Introduction to Machine Learning

Guillaume Tochon & Joseph Chazalon

LRE

Slides generated on Monday 24th February, 2025 at 19:57

Today's agenda

Motivation

What is learning?

Interviewer: What's your biggest strength?
Me: I'm an expert in machine learning.
Inteviewer: What's 6 + 10?
Me: Zero.
Interviewer: Nowhere near, it's 16.
Me: It's 16.
Interviewer: Ok... What's 10 + 20?
Me: It's 16.

Definition

Learning: Improve over experience to perform better in new situations.

Quoting S. Bengio

Learning is **not** learning by heart.
Any computer can learn by heart.
The difficulty is to **generalize** a behavior to a novel situation.

A new **science** with a goal and an object

Machine Learning Science

How can we **build computer systems** that automatically improve with experience, and **what** are the **fundamental laws** that govern all learning processes?

— *Tom Mitchell, 2006*

What is it good for?
According to Peter Norvig

The **3 main reasons** why you may want to use Machine Learning:

→ Avoid coding **numerous complex rules** by hand.
lower cost, more effective, faster reaction to changing problem

→ **Optimize the parameters** of your system given a dataset of yours.
better accuracy

→ Create systems for which you **do not know the rules consciously** (e.g. recognize a face).
greater potential

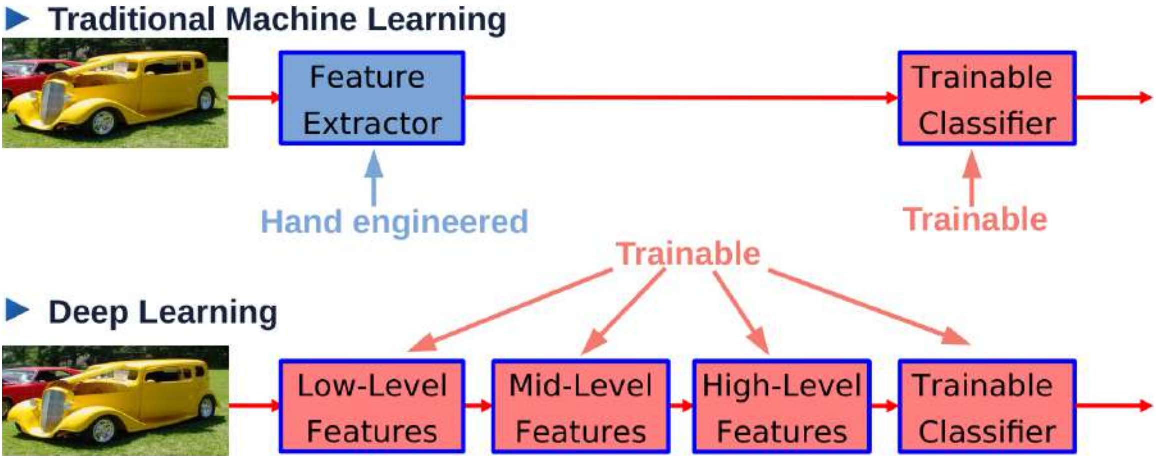
Artificial Intelligence vs Machine Learning

It's all about data

→ AI is a very fuzzy concept, much like "any computer program doing something useful". *Think "if-then" rules.*

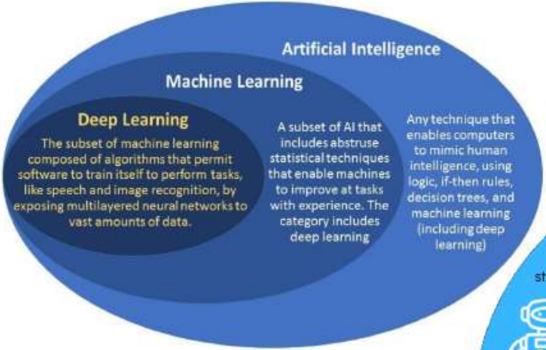
→ ML can be considered a **subfield of AI** since those algorithms can be seen as building blocks to make computers learn to behave more intelligently by somehow generalizing rather than just storing and retrieving data items like a database system would do.

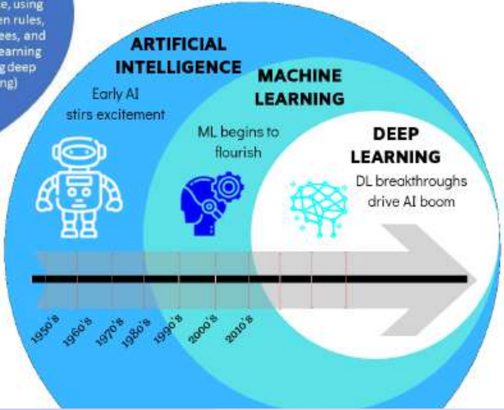
→ *Engineering point of view:* ML is about building programs with tunable parameters (typically an array of floating point values) that are **adjusted automatically** so as to improve their behavior by **adapting to previously seen data**.


Traditional Machine Learning
Feature Extractor
Trainable Classifier
Hand engineered
Trainable
Trainable
Deep Learning
Low-Level Features
Mid-Level Features
High-Level Features
Trainable Classifier

Remember this!

Artificial Intelligence $\supset$ Machine Learning $\supset$ Deep Learning



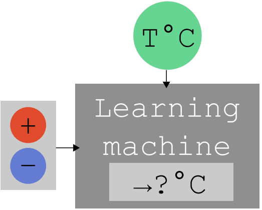


Can you list some examples of projects or products involving Machine Learning?

Machine Learning Problems

How can a machine learn?

A simple example

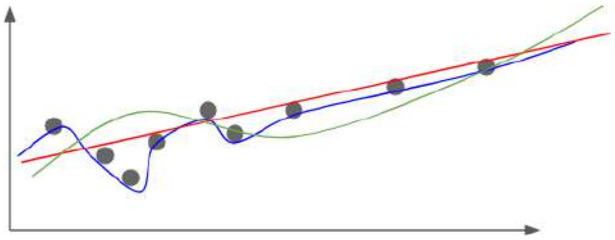


Date	Sensors	Button
8am	17°C	Warmer
9am	22°C	Colder
10am	18°C	Warmer
3pm	21°C	Colder

Given a **finite** amount of **training data**, you have to derive a **relation** for an **infinite** domain.
In fact, there is an infinite number of such relations.

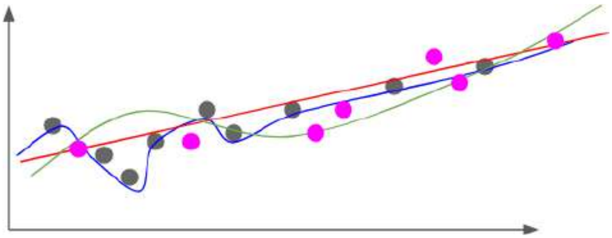
How should we draw the relation?

Given a **finite** amount of **training data**, you have to derive a **relation** for an **infinite** domain.
In fact, there is an infinite number of such relations.



Which relation is the most appropriate?

Given a **finite** amount of **training data**, you have to derive a **relation** for an **infinite** domain. In fact, there is an infinite number of such relations.



. . . the hidden test points (seen after the training). . .

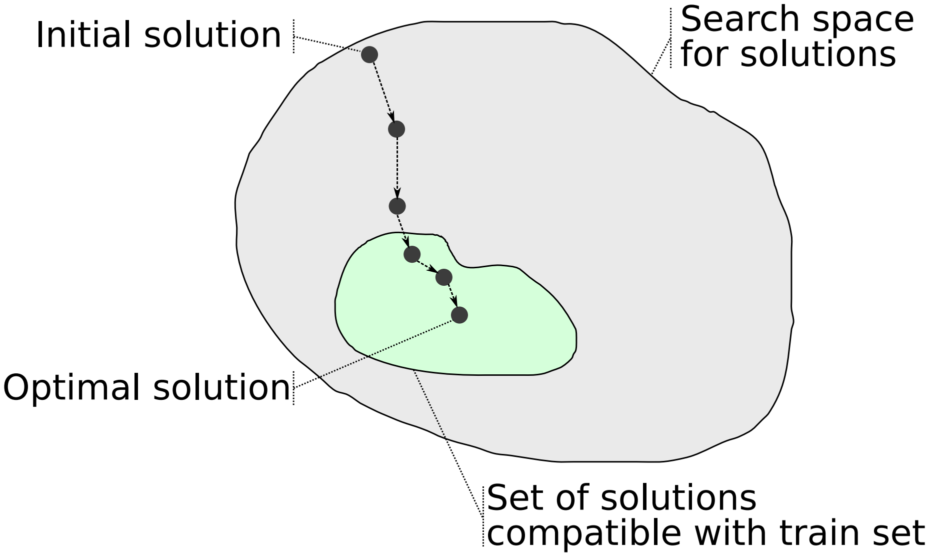
- → It is always possible to find a model complex enough to fit all the examples.
  Example: polynomial with very high degree
- → But how would this help us with new samples?
  It should not generalize well.
- → We need to define a family of acceptable solutions to search from.
  It forces to learn a "smoothed" representation.
  . . . but it should not smooth the representation too much!

Occam's Principle of Parsimony *(14th century)*

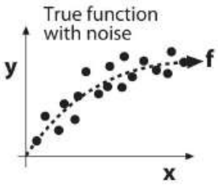
One should not increase, beyond what is necessary, the number of entities required to explain anything.

When many solutions are available for a given problem, we should select the simplest one.
But what do we mean by simple?
We will use prior knowledge of the problem to solve to define what is a simple solution.
Example of a prior: smoothness

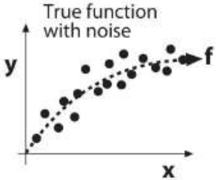
Hypothesis space / initial, compatible (with train set), optimal, and ideal solutions



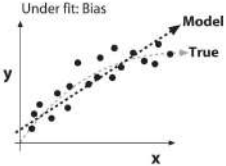
Noise, intrinsic error:
Your data is not perfect (can have noisy or erroneous labels). (or "Every model is wrong.") Even if there exist an optimal underlying model, the observations are corrupted by noise.



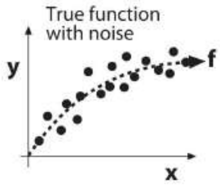
Noise, intrinsic error:
Your data is not perfect (can have noisy or erroneous labels). (or "Every model is wrong.") Even if there exist an optimal underlying model, the observations are corrupted by noise.



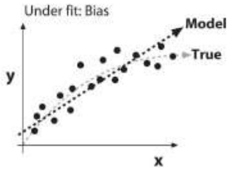
(Inductive) bias, approximation error:
We are exploring a restricted subset of all possible solutions. Your classifier needs to drop some information about the training set to have generalization power (simplify to generalize).



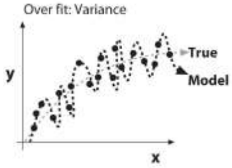
Noise, intrinsic error:
Your data is not perfect (can have noisy or erroneous labels). (or "Every model is wrong.") Even if there exist an optimal underlying model, the observations are corrupted by noise.



(Inductive) bias, approximation error:
We are exploring a restricted subset of all possible solutions. Your classifier needs to drop some information about the training set to have generalization power (simplify to generalize).



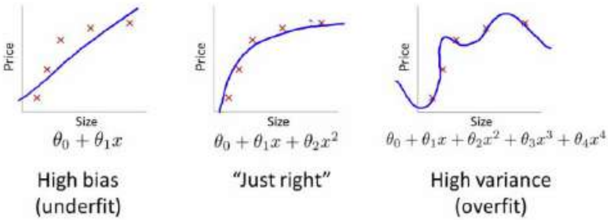
Variance, estimation error:
You have many ways to explain your training dataset. It is hard to find an optimal solution among those many possibilities. Our exploration is not very accurate, we are limited by data we see during training.

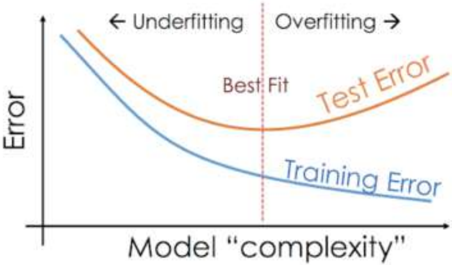


→ Low bias ⇔ high variance: large search set, can capture many useless details — overfitting.

→ High bias ⇔ low variance: small search set, limited exploration, solution too simple — underfitting.

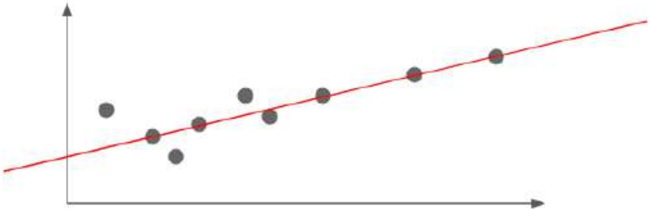
→ Solutions: regularization (penalize solutions which are too complex), early stopping (stop when no more progress). . .





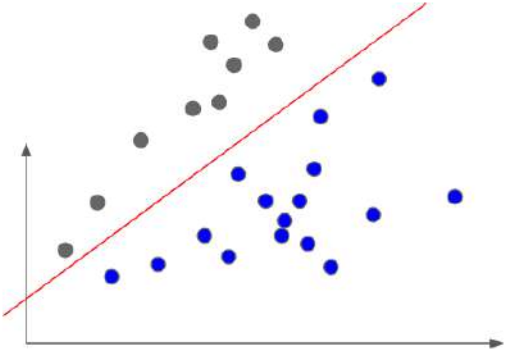
- → **Goal**: (*see next slides*)
- → **Protocol**: supervision? feedback? how many samples for each "experience"?
- → **Measure of success**: error cost? convergence? ...
- → **Inputs** (representation space): quality (noise, distribution) and nature (numerical, symbolical, mixed)
- → **Solutions** (space hypothesis/functions to explore): many approaches.

regression

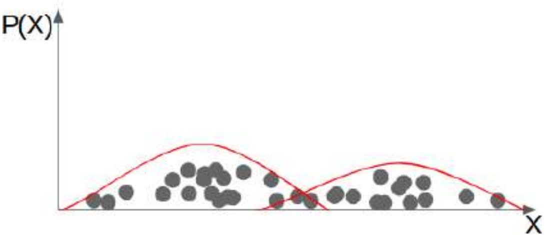


Three kinds of problems in traditional ML
According to Samy Bengio

regression, classification

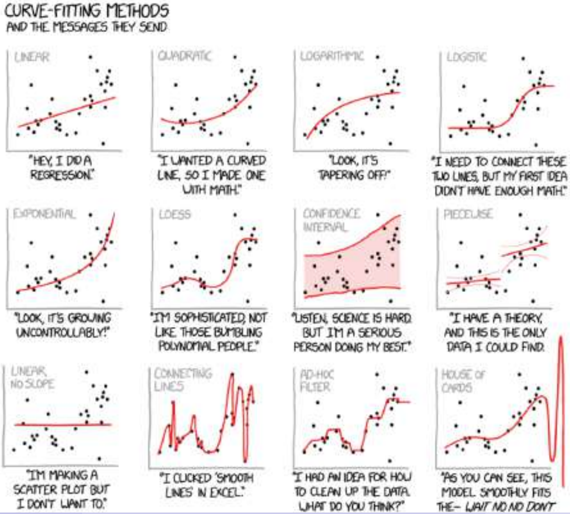


regression, classification, density estimation



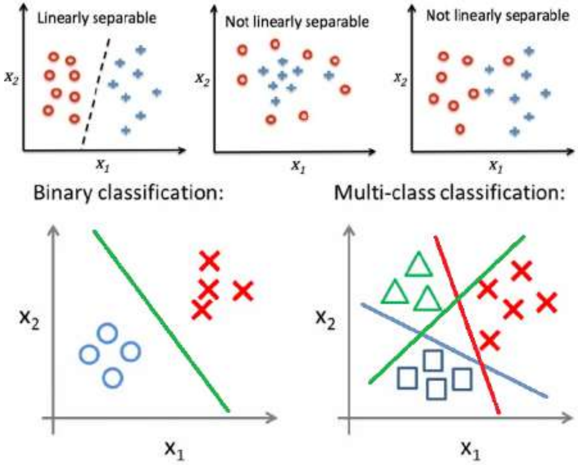
Regression input: samples described by several input variables (correlated)

Regression output: a quantitative variable (scalar)



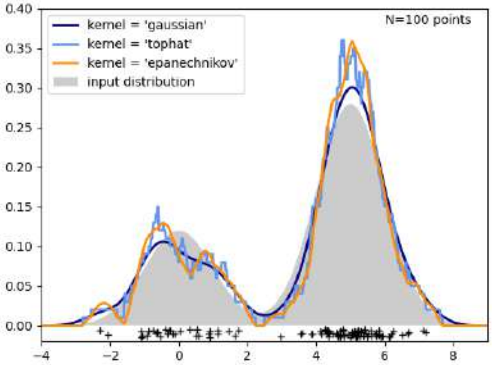
Classification input: samples described by several input variables (correlated)

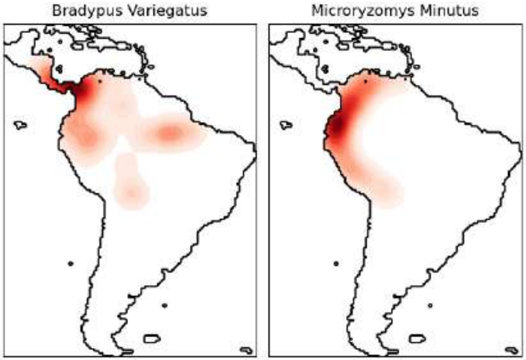
Classification output: a qualitative variable (class, category)



Density estimation input: samples described by several input variables (correlated)

Density estimation output: estimate of the probability distribution function over the feature space





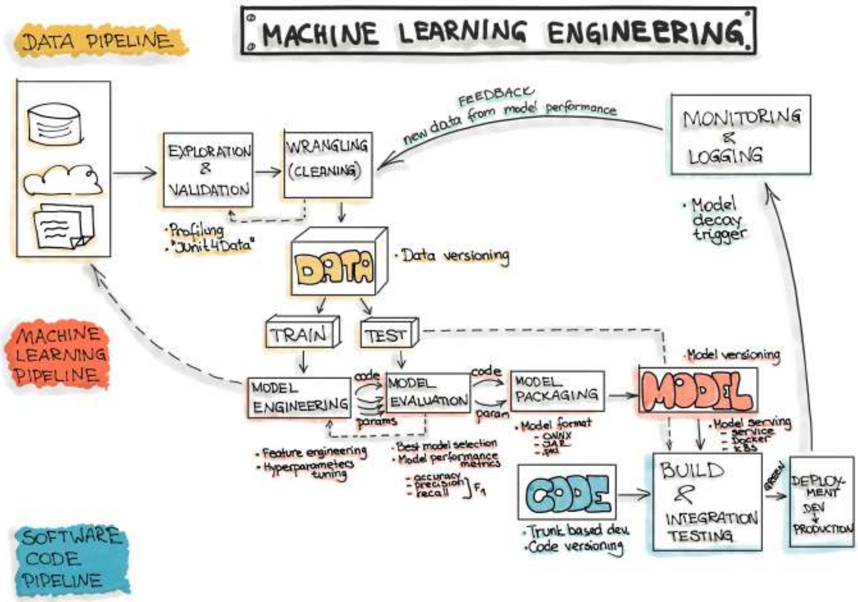
- → **Exploration-based**: Generalization or specialization of rules.
  Examples: Grammatical inference, heuristic discovery for SAT solvers…
- → **Optimization-based**: Topic of this course.
  Examples: linear separators and SVMs, neural networks, decision trees, Bayesian networks, HMMs…
- → **Approximation-based**: Data compression, analogy.
  Examples: KNN, embedding spaces

Can you list some applications of regression, classification or density estimation?
Which kind of supervision can be used for each of those?
Can you imagine how inputs can be represented? What is the is solution space? What could be a measure of success?

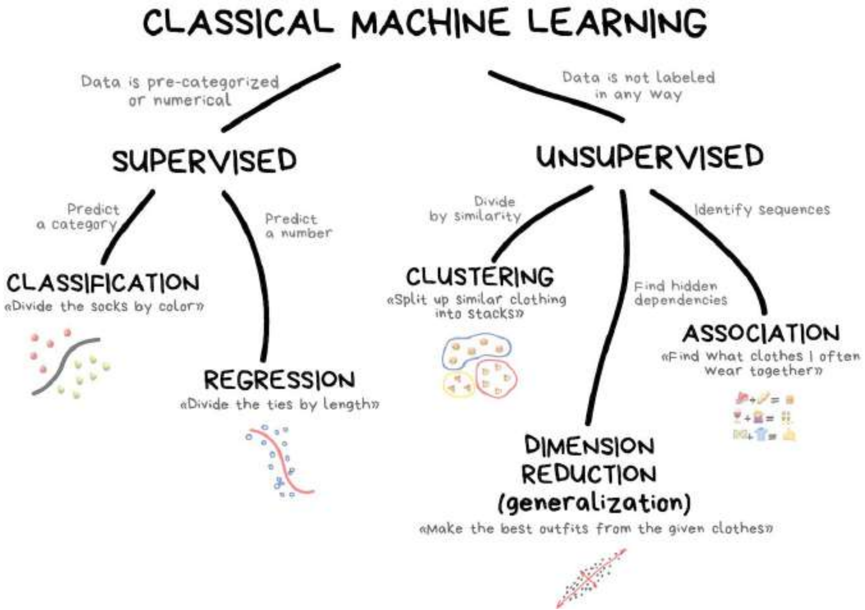
Machine Learning Engineering

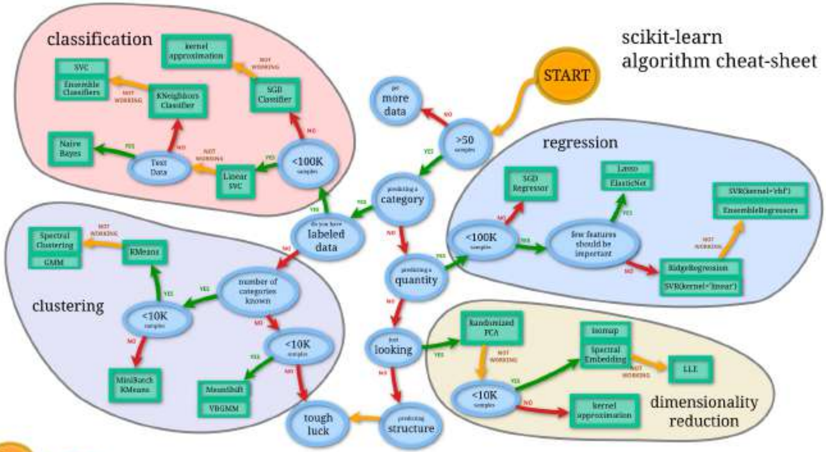
ML from an engineer point of view

Solve problems using the right tool



Simplified view of pre-2010 Machine Learning




scikit-learn
algorithm cheat-sheet
START
>50 samples
get more data
predicting a category
do you have labeled data
predicting a quantity
just looking
predicting structure
tough luck
classification
kernel approximation
SVC
Ensemble Classifiers
KNeighbors Classifier
SGD Classifier
Naive Bayes
Text Data
Linear SVC
<100K samples
clustering
Spectral Clustering
GMM
KMeans
number of categories known
<10K samples
MiniBatch KMeans
MeanShift
VBGMM
regression
SGD Regressor
Lasso
ElasticNet
SVR(kernel='rbf')
EnsembleRegressors
<100K samples
few features should be important
RidgeRegression
SVR(kernel='linear')
dimensionality reduction
Randomized PCA
Isomap
Spectral Embedding
LLE
<10K samples
kernel approximation
YES
NO
NOT WORKING

Back

one sample

$$X = \begin{pmatrix} 1.1 & 2.2 & 3.4 & 5.6 & 1.0 \\ 6.7 & 0.5 & 0.4 & 2.6 & 1.6 \\ 2.4 & 9.3 & 7.3 & 6.4 & 2.8 \\ 1.5 & 0.0 & 4.3 & 8.3 & 3.4 \\ 0.5 & 3.5 & 8.1 & 3.6 & 4.6 \\ 5.1 & 9.7 & 3.5 & 7.9 & 5.1 \\ 3.7 & 7.8 & 2.6 & 3.2 & 6.3 \end{pmatrix} \qquad y = \begin{pmatrix} 1.6 \\ 2.7 \\ 4.4 \\ 0.5 \\ 0.2 \\ 5.6 \\ 6.7 \end{pmatrix}$$

one feature

outputs / labels


scikit learn
Home
Installation
Documentation
Examples
Google Custom Search
scikit-learn
Machine Learning in Python
Simple and efficient tools for data mining and data analysis
Accessible to everybody, and reusable in various contexts
Built on NumPy, SciPy, and matplotlib
Open source, commercially usable - BSD license
Classification
Identifying to which category an object belongs to.
Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ...
— Examples
Regression
Predicting a continuous-valued attribute associated with an object.
Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ...
— Examples
Clustering
Automatic grouping of similar objects into sets.
Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ...
— Examples
Dimensionality reduction
Model selection
Preprocessing

1.9. Ensemble methods

The goal of **ensemble methods** is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator.

Two families of ensemble methods are usually distinguished:

- In **averaging methods**, the driving principle is to build several estimators independently and then to average their predictions. On average, the combined estimator is usually better than any of the single base estimator because its variance is reduced.

  Examples: *Bagging methods*, *Forests of randomized trees*, ...

- By contrast, in **boosting methods**, base estimators are built sequentially and one tries to reduce the bias of the combined estimator. The motivation is to combine several weak models to produce a powerful ensemble.

  Examples: *AdaBoost*, *Gradient Tree Boosting*, ...

sklearn.ensemble.RandomForestClassifier

class sklearn.ensemble.**RandomForestClassifier**(*n_estimators=10, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, bootstrap=True, oob_score=False, n_jobs=1, random_state=None, verbose=0, warm_start=False*) [source]

A random forest classifier.

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting.

Parameters: **n_estimators** : integer, optional (default=10)

The number of trees in the forest.

criterion : string, optional (default="gini")

The function to measure the quality of a split. Supported criteria are "gini" for the Gini impurity and "entropy" for the information gain. Note: this parameter is tree-specific.

Examples

Signal processing

Databases, information retrieval

Statistics

Pattern Recognition

Optimization

Data science, data mining

Practical considerations

Learning objectives

After this course, you should be able to

→ Identify problems that can be solved by machine learning.

→ Formulate your problem in machine learning terms.

→ Given such a problem, identify and apply the most appropriate classical algorithm(s).

→ Implement some of these algorithms yourself.

→ Evaluate and compare machine learning algorithms for a particular task.

See the course page on Moodle

Books, Online material, Research, Software...

Session 0: Intro Lesson (1h)

Session 1: Dimensionality reduction ($\sim$1h Lesson + $\sim$3h Lab)

Session 2: Clustering ($\sim$1h Lesson + $\sim$3h Lab)

Session 3: Supervised learning ($\sim$1h Lesson + $\sim$3h Lab)

Session 4: Supervised learning and project ($\sim$1h Lesson + $\sim$3h Lab)

- 16/20: project
  - /6: solutions proposed and their code (formalization, various approaches, motivation clear, sound pipeline, link with theoretical aspects)
  - /4: performance of your best approach (poor or good)
  - /3: evaluation protocol (data, metrics, results)
  - /3: report of 4-5 pages (complete, clean, precise): *what is the problem, what are the possible solutions, how we should evaluate them, how they perform, what is the best one*
  - **no defense**
- 4/20: weekly quizzes (1 point if you take it, 0 otherwise)

Project topic

To be announced.