Overview of Machine Learning and Pattern Recognition

Dipartimento di Automatica e Informatica
Politecnico di Torino, Torino, ITALY

Outline of this lecture(s)

- Pattern Recognition and Machine Learning: definitions and main concepts
- The classification paradigm
- Evaluating classification/prediction performance
- Feature Reduction
- Supervised/Unsupervised learning
  (overview and examples of approaches)
- Neural networks and Deep learning
  (overview and examples)

1. Introduction

Data provided by sequencing

- Genes/TFs/miRNAs/short or long non coding RNAs Expression
- Mutations (single base or group of bases)
- CNVs
- miRNAs and isomiRNAs
- miRNA/non-coding RNAs targets
- Binding sites for TFs, regulative proteins, regulative non coding RNAs
- Gene fusions
- Information about fusions structural mechanisms
- Epigenetics, conserved DNA regions
- Virus DNA/RNA damages
- Regulatory networks
- Proteins expression, networks etc.

Machine learning and sequencing

- These data can be used as input of classifiers in order to
  - Discovery drivers, features, very important molecular players or very relevant molecular interactions
  - Build living systems models for simulations
  - Build predictive models for pathology diagnosis, prognosis and therapy

A doll inside a doll

Computer Science

Artificial Intelligence

Machine Learning

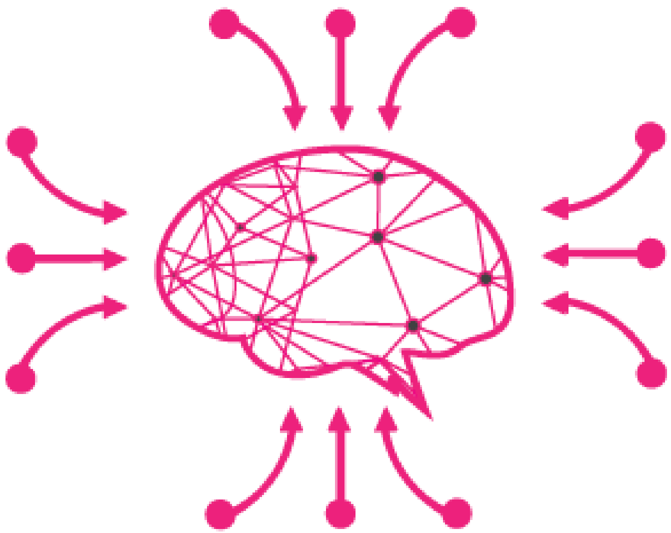
Pattern Recognition

Artificial Intelligence

- "**Intelligence**" (i.e. cognitive abilities typically associated to human mind, such as learning, making decisions, problem solving, etc.) exibited by a machine
- Branch of computer science aimed at developing artificial agents able to
  - **Perceive** the environment (i.e. receive data from the external world)
  - **Take actions** maximising the chance of success at some goal

Machine Learning

- Subfield of artificial intelligence that gives computers the ability to **learn** how to solve a specific problem without being explicitly programmed for it
- **Learning** = Deriving knowledge from experimental data



Pattern Recognition

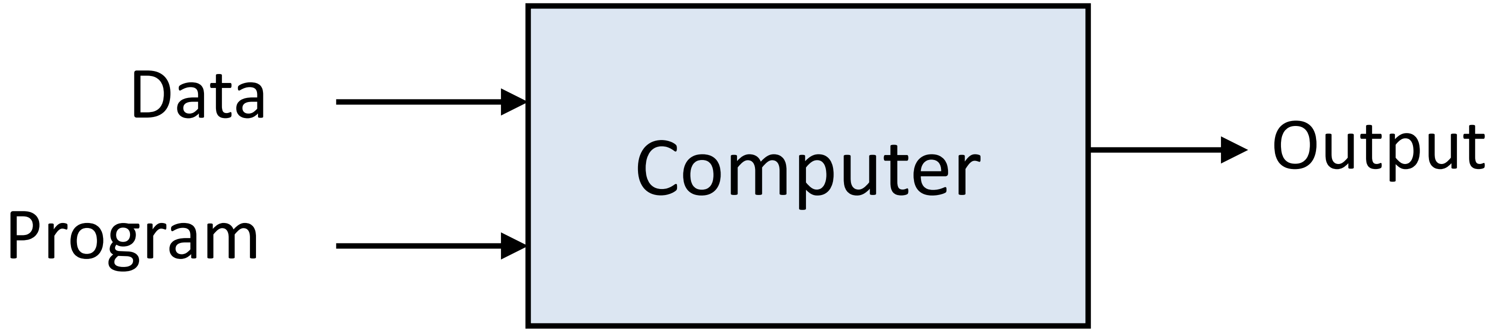
- Most of the times it is used as a **synonym** of machine learning
- More precisely, it is a branch of machine learning that focuses on the recognition of **patterns** and regularities in data
- **Pattern** = Any kind of discernible regularity in a set of data, whose elements repeat in a predictable manner
- Examples:
  - Specific **patterns of DNA sequences** (genes, protein coding regions, promoters, etc.) uncover functional aspects of cells
  - Specific **visual patterns of biological images** allow to identify cell types, unveil cancer, etc.

What is Machine Learning?

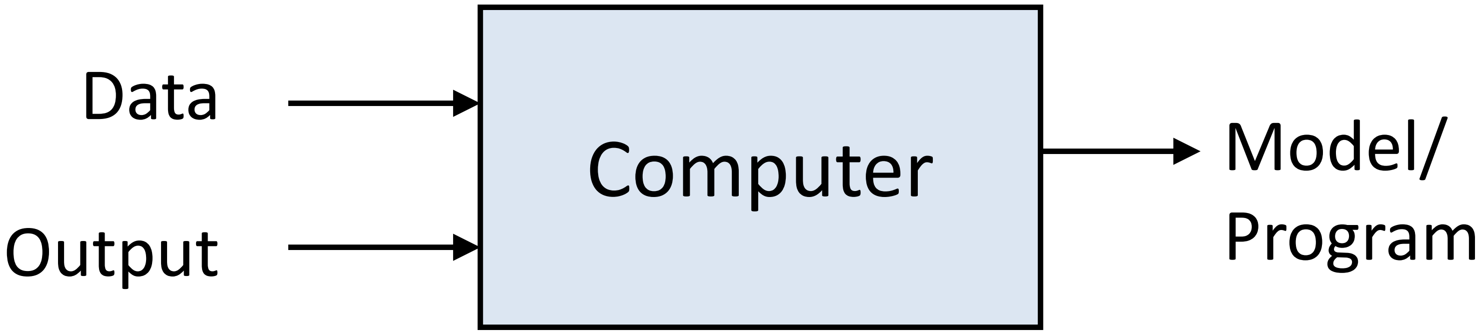
- Machine Learning: study of algorithms that improve their performance at some task with experience
- System that can **continuously self-improve** and thereby offer **increased** efficiency and **effectiveness by learning from experience**, analytical observation, and other means.
- It's a change of paradigm: Knowledge-based vs. Learning Systems

A change of paradigm

Traditional Programming



Machine Learning



Knowledge-based vs. Learning Systems

- **Knowledge-based Systems**: Acquisition and modeling of common-sense knowledge and expert knowledge
  ⇒ limited to given knowledge base and rule set

  ⇒ Inference: Deduction generates no new knowledge but makes implicitly given knowledge explicit

  ⇒ Top-Down: from rules to facts

- **Learning Systems**: Extraction of knowledge and rules from examples/experience

  ⇒ Learning as inductive process

  ⇒ **Bottom-Up: from facts to rules**

Why "Learn"?

- Learning is necessary when:
  - Human expertise does not exist
  - Humans are unable to explain their expertise
  - Human expertise exists, but it is unreliable (e.g. result may be affected by subjectivity)
  - Human expertise exists, but it is unfeasible (e.g. too many data to process, or too costly)
  - Solution may change in time
  - Solution may need to be adapted to particular cases
- Generally, data is cheap and abundant, while expertise is expensive and scarce

Growth of Machine Learning

- Machine learning is the preferred approach to
  - Speech recognition, Natural language processing
  - Computer vision
  - Robot control
  - Medical outcomes analysis
  - Computational biology
- This trend is accelerating
  - Improved machine learning algorithms
  - Improved data capture, networking, faster computers
  - Demand for self-customization to user
  - It turns out to be difficult to extract knowledge from human experts → failure of expert systems in the 1980's.

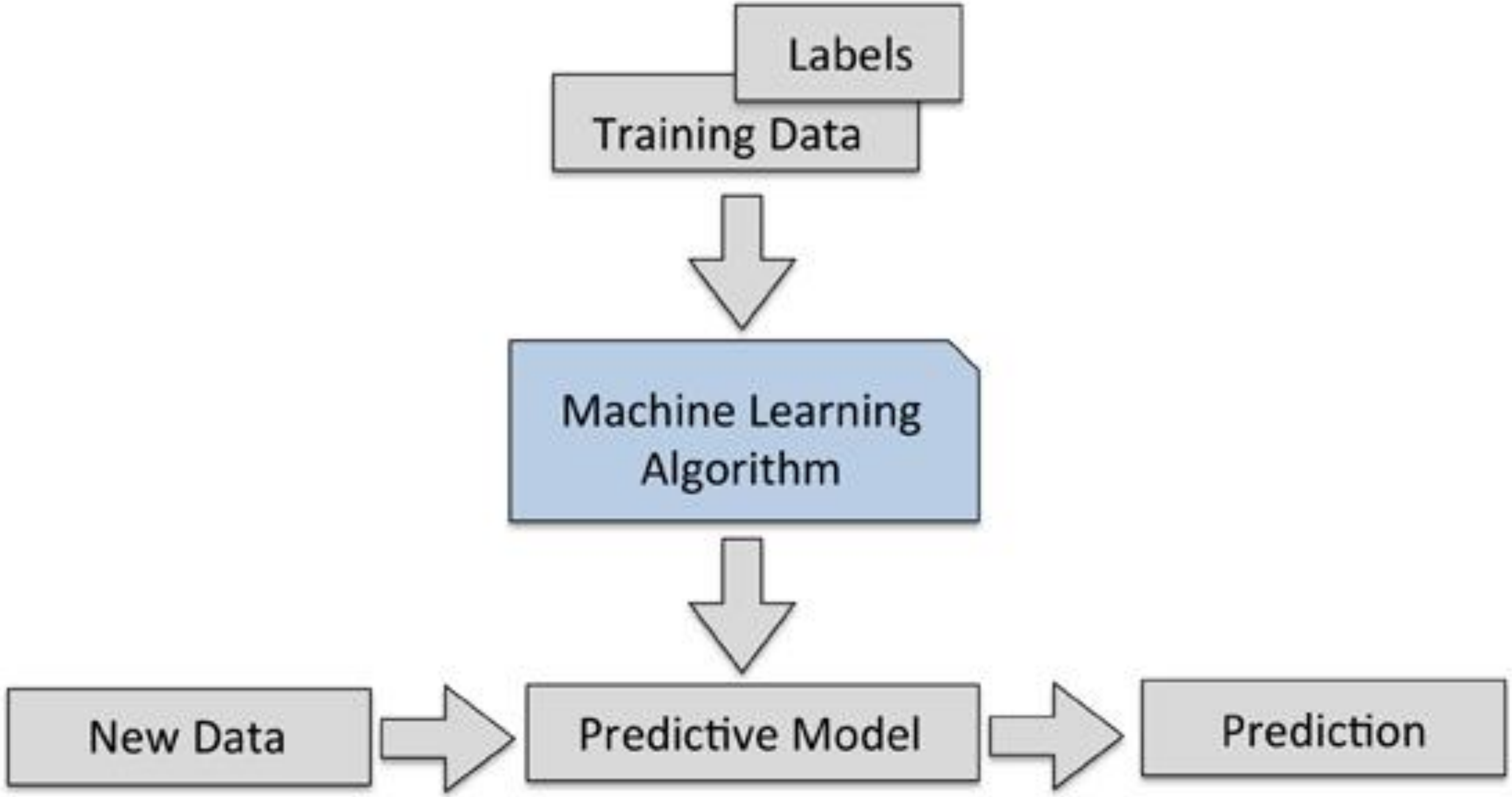
Types of machine learning

Supervised Learning

Unsupervised Learning

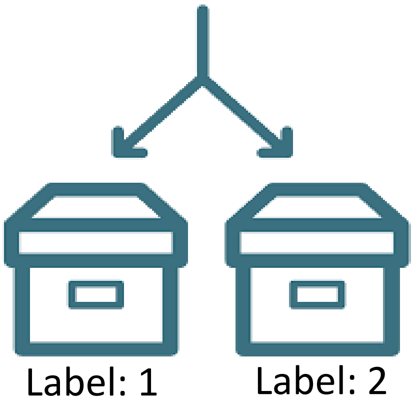
Reinforcement Learning

Supervised learning

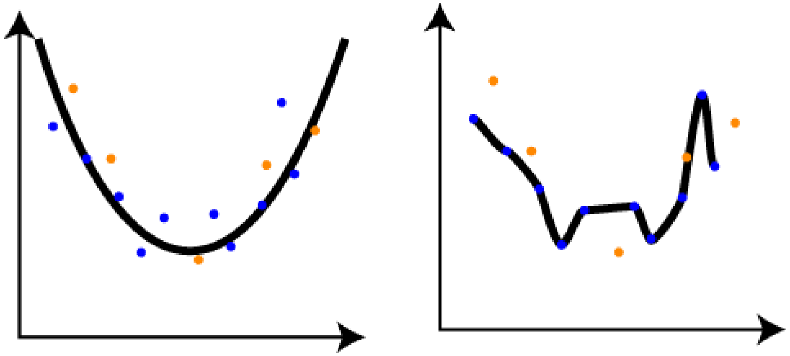


Two types of supervised learning

- Classification



- Regression

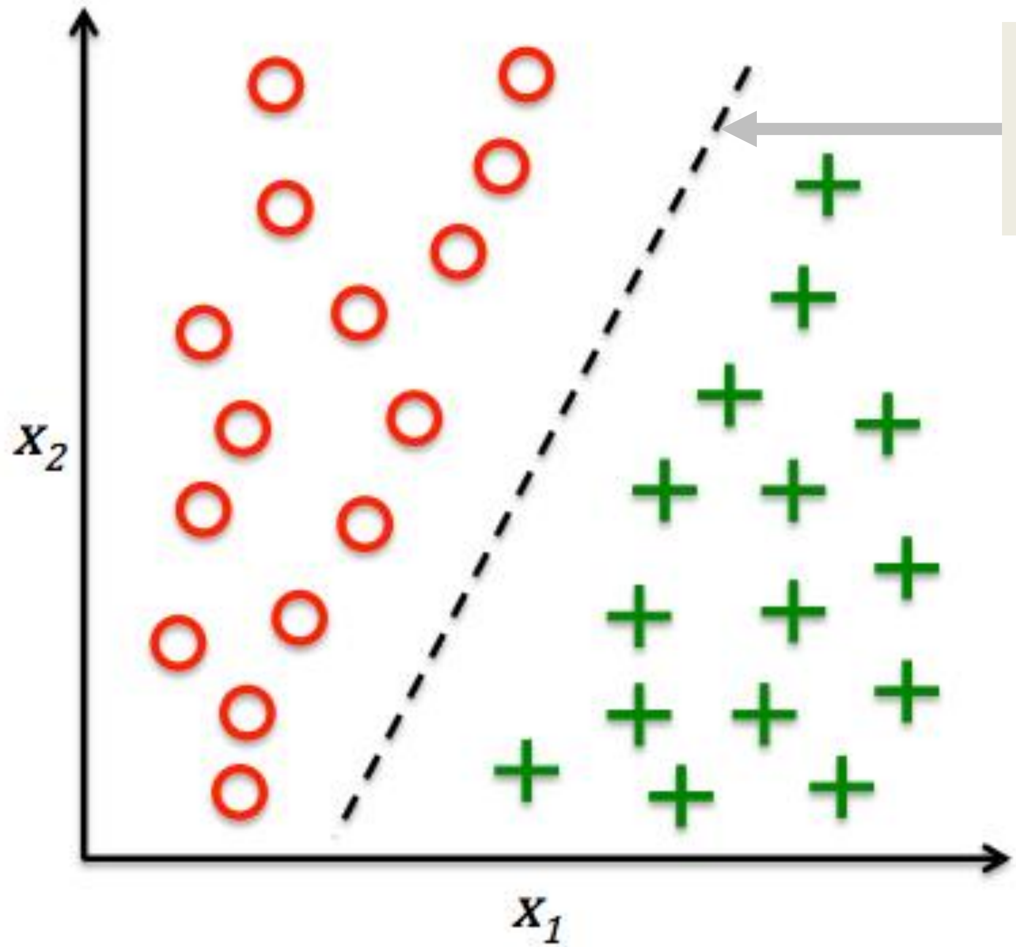


Classification

- A subcategory of supervised learning where the goal is to predict the **categorical class labels** of new instances, based on a training set of past observations
- **Class labels:** discrete, unordered values representing the *group memberships* of the instances
- **Traning set**: A set of instances with known class labels, based on which the classifier learns a classification strategy than can be applied to new unlabelled instances

- **Binary classification** → two classes
- **Multi-class classification** → more than two classes

Binary classification



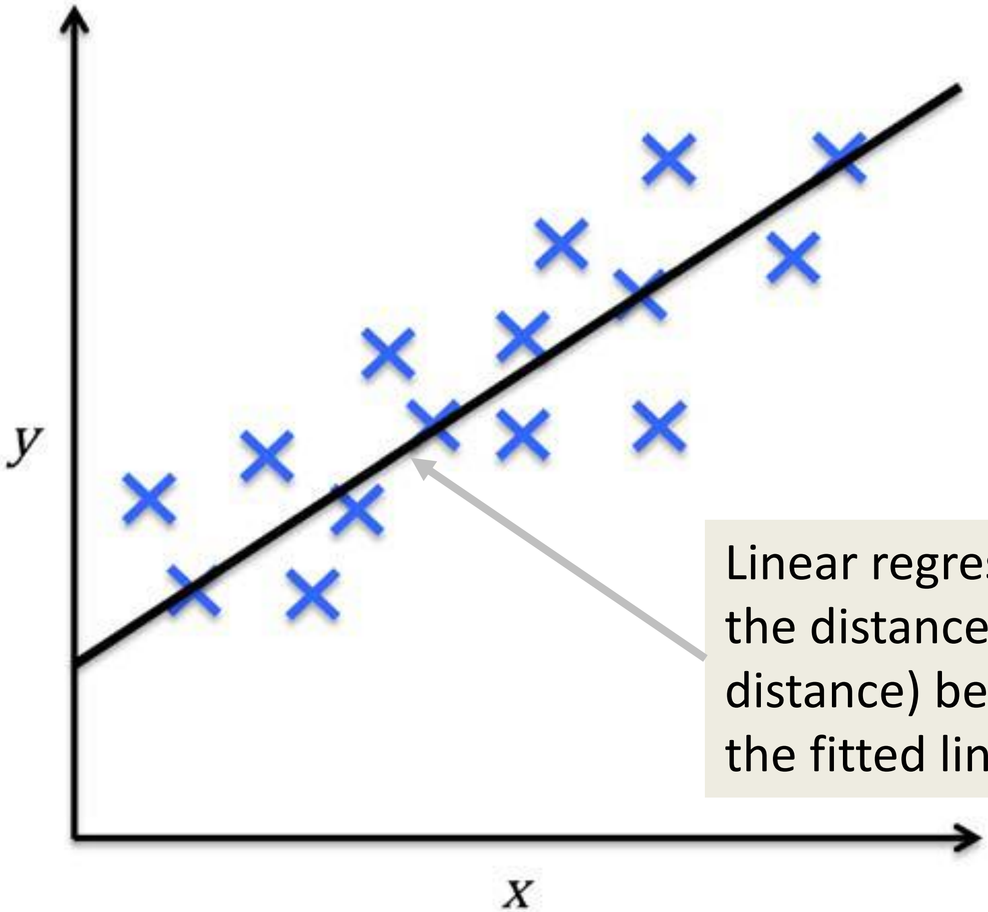
Learnt classification model:
DECISION BOUNDARY

+: Positive class
O: Negative class

Regression

- Differently from classification, that assigns categorical class labels to the instances, the aim of **regression analysis** is prediction of **continuous outcomes**
- Given a number of ***predictor*** (explanatory) variables and a continuous response variable (**outcome**), regression analysis tries to find a mathematical relationship between those variables, which can be used to predict the outcome with a resonable level of approximation.

Example: linear regression

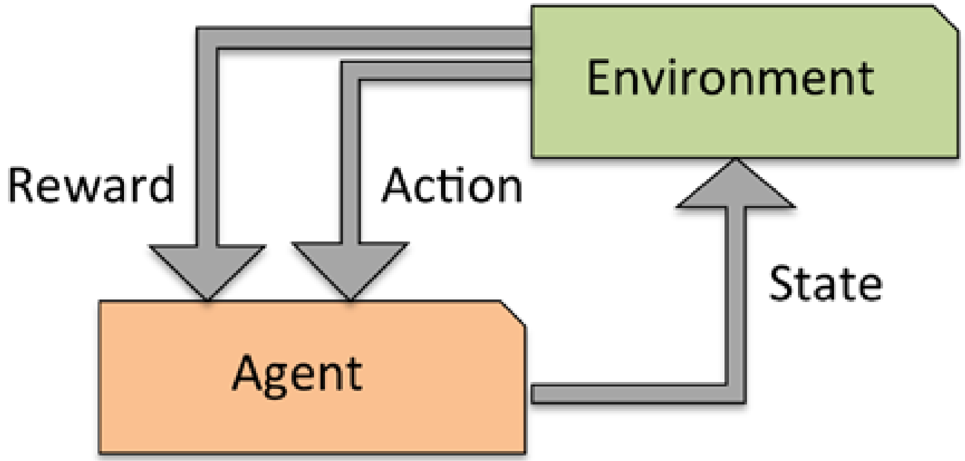


- x: predictor variable
- y: outcome (response variable)

Linear regression: find the line that minimizes the distance (typically, the mean squared distance) between all the sample points and the fitted line

Reinforcement learning

- The goal is to develop a system (***agent***) that improves its performance based on interactions with the *environment*.
- The agent takes an action on each time step and receives a **reward** value, which is a measure of how good the action is towards the goal to be achieved



Reinforcement learning

- Through the interaction with the environment, the agent learns a series of actions that maximizes this reward, typically via an **exploratory** trial-and-error approach.
- Most popular example: *chess engine*.

The agent decides upon a set of possible moves based on the current state of the board (environment) and the reward (win or lose at the end of the game)

Chess example (1)

- Let's say you start with a chess board set up for the start of a game. Each player has 16 pieces. Let's say that white starts. White has 20 possible moves:
  - The white player can move any pawn forward one or two positions.
  - The white player can move either knight in two different ways.
- The white player chooses one of those 20 moves and makes it.
- For the black player, the options are the same: 20 possible moves. So black chooses a move.

Chess example (2)

- In a world of "all possible moves," the program makes a big tree for all of those moves
- The total number of board positions is 10^{120}
- No computer is ever going to calculate the entire tree. What a chess computer tries to do is generate the board-position 5 or 10 or 20 moves into the future (e.g. a five-level tree contains 3,200,000 board positions)
- **Once it generates the tree**, then the computer needs to "evaluate the board positions." That is, **the computer has to** look at the pieces on the board and **decide whether that arrangement of pieces is "good" or "bad"**
- The way it does this is by using an **evaluation function**. The simplest possible function might just count the number of pieces each side has, and then compute the difference.

Chess example (3) –Evaluation function

- The previous formula can be more complicated by applying a weight to each type of piece.
- The evaluation function becomes more and more complicated by adding things like
  - board position,
  - control of the center,
  - vulnerability of the king to check,
  - vulnerability of the opponent's queen, and tons of other parameters.
- No matter how complicated the function gets, however, it is condensed down to a single number that represents the "goodness" of that board position.

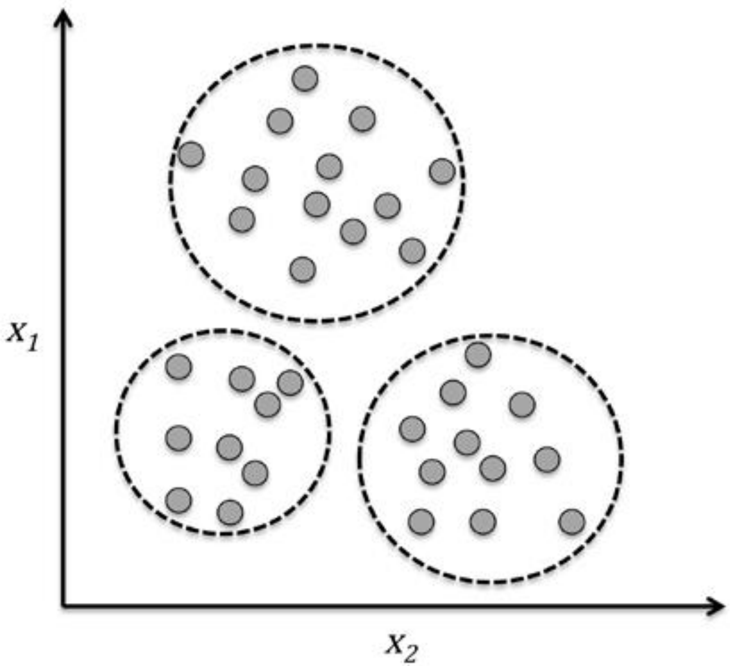
Reinforcement Learning

- Differently from supervised learning, it does not build a system based on a training set with apriori known outcomes. It just works towards the maximization of the cumulative reward.
- The sub-optimal actions (i.e. actions with negative reward) are not necessarily to be avoided: **exploration–exploitation tradeoff**
- Main application area:
  - game theory
  - robotics

Unsupervised Learning

- Deals with unlabeled data or data of *unknown structure*.
- The algorithm explores the inherent structure of the data, without the guidance of a labelled training set or reward function
- Typical applications :
  - **Clustering** (a.k.a. unsupervised classification)
  - **Dimensionality reduction**

Clustering



The instances are partitioned into a number of classes (clusters) based on the

- Maximization of similarity of instances of the same cluster
- Minimization of similarity of instances of different cluters

Examples of applications in bioinformatics:

- Exploration of recurrent sequence motifs
- Discrimination of tissue types in biological images
- Automatic summarization

Dimensionality reduction

- Data of high dimensionality—each observation comes with a high number of measurements— can present a challenge
  - limited storage space
  - computational performance of machine learning algorithms
- Dimensionality reduction projects the input instances into a new lower-dimensional space, in order to
  - Remove noise
  - Retain the most relevant information
- Most of the times, it is a pre-processing step

In a nutshell…

SUPERVISED	REINFORCEMENT	UNSUPERVISED
• Builds a classifier based on *input* and *output* data • Classifier is trained with a training set of data • Classifier is tested with a test set of data • Deployment if the *output* is satisfactory	• The algorithm presents a *state* and takes an *action* based on the *input* data • The action is rewarded or punished • The algorithm learns from the reward/punishment and updates itself, this continues	• Builds an algorithm based on *input* data • That algorithm is tested with a test set of data (in which the algorithm creates the classifier) • Deployment if the *classifier* is satisfactory

In a nutshell…

SUPERVISED

- Builds a classifier bas…
- …d …set of data
- Deployment if the *output* is satisfactory

"I know how to classify this data, I just need you (the classifier) to do it for me."

REINFORCEMENT

- The algorithm presents a *state* and takes an *action* based on the *input* data

The a…

"I have no idea how to classify this data, can you classify this data and I'll give you a reward if it's correct or I'll punish you if it's not."

UNSUPERVISED

- Builds an alg… b…
- …
- D… the class… …s satisfactory

"I have no idea how to classify this data, can you (the algorithm) create a classifier for me?"

Resources: Journals

- Journal of Machine Learning Research www.jmlr.org
- Machine Learning
- IEEE Transactions on Neural Networks
- IEEE Transactions on Pattern Analysis and Machine Intelligence
- Annals of Statistics
- Journal of the American Statistical Association
- Pattern Recognition
- …

Resources: Conferences

- International Conference on Machine Learning (ICML)
- European Conference on Machine Learning (ECML)
- Neural Information Processing Systems (NIPS)
- Computational Learning
- International Joint Conference on Artificial Intelligence (IJCAI)
- ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)
- IEEE Int. Conf. on Data Mining (ICDM)
- …

Resources: Software

Matlab: Easy to learn, very powerful and comprehensive, you can find nearly every high-level function you need and put them together to satisfy your needs. The speed is optimized if you use vectorized and matrix computation.

Weka (Java): Minimal programming skill required if you use GUI. Also Java is a powerful language for writing your own algorithms. Details (like input, output) can be handled easily.

R: Free and powerful. More powerful than Matlab if you are doing statistical modeling, but inferior in its general toolbox.

Python: There are libraries like *sklearn, numpy, pandas, and scipy* making python competent to do all kinds of scientific computing with its rich and powerful language features.