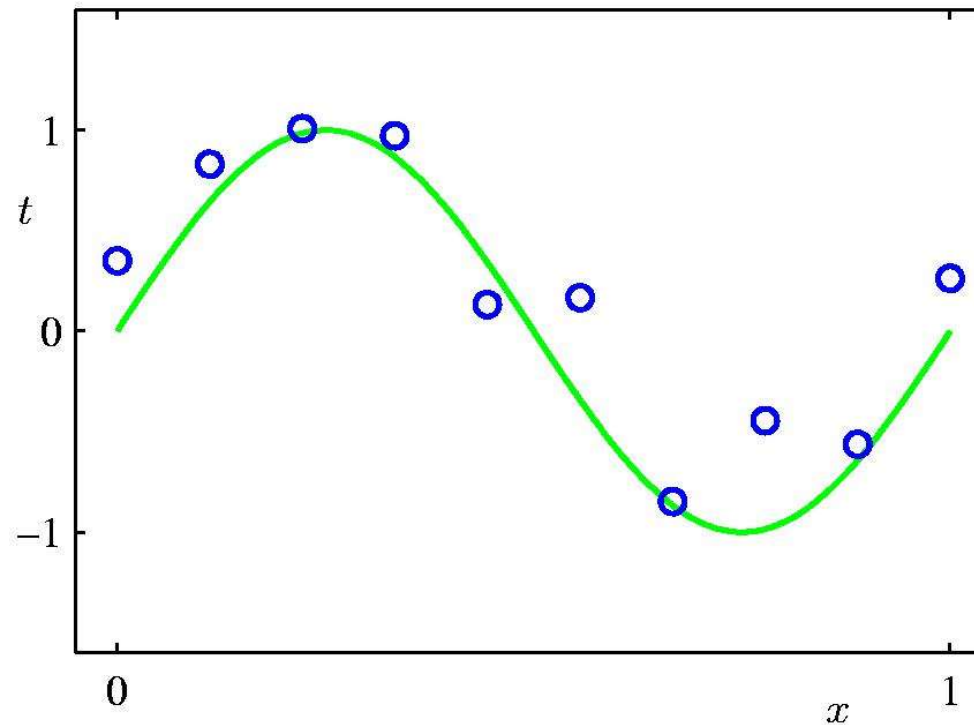
- An example on a parametric model for *regression*:
- The set of functions is assumed as polynomials with degree M
- The **complexity** of the hypothesis increases with the degree M
- l = number of examples
- Warning: This is an artificial simplified task (unrealistic due to the use of just 1 input variable, the fact that we know the target function in advance, ...)

Polynomial Curve Fitting



Dip. Informatica
University of Pisa

Target = $\sin(2\pi x)$ + random noise (gaussian)



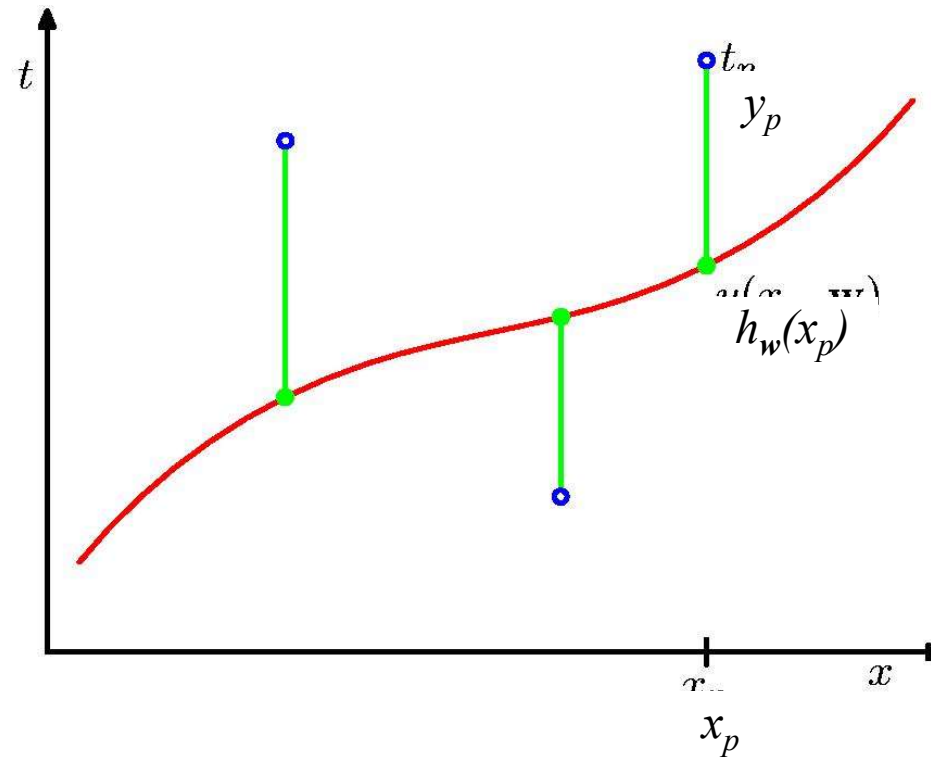
$$h_{\mathbf{w}}(x) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M = \sum_{j=0}^M w_jx^j$$

Samples affected by noise (not always on the green “true” line)

Sum-of-Squares Error Function



Dip. Informatica
University of Pisa



$$E(\mathbf{w}) = \sum_{p=1}^l (y_p - h_{\mathbf{w}}(x_p))^2$$

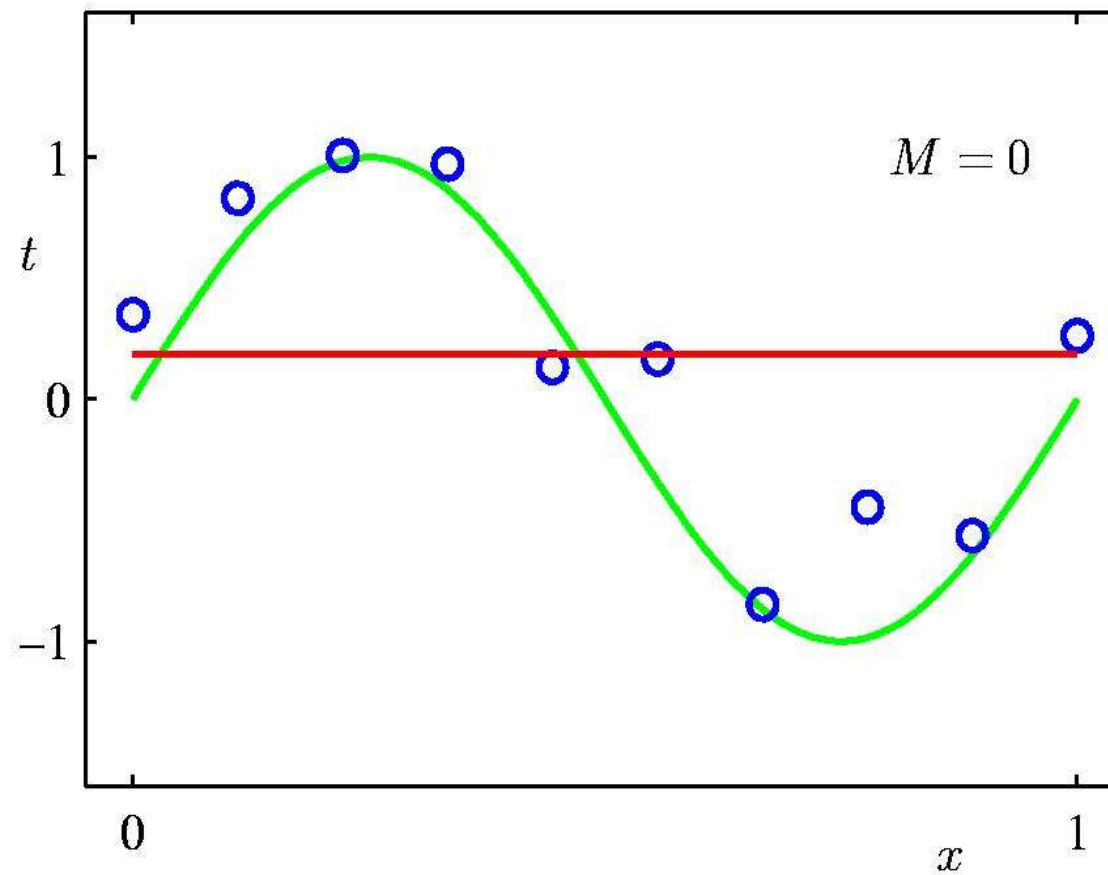
Note: p is the example, y_p the target for p
 l the total number of examples
 $h_{\mathbf{w}}(x_p)$ is the model output at the point x_p

Minimize $E(\mathbf{w})$ (Square Error) to find the best \mathbf{w} (fitting)

0th Order Polynomial



Dip. Informatica
University of Pisa

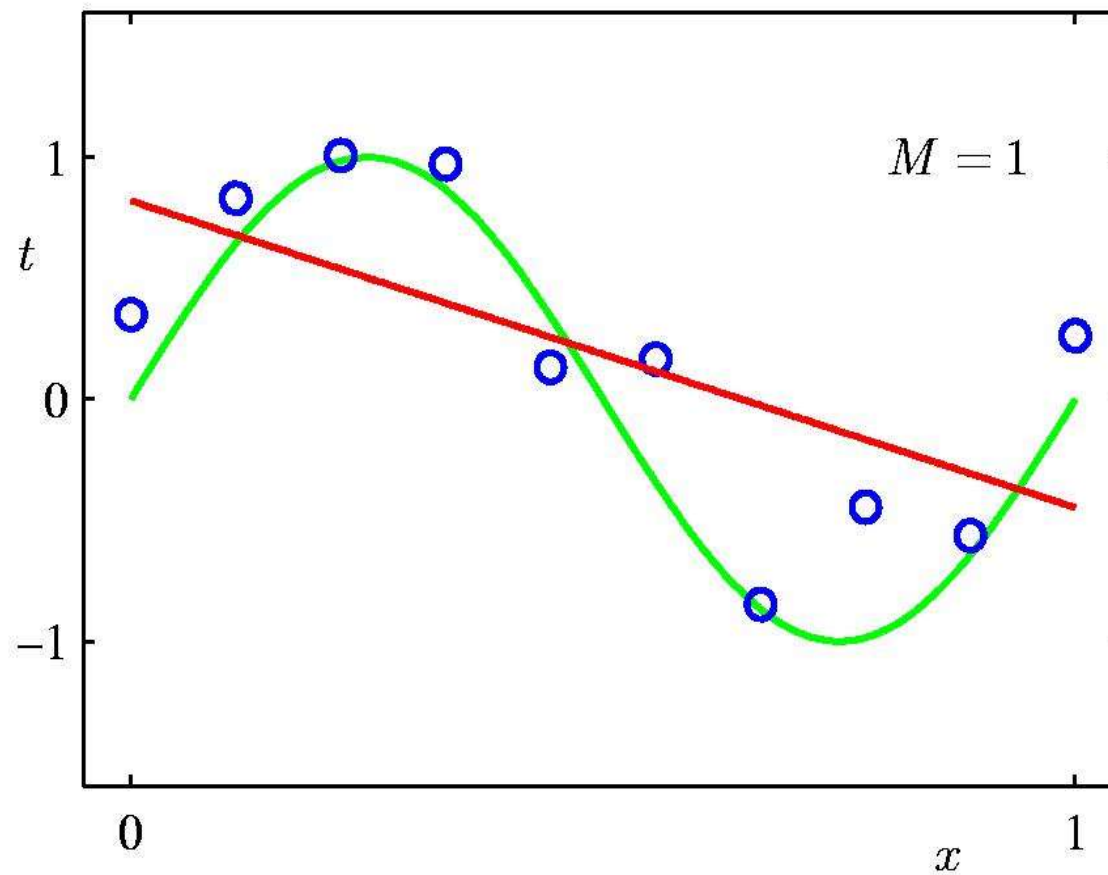


Underfitting: too simple model

1st Order Polynomial



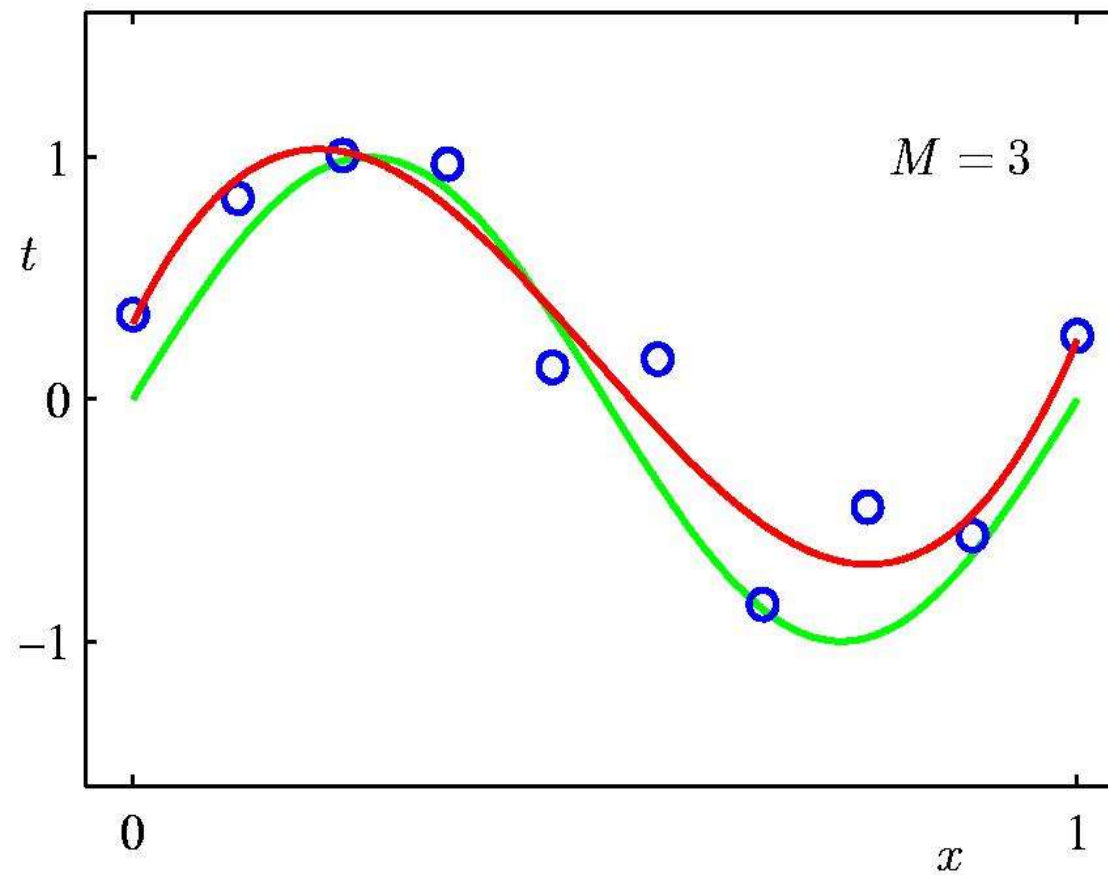
Dip. Informatica
University of Pisa



3rd Order Polynomial



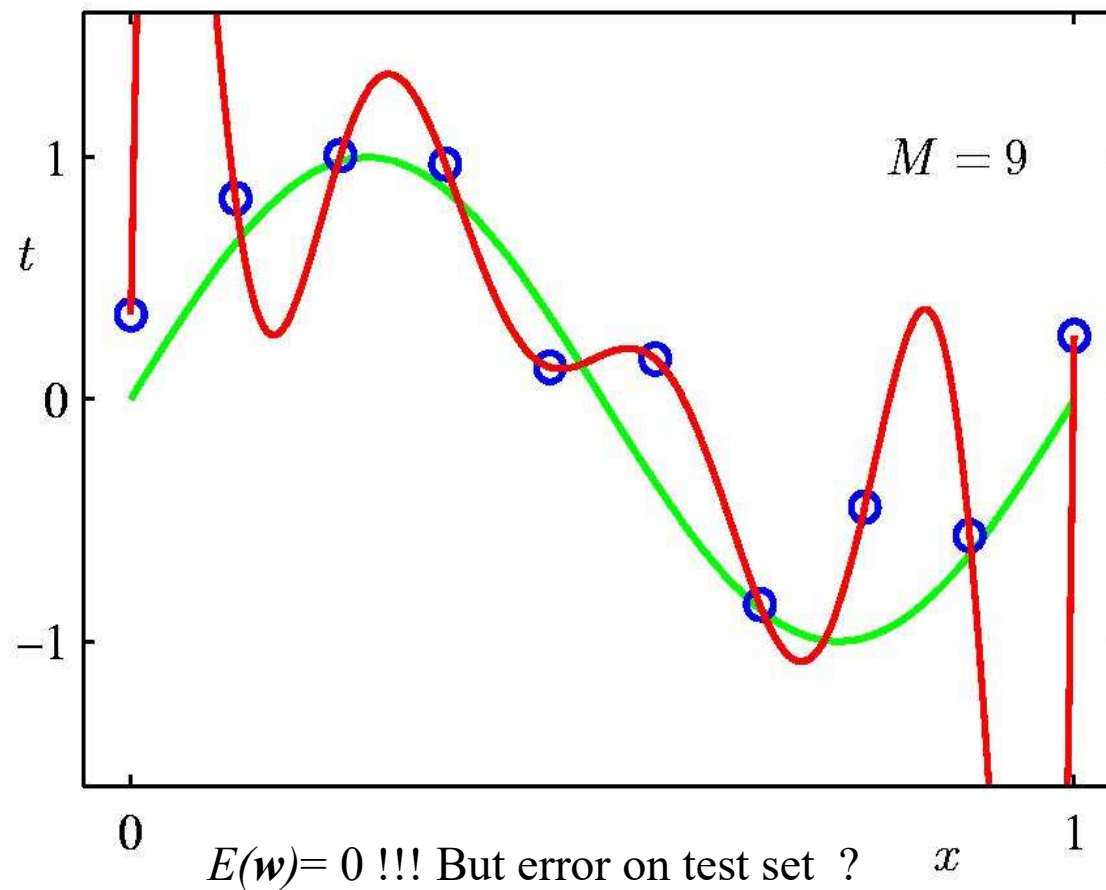
Dip. Informatica
University of Pisa



9th Order Polynomial



Dip. Informatica
University of Pisa



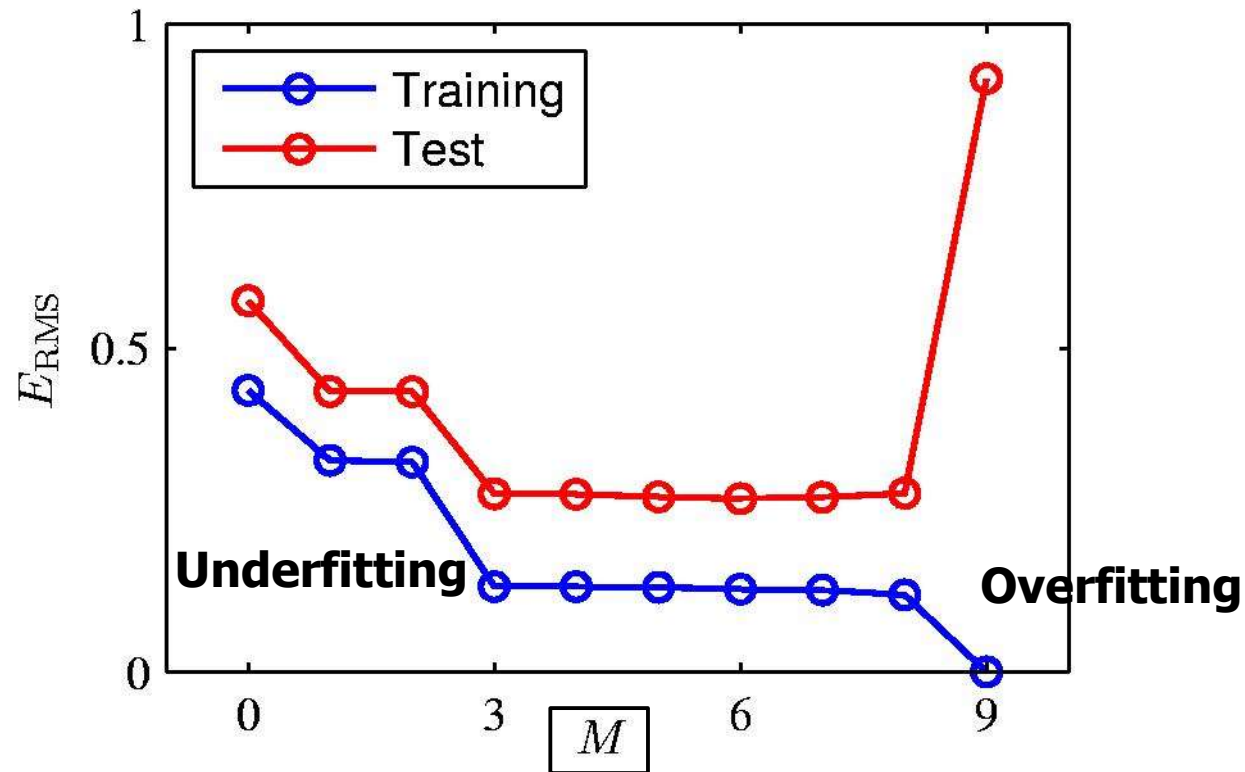
Too complex model: fit the noise!

Poor representation of the (green) true function (due to **overfitting**)

Underfitting and Overfitting with the complexity (M)



Dip. Informatica
University of Pisa



Root-Mean-Square (RMS) Error: $E_{\text{RMS}} = \sqrt{2E(\mathbf{w}^*)/l}$

Where $E(\mathbf{w}^*)$ is the error for the trained model

Polynomial Coefficients



Dip. Informatica
University of Pisa

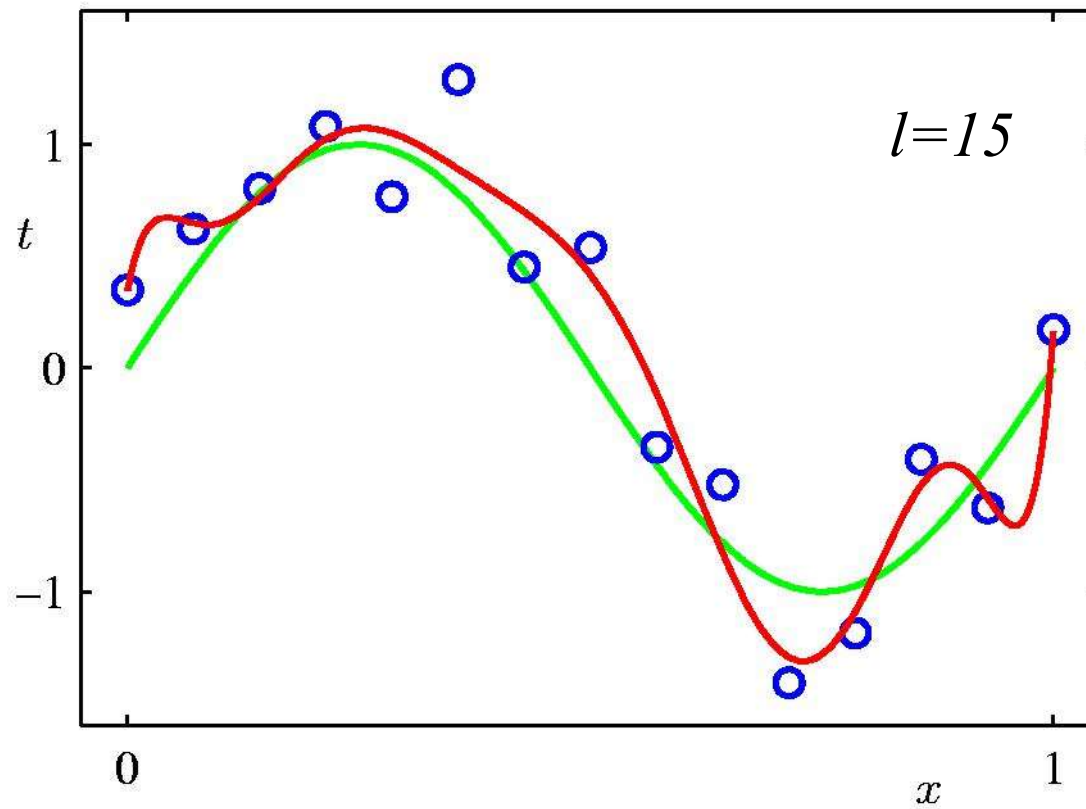
	$M = 0$	$M = 1$	$M = 3$	$M = 9$
w_0^*	0.19	0.82	0.31	0.35
w_1^*		-1.27	7.99	232.37
w_2^*			-25.43	-5321.83
w_3^*			17.37	48568.31
w_4^*				-231639.30
w_5^*				640042.26
w_6^*				-1061800.52
w_7^*				1042400.18
w_8^*				-557682.99
w_9^*				125201.43

Data Set Size: $l=15$
previous was 10



Dip. Informatica
University of Pisa

9th Order Polynomial

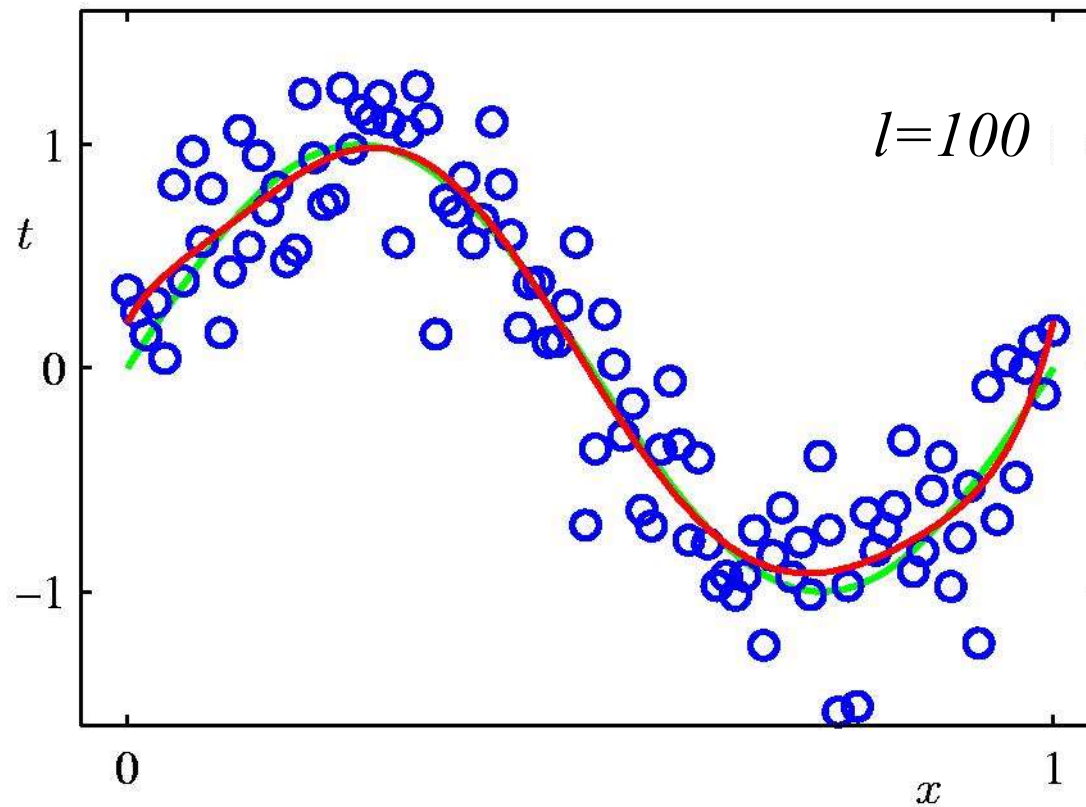


Data Set Size: $l=100$



Dip. Informatica
University of Pisa

9th Order Polynomial



We can use higher M with a higher number of data

Toward SLT

Putting all together:

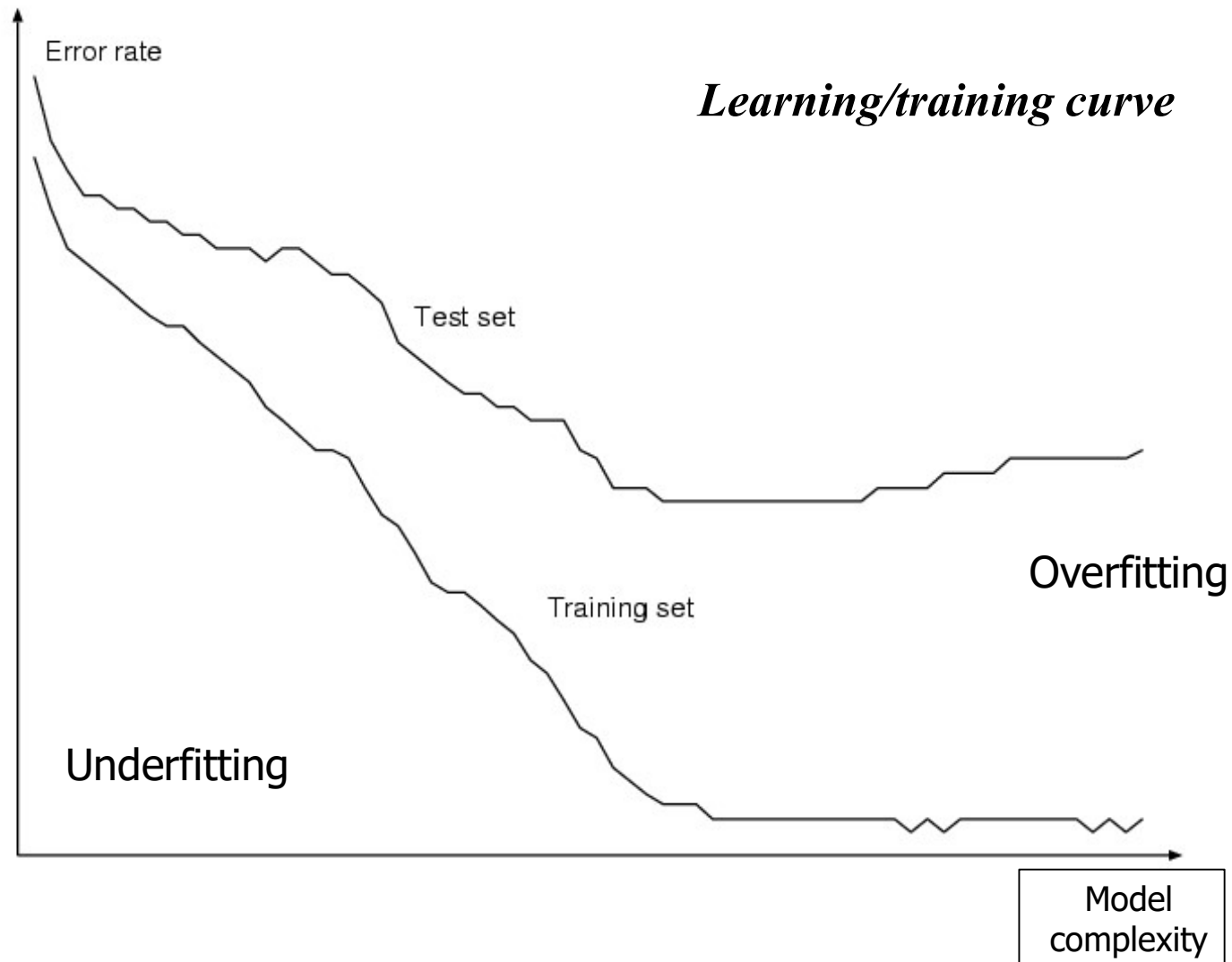
- We want to investigate on the *generalization* capability of a model (measured as a risk or test error)
 - with respect to the training error
 - overfitting and underfitting zones
- The role of model complexity
- The role of the number of data
- *Statistical Learning Theory (SLT)*: a general theory relating such topics

(Simplified) Formal Setting

- Approximate unknown $f(\mathbf{x})$, d target (*true f + noise*)
- Minimize *risk function* $R = \int L(d, h(\mathbf{x})) dP(\mathbf{x}, d)$ True Error
Over *all* the data
- Given
 - value from teacher (d) and the probability distribution $P(\mathbf{x}, d)$
 - a loss (or cost) function, e.g. $L(h(\mathbf{x}), d) = (d - h(\mathbf{x}))^2$
- Search h in H : Min R
- But we have only the finite data set $TR = (\mathbf{x}_p, d_p), \quad p = 1 \dots l$
- To search h : minimize empirical risk (**training error E**), finding the best values for the model free parameters

$$R_{emp} = \frac{1}{l} \sum_{p=1}^l (d_p - h(\mathbf{x}_p))^2$$
- Empirical Risk Minimization (ERM) Inductive Principle
- *Can we use R_{emp} to approximate R ?*

Typical behavior of learning



Vapnik-Chervonenkis-dim and SLT: a general theory (I)



Dip. Informatica
University of Pisa

- *VC-dim* measure complexity of H (*flexibility to fit data*)
(e.g. Num. of parameters for linear models/polynomials)

Repetita: Can we use R_{emp} to approximate R ?

Def.

VC-bounds in the form: it holds with probability $1-\delta$ that
guaranteed risk

$$R \leq \underbrace{R_{emp}} + \underbrace{\varepsilon (1/l, VC, 1/\delta)}_{VC\text{-confidence}}$$

- First (basic) explanation:
 - ε is a function directly proportional to VC ($VC\text{-dim}$), inversely proportional to l and δ .
 - We know that R_{emp} decrease using complex models (with high $VC\text{-dim}$) (e.g. the polynomial degree in the example)
 - δ is the confidence, it rules the probability that the bound holds (e.g. low δ 0.01, it holds with probability 0.99)
- Now we can see how it can “explain” the *underfitting* and *overfitting* and the aspects that control them.

Vapnik-Chervonenkis-dim and SLT: a general theory (II)



Dip. Informatica
University of Pisa

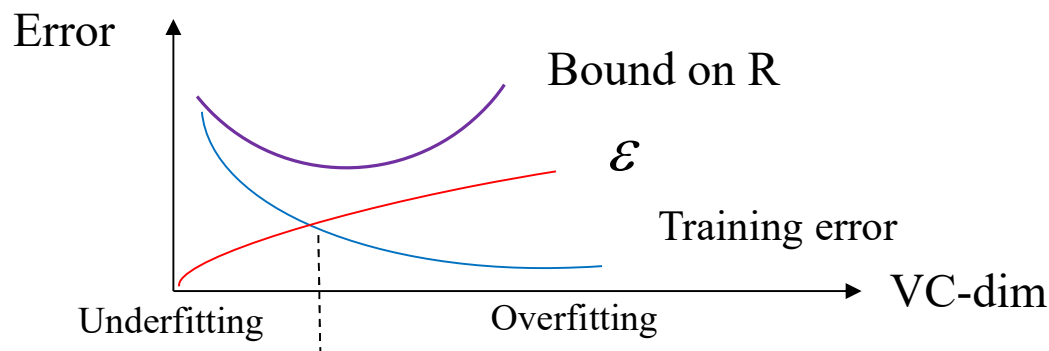
- *VC-dim* measure complexity of H (*flexibility to fit data*)
(e.g. Num. of free parameters for linear models/polynomials)

Repetita: Can we use R_{emp} to approximate R ?

- *VC-bounds in the form:* it holds with probability $1-\delta$ that

$$\text{guaranteed risk } R \leq \underbrace{R_{emp}}_{\text{Training error}} + \underbrace{\varepsilon (1/l, VC, 1/\delta)}_{\text{VC-confidence}}$$

- Intuition:
 - Higher l (data) \rightarrow lower VC confidence and bound close to R
 - Too simple model can be not suff. Due to high R_{emp} (underfitting)
 - Higher VC-dim (fix l) \rightarrow lower R_{emp} but R may increase (overfitting)
- Structural risk minimization: minimize the bound !



l is fixed
blue+red=purple

- Concept of control of the model complexity (flexibility):
A. Micheli trade-off between model complexity and TR accuracy (fitting)

An example

It is possible to derive an upper bound of the ideal error which is valid with probability $(1-\delta)$, δ being arbitrarily small, of the form:

- General:
$$R \leq R_{emp} + \varepsilon (1 / l, VC, 1 / \delta)$$
- Example:
$$R \leq R_{emp} + \varepsilon (VC / l, -\ln(\delta / l))$$
- There are different bounds formulations according to different classes of f , of tasks, etc.
- More in general, in other words (simplifying): we can make a good approximation of f from examples, provided we have a good number of data, and the complexity of the model is suitable for the task at hand.
 - Fit data as much as possible to avoid underfitting (high R_{emp}), but not too much to avoid overfitting (due to the increase of *VC-confidence* term)

Discussion

Complexity control



Dip. Informatica
University of Pisa

- SLT - Statistical Learning Theory:
 - It allows formal framing of the problem of generalization and overfitting, providing analytic upper-bound to the risk R for the prediction over all the data, regardless to the type of learning algorithm or details of the model..
 - *The ML is well founded*: the Learning risk can be analytically limited and only few concepts are fundamentals !
 - It leads to new models (SVM) (and other methods that directly consider the control of the complexity in the construction of the model)
 - It bases one of the inductive principles on the control of the complexity
 - It explains the main difference with respect to supporting methods from CM (providing the techniques to perform fitting), apart from modelling aspects

Open questions:

- What (other) principles are to found the control of the complexity?
How to work in practice?
 - How to measure the complexity (or fitting flexibility)?
 - How find the best trade-off between fitting and model complexity?

Validation

- Evaluation of performances for ML systems =
Predictive accuracy
- “*The performance on training data provide an overoptimistic evaluation*”
- Validation !
- Validation !!
- Validation !!!
- In the following: an introduction,
- Validation will be the topic of specific lectures later, in the “*Validation & SLT*” part of the course
- And it has a central role for the applications and the project

Validation: Two aims

- **Model selection:** estimating the performance (*generalization error*) of different learning models in order to choose the best one (to generalize).
 - this includes search the best *hyper-parameters* of your model (e.g. polynomial order, ...).

It returns a model

- **Model assessment:** having chosen a final model, estimating/evaluating its prediction error/ risk (*generalization error*) on new test data (measure of the quality/performance of the ultimately chosen model).

It returns an estimation

Gold rule: Keep separation between goals and use separate data sets

Validation: ideal world

- A large *training set* (to find the best hypothesis, see the theory)
- A large validation set for model selection
- A very large external unseen data *test set*

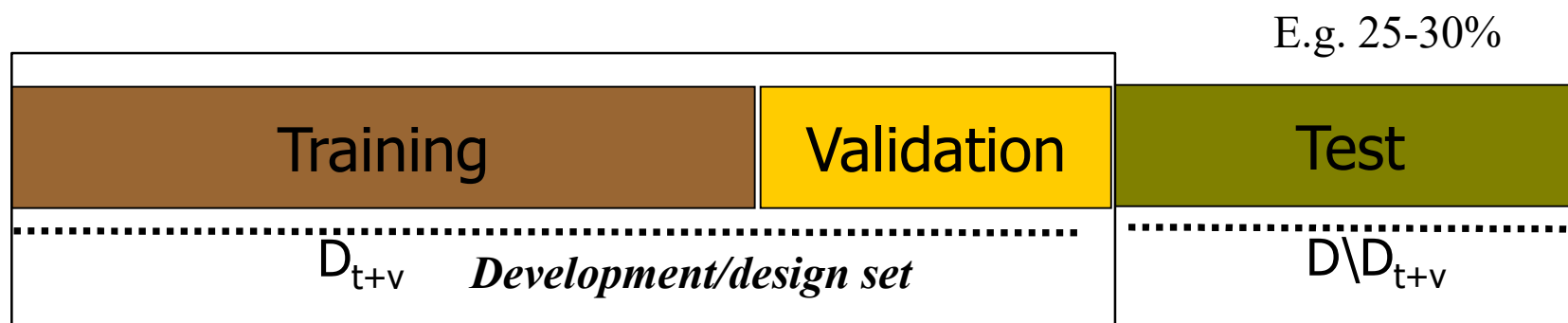
- *With finite and often small data sets?*
- *....Just estimation of the generalization performance*

- *We anticipate to basic techniques:*
 - *Simple hold-out (basic setting)*
 - *K-fold Cross Validation (just an hint in this lecture)*

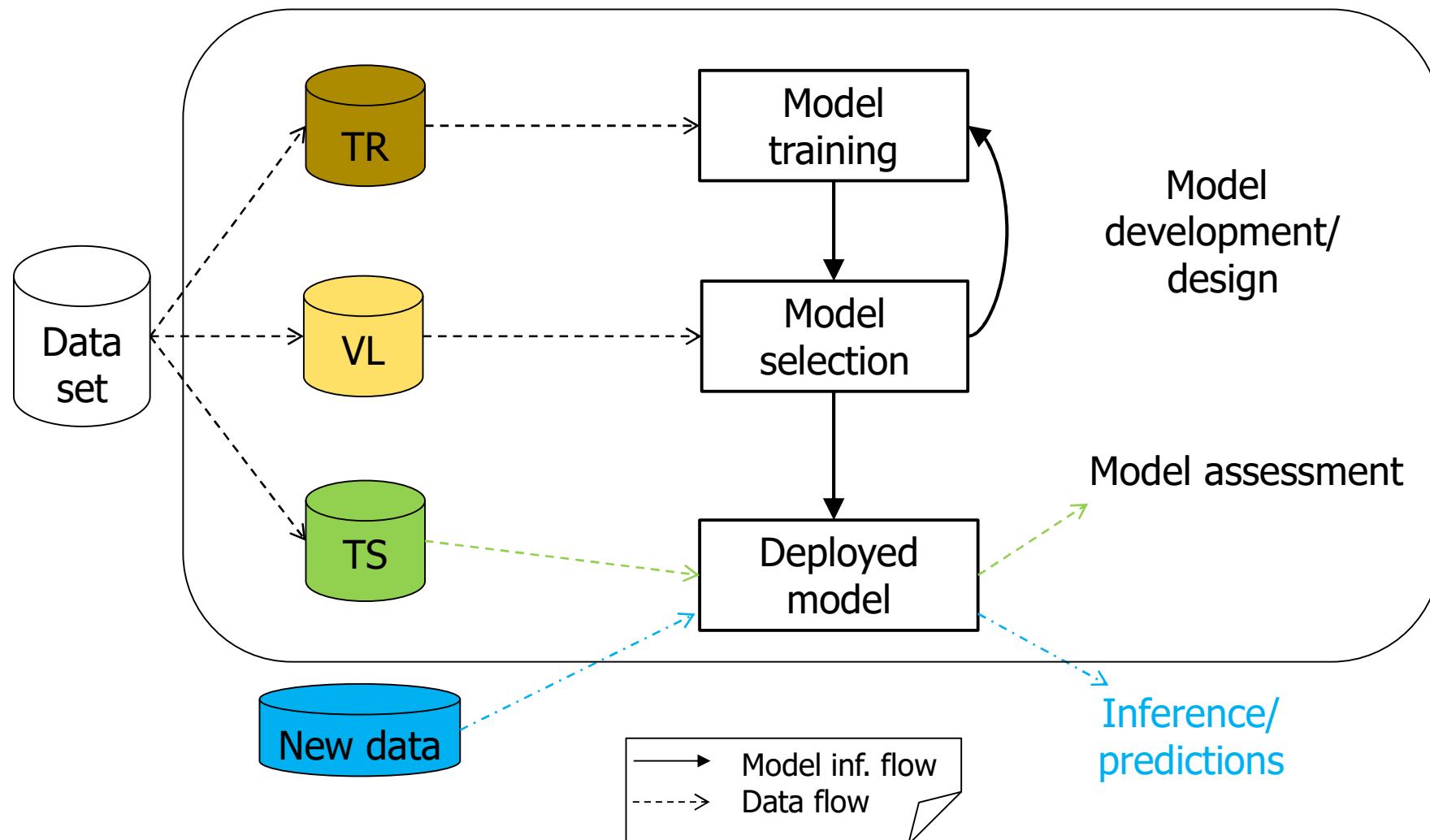
Hold out cross validation

Hold out: basic setting

- Partition data set D into *training set* (TR), *validation or selection set* (VL) and *test set* (TS)
 - All the three sets are disjoint sets !!!
 - Training is used to run the training algorithm
 - VL can be used to select the best model (e.g hyper-parameters tuning)
 - Test set is *not* to be used for tuning/selecting the best h : it is only for model assessment



TR/VL/TS by a schema



Hold out and **K-fold** cross validation

Hold out CV *can make insufficient use of data*



K-fold Cross-Validation

(we will see later during the course)

- Split the data set D into k mutually exclusive subsets D_1, D_2, \dots, D_k
- Train the learning algorithm on $D \setminus D_i$ and test it on D_i
- Can be applied for both VL or TS splitting
- *It uses all the data for training and validation/testing*

Issues:

- How many folds? 3-fold, 5-fold, 10-fold, ..., 1-leave-out
- Often computationally very expensive
- Combinable with validation set, double-K-fold CV,

Classification Accuracy



Dip. Informatica
University of Pisa

Def

Confusion matrix

Predicted \ Actual	Positive	Negative
Positive	TP	FN
Negative	FP	TN

$$\text{Specificity} = \text{TN} / (\text{FP} + \text{TN})$$

(True Negative rate = 1 - FPR)

$$\text{Sensitivity} = \text{TP} / (\text{TP} + \text{FN})$$

(True Positive rate or Recall)

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

false positive (FP) :eqv. with
false alarm

Accuracy: % of correctly classified patterns = $\text{TP} + \text{TN} / \text{total}$

Note: for binary classif.: 50% correctly classified = “coin” (random guess) predictor!

Other topics:

- unbalanced data (e.g. 99% +) \rightarrow trivial classifier exists ,
-

ROC curve

Confusion matrix

Predicted \ Actual	Positive	Negative
Positive	TP	FN
Negative	FP	TN

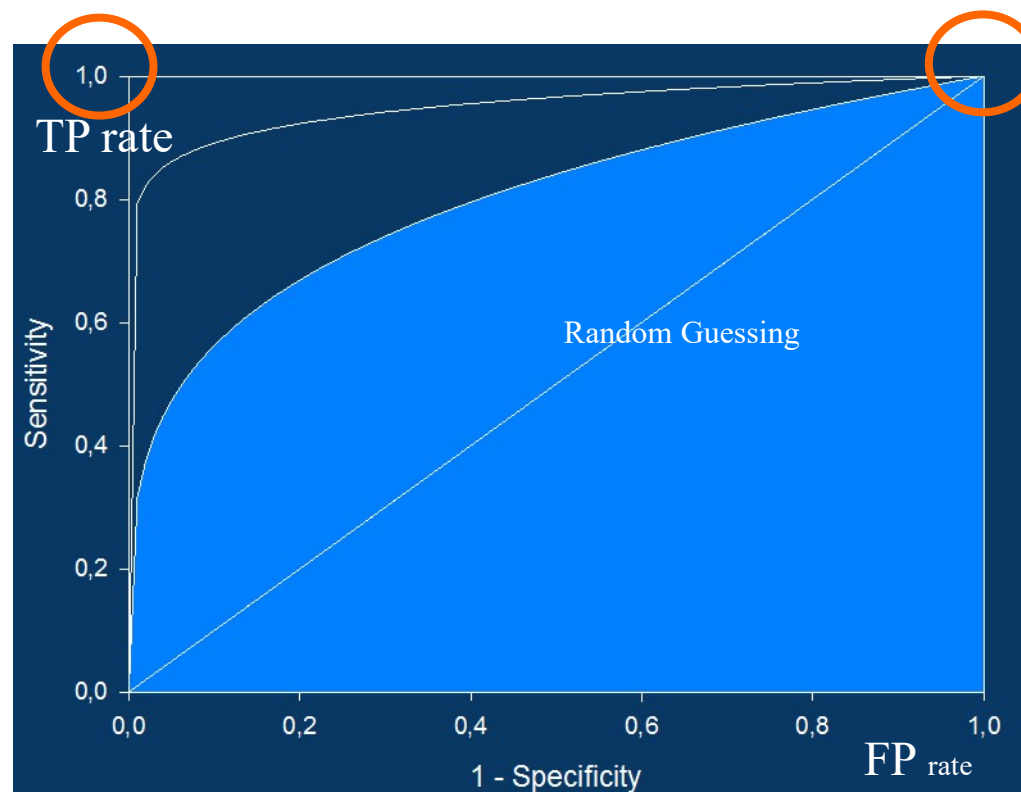
$$\text{Specificity} = \text{TN} / (\text{FP} + \text{TN})$$

$$\text{TPr or Sensitivity} = \text{TP} / (\text{TP} + \text{FN})$$

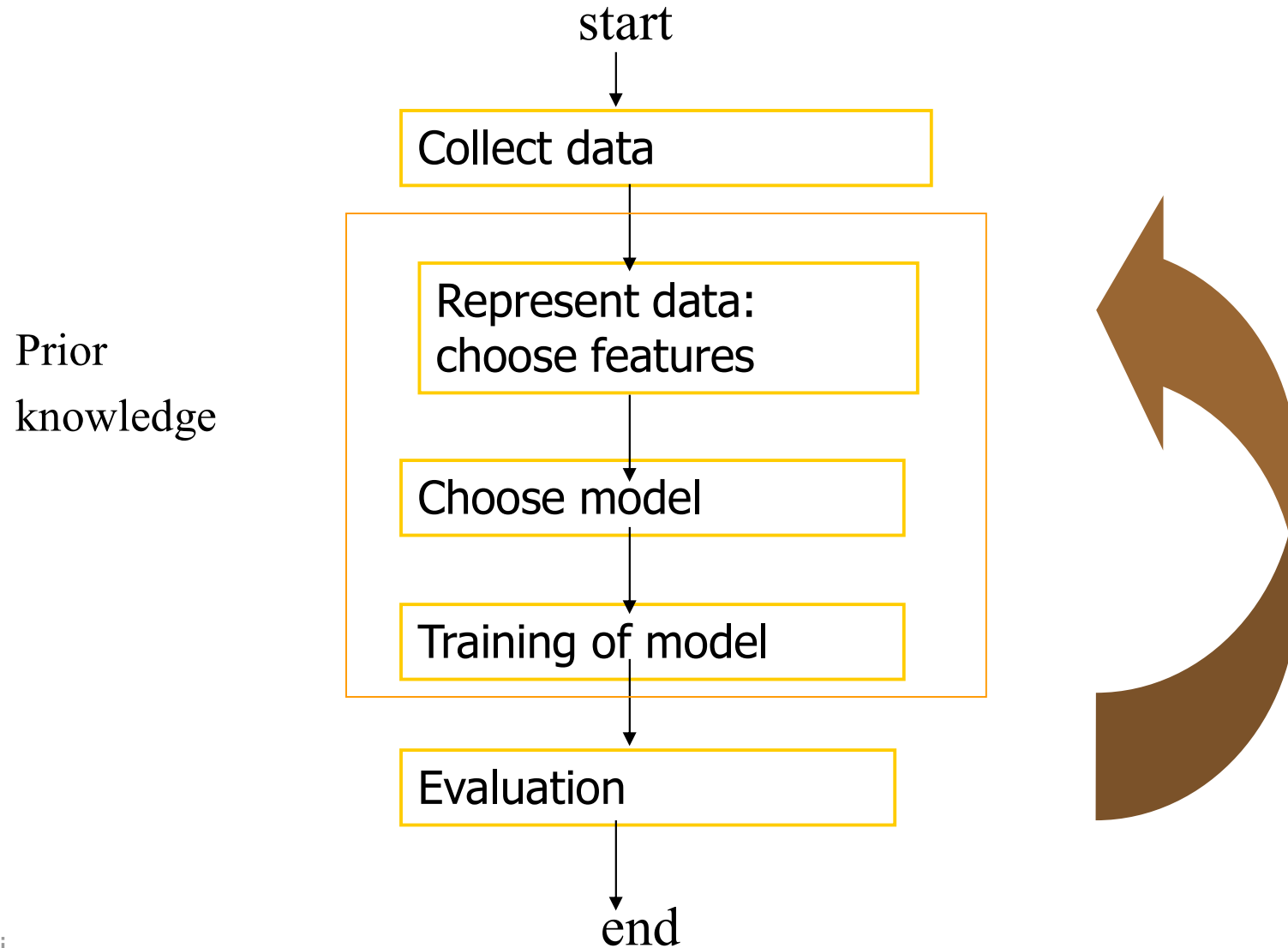
ROC curve

The diagonal corresponds to the worst classifier.

- Better curves have higher AUC (Area Under the Curve).



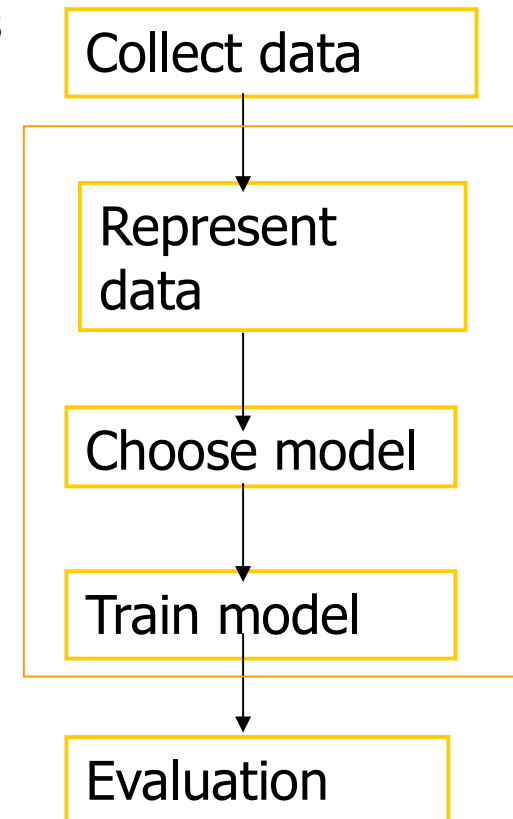
The Design Cycle



Design cycle (I)

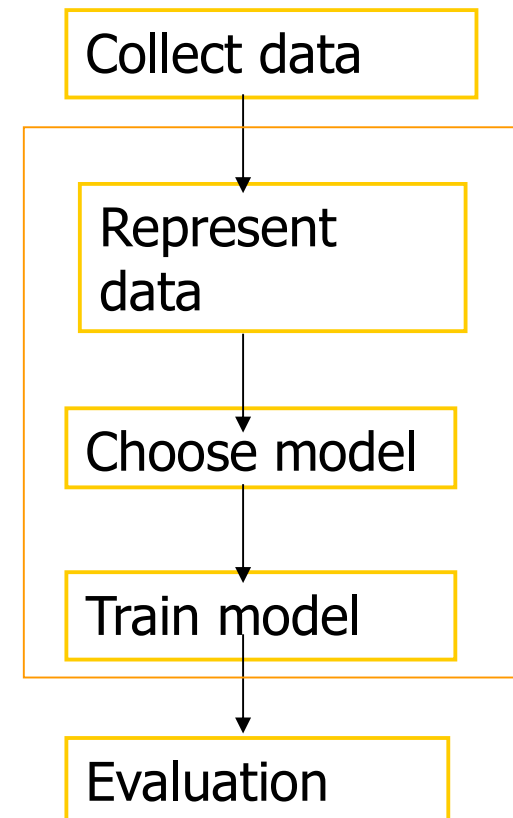
- Data collection:
 - adequately large and representative set of examples for training and test the system
- Data representation
 - domain dependent, exploit prior knowledge of the application expert
 - Feature selection
 - Outliers detection
 - Other preprocessing: variable scaling, missing data,..

Often the most critical phase for an overall success!
- Model choice:
 - statement of the problem
 - hypothesis formulation
 - You must know the limits of applicability of your model
 - complexity control



Design cycle (II)

- Building of the model (core of ML):
 - through the learning algorithm using the training data
- Evaluation:
 - performance = predictive accuracy !



Misinterpretations

For every statistical models (including DM applications)

- Causality is (often) assumed and a set of data representative of the phenomena is needed.
 - Not for unrelated variables and for random phenomena (lotteries)
 - Uninformative input variables → poor modeling → Poor learning results
- Causality cannot be inferred from data analysis alone:
 - People in Florida are older(on av.) than in other US states.
 - Florida climate causes people to live longer ?
- May be there is a statistical dependencies for reasons outside the data
- More specifically for ML:
- Powerful models (even for “garbage” data) → higher risk !
- Not-well validated results: the predicted outcome and the interpretation can be misleading.



Bibliographic references (lect 1-2-3)

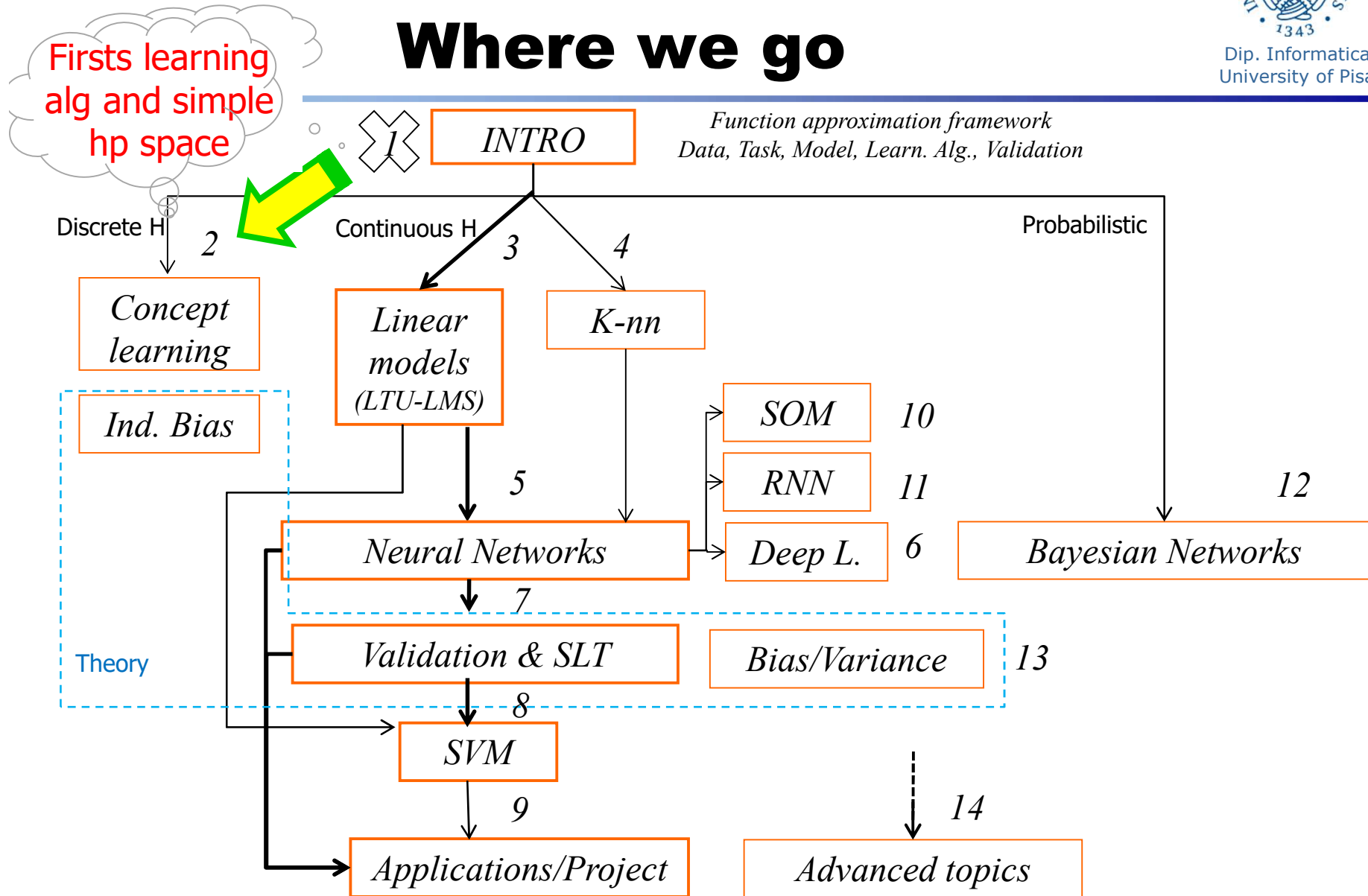
- **Course notes (slides copy): lectures 1-4** without specific textbook materials
 - *Readings* : Mitchell. *The discipline of Machine Learning*. July 2006. CMU-ML-06-108
 - And other in: <http://www.di.unipi.it/~micheli/DID/>
- *On the textbook:*
 - S. **Haykin**: *Neural Networks: a comprehensive foundation*, IEEE Press, 1998. (2nd. Ed.): **sez.1.7** (knowledge rep)
 - **(3rd ed): sez.1.7** (knowledge rep), **1.8, 1.9** (learning processes and tasks)
 - T. M. **Mitchell**, *Machine learning*, McGraw-Hill, 1997: **cap 1**.
- *Other references:*
 - Russell, Norvig: *Intelligenza artificiale (AIMA)*, 2005 (in vol. 2)
 - E.g. background appendix: <http://aima.cs.berkeley.edu/newchapa.pdf>
 - Hastie, Tibshirani, Friedman, *The Elements of Statistical Learning*, Springer Verlag, 2001 (esiste New Ed.): **cap 1** e sez. 7.10
 - Cherkassky, Mulier, *Learning from data : concepts, theory, and methods*, Wiley, 1998 (esiste New Ed.): **cap1 e sez.2.1** (loss e tasks)
 - C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer 2006: **sez. 1.1** (polynomial fitting example)
 - Duda, Hart, Stork, *Pattern Classification*, 2nd. ed. J. Wiley & Sons, 2001: **cap1** (design cycle)

ML Course structure

Where we go



Dip. Informatica
University of Pisa



For information

Alessio Micheli
micheli@di.unipi.it



Dipartimento di Informatica
Università di Pisa - Italy



**Computational Intelligence &
Machine Learning Group**

NP