

# Machine Learning

FIB, Master in Innovation and Research in Informatics

Marta Arias (minor mods. Mario Martin), Computer Science @ UPC

What is Machine Learning

# What is Learning?

A system (living or not) *learns* if it uses past experience to improve future performance

... and so in Machine Learning<sup>1</sup>:

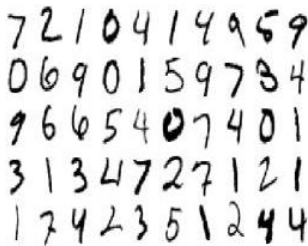
- ▶ “past experience” = data
- ▶ “improve future performance” = make good predictions


---

<sup>1</sup>learning by a *machine* or *computer* or *artificial device*

# What is Machine Learning good for?

An example: digit recognition



Input: image e.g.   
Output: corresponding class label [0..9]

Scenarios where ML can help:

1. Very hard to program yourself but easy to assign *labels*
2. Needs to be kept updated over time
3. Needs to be personalized for many different profiles ...

# Is Machine Learning useful?

## Applications of ML

- ▶ Web search
- ▶ Computational biology
- ▶ Finance
- ▶ E-commerce (recommender systems)
- ▶ Robotics
- ▶ Autonomous driving
- ▶ Fraud detection
- ▶ Information extraction
- ▶ Social networks
- ▶ Debugging
- ▶ Face recognition
- ▶ Credit risk assessment
- ▶ Medical diagnosis
- ▶ ... etc

# Types of ML tasks

- ▶ Supervised learning: uses *labelled* data
  - ▶ **regression**: predict real value label for each example
  - ▶ **classification**: predict discrete value (class, category) for each example
- ▶ Unsupervised learning: *has no labels*
  - ▶ **clustering**: discover homogeneous groups in data
  - ▶ **dimensionality reduction**: find lower-dimensional representation
  - ▶ association rule mining
  - ▶ outlier detection
  - ▶ ...
- ▶ Semi-supervised learning: only few labels
  - ▶ ranking: order examples according to some criterion
- ▶ Reinforcement learning: delayed rewards, learning to act in an environment

# ML in context

ML has strong ties to other disciplines

- ▶ **Statistics:** inferential statistics, distribution and sampling theory, mathematical statistics
  - ▶ in particular, in **multivariate statistics**, often goals and problems are similar
- ▶ **Data Mining:** very large data bases, interest in high-level knowledge
- ▶ **Mathematics:** optimization, numerical methods, asymptotics
- ▶ **Algorithmics:** correctness, complexity, scalability, ...
- ▶ **Artificial Intelligence:** aims at “intelligent” behaviour

# ML in the real world

1. Understand the domain, prior knowledge, goals
2. Data gathering, integration, selection, cleaning, *pre-processing*
3. **Create models from data**
4. Interpret results
5. Consolidate and deploy discovered knowledge
6. ... start again!

# Representing objects

Features or attributes, and target values in supervised learning

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	7.0	3.2	4.7	1.4	versicolor
4	6.1	2.8	4.0	1.3	versicolor
5	6.3	3.3	6.0	2.5	virginica
6	7.2	3.0	5.8	1.6	virginica

- Features or attributes: sepal length, sepal width, petal length, petal width
- Target value (class): species

Main objective in **classification**: predict class from feature values



# Elements of Supervised Learning

1. **Data.** A random sample collected from the problem that we want to model described with a set of attributes and their associated answer
2. **Models.** Description of how data are generated or behave in a general way using a specific language, for instance, logical formulas, mathematical functions or probability distributions.
3. **Learning.** The process by which concrete models are found so that they (1) explains observed data and also (2) can predict unseen data.

## On data

- ▶ Data is tabular: rows are **examples** (objects, instances, or data samples) and columns are the **attributes** (features, ..) describing the examples
- ▶ Features can be numerical (continues range of values) or categorical (discrete set of values)
- ▶ One special column corresponds to the supervised answer (numerical or categorical)
- ▶ So, each example is a  $d$ -dimensional vector  $\mathbf{x}_i$ , and a dataset is a set of labelled examples (input-output pairs):

$$\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$$

- ▶ It is convenient to place all the input features into a matrix  $\mathbf{X} \in \mathbb{R}^{n \times d}$ , and all the labels into a vector  $\mathbf{y}$

## On data pre-processing

Each problem requires a different approach in what concerns data cleaning and preparation. This pre-process can have a **deep impact on performance**; it can easily take a **significant amount of time**

1. Attribute coding (discretization, encoding)
2. Normalization (range, distribution)
3. Missing values (imputation)
4. Outliers
5. Feature selection
6. Feature extraction (feature engineering)
7. Dimensionality reduction and transformations

Non-tabular data (images, audio, text, time-series, graphs, ...) may need ad-hoc treatments and are out of the scope of this course.

# Models

Models are the artifact by which we describe the input data; can be understood as a **compression mechanism** with predictive abilities. They define how the learning is approached. In the course, we focus on two main groups of models: **functions**, and **probability distributions**

1. *Models as functions*, i.e. functions mapping input examples to target values

- ▶  $f : \mathbb{R}^d \rightarrow \{C_1, \dots, C_K\}$  for classification)
- ▶  $f : \mathbb{R}^d \rightarrow \mathbb{R}$  for regression, for example:

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$$

2. *Models as probability distributions* if we assume data come from a stochastic process it may be useful to allow models to represent/quantify uncertainty/noise; instead of only one function we model a distribution over possible functions; a model is then a **multivariate probability distribution**
  - ▶  $p(y|\mathbf{x})$  for *discriminative* models
  - ▶  $p(y, \mathbf{x})$  for *generative* models

# Learning

Learning is the process of finding good models from input data.

good model = a model that predicts well on **unseen data** (this is the *generalization ability* of a model)

Is learning possible?

**Complete the series:** 2, 4, 6, 8, ...

# Inductive bias

A fundamental concept in ML

**Complete the series:** 2, 4, 6, 8, ...

- ▶ Answer 1: **132**; *model 1*:  $f(n) = n^4 - 10n^3 + 35n^2 - 48n + 24$
- ▶ Answer 2: **10**; *model 2*:  $f(n) = 2n$

How can we rule out the more complex one? (and many others)

1. Supply more “training” data: 2, 4, 6, 8, 10, 12, 14, ...
2. Regularize: add a penalty to higher-order terms
3. Reduce the hypothesis space (e.g. restrict to quadratic models)

## Inductive bias (cont.)

*Based on training data only, there is no means of choosing which function  $f$  is better (generalization is not “guaranteed”)*

So ... what do we do?

**Complexity control:** we must add control to the “fitting ability” of our methods

$$\text{true error}(f) \leq \text{training error}(f) + \text{complexity}(f)$$



# On error

True error / generalization error

Suppose that the learning process has a candidate model  $f$ . How can we assess its quality?

We have several *notions* of errors we can compute or at least estimate.

Assume that the input dataset is given by  $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_n)\}$ , then we denote:

- ▶  $\hat{y} = f(\mathbf{x})$  is the *prediction* on object  $\mathbf{x}$  by model  $f$
- ▶ the **error function** (or *loss function*)  $l(y, \hat{y})$  measures how “off” are predictions from true values

We call the true error of a model  $f$  (i.e. its **generalization error**), the expected error<sup>2</sup> that the model will make on a random, possibly unseen example  $(\mathbf{x}, y)$  drawn from distribution  $p(\mathbf{x}, y)$ :

$$E_{true}(f) = \mathbb{E}_{\mathbf{x}, y}[l(y, f(\mathbf{x}))] = \int_{\mathbf{x}, y} l(y, f(\mathbf{x}))p(\mathbf{x}, y)d\mathbf{x}dy$$

---

<sup>2</sup>also called **expected risk**

## On error, cont.

Empirical error / training error

We only see a partial view of the process we are modelling through a **finite dataset**, so we cannot compute the true error directly, and therefore we resort to estimates/approximations for it.

Assuming that the examples are independent and identically distributed (iid), the empirical mean of the loss is a good estimate of the population loss. So we define the **empirical error**<sup>3</sup>:

$$E_{emp}(f, \mathbf{X}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^n l(y_i, f(\mathbf{x}_i))$$

**Empirical risk minimization** is a common learning process by which one finds a model with minimum empirical error. This model is such that will be the one that best explains the input (training) data.

---

<sup>3</sup>also called **empirical risk**

# On error, cont.

## Regularization

The *first law of Machine Learning* states:

Small training error does not imply small generalization error.

Minimizing training error excessively may lead to the famous notion of **overfitting**. This is particularly dangerous for **complex**  $f$ s, so the natural way to fix this is by limiting complexity or **penalizing complexity** (this is called *regularization*).

So, now the learning process should seek an  $f$  that minimizes this empirical risk instead:

$$E_{reg}(f, \mathbf{X}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^n l(y_i, f(\mathbf{x}_i)) + \lambda |f|$$

- ▶ where  $|f|$  stands for some function of the parameters of  $f$  capturing  $f$ 's complexity
- ▶ note the introduction of a new **hyper-parameter**  $\lambda$  which we will need to set appropriately

## On error, cont.

Validation error, a better estimate of true error

Training error underestimates generalization error.

This is because the examples used in the estimate *have been seen by the learning process*, and so memorization could have taken place especially for complex *fs*.

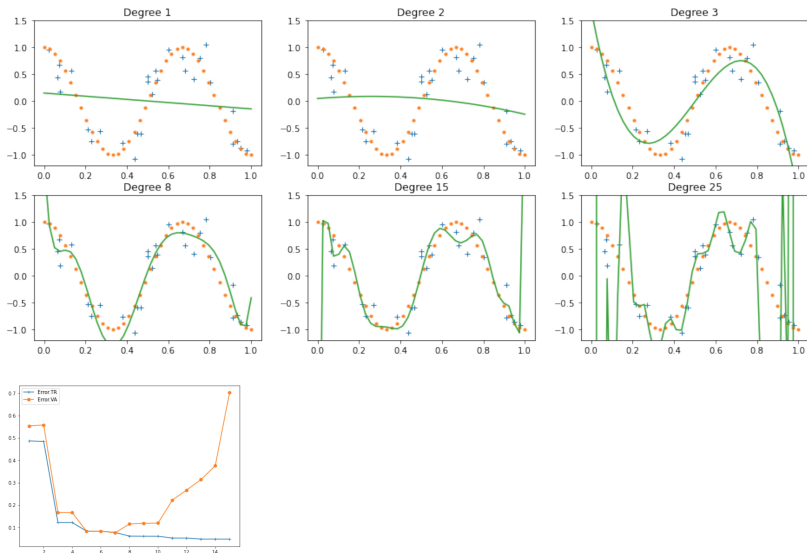
A validation set is used to better estimate true error. The particularity of a validation set is that the learning process **does not have access to it** during learning.

So, generalization error – i.e. quality of model – can be assessed by computing the empirical error on an independent **validation set**.

Fancier methods exist (e.g. cross-validation, loocv, ...); these are known as **resampling methods**) which in general are different ways of sampling input data to better estimate generalization error.

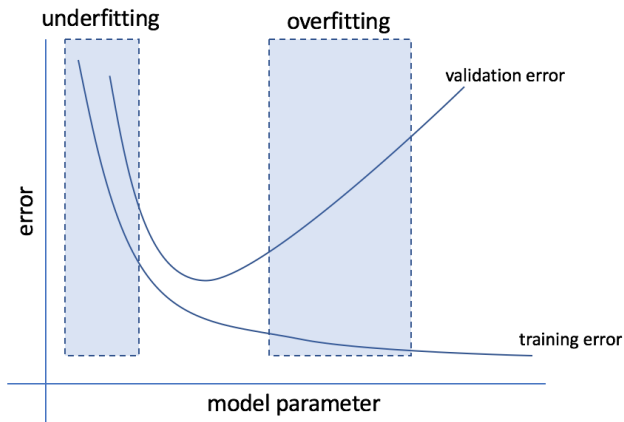
# On error, cont.

Under- and overfitting, unregularized empirical risk minimization



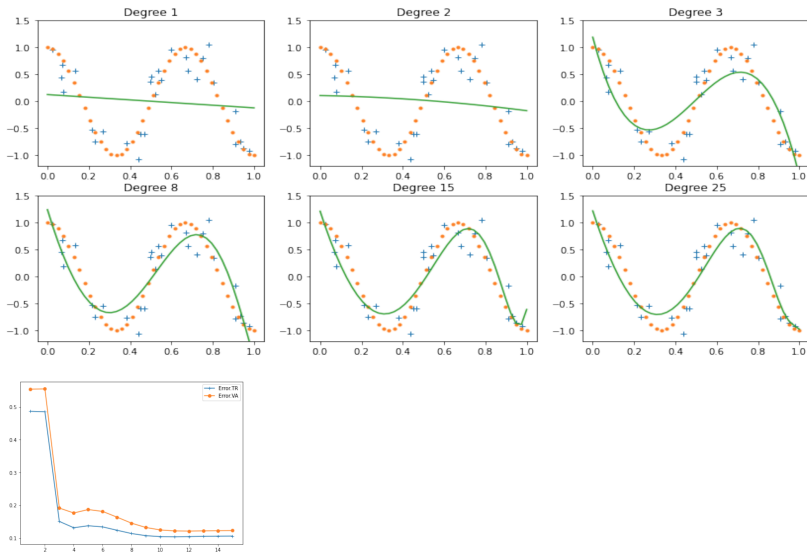
## On error, cont.

Under- and overfitting, unregularized empirical risk minimization



# On error, cont.

Under- and overfitting, regularized empirical risk minimization



## In the next lecture

We will see all of these concepts and more in the context of **linear regression**.

Please refresh concepts from linear algebra, vector calculus, probability theory and statistics.

The book **Mathematics for Machine Learning** contains good coverage of these topics:

- ▶ Eigendecomposition and the SVD (chapter 4)
- ▶ Partial differentiation, gradients of vector-valued functions (chapter 5)
- ▶ Probability and Distributions (chapter 6)



## Some concepts to check

- ▶ Data (tabular form, preprocess,...)
- ▶ Learning and kinds of learning
- ▶ Model: Generative/Discriminative
- ▶ Parameters
- ▶ Inductive bias
- ▶ Loss function
- ▶ Empirical Risk minimization
- ▶ Underfitting / Overfitting
- ▶ Simplicity and Regularization