





cnrs formation entreprises

Fundamentals of Machine Learning

Direction de l'innovation et des relations avec les entreprises

June 22nd-29th

Aurélien Garivier, Rémi Gribonval, Mathurin Massias, Titouan Vayer

Plan

1 Organization details

2 Introduction to ML

Practical information

- 17:00 UTC+2 – 20:30 UTC+2 (except on Friday: 16:00 UTC+2), 15 minutes break in the middle
- 6 days, from Thursday 22nd to Thursday 29th
- On zoom: <https://cnrs.zoom.us/j/95024210467?pwd=QVNaUmVFaUZpbGViL2lUM2JXMGw5UT09>
  - cameras on please
  - do ask questions and interrupt
- Discord server: <https://discord.gg/pkNQSBeH>

Take 3 minutes to join Discord now as we'll send the material through this channel

Goals

- Target audience: engineers, non specialists, with mathematical background (linear algebra, statistics)
- Content:
  - Machine Learning/AI/Datascience
  - non exhaustive list of the techniques and applications
- What you'll learn:
  - analyze a concrete problem in terms of ML
  - use adequate ML tools and reasoning
  - setup a ML pipeline to address a real-life problem

  ... but also from the scientific point of view: get to know what goes on inside ML black boxes
- Approach:
  - 2/3 class, 1/3 Lab (Python)

Week plan

- Day 1: What is ML? Data, methodology, visualization and feature engineering
- Day 2: Unsupervised learning, dimension reduction
- Day 3: Classification, regression, ensemble methods (logistic regression, random forests)
- Day 4: Model selection, model optimization
- Day 5: Deep learning basics
- Day 6: Advanced deep learning

After each session we send the slides and notebooks of the day so that you can go over it.

Organization for practical sessions

- Why using Python?
  - language of choice for ML (fast prototyping, fast learning)
  - richness of ecosystem: scikit-learn, pytorch, pandas, matplotlib...
- Support: jupyter notebooks
  - great to mix code, images and text
  - very fast prototyping feedback loop
  - beware of potential dangers, see "*I don't like notebooks*" talk by Joel Grus.
- Notebook check: everybody should be able to launch a Jupyter notebook now.
  - launch notebook
  - check interpreter being used
  - import packages
  - check working directory

- Teaching team: Aurélien Garivier, Rémi Gribonval, Mathurin Massias, Titouan Vayer
- Attendees: background, expectations?
  - Uma SHANKAR
  - Hsin-Chi CHANG
  - Larry HALL
  - Satish KANJARKAR
  - Rob LOEBER
  - Chetan NATARAJA
  - Rhythm PATEL
  - Lelass SARAKBI
  - Kiran SHAKYA
  - Ryan TALK
  - Sophie WONG
  - Snehal ZAGADE

Plan

1 Organization details

2 Introduction to ML

What is Machine Learning

Informally

- learning = find the good value for some numbers
- using training data: learn from examples
- use "computation" rather than abstract reasoning

How does a model learn?

- uses examples
- a performance measure (when is the model doing good? bad?)
- model parameters are tuned to have good performance on the training data (trial and error)

What is Machine Learning

ML is data-driven

- use as little expert input as possible
- but practitioner (you) should still make choices:
  - which model?
  - which algorithm?
  - what to gather in the data?

What does ML need?

- data (more and more with DL)
- optimization algorithms
- computing power (more and more with DL)

Diversity of ML data

Inputs:

- Sound
- Images
- Raw number(s)
- Text
- Graphs
- ...

Outputs:

- Sound
- Images
- Raw number(s)
- Text
- Graphs
- ...

Many mappings are possible! (text translation, captioning, image generation, image denoising...)

ML models work with numbers:

- each input is $x \in \mathbb{R}^d$
  - raw data (pixel value, numerical values (size, age, ...))
  - encoded data (feature engineering)
- each output is $y \in \mathbb{R}^k$

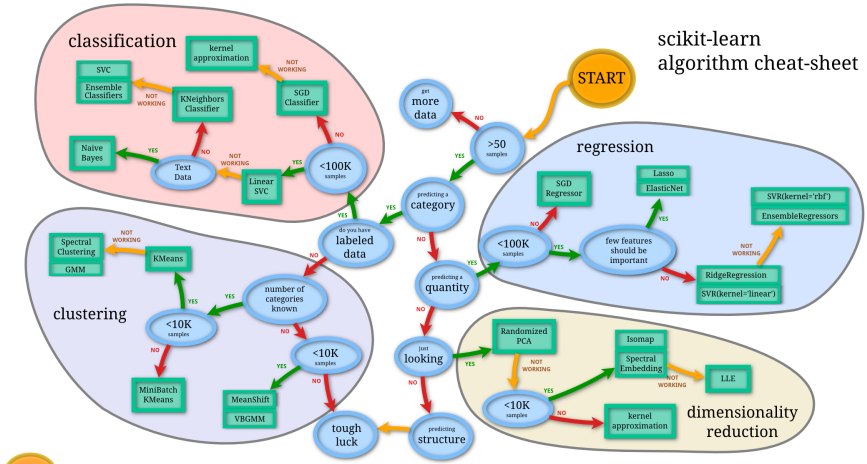
Diversity ML tasks

Two families: **supervised** vs **unsupervised** learning, depending on if **true output y is observed**.

- supervised
  - regression: predict a number
  - classification: predict a class (discrete set of outputs)
    - binary (is this image a dog or a cat?)
    - multiclass (in which language is this text?)
    - multilabel (which objects are present in this image?)
- unsupervised
  - clustering: unsupervised classification (no observed classes)
  - dimensionality reduction: project data into smaller data
  - dictionary learning: find meaningful representation of the data

Unsupervised is way harder! But labelled data costs (cf ImageNet)

ML map


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

ML formalism

- a ML model is just a **parametric function**: $F_\theta(x)$
  - x is the input data
  - θ are the unknown parameters, that we need to fit

- training a ML model is (almost always):
  - choosing a loss function adapted to the task
  - minimizing the loss on training data $(x_1, \dots, x_n)$:

$$\theta^* = \arg\min_\theta \sum_{i=1}^{n} \mathcal{L}(F_\theta(x_i))$$ (unsupervised)

$$\theta^* = \arg\min_\theta \sum_{i=1}^{n} \mathcal{L}(F_\theta(x_i), y_i)$$ (supervised)

The learned parameters are θ^* and the learned model/decision function is F_{θ^*}

Example of ML models

- Linear regression: $F_\theta(x) = \theta^\top x$
- Linear classification: $F_\theta(x) = \text{sign}(\theta^\top x) \in \{-1, 1\}$
- Decision and regression trees: $\theta =$ tree and split thresholds (see Lab 1)
- Random forests: $\theta =$ set of trees
- K-Means (clustering): $\theta = \{c_1, \ldots, c_K\}, \quad F_\theta(x) = \arg\min_k \|x - c_k\|$
- PCA (dimension reduction): $\theta = U \in \mathbb{R}^{d \times D}, \quad F_\theta(x) = U^\top x \in \mathbb{R}^D$

Example of losses

- Regression
  - ℓ_2, mean squared error $\mathcal{L}(\theta) = \frac{1}{2n} \sum_1^n (y_i - F_\theta(x_i))^2$
  - ℓ_1, mean absolute deviation $\frac{1}{n} \sum_1^n |y_i - F_\theta(x_i)|$
- Binary classification
  - accuracy $\sum_1^n 1_{y_i \neq \text{sign } F_\theta(x_i)}$
  - logistic loss $\sum_1^n \log(1 + \exp(-y_i F_\theta(x_i)))$
- Trees (various impurity measures in cells)
- Random Forests (same)
- K-Means (sum of distances to nearest centroid)
- PCA (reconstruction error) $\min_U \sum_1^n \|x_i - UU^\top x_i\|^2$

First ML hands-on: classification

Data and model

- stacked vertically in the **design matrix** $X \in \mathbb{R}^{n \times p}$: n observations in dimension p
- observation vector $y \in \{-1, 1\}^n$
- 2 model choices:
  - decision tree
  - logistic regression (linear model + logistic loss)

$$\mathbb{P}(y = 1|x) = \frac{1}{1 + \exp(-\theta^\top x)}$$

Overfitting

- learner's enemy when $n \leq$ #parameters
- a perfect fit on the data is usually not desirable (noise)
- solution here: regularization

$$\min_{\theta} \mathcal{L}(\theta) + \lambda R(\theta)$$

- $\mathcal{L}$ datafit or data fidelity term, R regularizer or penalty
- $\lambda > 0$ controls the trade-off
- what choice for R?

First regularisation for linear models: Ridge

(Generalized) linear model: $F_\theta(x) = f(\theta^\top x)$

Squared Euclidean norm

- penalizes squared ℓ_2 norm:

$$\min_\theta \mathcal{L}(\theta) + \frac{\lambda}{2}\|\theta\|_2^2$$

- smooth, amenable to 1st and 2nd order methods (gradient descent, Newton method)
- always has a unique solution if $\mathcal{L}$ is **convex**
- improves convergence speed of algorithms

Second regularisation: Lasso

(Generalized) linear model: $F_\theta(x) = f(\theta^\top x)$

Norme L1: $\|\theta\|_1 = \sum_1^p |\theta_i|$

- penalizes ℓ_1 norm:

$$\min_\theta \mathcal{L}(\theta) + \lambda \|\theta\|_1$$

- non-smooth
- but efficient algorithms (proximal methods)
- solution is sparse! $\|\theta^*\|_0 \ll p$

Why the ℓ_1 norm?

Sparsity

- We would like:

$$\min_{\theta} \frac{1}{2}\|X\theta - y\|^2 \quad \text{subject to} \quad \|\theta\|_0 \leq k$$

- issues with this formulation?
- relax it to

$$\min_{\theta} \frac{1}{2}\|X\theta - y\|^2 \quad \text{subject to} \quad \|\theta\|_1 \leq k$$

- under some assumptions, same solutions
- generalisations: group Lasso, Elastic Net, sparse Logistic regression, ...
- other non convexes penalties: MCP, SCAD, ℓ_q $(0 < q < 1)$: better properties