Machine Learning: Supervised Approaches

Team: Arthur Bit-Monnot, M. Siala,
MJ Huguet, E. Chantery, MV LeLann

Course Objectives

At the end of this module, the student will have understood and be able to explain:

- The **characteristics of supervised learning problems** (data sets, classification / regression, learning process, evaluation of learning models)
- the **main basic methods and algorithms** to deal with these problems (neural networks and interpretable models)

Planning

Courses: 6 CM

- Introduction on AI and supervised learning (1CM – A.Bit-Monnot)
- Learning with Artificial Neural Networks (2CM – A. Bit-Monnot)
- Learning with Interpretable Models (3CM - M. Siala)

Labs (3 TP)

- Perceptron: linear regression & gradient descent (hand-rolled, python)
- Neural Networks (scikit-learn)
- Decision Trees (scikit-learn)

Evaluation: 1 written exam (QCM), lab reports

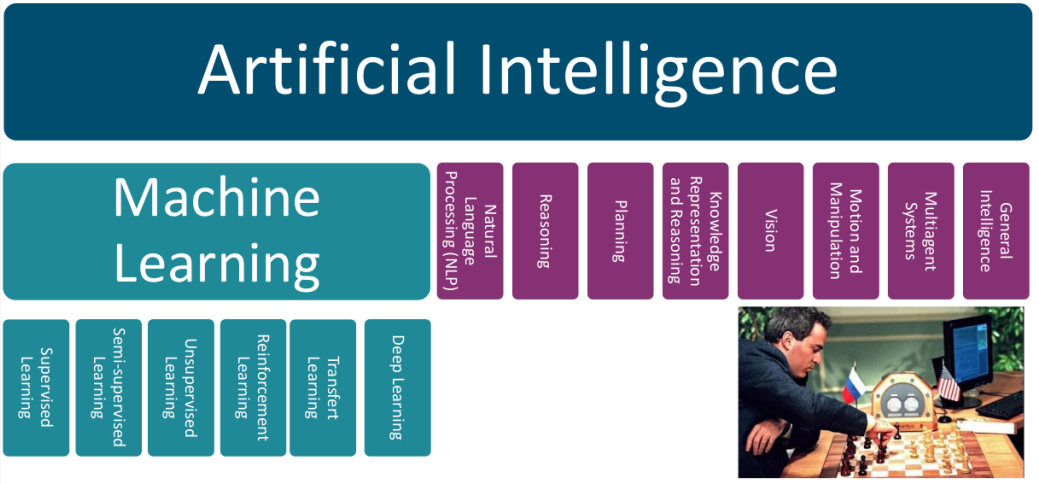
Artificial Intelligence[1]

Systems that think like humans	Systems that think rationally
"The exciting new effort to make computers think . . . *machines with minds,* in the full and literal sense." (Haugeland, 1985)	"The study of mental faculties through the use of computational models." (Chamiak and McDermott, 1985)
"[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . ." (Bellman, 1978)	"The study of the computations that make it possible to perceive, reason, and act." (Winston, 1992)
Systems that act like humans	**Systems that act rationally**
"The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil, 1990)	"Computational Intelligence is the study of the design of intelligent agents." (Poole *et al.*, 1998)
"The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight, 1991)	"AI …is concerned with intelligent behavior in artifacts." (Nilsson, 1998)

Figure 1.1 Some definitions of artificial intelligence, organized into four categories.

[1] From *Artificial Intelligence: A Modern Approach*

Machine Learning a subset of AI



Not all AI systems involve machine learning: Deep Blue (1996) executes the alpha-beta search

Traditional Programming vs ML

Traditional programming:

$$program + data \rightarrow output$$

Machine Learning:

$$data + outputs \rightarrow program$$

"Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed", Arthur Samuel (1959)

Why Machine Learning?

- Reduce time programming
  - a spell checker would require many complex rules
  - rules must be changed for each language
- Solve unprogrammable tasks
  - associating feelings to facial expressions
  - associating meaning to whale songs
- Provide approximations of expensive/untractable computation
  - E.g.: heuristics in graph search (A^*)

Some applications of machine learning

- Computer Vision: object recognition, scene understanding
- Natural Language Processing (NLP): translation, generation (DeepL, ChatGPT)
- Games
- Pattern recognition: email spam detection, fingerprint/face detection and matching
- System control: self-driving cars (Uber, Tesla), automatic control, sort (post office)
- Voice synthesizer, smart assistants (Apple Siri, Amazon Alexa...)
- Finance/industry: cost of living forecasting, stock predictions
- Sports prediction, product recommendation (Netflix, Amazon...)
- Drug design, medical diagnoses (EEC and ECG analysis)

Section 1

Main Categories of Machine Learning

Main Categories of Machine Learning

- Supervised learning
- Unsupervised learning (uncovered)
- Reinforcement learning (uncovered)

Supervised Learning

- Labeled data
- Direct feedback
- Predict outcome/future

Input: a set of examples annotated/labeled with the expected output of the function to learn.

Goal: an algorithm/model that, for a new example x^*, can predict from the correct value y^*.

Example (spam detector): after training the system with a set of email explicitly labeled as *spam* or *not-spam*, the system should correctly classify new emails as spam or not.

Unsupervised learning (uncovered)

- No labels/targets
- No feedback
- Find hidden structure in data

Input: unlabeled data, targets are unknown

Goal: group the data $x_1, x_2, x_3, \ldots$ by similarity and discover the relationship with structural hidden variables

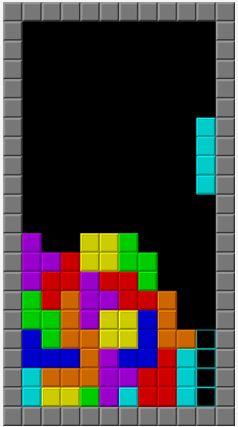
Example: recommendation systems, similarity detection, customer segmentation, data-set labeling.

Reinforcement learning (uncovered)

Learns a policy: association of each possible state of the system to an action to take.

Input: The learning system is getting *sparse (delayed) rewards* for the action it takes.
Goal: produce a policy that maximizes long-term reward.

Examples: computer games, motion control



Section 2

Data

Some ML Terminology

Running example: house prices

- **Label**: are what we are trying to predict
  - the price at which the house was or will be sold
- **Feature:** (aka attribute/variable) an input of the learned function
  - number of rooms
  - surface
  - age of the building
- **Example:** (aka sample, instance) is an instance of the data that combines the set of attributes with a label

Dataset

Features			Label
m^2	Num Rooms	Floor	Price (€)
24	1	4	102 000
46	3	2	140 000
50	3	6	353 600
211	5	3	892 000
74	3	1	198 000

Labeled examples of apartment prices[2]

[2] Source: leboncoin.fr

Quiz: Iris recognition

sepal_length	sepal_width	petal_length	petal_width	Iris_class
5	2	3.5	1	versicolor
6	2.2	4	1	versicolor
6.2	2.2	4.5	1.5	versicolor
6	2.2	5	1.5	virginica
4.5	2.3	1.3	0.3	setosa
5.5	2.3	4	1.3	versicolor
6.3	2.3	4.4	1.3	versicolor
5	2.3	3.3	1	versicolor
4.9	2.4	3.3	1	versicolor
5.5	2.4	3.8	1.1	versicolor
5.5	2.4	3.7	1	versicolor
5.6	2.5	3.9	1.1	versicolor
6.3	2.5	4.9	1.5	versicolor
5.5	2.5	4	1.3	versicolor
5.1	2.5	3	1.1	versicolor
4.9	2.5	4.5	1.7	virginica
6.7	2.5	5.8	1.8	virginica
5.7	2.5	5	2	virginica
6.3	2.5	5	1.9	virginica
5.7	2.6	3.5	1	versicolor
5.5	2.6	4.4	1.2	versicolor
5.8	2.6	4	1.2	versicolor

What does the purple/red/green boxes represent:

- an example
- several examples
- an attribute
- a label

Quiz:

Suppose you want to develop a supervised ML model to predict whether a given email is “spam” or “not spam”. Which of the following statements is true?

1. Word in the subject header will make good labels
2. Unlabeled examples will be used to train the model
3. Emails not marked as “spam” or “not spam” are unlabeled examples
4. By hypothesis, all the labels applied to examples are reliable

Essential Categories of data

- Quantitative/Numerical data
  - Can be counted, data are exact numbers, but they are not ordered
  - Ex: house prices, speed, frequency
  - Can be continuous (temperature, speed) or discrete/binary (number of cycles)

- Qualitative/Categorical data
  - Can't be counted, represents characteristics
  - Ex: gender, color, team
  - Can take numerical values that do not have mathematical meaning
  - Can be nominal (not ordered, ex: gender, color) or ordinal (small<medium<large)

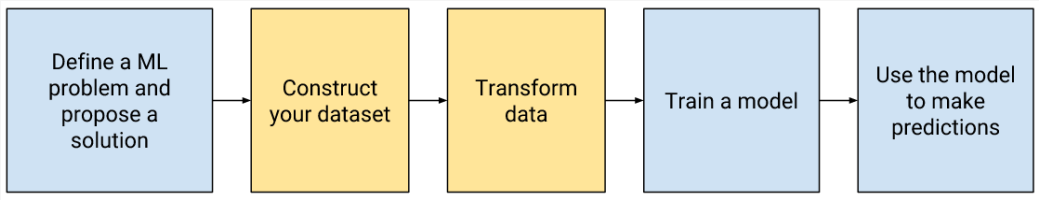
Other categories of data

- Time series data
  - A sequence of numbers collected at regular intervals over some period of time
  - Values are ordered: there is a first data point and a last data point collected
  - Ex: the voltage value during 10 sec
- Text
- Video
- Audio
- Image

Typically handled with:

- translation into quantitative data,
- with specialized learning models (e.g. convolutional neural network structure for images)

The ML workflow: data-centric



- Construct your dataset
  - Collect raw data
  - Identify feature and label sources
  - Select sampling strategy
  - Split the data
- Transform data
  - Explore and clean the data set
  - Feature engineering

(Note: typical process but may differ)

Dataset size

Data set	Size (number of examples)
Iris flower data set	150 (total set)
MovieLens (the 20M data set)	20,000,263 (total set)
Google Gmail SmartReply	238,000,000 (training set)
Google Books Ngram	468,000,000,000 (total set)
Google Translate	trillions

Rule of thumb: your model should train on at least an order of magnitude more examples than trainable parameters.

Data quality: Garbage-in, garbage-out

Google Translate

The Google Translate team has more training data than they can use. Rather than tuning their model, the team has earned bigger wins by using the best features in their data.

"...one of our most impactful quality advances since neural machine translation has been in identifying the best subset of our training data to use"

- Software Engineer, Google Translate

Several (uncovered) techniques to clean up the data:

- select/prune features
- remove outliers/noisy data

Section 3

Supervised learning

Supervised learning

We are given a **training set** of N examples input-output pairs

$$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$$

where each pair was generated by an unknown function f:

$$y = f(x)$$

Objective: discover a function h, the **hypothesis** that approximates the true function f.

Running example

Let say that I want to sell my apartment in the center of Toulouse.

My apartment[3] has:

- 165 m^2
- 4 rooms
- on the 6th floor

Question: what is the market price for such an apartment?

[3] completly fictional, I am just an associate professor

Dataset

I look at several apartments on sell in the neighborhood and come up with the following dataset:[4]

X			Y
m^2	Num Rooms	Floor	Price (€)
24	1	4	102 000
46	3	2	140 000
50	3	6	353 600
211	5	3	892 000
74	3	1	198 000

[4] Source: leboncoin.fr

Vocabulary

Each example i has 3 **features** (m^2, #rooms, floor)

$$x_i = [x_{i,1}, x_{i_2}, x_{i,3}]$$

and associated **ground truth** y_i.

The *unknown* function f associates each example with its ground truth:

$$f(x_i) = y_i$$

For instance: $f([24, 1, 4]) = 102\text{k€}$

I want to know the market price for my apartment: $f([165, 4, 6]) = \ ??$

Hypothesis space

I want to **learn** a function h that closely approximates f.

$$\text{i.e.} \quad h(x) \approx f(x), \forall x \qquad \text{(more complex in practice)}$$

Among the set of all possible functions $\mathcal{H}$, I want to choose the one that has the least different behavior from f:

$$h^* = \underset{h \in \mathcal{H}}{\arg\min} \ \textit{diff}(h, f)$$

$\mathcal{H}$, the set of all possible functions, is called the **hypothesis space**.

What's a suitable hypothesis space?

The set of possible python functions:

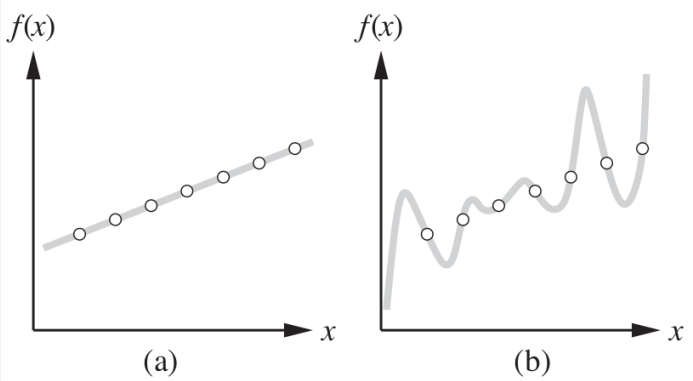
```
def h(sqm, nroom, floor):
    price = 4000 * sqm + 10000 * nrooms
    if floor == 1:
        price -= 30000
    return price
```

The set of linear functions:

$$h(sqm, nrooms, floor) = 4000 * sqm + 10000 * nrooms + 10000 * floor$$

. . . or the set of possible decision trees.

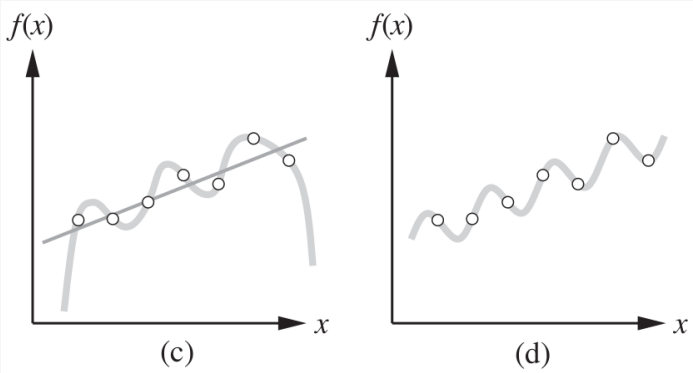
Hypothesis: Case study 1



A set of examples $(x, f(x))$ pairs, consistent with a linear hypothesis (a) and degree-7 polynomial hypothesis (b).

Note: A **consistent hypothesis** exactly matches the data.

Hypothesis: Case study 2



A set of examples $(x, f(x))$ pairs, consistent with a degree-6 polynomial hypothesis (c) and simple sinusoidal hypothesis (d).

What's a suitable hypothesis space?

Each hypothesis space has its own characteristics:

- **bias:** tendency to underfit the data.
  - linear function strongly limit the possible hypotheses which could result in a failure to fit the data
- **variance**: tendency to overfit the data
  - python is turing complete and could be made to produce exactly the ground truth for each example

Rule of thumb:

- simple model:
  - high-bias, low variance / poor-fit but generalizes well
- complex model:
  - low-bias, high variance / great fit but generalizes poorly
- (Deep) neural network: complex model that (sometimes) generalizes well

Section 4

Evaluating hypothesis: Loss functions

What's a good hypothesis in $\mathcal{H}$?

Given a prediction

$$\hat{y} = h(x)$$

The **loss function** measures how bad it is to have the prediction $\hat{y}$ instead of the true value y for the example x.

$$L(x, y, \hat{y})$$

It is often stated independently of x: $L(y, \hat{y})$

Intuitively, our objective is to **minimize the average loss** of our predictions.

Some common loss functions: error minimization

Absolute-value loss	$L_1(y, \hat{y}) = \lvert y - \hat{y} \rvert$
Squared-error loss	$L_2(y, \hat{y}) = (y - \hat{y})^2$
0/1 loss	$L_{0/1}(y, \hat{y}) = 0$ if $y = \hat{y}$, else 1

Note that for any (well-formed) loss function:

$$L(y, y) = 0$$

i.e., nothing is lost if the prediction is perfect.

- Question: which losses are adapted for regression? For classification?

Loss: beyond error minimization

I am selling my apartment and will decide the price at which I sell based on a learned function.

Hypothesis:

- the apartment will not be sold above its *true price*
- if the price is lesser than or equal to the *true price* it will be sold immediately
- if the apartment is not sold for 1 week, I will decrease the price by 10k€
- I evaluate my prejudice for a 1 week delay to 2000€

Exercise: What is the loss function that I should minimize (in order to maximize my profits)?

Evaluating a hypothesis: the perfect measure

An agent should choose the hypothesis that minimizes the expected loss over all input-output pairs it **will** see.

Let $\mathcal{E}$ be the set of all possible examples and $P(X, Y)$ be a probability distribution over examples.

We can define the **generalization loss** for a hypothesis h and a loss function L:

$$GenLoss_L(h) = \sum_{(x,y)\in\mathcal{E}} L(y, h(x)) \times P(x, y)$$

The best hypothesis is the one with the minimum expected generalization loss:

$$h^* = \arg\min_{h\in\mathcal{H}} GenLoss_l(h)$$

Evaluating a hypothesis: the empirical measure

Problem: $P(X, Y)$ is usually unknown. Instead we only have a set of examples E.

The **empirical loss** is an estimate of the generalization loss on a set of examples E:

$$EmpLoss_{L,E}(h) = \sum_{(x,y)\in E} L(y, h(x)) \times \frac{1}{|E|}$$

The estimated best hypothesis $\hat{h}^*$ is the one with the minimum empirical loss:

$$\hat{h}^* = \arg\min_{h\in\mathcal{H}} EmpLoss_{L,E}(h)$$

Quiz: hypothesis selection

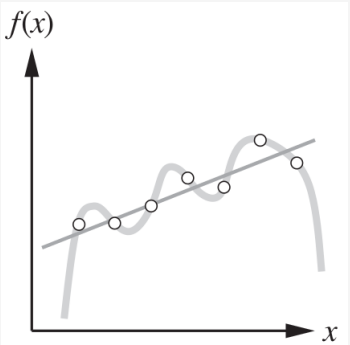
Which has the best empirical loss?

1. linear hypothesis
2. polynomial hypothesis

Which would you be more confident in using for prediction?

1. linear hypothesis
2. polynomial hypothesis

What is the problem with our methodology?



Section 5

Model evaluation & validation

Model validation/evaluation

Suppose we must choose between two possible ways to fit some data. How do we choose between them?

- Naïve solution: pick the one that best fits the data.
  Problem: generalization to new measurements?
- Correct solution: evaluate models by on a new (unseen) data set (the "test set"), distinct from the training set.

Several methodologies:

1. "test-set validation" or "hold-out validation"
2. "k-fold cross-validation"
3. "leave-one-out cross-validation" (LOOCV).

"hold-out validation"

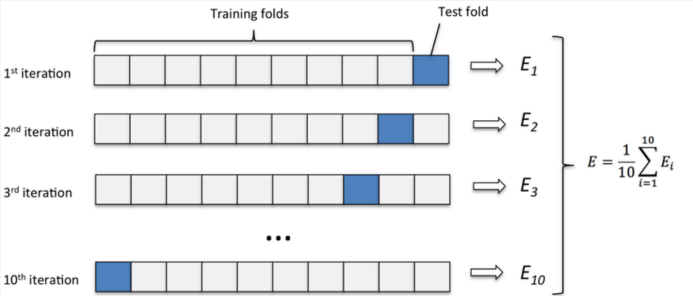
The simplest method:

1. Partition the data randomly into a training set (usually $> 60\%$) and a validation set (hold-out set)
2. For a set of chosen values for hyperparameters[5], learn a model on the training set
3. Compute the model's error on the validation set (see metrics)
4. Pick the best hyperparameter which has the smallest validation set error and retrain the model

[5] Parameters of learning method. each combination of hyperparameters will result in a separate hypothesis.

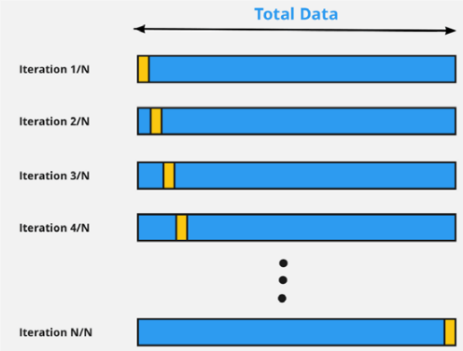
k-fold cross-validation

1. Randomly partition the training data into K sets of equal size
2. Run the learning algorithm K times: each time, a different one of the K sets is deemed the test set, and the model is trained on the remaining K-1 sets
3. The hyperparameter score is the average of the error across the K tests
4. Pick the best hyperparameters and retrain the model



Leave-One-Out Cross-Validation (LOOCV)

K-fold cross validation with $K = M - 1$, with M the number of data points



Kinds of datasets

- **Training Dataset**: The sample of data used to fit (train) the model.
- **Validation Dataset**: The sample of data used select among several hypothesis. Used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. The evaluation becomes more biased as skill on the validation dataset is incorporated into the model configuration.
- **Test Dataset**: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

Data management rules

1. Never evaluate a model on data that be used for its training and/or selection
2. The data on which the model is trained and evaluated must be representative of the data on which it will be exploited.