ML@M1 — MACHINE LEARNING

MASTER 1 INFORMATIQUE FONDAMENTALE

YOHANN DE CASTRO AND AURÉLIEN GARIVIER*

†

Year 2020-2021

Amphi E, ENS de Lyon site Monod

These notes are written by the students of the course after each lecture.

CONTENTS

For more visit `http://informatique.ens-lyon.fr/fr`.

*yohann.de-castro@ec-lyon.fr, aurelien.garivier@ens-lyon.fr (speaker)
†email@email.com (scribe)

I STATISTICS 101

Lecture 1 (2 hours, notes by Eliot Tron)
17th January 2020

1 DEFINITION (Statistical model). A statistical model is a pair $(\mathcal{X}^n, \mathcal{P})$ where n is the sample size (integer), $\mathcal{X}$ is a set, and $\mathcal{P}$ is the set of probability distributions on $\mathcal{X}^n$.

I.1 EXAMPLE. n = sample size, $\mathcal{X} = \{0,1\}$ and $\mathcal{P} = \{\mathcal{B}(p)^{\otimes n},\ p \in [0,1]\}$ where $\mathcal{B}(p)^{\otimes n}$ correspond to the distribution $(x_1, ..., x_n)$ on $\{0,1\}^n$, $\mathbb{P}_p(\mathrm{X}_1 = x_1, ..., \mathrm{X}_n = x_n) = \prod_{i=1}^{n} p^{x_i}(1-p)^{1-x_i}$.

2 Definition. A model is parametric if $\mathcal{P} = \{P_\theta,\ \theta \in \Theta\}$ where $\Theta \subset \mathbb{R}^d$ for some $d \in \mathbb{N}$.

3 Definition. A product model is when

$$\forall \theta \in \Theta,\ P_\theta = Q_\theta^{\otimes n}$$

A parametric product model is described by:

- a sample size n
- a set of observation $\mathcal{X}$
- a family $\{Q_\theta,\ \theta\Theta\}$ of probabilities distributions on $\mathcal{X}$

4 Definition. a sample $(X_1, ..., X_n)$ of distribution $P \in \mathcal{P}$ is a random vector $(X_1, ..., X_n) \sim \mathcal{P}$. In a product model, the (X_i) are independent.

5 Definition (Statistic). A statistic is a function of the sample $(X_1, ..., X_n)$.

I.2 example. $T_1 = X_1 + ... + X_n$ is a statistic but not $T_2 = p - X_1$.

6 Definition (Estimator). An estimator is a statistic. It aims at approximating a quantity of interest $g(\theta)$, for a parametric model.

I.3 example. $E_1 = X_1$, $E_2 = \overline{X}_n = \frac{X_1+...+X_n}{n}$ and $E_3 = \frac{X_1+...+X_{\lfloor n/2 \rfloor}}{n/2}$ are estimators.

7 Definition. An estimator T_n is called:

- unbiased if $\forall \theta \in \Theta$, $\mathbb{E}_\theta[T_n] = g(\theta)$.
- consistant (strongly) if $\forall \theta \in \Theta$, $\lim_{n\to\infty} T_n \overset{P-a.s.}{=} g(\theta)$. It is weakly consistant if the convergence is in law.

I.4 example. E_1, E_2 and E_3 are unbiased, and E_2, E_3 are (strongly) consistant.

8 Definition. The quadratic risk of estimator T_n for $g(\theta)$ is:

$$\mathcal{R}_n(T_n, \theta) = \mathbb{E}_\theta\left[\|T_n - g(\theta)\|^2\right]$$

The maximal risk of T_n for $g(\theta)$ is:

$$\overline{\mathcal{R}}_n(T_n) = \sup_{\theta\in\Theta} \mathbb{E}_\theta\left[\|T_n - g(\theta)\|^2\right]$$

I.5 example. Bernoulli model:

- E_1: $\mathbb{E}_P\left[(X_1 - p)^2\right] = Var_P(X_1) = p(1-p)$
- E_2: $\mathbb{E}_P\left[(\overline{X}_n - p)^2\right] = Var_P(\overline{X}_n) = \frac{p(1-p)}{n} \xrightarrow[n\to\infty]{} 0$
- E_3: $\mathbb{E}_P\left[(\overline{X}_{n/2} - p)^2\right] = \frac{p(1-p)}{n/2}$

- $E_4 = \frac{1}{2}$

9 DEFINITION (Minimax estimator). An estimator T_n for $g(\theta)$ is called *minimax* if, whatever another estimator U_n,

$$\overline{\mathcal{R}}_n(T_n) \leq \overline{\mathcal{R}}_n(U_n)$$

or

$$\max_{\theta\in\Theta} \mathcal{R}_n(T_n, \theta) = \min_{U_n \text{estimator}} \max_{\theta\in\Theta} \mathcal{R}_n(U_n, \theta)$$

QUESTION: is E_2 minimax-optimal ? No.

I.1 *Building estimators*

METHOD 1: Method of moments. For a product model, if $g(\theta) = \phi\left(\mathbb{E}_\theta[X_1], ..., \mathbb{E}_\theta\left[X_1^k\right]\right)$ then $\hat{g}_n = \phi\left(\frac{1}{n}\sum_{i=1}^n X_i, ..., \frac{1}{n}\sum_{i=1}^n X_i^k\right)$.

1 PROPOSITION. The moment estimator is strongly consistent if ϕ is continuous and $\mathbb{E}_\theta\left[|X_1|^k\right] < \infty$.

METHOD 2: Maximum Likelihood method

10 DEFINITION. the likelihood function

$$l(\theta, X_1, ..., X_n) = \begin{cases} \mathbb{P}_\theta(X_1, ..., X_n) & \text{in a discret model} \\ f_\theta(X_1, ..., X_n) & \text{when } \mathbb{P}_\theta \text{ has density } f \text{ over } \mathcal{X}^n \end{cases}$$

The **Maximum Likelihood Estimator** (MLE) of θ is

$$\hat{\theta}_n = \operatorname*{argmax}_{\theta\in\Theta} l(\theta, X_1, ..., X_n)$$

I.6 EXAMPLE. Bernoulli likelihood:

$$\begin{aligned} l(p, X_1, ..., X_n) &= \prod_{i=1}^n p^{X_i}(1-p)^{1-X_i} \\ &= p^{\sum_{i=1}^n X_i}(1-p)^{\sum_{i=1}^n 1-X_i} \end{aligned}$$

Maximizing $l(\cdot, X_1, ..., X_n) \iff$ minimizing $g(p) = \ln(l(p, X_1, ..., X_n))$

$$g(p) = \left(\sum_{i=1}^{n} \mathrm{X}_i\right)\ln(p) + \left(n - \sum_{i=1}^{n} \mathrm{X}_i\right)\ln(1-p)$$

$$g'(p) = \left(\sum_{i=1}^{n} \mathrm{X}_i\right)\frac{1}{p} - \left(n - \sum_{i=1}^{n} \mathrm{X}_i\right)\frac{1}{1-p}$$

$g'(p)$ equals zero for $p = \overline{\mathrm{X}_n} = \frac{\sum_{i=1}^{n} \mathrm{X}_i}{n}$, so $\overline{\mathrm{X}}_n$ is the MLE.

II UNSUPERVISED LEARNING: CLUSTERING

Lecture 2 (2 hours, notes by Jérôme Boillot) 23rd January 2020

We have training data $\mathrm{D}_n = \{x_1, \ldots, x_n\}$ with $x_i \in \mathbb{R}^d$ and we want to recover Latent groups.

The idea is to construct $f : \mathbb{R}^d \to \{1, \ldots, \mathrm{K}\}$, $f : x_i \to k_i$ which affects cluster number to x_i.

II.1 *K-means problem*

11 Definition. Quantification error

$$\sum_{i=1}^{n} \min_{k=1,\ldots,\mathrm{K}} \|x_i - c_k\|_2^2$$

12 Definition. K-means algorithm

- Fix $\mathrm{K} \geq 2$, n data points $x_i \in \mathbb{R}^d$.
- Find centroids $c_1, \ldots, c_k$ that minimize the quantification error.

Impossible to find the exact solution (NP Complete).

13 Definition. K-means algorithm (Lloyd, 1981)

- Choose at random K centroids $\{c_1, \ldots, c_{\mathrm{K}}\}$
- For each $k \in \{1, \ldots, \mathrm{K}\}$, find the set C_k of points that are closer to c_k than any $c_{k'}$ for $k' \neq k$
- Update the centroids:
$$c_k = \frac{1}{|\mathrm{C}_k|} \sum_{i \in \mathrm{C}_k} x_i$$
- Repeat the two previous steps until the sets C_{K} don't change

II.1 remark. K-means computes a Voronoi partitioning, it implicitly assumes convex clusters that are uniquely defined by their centroids.

II.2 REMARK. Initialization problem: K-means is very sensitive to the choice of initial points.

Easy solution :

- Pick a first point at random
- Choose the next point the farthest from the previous ones

II.3 REMARK. However, this solution is very sensitive to outliers

II.2 *K-means++*

We want a solution which is less sensitive to outliers: K-means++.

The idea is to pick the first centroid uniformly at random among the x_i.

Then, the next centroids will be taken among the x_i with a probability linear in their distance to the closer centroid already chosen.

The formal algorithm is:

1. $k \leftarrow 1$
2. Pick uniformly at random $x_i \in \{x_1, \dots, x_n\}$, put $c_1 \leftarrow x_i$
3. $k \leftarrow k + 1$
4. Sample $x_i \in \{x_1, \dots, x_n\}$ with probability
$$\frac{\min_{k'=1,\dots,k-1} \|x_i - c_{k'}\|_2^2}{\sum_{j=1}^{n} \min_{k'=1,\dots,k-1} \|x_j - c_{k'}\|_2^2}$$
5. Put $c_k \leftarrow x_i$
6. If $k < \mathrm{K}$ go back to step 3.

Then we use K-means based on these initial centroids.

This algorithm is between random initialization and furthest point initialization.

II.4 REMARK. The complexity of the K-means++ algorithm is in $\mathcal{O}(n \times \mathrm{K} \times n_{it})$ with n_{it} the number of iterations of the algorithm.

2 PROPOSITION. Arthur and Vassilvitskii (2006)

If $c_1, \dots, c_{\mathrm{K}}$ are centroids obtained with K-means++, then:

$$\mathbb{E}[\mathrm{Q}_n(c_1, \dots, c_{\mathrm{K}})] \leq 8(\log \mathrm{K} + 2) \min_{c'_1,\dots,c'_{\mathrm{K}}} \mathrm{Q}_n(c'_1, \dots, c'_{\mathrm{K}})$$

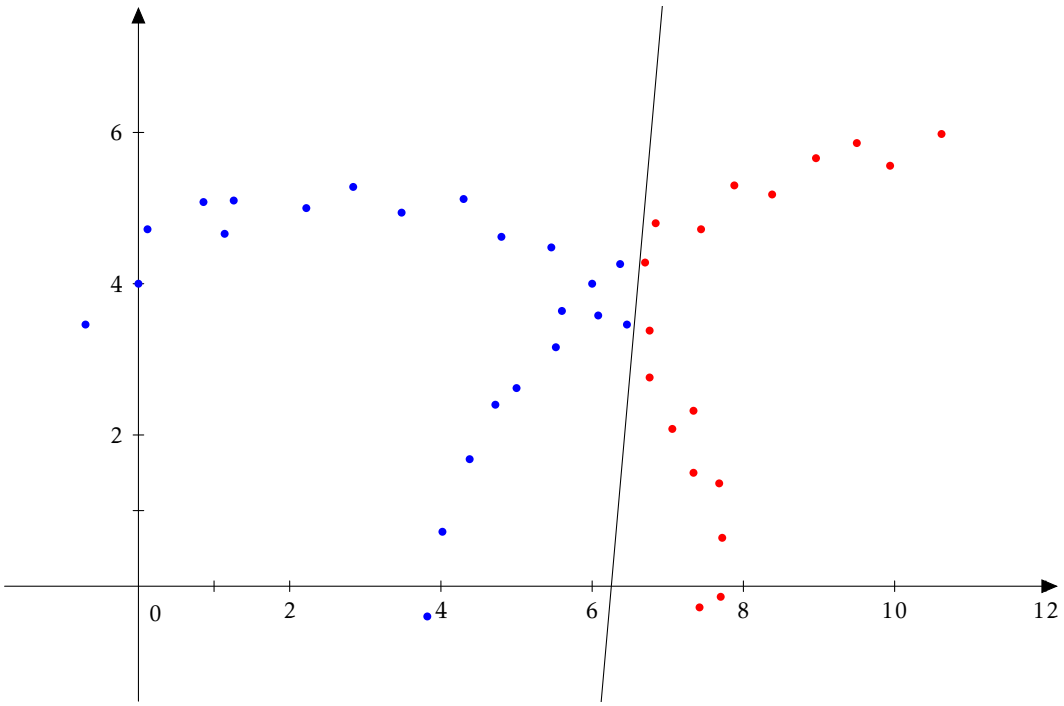
II.5 REMARK. Pro and Cons of K-means++

Pros:

- Simple: easy to implement
- Efficient: cf complexity
- Popular

Cons:

- Notion of mean which is not defined
- Number of clusters K which needs to be specified
- Sensitive to outliers (can be fixed by subsampling and/or outlier detection)
- Roundish clusters: not suited for spherical data, fails if clusters are not convex/round



II.3 *Model-based clustering*

Idea:

- Use a model on data with clusters
- Using a mixture of distributions with different location/mean eg Gaussian mixture

II.3.1 Gaussian mixture models (GMM)

We have a mixture of densities $f_1, \dots, f_K$ and $(p_1, \dots, p_K) \in (\mathbb{R}^+)^K$ st $\sum_{k=1}^{K} p_k = 1$.

14 DEFINITION. Mixture density

$$f = \sum_{k=1}^{K} p_k f_k$$

15 DEFINITION. Gaussian Mixtures Model (GMM)

$$\begin{aligned} f_k = \varphi_{\mu_k, \Sigma_k} &= \text{density of } N(\mu_k, \Sigma_k) \\ &\triangleq \frac{1}{(2\pi)^{d/2}\sqrt{\det(\Sigma_K)}} \exp\left(-\frac{1}{2}(x-\mu_k)^\top \Sigma_k^{-1}(x-\mu_k) \right) \end{aligned}$$

with $\Sigma_k > 0$

16 DEFINITION. Latent variable

- Let $\{P_\theta = f_\theta \mu : \theta \in \Theta\}$ be a statistical model dominated by μ and $(\theta_1, \dots, \theta_K) \in \Theta^K$.
- Let $(p_1, \dots, p_K)$ be a probability vector.
- Let $Z \sim \mathcal{M}(1, p_1, \dots, p_K)$ be a multinomial variable.
- $Y := \sum_{k=1}^{K} k \mathbb{1}_{Z_k = 1}$

  Then
  $$\forall k \in [K] : \mathbb{P}(Y = k) = p_k$$
  In addition, let X be a random variable such that $X|Y \sim P_{\theta_Y}$. Then,
  $$X \sim \sum_{k=1}^{K} p_k P_{\theta_k}$$

We have properties that link together mixture models and clustering:

- When X is distributed with respect to a mixture model with K components, we describe it with K clusters defined by a latent variable $Y \in [K]$.
- Conversely, clustering is naturally modeled by a mixture model: clusters are distributed with respect to conditional variables X|Y.

3 Proposition. The marginal distribution of X is, by Bayes' theorem :

$$\forall x \in \mathbb{R}^d : f(x) = \sum_{k=1}^{\mathrm{K}} p_k f_{\theta_k}(x)$$

where $p_k = \mathbb{P}(\mathrm{Y} = k)$.

17 Definition. Bayes rule for clustering

$$g^* : x \to \underset{1\leq k\leq \mathrm{K}}{\operatorname{argmax}}\, \mathbb{P}_{(p_1,\dots,p_\mathrm{K},\theta)}(\mathrm{Y} = k|\mathrm{X} = x) = \underset{1\leq k\leq \mathrm{K}}{\operatorname{argmax}}\, p_k f_{\theta_k}(x)$$

18 Definition. Then the partitioning $(\mathrm{C}_1, \dots, \mathrm{C}_\mathrm{K})$ is:

$$\forall k \in [\mathrm{K}] : \mathrm{C}_k = \{x \in \mathbb{R}^d : g^*(x) = k\}$$

II.3.2 Study of the GMM

In the case of the model GMM we study the parameter $\theta = (p_1, \dots, p_\mathrm{K}, \mu_1, \dots, \mu_\mathrm{K}, \Sigma_1, \dots, \Sigma_\mathrm{K})$. We define the goodness of the statistical model with density $f_\theta = \sum_{k=1}^{\mathrm{K}} p_k \varphi_{\mu_k,\Sigma_k}$ by :

$$\mathrm{R}_n(\theta) = -\log\text{-likelihood} = -\sum_{i=1}^{n} \log\left(\sum_{k=1}^{\mathrm{K}} p_k \varphi_{\mu_k,\Sigma_k}(x_i)\right)$$

In fact, a local minimizer $\hat{\theta}$ can be obtained using an algorithm called Expectation-Minimization (EM) algorithm.

II.4 *EM Algorithm*

II.4.1 When do we use it?

It is an algorithm that allows to optimize a likelihood with missing or latent data. For a mixture distribution we come up with natural latent variables that simplify the original optimization problem.

II.4.2 The idea

Their is a hidden structure in the model and knowing this structure the optimization problem is easier.

Indeed each point X_i belongs to an unknown class $k \in [\mathrm{K}]$.

So, we can define $\mathrm{C}_{i,k} = 1$ when i belongs to class k, $\mathrm{C}_{i,k} = 0$ otherwise.

These variables are unknown, we say that they are latent variables.

Then, $\mathrm{C}_k := i : \mathrm{C}_{i,j} = 1$ and $\mathrm{C}_1, \dots, \mathrm{C}_\mathrm{K}$ is a partition of $[n]$.

II.4.3 Generative model

i belongs to class C_k with probability p_k, namely:

$C_i = (C_{i,1}, \dots, C_{i,K}) \sim \mathcal{M}(1, p_1, \dots, p_K)$

and $X_i \sim \varphi_{\mu_k, \Sigma_k}$ if $C_{i,k} = 1$.

Then the joint distribution of (X, C) is:

$$f_\theta(x, c) = \prod_{k=1}^{K} (p_k \varphi_{\mu_k, \Sigma_k}(x))^{c_k}$$

We know that $c_k = 1$ for only one k and 0 elsewhere so the marginal density in X is the one of the mixture:

$$f_\theta(x) = \sum_{k=1}^{K} p_k \varphi_{\mu_k, \Sigma_k}(x)$$

II.6 REMARK. C is indeed a latent variable, in such way that the marginal distribution of (X, C) in X is indeed the one of the mixture f_θ.

II.7 REMARK. Complete Likelihood Let $\mathbf{X} = (X_1, \dots, X_n)$ and $\mathbf{C} = (C_1, \dots, C_n)$.

Do as if we observed the latent variable $\mathbf{C}$. Then, the completed likelihood for these "virtual" observations is:

$$L_c(\theta, \mathbf{X}, \mathbf{C}) = \prod_{i=1}^{n} \prod_{k=1}^{K} (p_k \varphi_{\mu_k, \Sigma_k}(X_i))_{i,k}^{C}$$

and the completed log-likelihood:

$$l_c(\theta, \mathbf{X}, \mathbf{C}) = \sum_{i=1}^{n} \sum_{k=1}^{K} C_{i,k} (\log p_k + \log \varphi_{\mu_k, \Sigma_k}(X_i))$$

II.4.4 Description of the EM-Algorithm - Dempster et al. (1977)

1. Initialize $\theta^{(0)}$
2. for $t = 0$ until convergence, repeat:
   (a) E-step: Expectation with respect to the latent variables, for the previous value of θ. Compute:
   $$\theta \to Q(\theta, \theta^{(t)}) = \mathbb{E}_{\theta^{(t)}}\left[l_c(\theta, \mathbf{X}, \mathbf{C}) \middle| \mathbf{X} \right]$$
   (b) M-step: Maximize this expectation. Compute:
   $$\theta^{(t+1)} \in \operatorname*{argmax}_{\theta \in \Theta} Q(\theta, \theta^{(t)})$$

1 THEOREM. The sequence $\theta^{(t)}$ obtained using EM-Algorithm satisfies:

$$l(\theta^{(t+1)}, \mathbf{X}) \geq l(\theta^{(t)}, \mathbf{X})$$

II.8 REMARK. The initialization step will be very important for the EM-Algorithm. It is usually done using K-Means or K-Means++.

Proof. We have to check that $Q_1(\theta^{(t+1)}, \theta^{(t)}) - Q_1(\theta^{(t)}, \theta^{(t)}) \leq 0$.

$$\begin{aligned} Q_1(\theta^{(t+1)}, \theta^{(t)}) - Q_1(\theta^{(t)}, \theta^{(t)}) &= \mathbb{E}_{\theta^{(t)}}[l(\theta^{(t+1)}, \mathbf{X}|\mathbf{X}) - l(\theta^{(t)}, \mathbf{X}|\mathbf{X})|\mathbf{X}] \\ &= \int \log\left(\frac{f_{\theta^{(t+1)}}(x|\mathbf{X})}{f_{\theta^{(t)}}(x|\mathbf{X})}\right) f_{\theta^{(t)}}(c|\mathbf{X})\mu(dc) \\ &\underset{\text{(Jensen)}}{\leq} \log \sum f_{\theta^{(t+1)}}(x|\mathbf{X})\mu(dc) = 0 \end{aligned}$$

This proves $l(\theta^{(t+1)}, \mathbf{X}) \geq l(\theta^{(t)}, \mathbf{X})$ for any t. □

II.5 *EM and GMM (soft K-means)*

With GMM the completed log-likelihood is:

$$l_c(\theta, \mathbf{X}, \mathbf{C}) = \sum_{i=1}^{n} \sum_{k=1}^{K} C_{i,k}(\log p_k + \log \varphi_{\mu_k, \Sigma_k}(X_i)$$

where

$$\theta = (p_1, \ldots, p_K, \mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K)$$

The E-Step is:

$$\begin{aligned} \mathbb{E}_{\theta^{(t)}}\left[l_c(\theta, \mathbf{X}, \mathbf{C})\middle|\mathbf{X}\right] &= \sum_{i=1}^{n} \sum_{k=1}^{K} \mathbb{E}_{\theta^{(t)}}[C_{i,k}|\mathbf{X}](\log p_k + \log \varphi_{\mu_k, \Sigma_k}(X_i)) \\ &= \mathbb{P}_\theta(C_{i,k} = 1|X_i) =: \pi_{i,k}(\theta) \end{aligned}$$

where

$$\pi_{i,k}(\theta) = \frac{p_k \varphi_{\mu_k, \Sigma_k}(X_i)}{\sum_{k'=1}^{K} p_{k'} \varphi_{\mu_{k'}, \Sigma_{k'}}(X_i)}$$

We call $\pi_{i,k}(\theta)$ the "soft-assignment" of i in class k.

The M-Step is:

$$\theta^{(t+1)} = (p_1^{(t+1)}, \dots, p_K^{(t+1)}, \mu_1^{(t+1)}, \dots, \mu_K^{(t+1)}, \Sigma_1^{(t+1)}, \dots, \Sigma_K^{(t+1)})$$

where

$$p_k^{(t+1)} = \frac{1}{n} \sum_{i=1}^{n} \pi_{i,k}(\theta^{(t)})$$

$$\mu_k^{(t+1)} = \frac{\sum_{i=1}^{n} \pi_{i,k}(\theta^{(t)}) X^i}{\sum_{i=1}^{n} \pi_{i,k}(\theta^{(t)})}$$

$$\Sigma_k^{(t+1)} = \frac{\sum_{i=1}^{n} \pi_{i,k}(\theta^{(t)})(X^i - \mu_k^{(t+1)})(X^i - \mu_k^{(t+1)})^\top}{\sum_{i=1}^{n} \pi_{i,k}(\theta^{(t)})}$$

II.9 REMARK. The complexity of this algorithm is $\mathcal{O}(n \times K \times n_{it})$ with n_{it} the number of iterations.

19 DEFINITION. Maximum a posteriori (MAP) rule

Given $\hat{\theta}$, how to affect a cluster number k to a given $x \in \mathbb{R}^d$?

We compute the soft-assignments $\pi_k(x)$ and then $i \in C_k$ if $\pi_k(x) > \pi_{k'}(x)$ for any $k' \neq k$.

II.6 *Point-based objectives*

Contrarily to center-based objectives, point-based objectives do not require to compute a cluster center. With center-based approach we are making sure that points in the same cluster are similar whereas with point-based objectives approach we are making sure that points separated into different clusters should be dissimilar. We use a distortion $D(C_1, \dots, C_K)$ to minimize which is computed on pair of points belonging to clusters. For example, the sum of in-cluster distances is:

$$\hat{D}(C_1, \dots, C_K) = \sum_{k=1}^{K} \sum_{X,Y \in \hat{C}_k} d(X, Y)$$

with $\hat{C}_k = C_k \cap \{X_i : i \in [n]\}$.

We also define a similarity measure $s : \mathcal{X} \times \mathcal{X} \to [0, 1]$. Another example

of distortion is defined by the sum of interclass similarities:

$$D(C_1, \dots, C_K) = \mathbb{E}\left(\sum_{k=1}^{K} s(X, Y) \mathbb{1}_{X \in C_k \cap Y \notin C_k} \right)$$

We can represent this similarity measure by a similarity graph:

- each vertex represents a data point X_i
- vertices are connected by an edge whose weight is their similarity

Such a graph can be defined by the similarity (or adjacency) matrix $W = (s(X_i, X_j))_{1 \leq i,j \leq n}$. Then:

$$\hat{D}(C_1, \dots, C_K) = \sum_{j=1}^{k} \sum_{i \in I_j \wedge l \notin I_j} W_{i,l}$$

with I_k the index sets of each empirical cluster $\hat{C}_k$.

Minimizing $\hat{D}(X_1, \dots, C_K$ is often referred as the graph cut problem.

II.10 REMARK. Generalities about similarity graphs ($A \subset V$)

- Two vertices are connected their similarity is > 0 or $> \tau$ with τ a threshold.
- The graph is undirected: the similarity measure is symmetric.
- The size of $A = |A|$ = the number of its vertices.
- The volume of A, $\text{vol}(A) = \sum_{i \in [n]: X_i \in A} d_i$ (with d_i the degree of X_i).
- A is said connected if any two vertices of A can be joined by a path such that all intermediate points also lie in A.
- A is called a connected component if it is connected and if there are no connections between vertices in A and $V - A$.

20 DEFINITION. The ϵ-neighborhood graph

- X_i and X_j are connected iff $d(X_i, X_j) \leq \epsilon$.
- If ϵ is small enough, all connected points are roughly at the same distance then we put a weight of 1 if $d(X_i, X_j) \leq \epsilon$ and 0 otherwise. It is usually considered as an unweighted graph.

21 DEFINITION. k-nearest neighbor (kNN) graph

X_i and X_j are connected iff $X_i \in k\text{NN}(X_j) \vee X_j \in k\text{NN}(X_i)$. We put a weight of 1 if two vertices are connected and 0 otherwise.

22 DEFINITION. Mutual k-nearest neighbor (kNN) graph

X_i and X_j are connected iff $X_i \in k\text{NN}(X_j) \wedge X_j \in k\text{NN}(X_i)$. We put a weight of 1 if two vertices are connected and 0 otherwise.

23 DEFINITION. Fully connected graph

X_i and X_j are connected iff they have a similarity $s(X_i, X_j) > 0$. Then, the edges are weighted by $s(X_i, X_j)$.

A popular choice of similarity is the Gaussian similarity:

$$s(x, x') = \exp\left(-\frac{d(x, x')^2}{2\sigma^2}\right)$$

In this definition, σ^2 plays a role similar to ϵ and k.

II.7 *Spectral clustering*

For $k = 2$, finding a minimal cut of a graph can be done efficiently with the Stoer-Wagner algorithm. A problem is often results in separating a vertex from the rest.

A solution is to normalize the empirical distorsion either by the size of the clusters or by their volume:

$$\hat{D}_r(C_1, \dots, C_K) = \sum_{k=1}^{K} \frac{1}{|\hat{C}_k|} \sum_{i \in I_k \wedge l \notin I_k} W_{i,l} \qquad \text{(Ratio cut)}$$

$$\hat{D}_n(C_1, \dots, C_K) = \sum_{k=1}^{K} \frac{1}{\text{vol}(\hat{C}_k)} \sum_{i \in I_k \wedge l \notin I_k} W_{i,l} \qquad \text{(Normalized cut)}$$

A problem remains: the balancing introduced by the cluster importance makes the minimization problem computationally hard to solve. However, a can study a relaxation procedure: the spectral clustering algorithm.

24 DEFINITION. Unnormalized Laplacian graph

Let $W \in \mathbb{R}^{n \times n}$ be a symmetric matrix.

- The diagonal matrix $D \in \mathbb{R}^{n \times n}$ such that $\forall i \in [n], D_{i,i} = \sum_{j=1}^{n} W_{i,j}$ is called the degree matrix of the graph defined by W.
- $L = D - W$ is called the Laplacian of the graph defined by W.

II.11 REMARK. Let W and L be respectively the adjacency matrix and Laplacian of the similarity graph of $(X_1, \dots, X_n)$. For any positive integer K and for all partitioning $(C_1, \dots, C_K)$ of $(X_1, \dots, X_n)$ we have:

$$\hat{D}_r(C_1, \dots, C_K) = \text{tr}(H^\top L H)$$

where $H = \left(\frac{1}{\sqrt{|I_k|}}\mathbb{1}_{i\in I_k}\right)_{1\le i\le n\wedge 1\le k\le K}$.

[1em] In addition, the columns of H are orthonormal to each other ($H^\top H = I$).

Proof. Let's denote $h_j \in \mathbb{R}^n$ the columns of H (for $j \in [K]$. We have:

$$\operatorname{tr}(H^\top L H) = \operatorname{tr}((L^{1/2}H)^\top (L^{1/2}H)) = \sum_{j=1}^{K} (L^{1/2}h_j)^\top (L^{1/2}h_j) = \sum_{j=1}^{K} h_j^\top L h_j$$

In addition, for all $u \in \mathbb{R}^n$:

$$\begin{aligned}
u^\top L u &= u^\top D u - u^\top W u \\
&= \sum_{i=1}^{n} D_{i,i}u_i^2 - \sum_{1\le i,l\le n} W_{i,l}u_i u_l \\
&= \frac{1}{2}\left(\sum_{i=1}^{n} D_{i,i}u_i^2 + \sum_{l=1}^{n} D_{l,l}u_l^2 - 2\sum_{1\le i,l\le n} W_{i,l}u_i u_l\right) \\
&= \frac{1}{2}\left(\sum_{1\le i,l\le n} W_{i,l}u_i^2 + \sum_{1\le i,l\le n} W_{i,l}u_l^2 - 2\sum_{1\le i,l\le n} W_{i,l}u_i u_l\right) \quad (W_{i,l} \text{ symmetric}) \\
&= \frac{1}{2}\sum_{1\le i,l\le n} W_{i,l}(u_i - u_l)^2
\end{aligned}$$

Therefore, for all $j \in [K]$:

$$\begin{aligned}
h_k^\top L h_j &= \frac{1}{2}\sum_{1\le i,l\le n} W_{i,l}(H_{i,j} - H_{l,j})^2 \\
&= \frac{1}{2}\sum_{i\in I_j \wedge l\notin I_j} W_{i,l}
\end{aligned}$$

since $H_{i,j} - H_{l,j}$ is nonzero only if $i \in I_j$ and $l \notin I_j$ or the other way around.

Gathering everything we have:

$$\operatorname{tr}(H^\top L H) = \sum_{j=1}^{K} h_j^\top L h_j = \sum_{j=1}^{K} \frac{1}{|I_j|}\sum_{i\in I_j \wedge l\notin I_j} W_{i,l} = \hat{D}_r(C_1, \dots, C_K)$$

□

Up to normalization, H represents the one-hot-encoding. For example, for $K = 3$ if we recognize the sample $(X_1, \dots, X_n)$ such that $\hat{C}_1$ appears first, then

$\hat{C}_2$ and so on, we get:

$$H = \begin{pmatrix} \frac{1}{|\hat{C}_1|} & 0 & 0 \\ \vdots & \vdots & \vdots \\ \frac{1}{|\hat{C}_1|} & 0 & 0 \\ 0 & \frac{1}{|\hat{C}_2|} & 0 \\ \vdots & \vdots & \vdots \\ 0 & \frac{1}{|\hat{C}_2|} & 0 \\ 0 & 0 & \frac{1}{|\hat{C}_3|} \\ \vdots & \vdots & \vdots \\ 0 & 0 & \frac{1}{|\hat{C}_3|} \end{pmatrix}$$

II.12 REMARK. We have shown that the ratio-cut problem:

$$\min_{(\hat{C}_1,\ldots,\hat{C}_K)\in P(\{X_1,\ldots,X_n\})} \sum_{k=1}^{K} \frac{1}{|I_k|} \sum_{i\in I_k \wedge l\notin I_k} W_{i,l}$$

is equivalent to:

$$\min_{H\in\mathbb{R}^{n\times K}} \operatorname{tr}(H^\top LH)$$

$$\text{s.t.} \begin{cases} H^\top H = I \\ \forall j \in [K], \forall i \in [n] : H_{i,j} \in \left\{0, \frac{1}{\sqrt{|I_j|}}\right\} \end{cases}$$

II.7.1 Relaxation for ratio-cut

The equivalent program that we have found is an integer programming problem which we may not be able to solve efficiently. The idea to be able to approach it is to discard the last constraint (the values $(|I_1|,\ldots,|I_K|)$ are known in advance):

$$\min_{H\in\mathbb{R}^{n\times K}} \operatorname{tr}(H^\top LH)$$

$$\text{s.t.} H^\top H = I$$

This problem is solved by the matrix H for which the columns are the minor eigenvectors of L. We have an algorithm:

25 DEFINITION. Unnormalized spectral clustering

Require: $W \in \mathbb{R}^{n \times n}$

1. $L \leftarrow$ Laplacian of W
2. $H \leftarrow$ K minor eigenvectors of L as columns
3. $Y_i \leftarrow i^{th}$ row of H $\forall i \in [n]$. $Y_i \in \mathbb{R}^K$.
4. $(\hat{C}_1, \ldots, \hat{C}_K \leftarrow$ output of K-means algorithm based on $(Y_1, \ldots, Y_n)$

Ensure: $(\hat{C}_1, \ldots, \hat{C}_K)$

4 Proposition. Reformulation for normalized cut

Let W and L be the usual matrices of the similarity graph of $(X_1, \ldots, X_n)$. For any positive integer k and for all partitioning $(C_1, \ldots, C_K)$ we have:

$$\hat{D}_n(C_1, \ldots, C_K) = \mathrm{tr}(H^\top L H)$$

where $H = \left(\frac{1}{\sqrt{\mathrm{vol}}} \mathbb{1}_{i \in I_j} \right)_{1 \leq i \leq n \wedge 1 \leq j \leq K}$.

[1em]
In addition, the columns of $D^{1/2}H$ are orthonormal to each other ($H^\top DH = I$).

The proof of this property is similar to the one of the previous proposition except that we have for all $j \in [K]$:

$$h_j^\top L h_j = \frac{1}{\mathrm{vol}(\hat{C}_j)} \sum_{i \in I_j \wedge l \notin I_j} W_{i,l}$$

In fact, the normalized cut problem:

$$\min_{(\hat{C}_1, \ldots, \hat{C}_K) \in P(\{X_1, \ldots, X_n\})} \sum_{k=1}^{K} \frac{1}{\mathrm{vol}(\hat{C}_j)} \sum_{i \in I_k \wedge l \notin I_k} W_{i,l}$$

is equivalent to:

$$\min_{H \in \mathbb{R}^{n \times K}} \mathrm{tr}(H^\top L H)$$

$$\text{s.t.} \begin{cases} H^\top DH = I \\ \forall j \in [K], \forall i \in [n] : H_{i,j} \in \left\{ 0, \frac{1}{\sqrt{\mathrm{vol}(\hat{C}_j)}} \right\} \end{cases}$$

and can be relaxed to:

$$\min_{H \in \mathbb{R}^{n \times K}} \mathrm{tr}(H^\top L H)$$

$$\text{s.t.} H^\top DH = I$$

what can be reformulated in:

$$\min_{H \in \mathbb{R}^{n \times K}} \operatorname{tr}(U^\top L_S U)$$
$$\text{s.t.} \begin{cases} H = D^{-1/2} U \\ U^\top U = I \end{cases}$$

where $L_S = D^{-1/2} L D^{-1/2}$.

This problem is solved by U for which the columns are minor eigenvectors of L_S. There is a correlation to H for which columns are minor eighenvectors of $L_W = D^{-1} L$. The resulting algorithm is the following:

26 DEFINITION. Normalized spectral clustering

Require: $W \in \mathbb{R}^{n \times n}$

1. $L_W \leftarrow$ Laplacian of W
2. $H \leftarrow$ K minor eigenvectors of L_W as columns (similar to the generalized eigenproblem $Lu = \lambda Du$)
3. $Y_i \leftarrow i^{th}$ row of $H \forall i \in [n]$. $Y_i \in \mathbb{R}^K$.
4. $(\hat{C}_1, \ldots, \hat{C}_K \leftarrow$ output of K-means algorithm based on $(Y_1, \ldots, Y_n)$

Ensure: $(\hat{C}_1, \ldots, \hat{C}_K)$

II.13 REMARK. $\lambda \in \mathbb{R}_+$ is eigenvalue of L_W with eigenvector u iff λ and u solve the generalized eigenvalue problem $Lu = \lambda Du$.

II.7.2 Comparison of Ratio cut vs. normalized cut i

Both have objective functions such that points separated into different clusters are dissimilar and take into account the importance of the clusters (by their size or their volume).

However, they have different behavior on cluster importance:

$$\sum_{i \in I_j \wedge l \in I_j} W_{i,l} = \operatorname{vol}(\hat{C}_j) - \sum_{i \in I_j \wedge l \notin I_j} W_{i,l}$$

In other words, the intra-cluster similarity is maximized as soon as the volume is maximized and the cut with rest of the vertices is minimized; which is achieved by normalized cut minimization.

On the other hand, the size $|\hat{C}_j|$ of a cluster is not necessarily related to the intra-cluster similarity.

II.14 REMARK.

(+) Normalized spectral clustering: L_W behaves as expected when $n \to \infty$.

(-) L can lead to completely unreliable results, even for small sample size (cf Von Luxburg, 2007).

27 DEFINITION. Another Normalized spectral clustering (with L_S)

Require: $W \in \mathbb{R}^{n \times n}$

1. $L_S \leftarrow$ Laplacian of W
2. $H \leftarrow$ K minor eigenvectors of L_S as columns
3. $Y_i \leftarrow i^{th}$ row of H normalized to 1 for all $i \in [n]$. $Y_i \in \mathbb{R}^K$, $\sum_{j=1}^{K} (Y_i)_j^2 = 1$.
4. $(\hat{C}_1, \ldots, \hat{C}_K \leftarrow$ output of K-means algorithm based on $(Y_1, \ldots, Y_n)$

Ensure: $(\hat{C}_1, \ldots, \hat{C}_K)$

II.15 REMARK.

- There is no theorical guaratees concerning the "quality" of these two relaxations.
- There exists many other relaxations: relying on semidefinite programming.
- Spectral relaxations are not appealing for the quality of the solutions they provide but for the simplicity of the problem in which they result (standard linear algebra - eigenvalue - problems).

28 DEFINITION. Definitions of Laplacian graph

- Unnormalized Laplacian: $L = D - W$
- Normalized Laplacian 1: $L_S = D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$
- Normalized Laplacian 2: $L_W = D^{-1} L = I - D^{-1} W$

For the two last definitions, they are respected symmetrically normalized by $D^{-1/2}$ and whitened by D.

5 PROPOSITION. Properties of Laplacian graph

1. $\forall u \in \mathbb{R}^n$:

$$u^\top L u = \frac{1}{2} \sum_{1 \leq i,l \leq n} W_{i,l} (u_i - u_l)^2$$

$$u^\top L_S u = \frac{1}{2} \sum_{1 \leq i,l \leq n} W_{i,l} \left(\frac{u_i}{\sqrt{D_{i,i}}} - \frac{u_l}{\sqrt{D_{l,l}}} \right)^2$$

2. 0 is eigenvalue of L and L_W with eigenvector $\mathbb{1}$. 0 is eigenvalue of L_S with eigenvector $D^{1/2}\mathbb{1}$.

3. $\lambda \in \mathbb{R}_+$ is eigenvalue of L_W with eigenvector u iff λ is eigenvalue of L_S with eigenvector $D^{1/2}u$.

4. L, L_S and L_W are symmetric SDP (semi-definite positive) matrices.

Proof.

1. See above

2. Obvious

3. $\lambda u = L_W u \Leftrightarrow \lambda u = D^{-1}Lu \Leftrightarrow \lambda(D^{1/2}u = D^{-1/2}LD^{1/2}(D^{1/2}u)$

4. Symmetry comes from symmetry of W and SDPness comes from Point 1 and Point 3.

□

6 Proposition. Let G be an undirected graph with non-negative weights.

- The multiplicities of the eigenvalues 0 of L, L_S and L_W are the same and equal the number k of connected components $(A_1, \ldots, A_k)$ in G.
- The eigenspace of 0 for both L and L_W is spanned by $\{\mathbb{1}_{A_1}, \ldots, \mathbb{1}_{A_k}\}$ and the eigenspace of 0 for L_S is spanned by $\{D^{-1/2}\mathbb{1}_{A_1}, \ldots, D^{-1/2}\mathbb{1}_{A_k}\}$.

II.8 *Hierarchical clustering*

With the method K-means we have a lack of hierarchy in clusters: decreasing K does not lead to merging clusters.

Then we will focus on agglomerative approaches (which is based on merging clusters) ∼ bottom-up or on divisive ones (based on splitting clusters) ∼ top-down.

29 Definition. Agglomerative approach

We start from the partitioning of the training set $(X_1, \ldots, X_n)$ in which each cluster is a unit set $\{X_i\}$ and then we will merge successively the closest clusters.

Then, the number of clusters decreases at each iteration, the clusters are nested and a cluster at iteration t denoted by $\hat{C}^t$ is either the same ($\hat{C}^t = \hat{C}^{t-1}$) or the union of two previous clusters ($\hat{C}^t = \hat{C}_1^{t-1} \cup \hat{C}_2^{t-1}$).

Two parameters have to be defined in such a procedure :

- the (dis)similarity (or linkage) between two clusters

- the merging stopping rule

II.16 REMARK. Examples of cluster dissimilarities

The dissimilarities will be denoted by $D : P(\{X_1, \dots, X_n\})^2 \to \mathbb{R}_+$.

- Simple linkage:
$$D(A, B) = \min_{x \in A, y \in B} d(x, y)$$

- Complete linkage:
$$D(A, B) = \max_{x \in A, y \in B} d(x, y)$$

- Average linkage:
$$D(A, B) = \frac{1}{|A||B|} \sum_{x \in A, y \in B} d(x, y)$$

- Ward's minimum variance:

  Given the intraclass inertia for a generic subset $C \subset (X_1, \dots, X_n)$:
$$I(C) = \sum_{x \in C} d(x, m_C)^2$$
  where $m_C = \frac{1}{|C|} \sum_{u \in C} y$,
$$D(A, B) = I(A \cup B) - I(A) - I(B)$$
  which is the increase of intraclass inertia when merging A and B.

- The Euclidian distance:
$$D(A, B) = \frac{|A||B|}{|A| + |B|} \|m_A - m_B\|^2$$
  This method is very similar to K-means but with a greedy procedure since Ward's method merges clusters by minimizing the increase in the total intraclass inertia.

- We can also use the Manhattan distance (or Cityblock, or l1), the cosine distance or any precomputed affinity matrix.

30 DEFINITION. Stopping rules

We can choose to use a fixed number of clusters or a distance upper bound $\bar{D}$ (or alternatively a scaled distance upper bound $\alpha \in \mathbb{R}_+$ such that $\bar{D} = \alpha \max_{1 \le i,j \le n} d(X_i, X_j)$ for single, complete and average linkages).

II.17 REMARK. The resulting sequence of partitioning can be represented as a tree, called dendrogram. In this tree the root is the unique cluster that

gathers all points (the final cluster) and the leaved are the unit set clusters (the algorithm initialization).

II.18 REMARK. The complexity of such an algorithm is $\mathcal{O}(n^3)$ if there is no restriction on the merging possibilities and $\mathcal{O}(n^2)$ if there is only a bounded number of merging possible for a given cluster.

Lecture 3 (2 hours, notes by Guillaume Rousseau) 31st January 2020

III DIMENSIONALITY REDUCTION

31 DEFINITION. Random Gaussian Vector A random vector $X \in \mathbb{R}^k$ is Gaussian if, for all $a \in \mathbb{R}^k$, $\langle X, a \rangle$ is a real Gaussian variable. In particular, if the components of X are independent Gaussian variables, then X is a Gaussian vector. If the covariance Σ of X is invertible, then the density of X is:

$$f(x_1, \dots, x_k) = \frac{1}{\sqrt{2\pi det(\Sigma)}} e^{\frac{-\tilde{x}^T \Sigma \tilde{x}}{2}}$$

Where $\tilde{x} = x - u$, with $u = (E[X_1], \dots, E[X_n])$

Data:

$$X = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix} \in \mathcal{M}_{n,p}(\mathbb{R})$$

i.e. n points in $\mathbb{R}^p$. We assume p to be extremely large. We want to reduce the dimension, i.e. replace each x_i by a $y_i \in \mathbb{R}^d$ with $d << p$. In this section we will look at linear reduction:

$$y_i = W x_i$$

with $W \in (M)_{d,p}(\mathbb{R})$.

We also want to formalize the fact that we do not loose too much information. We introduce two notions:

- Quasi-invertibility: there exists a pseudo-inverse $U : \mathbb{R}^d \to \mathbb{R}^p$ such that $\forall i \in \{1, \dots, n\}, \tilde{(x_i)} = U y_i \approx x_i$.
- Distance preserving property: $\forall i, j \in \{1, \dots, n\}, \|y_i - y_j\| \approx \|x_i - x_j\|$.

III.1 *First approach: Principal Component Analysis*

We consider the following optimization problem:

$$\underset{W \in (M)_{d,p}(\mathbb{R}), U \in (M)_{p,d}(\mathbb{R})}{\arg\min} \sum_{i=1}^{n} \|x_i - UWx_i\|^2 \quad (1)$$

7 PROPOSITION. A solution (W, U) of (1) is of the form ($U = V, W = V^T$) such

that $V^T V = I_d$. In other words:

$$(1) \iff \underset{V \in (M)_{p,d}(\mathbb{R}), V^T V = I_d}{\arg\min} \sum_{i=1}^{n} \|x_i - VV^T x_i\|^2$$

Proof. take $W \in (M)_{d,p}(\mathbb{R})$, $U \in (M)_{p,d}(\mathbb{R})$. Define:

$$R = \{UWx | x \in \mathbb{R}^p\} = Im(UW) \subseteq \mathbb{R}^p$$

$dim(R) \leq d$. Let us assume that $dim(R) = d$, without loss of generality. Let $v_1, \ldots, v_d$ be an orthonormal basis of R, and $V = (v_1|v_2|\ldots|v_d) \in (M)_{p,d}(\mathbb{R})$.

Then $V^T V = I_d$. Besides, for all $\tilde{x} \in R$, there exists $y \in \mathbb{R}^d$ such that $\tilde{x} = Vy$. Then for every $x \in \mathbb{R}^p$ and $\tilde{(x)} \in R$ such that $\tilde{x} = Vy$:

$$\|x - \tilde{x}\|^2 = \|x - Vy\|^2 = \|x\|^2 - 2x^T Vy + y^T V^T Vy = \|x\|^2 + \|y\|^2 - 2x^T Vy$$

Let us denote this quantity $g(y)$. g is minimal at the point y^* such that $\nabla g(y^*) = -2V^T x + 2y^* = 0$, i.e. in $y^* = V^T x$. So $g(y) \geq g(y^*) = \|x - VV^T x\|^2$ Hence the result. □

III.1 REMARK. $VV^T x_i$ is the orthogonal projection of x_i onto R.

Since $\|x_i - VV^T x_i\|^2 = \|x_i\|^2 - 2x_i^T VV^T x_i + x_i^T VV^T VV^T x_i = \|x_i\|^2 - Tr(x_i^T VV^T x_i) = \|x_i\|^2 - Tr(V^T x_i x_i^T V)$, we have:

$$(1) \iff \underset{V \in (M)_{p,d}(\mathbb{R}), V^T V = I_d}{\operatorname{argmax}} Tr(V^T ATV)$$

with $A = \sum_{i=1}^{n} x_i x_i^T$.

The matrix A is in $S_p^+(\mathbb{R})$, so by spectral theorem:

$$\exists W \in O_p(\mathbb{R}), D = \begin{bmatrix} d_1 & & (0) \\ & \ddots & \\ (0) & & d_n \end{bmatrix}, A = WDW^{-1} = WDW^T.$$

III.2 THEOREM. *Let $v_1, \ldots, v_d$ be the eigenvectors of A associated to the d largest eigenvalues of* A, *and* $V = (v_1, \ldots, v_d)$. *Then* $(U = V, W = V^T)$ *is a solution to (1).*

Proof. Let $U \in (M)_{p,d}(\mathbb{R})$ such that $U^T U = I_d$ and let $B = W^T U$.

Then:

$$WB = U \text{and} U^T AU = B^T W^T AWB = B^T BD$$

Hence:

$$Tr(U^T AU) = \sum_{j=1}^{p} d_j \sum_{i=1}^{p} B_{ji}^2$$

But:

$$B^T B = U^T W W^T U = U^T U = I_d$$

Hence the columns of B are orthonormal, and so

$$\sum_{j=1}^{p} \sum_{i=1}^{d} B_{ji}^2 = \sum_{i=1}^{d} (\sum_{j=1}^{p} B_{ji}^2) = d$$

In addition, the columns of B can be completed into an orthonormal basis of $\mathbb{R}^p$ which yields a matrix $\tilde{B} \in (M)_p(\mathbb{R})$ that is orthonormal. In particular:

$$\sum_{i=1}^{d} B_{ji}^2 \leq \sum_{i=1}^{p} B_{ji}^2 = 1$$

Hence :

$$Tr(U^T A U) \leq max_{\beta \in [0,1]^p, \|\beta\|_1 = d} \sum_{j=1}^{p} d_j \beta_j$$

The problem is now easy to solve: we assign 1 to the d first β_i. This yields $Tr(U^T A U) = \sum_{j=1}^{d} d_j$, reached by $U = V = (v_1, \ldots, v_d)$, hence the result. □

III.3 REMARK. $A = \sum_{i=1}^{n} x_i x_i^T \in (M)_p(\mathbb{R})$. Denote:

$$X = \begin{pmatrix} x_1^T \\ \vdots \\ x_n^T \end{pmatrix}$$

if $p > n$, $B = XX^T \in (M)_n(\mathbb{R})$ is smaller than A. But if $Bu = \lambda u$ then for $v = \frac{X^T U}{\|X^T U\|}$, $Av = \lambda v$.

PCA looks for the direction where the variance of the data is maximal.

III.2 *Approach 2: Johnson-Lindenstrauss lemma*

III.4 THEOREM. *There exists* $A \in (M)_{d,p}(\mathbb{R})$ *such that* $\forall i, j \in 1, ..., n$:

$$(1 - \epsilon)\|x_i - x_j\|^2 \leq \|Ax_i - Ax_j\|^2 \leq (1 + \epsilon)\|x_i - x_j\|^2$$

as soon as $d \geq \frac{4log(n)}{\epsilon - log(1+\epsilon)}$

In fact, if $d \geq \frac{4log(n)+2log(\frac{1}{\delta})}{\epsilon - log(1+\epsilon)}$, *a matrix* $A \in (M)_{d,p}(\mathbb{R})$ *such that the* A_{ij} *are independent, identically distributed, following* $\mathcal{N}(0, \frac{1}{\delta})$, *has probability at least* $1 - \delta$ *of satisfying the theorem. In other words, picking a random matrix will provide a good projection with high probability.*

Proof. Let $y \in \mathbb{R}^n$ and $Y = Ay$. For $i \in \{1, ..., d\}$, $Y_i = \sum_{j=1}^{p} A_{ij} y_j \sim \mathcal{N}(0, \frac{\|y\|^2}{d})$. Thus:

$$E[\|Y\|^2] = E[\sum_{i=1}^{d} Y_i^2 = \|y\|^2$$

III.5 LEMMA. *If* $U_i \sim \mathcal{N}(0, \frac{1}{\delta})$ *i.i.d., then:*

$$P\left(\sum_{i=1}^{d} U_i^2 \geq (1+\epsilon)d\right) \leq e^{-d\phi^*(\epsilon)}$$

with $\phi^*(\epsilon) = \frac{\epsilon - log(1+\epsilon}{2}$

Proof.

$$P(\sum_{i=1}^{d} U_i^2 \geq (1+\epsilon)d)$$

$$= P(e^{\lambda \sum_{i=1}^{d} U_i^2} \geq e^{\lambda(1+\epsilon)d})$$

$$\leq \frac{E[e^{\lambda \sum_{i=1}^{d} U_i^2}]}{e^{\lambda(1+\epsilon)d}}$$

$$\leq \frac{E[e^{\lambda U_1^2}]^d}{e^{\lambda(1+\epsilon)d}}$$

$$\leq e^{-d(\lambda(1+\epsilon) - \phi(\lambda))}$$

where $\phi(\lambda) = log(E[e^{\lambda U_i^2}]) = log(\int_{-\infty}^{+\infty} e^{\lambda u^2} \frac{e^{-\frac{u^2}{2}} du}{\sqrt{2\pi}}) = \frac{-1}{2} log(1 - 2\lambda)$

So for $\lambda > 0$, $P(\sum_{i=1}^{d} U_i^2 \geq (1+\epsilon)d) \leq e^{-dg(\lambda)}$. Now we choose λ so as to minimize the upper bound found, i.e. such that $g'(\lambda) = 1 + \epsilon - \frac{1}{2}\frac{2}{1-2\lambda} = 0$.

This yields $\lambda^* = \frac{1}{2}\frac{\epsilon}{1+\epsilon}$, and so:

$$\phi^*(\epsilon) = g(\lambda*) = \frac{1}{2}(\epsilon - log(1+\epsilon))$$

□

So, by applying this lemma to the components of Y:

$$\begin{aligned}
&\mathrm{P}(\|\mathrm{Y}\|^2 \geq (1+\epsilon)\|y\|^2) \\
&\qquad = \mathrm{P}(\sum_{i=1}^{d} \mathrm{Y}_i^2 \geq (1+\epsilon)\|y\|^2) \\
&\qquad = \mathrm{P}(\sum_{i=1}^{d} \left(\frac{\sqrt{d}\mathrm{Y}_i}{\|y\|}\right)^2 \geq d(1+\epsilon)) \\
&\qquad \leq e^{-d\phi^*(\epsilon)}
\end{aligned}$$

Similarly, $\mathrm{P}(\|\mathrm{Y}\|^2 \geq (1+\epsilon)\|y\|^2) \leq e^{-d\phi^*(\epsilon)}$

Now let us look at the following probability:

$$\begin{aligned}
p &= \mathrm{P}(\bigcap_{1\leq i\neq j\leq n} (\|\mathrm{A}(\mathrm{X}_i-\mathrm{X}_j)\|^2 \leq (1+\epsilon)\|\mathrm{X}_i-\mathrm{X}_j\|^2) \cap (\|\mathrm{A}(\mathrm{X}_i-\mathrm{X}_j)\|^2 \geq (1-\epsilon)\|\mathrm{X}_i-\mathrm{X}_j\|^2)) \\
&= 1-\mathrm{P}(\bigcup_{1\leq i<j\leq n} (\|\mathrm{A}(\mathrm{X}_i-\mathrm{X}_j)\|^2 > (1+\epsilon)\|\mathrm{X}_i-\mathrm{X}_j\|^2) \cap (\|\mathrm{A}(\mathrm{X}_i-\mathrm{X}_j)\|^2 < (1-\epsilon)\|\mathrm{X}_i-\mathrm{X}_j\|^2)) \\
&\geq 1-\sum_{1\leq i<j\leq n} (\mathrm{P}(\|\mathrm{A}(\mathrm{X}_i-\mathrm{X}_j)\|^2 > (1+\epsilon)\|\mathrm{X}_i-\mathrm{X}_j\|^2) + \mathrm{P}(\|\mathrm{A}(\mathrm{X}_i-\mathrm{X}_j)\|^2 < (1-\epsilon)\|\mathrm{X}_i-\mathrm{X}_j\|^2)) \\
&\geq 1-n(n-1)e^{-d\phi^*(\epsilon)}
\end{aligned}$$

So, for d such that $e^{-d\phi^*(\epsilon)} < \frac{1}{n^2}$, $p \geq 1 - \frac{n(n-1)}{n^2} > 0$ □

Lecture 4 (2 hours, notes by Gabriel Bathie)
14th February 2020

IV ON OVERFITTING AND HOW TO AVOID IT

The course starts with a remark on expectation of maximum and maximum of expectation :

IV.1 REMARK. Let $\mathrm{X}_1, \ldots, \mathrm{X}_n$ be i.i.d. random variables.
Then $\mathbb{E}[\max_i \mathrm{X}_i] \geq max_i \mathbb{E}[\mathrm{X}_i]$.

IV.2 EXAMPLE. If the $\mathrm{X}_i \sim \mathcal{N}(0,1)$, then $\forall i, \mathbb{E}[\mathrm{X}_i] = 0 \Rightarrow \max_i \mathbb{E}[\mathrm{X}_i] = 0$.
Yet, one can show (Exercise) that $\mathbb{E}[\max_i \mathrm{X}_i] \sim \sqrt{2\log(n)}$.

IV.3 EXAMPLE. Bit Guessing game

Consider the following game : the professor chooses secretely a uniformly random 10-bit string. Then, every student tries to guess the bit string.

Let us consider the random variable X_i = number of bits that student i guessed correctly. On average, a student will get 5 bits correctly. However, on average, the student who guesses the most bits correctly will have guessed correctly 7 or 8 bits.

However, if we restart the game, that student will most likely not be the have the highest number of correct guesses again. It can be seen as a kind of overfitting for a specific bit-string.

IV.1 *Case of the k-Nearest Neighbors algorithm*

When doing k-Nearest Neighbors classification, we must *a priori* choose a good value for k. However, if we take k too small (e.g. 1 or 2), we will be overfitting (the model does not generalize), and if we take k too large, we will be underfitting (the model can not learn).

Let us first look at a specific case. The dataset[1] is data about US citizens in 1994, and contains information like age and number of years of education. They are classified in two categories, depending on whether their annual income is lower than \$50.000.

First, one can remark that there is no absolute answer for a given pair (age, education), but rather a probability. Therefore, the Bayes classifier is not perfect (but still optimal).

We would like to get a classifier that is as close as possible to the Bayes classifier. However, we do not know the Bayes classifier for the dataset.

Instead, we can do the following to evaluate a model $\mathcal{M}$ (that will give us a classifier): we create a similar problem, and use it to generate fake data for which we will know the Bayes classifier. In our setting, creating a similar problem means define a probability $p = p(age, education)$ such that the dataset is a likely outcome of this probability distribution.

We then run our model on the fake data, and obtain a classifier $\mathcal{C}$.

Remember that we have seen that the risk of a classifier can be expressed as a function of the risk of the Bayes classifier plus the difference between $\mathcal{C}$ and the Bayes classifier (BC):

$$R(\mathcal{C}) = L^* + \mathbb{E}\left[2\left|\eta(X) - \frac{1}{2}\right| \mathbb{1}_{\mathcal{C}(X)\neq BC(X)}\right]$$

We use this formula to compare $\mathcal{C}$ to the Bayes classifier: if they differ a lot, i.e. if our model did not manage to learn the model of the fake (yet similar) data, it is unlikely that it will learn the model of the actual data.

IV.2 *Computing the risk of a classifier*

We would now like to compute the risk of our k-Nearest Neighbors classifier, as a function of k, to find an optimal value. There are multiple methods to do so, and we will introduce two of them here: using a validation set, and cross validation.

[1] *Adult Data Set*, UCI ML repository `http://archive.ics.uci.edu/ml/datasets/Adult`

IV.2.1 Method 1: Validation set

Instead of using the whole dataset to train, we split it in two parts: the training set and the validation set.

We use the training set to, well, train the model, and then use the validation set to measure its accuracy. It is important that these two sets are disjoint because a lot of models are 100% accurate on all the examples they have seen (e.g. nearest neighbor classifiers).

There are two main questions:

- How to choose the validation set in the dataset?
- How big should the validation set be?

How to choose the validation set: it may be tempting to take the first p% of the dataset as the training set, and the remainder as the validation set, but this should be avoided: the layout of the data may contain some bias. Instead, the choice should involve randomization.

Size of the validation set: there is no absolute rule. The only requirement is that it should be large enough to allow the evaluation of the mean of a Bernoulli random variable ($\mathbb{1}_{\text{prediction(X)=label(X)}}$) to the desired accuracy.

Problem: When the dataset is small (because measurements are expensive, for example), this does not feel right. There is a workaround:

IV.2.2 Cross Validation

Cross validation (CV) aims to solve the previous problem, using all of the data, but yet, having disjoint training and validation sets.

The idea is the following : choose v such that the dataset $\mathcal{D}$ can be partitioned in v parts of equal size, $\mathrm{D}_1, \ldots, \mathrm{D}_v$.

Then, for $i = 1$ to v, build a classifier $\mathcal{C}_i(k)$ with validation set D_i and training set $\mathcal{D} \setminus \mathrm{D}_i$, and compute its loss $l_i(k)$.

One can compute which $\hat{k}$ has the highest accuracy by computing $\mathrm{L}(k) = \frac{1}{v} \sum_{i=1}^{v} l_i(k)$: we then have $\hat{k} \in \arg\min_k \mathrm{L}(k)$.

We then have to build a final classifier to solve the problem. A first idea could be to build a k-Nearest Neighbors classifier with parameter $\hat{k}$ and train it on the whole dataset, but there might be issues: perhaps $\hat{k}$ is optimal only when the dataset has size $\frac{v-1}{v}|\mathcal{D}|$.

Another idea is to keep the v classifier that we built for each k, and when we must make a prediction, use all of them and aggregate the results: in the case of classification, do a majority vote (there is no particular rule to handle ties). In the case of regression, take the average of all the outputs.

Finally, we have to choose v to run such an algorithm. Again, there is no universal rule: the most standard is to pick $v = 5$ to 10, but one may try low values such as $v = 2, 3$ or $v = |\mathcal{D}|$ (called "LOO", for Leave One Out, as the training composed of all the entries of the dataset bar one).

V OTHER ALGORITHM FOR CLASSIFICATION: CLASSIFICATION AND REGRESSION TREE (CART)

The CART algorithm is a classification algorithm built using basic blocks: simple binary decision rules.

Decision rules have the following form : for a variable v_j:

- if v_j is quantitative (i.e. it is a number), choose a threshold s and decide $v_j \leq s$ versus $v_j > s$.
- if v_j takes values in a set R, choose S $\subseteq$ R, and decide $v_j \in$ S versus $v_j \notin$ S.

In other words, we look at a variable and we split the dataset in two, according to a simple criterion on that variable. We then want to apply recursively and independently the process on each part of the dataset (that is, not necessarily using the same decision rules on each part) until we obtain only homogeneous datasets.

This process creates a decision tree that can be easily interpreted, where the leaves correspond to a set of individual mapped to a unique prediction.

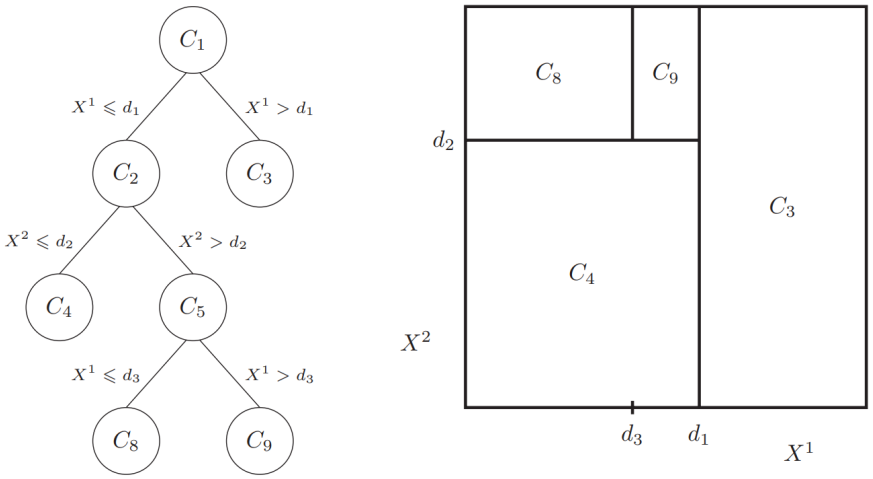


Figure 1: Example of classification tree

We would then like to know how to obtain the smallest tree that gives the highest accuracy on our dataset. However, this problem is really hard. Brieman[2] proposes a nontrivial greedy algorithm that yields good results.

The two main questions that we need to answer to run the algorithm are :

- How to do the first cut ? This is sufficient because every cut can be seen as the first cut on its own part of the dataset.

[2] *Classification and regression trees*, 1984

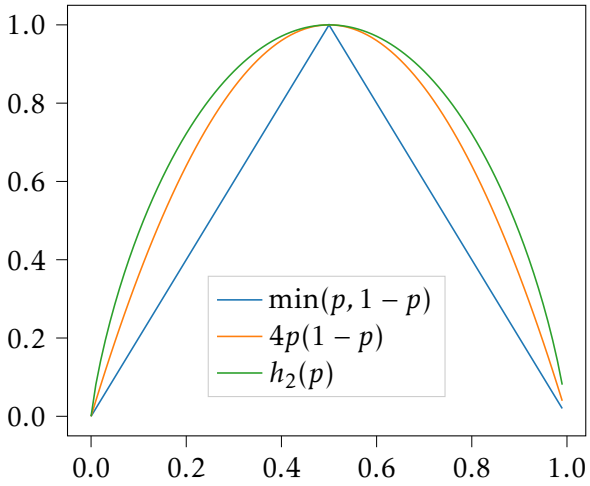


Figure 2: Plot of candidate heterogeneity measures : Shannon's surprisal, variance of Bernoulli RV, etc.

- When do we stop ?

In order to find the best cut, we first define *admissible cuts* : cuts such that it splits the dataset into two non-empty datasets. If v_j is quantitative and takes c_j values over the dataset, there are $c_j - 1$ admissible cuts. If v_j takes values in R of cardinality c_j, there are $2^{c_j} - 2$ admissible cuts (every S $\subseteq$ R except $\emptyset$ and R define admissible cuts).

We must then define a quality measure on a cut. We do that by defining a notion of heterogeneity of a node of the tree: a function such that

- it has value 0 if and only if a modality appears with frequency 1 in that node.
- it is maximal when the variance is large.

V.1 example. Binary classification :

We assume that the variable we are interested in takes two values, 0 or 1. When the frequency of 1 in a node is 0 or 1, our function must have value 0. Furthermore, when the frequency of 1 if 1/2, the variance is maximized, therefore the function should have value 1. Any function that satisfies these requirements may work, and using different functions yields different results in the algorithm.

V.2 example. Regression :

If we are interested in regression, we instead want the heterogeneity measure to be large when the points are far, for example we can use the empirical variance.

First cut: The algorithm to find the cut is then: try all possible cut for all possible variables, and choose the one that minimizes $h(c_1) - h(c_2)$, where c_1,c_2

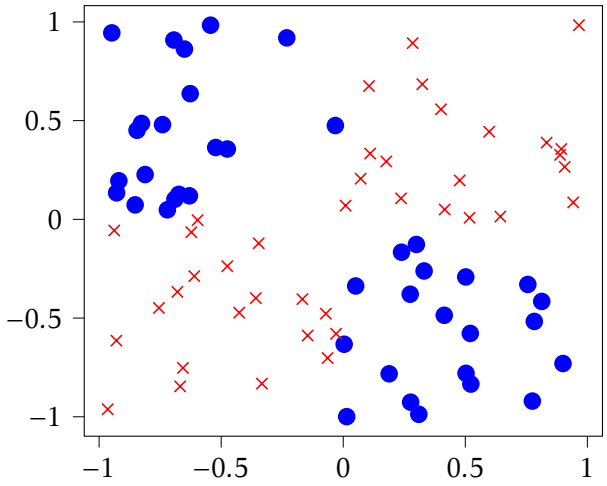


Figure 3: Example of dataset for which the first cut is very heterogeneous, but continuing the process yields a high quality estimator.

are the nodes obtained after cutting, and h is the heterogeneity function.

Stopping: A node is terminal if it is homogeneous (i.e. $h(node) = 0$) or if its size is smaller than some predetermined threshold.

Affectation: A terminal node has the value of the class that is the most represented inside.

After growing the tree fully, we have to prune it, otherwise we might overfit, for examples if the leaves contain one individual each.

To do that, we fix a number of leaves, and we remove exceeding nodes: starting from the leaves, we remove a split if $\Delta h = h(root) - h(left) - h(right)$ is small, that is if the split did not improve the homogeneity.

It is important to grow the tree and then prune it, and not to grow it and stop whenever we do a split with low Δh. If we did that, we might stop on first split of a highly heterogeneous dataset, where the second or third splits could have had a high Δh. See for example Figure 3.

VI STATISTICAL LEARNING FOR BINARY CLASSIFICATION

Lecture 5 (1h30 hours, notes by Justine Sauvage) 21st February 2020

Reference : *Mathematical Foundations of Statistical Learning*, Christophe Giraud (http://www.math.polytechnique.fr/xups/textes-provisoires13/giraud.pdf)

VI.1 *Modeling*

In this section, we will fix notations and framework for the following subsection(s).

First, note that we restrict our-self to binary classification.

Let :

- $\mathcal{X}$ be a measurable space
- $\mathcal{Y} = \{-1, 1\}$.
- (X, Y) be a random variable over $\mathcal{X} \times \mathcal{Y}$

Intuitively (X, Y) is a tuple ("input data","output data") and, as a remainder, or goal in to predict y given a x input by X.

32 DEFINITION (Classifier, Risk). A classifier is a function, that we will often note h from $\mathcal{X}$ to $\mathcal{Y}$. h can be seen as a "prediction" functions, try to class the element of $\mathcal{X}$ by their output. Hence, we defined the probability of misclassification, or risk :
$\mathrm{R}(h) = \mathbb{P}(\mathrm{Y} \neq h(\mathrm{X}))$

REMARK. The loss function in our case is $l : y, y' \to \mathbb{1}_{y \neq y'}$. If you recall the general definition of the risk was $\mathbb{E}(l(\mathrm{Y}, h(\mathrm{X}))$. In our case this is equal to $\mathbb{E}(\mathbb{1}_{\mathrm{Y} \neq h(\mathrm{X})} = \mathbb{P}(\mathrm{Y} \neq h(\mathrm{X}))$, hence the two definitions agree.

33 DEFINITION (Bayes classification). Let h_x be such that $h_x \in argmin_{h\ measurable} \mathrm{R}(h)$. h_x is called the Bayes classification.

8 PROPOSITION. Recall that we have $h_x(x) = sign(\mathbb{E}(\mathrm{Y} \| \mathrm{X} = x))$, or, equivalently,
$h_x(x) = \begin{cases} 1 \ if \ \mathbb{P}(\mathrm{Y} = 1 \| \mathrm{X} = x) \geq \mathbb{P}(\mathrm{Y} = -1 \| \mathrm{X} = x) \\ -1 \ otherwise \end{cases}$

Proof. Immediate with the fact that $\mathbb{E}(\mathrm{Y}|\mathrm{X} = x) = \mathbb{P}(\mathrm{Y} = 1 \| \mathrm{X} = x) - \mathbb{P}(\mathrm{Y} = -1 \| \mathrm{X} = x)$. □

VI.2 *Empirical risk minimization*

Our goal is to find a classifier that minimize the risk, ie a classifier that predict with a minimum amount of mistakes the label y of a given label x.

VI.2.1 Simplifying the problem

We first need to be able to compute the risk of a classifier, or at least to give a "good" bounding or estimation of it. Unfortunately, in "real life", we don't have access to the law of X and Y, but to a set of inputs associated to their respective outputs. To model that, we thus consider that we have a data set $\mathrm{D}_n = \{(\mathrm{X}_i, y_i), i \in [n]\} \in \mathcal{X} \times \mathcal{Y}$ where for all $i \in [n]$, $(\mathrm{X}_i, \mathrm{Y}_i) \sim (\mathrm{X}, \mathrm{Y})$ and all $(\mathrm{X}_i, \mathrm{Y}_i)_{i \in [n]}$ mutually independent.

Hence, there is not point in trying to compute the risk directly. So we focused on a similar quantity : the empirical risk (and hope for a way to link them with each other).

34 DEFINITION (Empirical risk, empirical law). Let h be a classifier. Its empirical risk is : $\mathrm{R}_n(h) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{(h(\mathrm{X}_i) \neq \mathrm{Y}_i}$.

Note that, if we define the empirical law $\mathbb{P}_n$ by : $\mathbb{P}_n = \frac{1}{n}\sum_{i=1}^{n}\delta_{(X_i,Y_i)}$, then : $R_n(h) = \mathbb{P}_n(Y \neq h(X))$

Then, we want to optimize the empirical risk over all measurable function of $\mathcal{Y}^{\mathcal{X}}$. But even it is usually still too complicated. So, the idea will be to minimize the risk on a small set of measurable functions. That why we introduce the dictionaries.

35 Definition (Dictionary). A dictionary $\mathcal{H}$ is a set of classifier.

Now, assume that we have found a $\mathcal{H}$ such that it is not "too complicated" to optimize on it. Let then :

36 Definition ($h_{\mathcal{H}}$). $h_{\mathcal{H}} = argmin_{h\in\mathcal{H}} R_n(h)$

Remark. In practice, finding a such dictionary is also a very important and difficult problem, for the rest of this part, we assume that we have found such a dictionary.

Now, the question is : How to certificate that our solution - say $h_{\mathcal{H}}$ - is "good" ?

VI.3 *Bias - Variance decomposition*

9 Proposition. $R(h_{\mathcal{H}}) - R(h_X) \geq 0$

Our goal is then to make this difference get as close as possible to zero (more the risk over the dicitonary is close to the risk over all measurable function, "better" it is).

But, event by considering this differences, the problem is still too complicated : we hence cut it into too error expression, and we will study both separately.

10 Proposition. $R(h_{\mathcal{H}}) - R(h_X) \leq [min_{h\in\mathcal{H}} R(h) - R(h_X)]_{approximation\ error} + [R(h_{\mathcal{H}}) - min_{h\in\mathcal{H}} R(h)]_{stochastic\ error}$

Let first concentrate on the stochastic error.

VI.4 *Misclassification probability of $h_{\mathcal{H}}$*

37 Definition (Shattering coefficient). Let $S_n(\mathcal{H}) = max_{x_1,x_2,..x_n\in\mathcal{X}^n} \#\{h(x_1), ..., h(x_n), h \in \mathcal{H}\}$.

$S_n(\mathcal{H})$ is the shattering coefficient. It gives the max number of different labelling of n points that the classifiers in $\mathcal{H}$ can produce.

38 Definition. Let now, $\mathcal{H}_l im = \{sign(\langle\beta, x\rangle)\beta \in \mathcal{R}^p\}$

11 Proposition. $S_n(\mathcal{H}_l im) \leq (n+1)^p$

VI.1 THEOREM (Control of the stochastic error). $\forall t > 0$, *with probability at least* $1 - e^{-t}$ *we have :*

$R(h_{\mathcal{H}}) - min_{h\in\mathcal{H}} R(h) \leq 4 * \sqrt{\frac{2log(2S_n(\mathcal{H}))}{n}} + \sqrt{\frac{2t}{n}}$

VI.5 *Dictionary selection*

Let $\{\mathcal{H}_1, \mathcal{H}_2, ...\mathcal{H}_M\}$ collection of classifiers. We would like to select among this collection, the dictionary with the smallest misclassification probability dictionaries.

VI.2 THEOREM. *Consider the following procedure :*

$\hat{m} = argmin_{m=1,..,M} R(h_{\mathcal{H}_{\Updownarrow}} + pen(\mathcal{H}_{\Updownarrow}$ *where*

$pen(\mathcal{H}) = 2\sqrt{\frac{2log(2S_n(\mathcal{H})}{n}}$.

Then, $forall t > 0$, *with probability* $1 - e^{-t}$, *we have*

$R(h_{\mathcal{H}_{\hat{m}}}) \leq min_{m=1,..,M}\{inf_{h\in\mathcal{H}_m} R(h) + 2pen(\mathcal{H}_m)\} + \sqrt{\frac{2log(M)+2t}{M}}$

REMARK.
- The second term is negligible as soon as $M = o(e^n)$
- Since we have $min_{h\in\mathcal{H}} R(h) \leq R(h_{\mathcal{H}})$, we have that : $R(h_{\mathcal{H}_{\hat{m}}}) \leq R(h_{m*}) + 2pen(\mathcal{H}_{m*}) + \sqrt{\frac{2log(M)+2t}{M}}$ with m* the argmin of the inf in the expression above.

REMARK. If you check the proof in the reference, it appears that we obtain the confidence interval for the misclassification probability :

$\mathcal{P}(R(h_{\mathcal{H}_{\hat{m}}}) \in [R_n(h_{\mathcal{H}_{\hat{m}}}) + / - \delta(\hat{m}, t)]) \geq 1 - e^{-t}$

with $\delta(m, t) = pen(\mathcal{H}_m) + \sqrt{\frac{log(M)+t}{2M}}$

VI.6 *VC dimension*

39 DEFINITION (Vapnik-Chervonenski dimension). $VC - dim(\mathcal{H}) = sup\{d \in \mathcal{N}, S_d(\mathcal{H}) = 2^d\}$

(translate : to which extent, $\mathcal{H}$ perfectly model the set of measurable functions in $\mathcal{X}^{\{+1,-1\}}$).

12 PROPOSITION (Saver's Lemma). If $d_{\mathcal{H}} = VC(\mathcal{H})$ is finite.

Then, $\forall n \geq 0, S_n(\mathcal{H}) \leq (n+1)^{d_{\mathcal{H}}}$.

Lecture 6 (2 hours, notes by Dina El Zein and Emile Sorci)
28th February 2020

VII BAGGING

VII.1 *Bootstrap Aggregation*

INTRODUCTION TO BOOTSTRAP:

$X_1, ..X_n \sim P$ independent.

$\mu = \mathbf{E}[\mathbf{X}]$

$\widehat{\mu} = \bar{X}n = \frac{x_1+x_2+...x_n}{n}$

θ = sample property of X

$\widehat{\theta}n = \psi n(x_1, ...x_n)$

How to give an idea of the precision of my estimator?

case of μ CLT:

$\sigma^2 = \mathbf{E}[(\mathbf{X} - \mu^2)]$

$\sqrt{n}, \frac{\widehat{\mu}_n - \mu}{\sqrt{\widehat{\sigma}_n^2}} \rightarrow \mathrm{N}(0,1)$, with probability 95%

$\left|\sqrt{n}, \frac{\widehat{\mu}_n - \mu}{\sqrt{\widehat{\sigma}_n^2}}\right| \leq 1.96$ so $\mu \in \left[\mu_n \pm \frac{1.96\sqrt{\widehat{\sigma}_n^2}}{\sqrt{n}}\right]$

The problem here is that for distribution with heavy tails, the convergence of the CCT is slow.

$f_\mu(x) = \frac{c}{|1+(\mu-x)|^{\alpha+1}}$ s.t $\alpha > 2$

CASE OF θ OR IF THE CENTRAL LIMIT THEOREM IS NOT RELIABLE:

- reproduce the experiment k times, $\widehat{\theta}_n^1, \widehat{\theta}_n^2, ...\widehat{\theta}_n^k$
- the idea here is as follows: for n sufficiently large $\mathrm{P}_n \approx \mathrm{P} : for every \mathrm{A} \in \rho, \mathrm{P}_n(\mathrm{A}) \xrightarrow[n\to\infty]{} \mathrm{P}(\mathrm{A})$

BOOTSTRAP: (RE-SAMPLING):

- $\mathrm{X}_1^{\star,1}, ...\mathrm{X}_n^{\star,n}$ identically distributed $\mathrm{P}_n \rightarrow \widehat{\theta}_n^{\star,1} = \psi_n(\mathrm{X}_1^{\star,1}, ...\mathrm{X}_n^{\star,1})$
- $\mathrm{X}_1^{\star,k}, ...\mathrm{X}_n^{\star,k}$ identically distributed $\mathrm{P}_n \rightarrow \widehat{\theta}_n^{\star,k}$

VII.2 *Application to Supervised Learning*

here we have a weak learner (meaning doesn't have to be very efficient)

- $\psi_n(x,y)^n \rightarrow y^x$

  $(x_1, y_1), ...(x_n, y_n) \rightarrow h_n : x \rightarrow y$

- Meta Learner: Bagging (ψ_1)

  In: $(x_1, y_1)...(x_n, y_n)$

  out:$\bar{h}_n : x \rightarrow \mathrm{y}$

  for k=1 to K

compute a bootstrap sample $(x_1^{\star k}, y_1^{\star k})....(x_n^{\star} k, y_n^{\star k})$

call ψ_n on it to obtain $h_n^{\star k}$

end for

- $\widehat{h}_n = aggregation(h_n^{\star 1}, ...h_n^{\star k})$

- for Regression:

$$\text{aggregation}(h_n^{\star 1}, ...h_n^{\star k})(x) = \frac{1}{k} \sum_{i=1}^{k} h_n^{\star,k}(x)$$

- for classification (majority vote):

$$\text{aggregation}(h_n^{\star 1}, ...h_n^{\star k})(x) = \frac{1}{k} \underset{y \in \mathrm{Y}}{argmax} \sum_{i=1}^{k} \mathbb{1} h_n^{\star,k}(x) = y$$

Random Forest Algorithm: (bagging classification or regression trees)

- no prunning
- subsample the variables candidate for splits at every step
- grow the tree only up to depth d (d=2, 3, 4)

VIII BOOSTING

Read the slides

Lecture 7 (2 hours, notes by Zoé Varin)
13th March 2020

IX SUPPORT VECTOR MACHINES

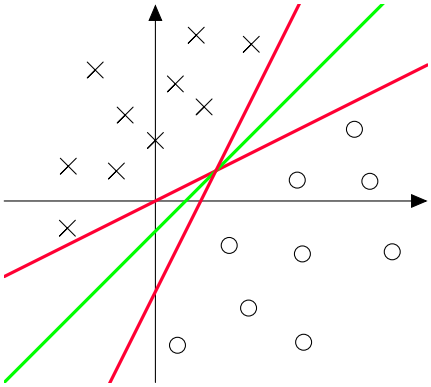
IX.1 *The linearly-separable case*

Let $\mathcal{X} \subseteq \mathbb{R}^p$ and $\mathcal{Y} = \{-1, 1\}$.
We assume that : $\exists w^* \in \mathbb{R}^p : \mathbb{P}(\mathrm{Y}\langle w^*, \mathrm{X}\rangle > 0) = 1$.

Remarks.
- We saw that this separation can be done using the perceptron algorithm
- We also remarked that affinely separable data can be reduced to the linear case by the mapping :

$$\begin{array}{ccc} \mathbb{R}^p & \to & \mathbb{R}^{p+1} \\ x = \begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix} & \longmapsto & \tilde{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_p \\ 1 \end{pmatrix} \end{array}$$

- illustration of the main objective :



We see that there can be many separators with no error, but some seem more robust (see the one associated to the green line). So we would like to have confidence margin.

40 Definition. A linear separator w is *compatible* with the dataset if :

$$\forall i \in \{1, \dots, n\},\ y_i \langle w, x_i \rangle > 0$$

41 Definition. The *margin* is :

$$m(w) = \min_{1 \le i \le n} d(x_i, \mathrm{H}_w)$$

where $\mathrm{H}_w = \{x \in \mathbb{R}^p : \langle w, x \rangle = 0\}$.

Among all the linear separators compatible with the dataset, we look for the one with highest margin.

13 Proposition. For every $x \in \mathbb{R}^p$ and $w \in \mathbb{R}^p \backslash \{0\}$,

$$d(x, \mathrm{H}_w) = \frac{|\langle w, x \rangle|}{\|w\|}$$

Proof. Let $y = x - \langle \frac{w}{\|w\|}, x \rangle \frac{w}{\|w\|}$ (y is the projection of x onto the hyperplane H_w), and let $z \in \mathrm{H}_w$.

First, denote that $y \in \mathrm{H}_w$, since $\langle w, y \rangle = \langle w, x \rangle - \frac{\langle w, x \rangle}{\|w\|^2} \langle w, x \rangle = 0$.

$$\begin{aligned}\|x-z\|^2 &= \|x-y+y-z\|^2 \\ &= \|x-y\|^2 + \|y-z\|^2 + 2\langle x-y, y-z\rangle \\ &= \frac{\langle w,x\rangle^2}{\|w\|^4}\|w\|^2 + \underbrace{\|y-z\|^2}_{>0} + 2\langle w,x\rangle \underbrace{\langle w, y-z\rangle}_{=0}\end{aligned}$$

So $d(x, \mathrm{H}_w) = \|x-y\| = \frac{|\langle w,x\rangle|}{\|w\|}$.

□

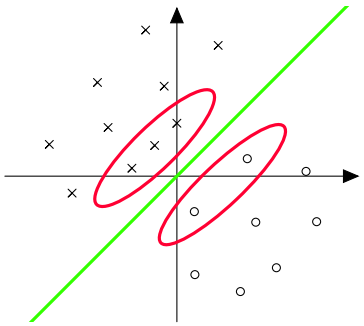
The optimal linear separator is then the solution of

$$\max_{w\in\mathbb{R}^p, \forall i, y_i\langle w,x_i\rangle=1} \min_{1\le i\le n} \frac{|\langle w, x_i\rangle|}{\|w\|} \tag{OPT1}$$

$$\begin{aligned}(\text{OPT1}) &= \max_{w\in\mathbb{R}^p, \min_{1\le i\le n} y_i\langle w,x_i\rangle=1} \min_{1\le i\le n} \frac{|\langle w, x_i\rangle|}{\|w\|} \\ &= \max_{w\in\mathbb{R}^p, \min_{1\le i\le n} y_i\langle w,x_i\rangle=1} \frac{1}{\|w\|} \underbrace{\min_{1\le i\le n} |\langle w, x_i\rangle|}_{=1} \\ &\Leftrightarrow \min_{w\in\mathbb{R}^p, \min_{1\le i\le n} y_i\langle w,x_i\rangle=1} \frac{\|w\|^2}{2} \qquad \text{(OPT2)}\end{aligned}$$

(OPT2) can be solved numerically (this is quadratic program, under linear constraints).

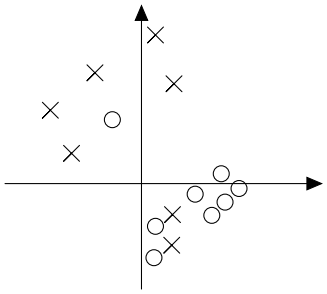
Remark.



One can see that the solution only depends on the points that are close to the hyperplane. The others can be moved a lot without changing the solution.

IX.2 *Extension to the non-linearly separable case*

In the first model (Hard-SVM) we didn't allow for exceptions. But now we would like to. This is the Soft-SVM model.

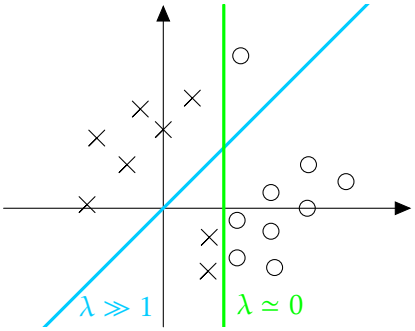


Then we define a new problem :

$$\max_{w\in\mathbb{R}^p,\forall i, y_i\langle w,x_i\rangle\geq 1-\alpha_i,\alpha_i\in[0,+\infty[} \lambda\frac{\|w\|^2}{2}+\frac{1}{n}\sum_{i=1}^{n}\alpha_i \qquad \text{(OPT3)}$$

The idea with the α_i's is that we pay by how much a point is far from the margin. Then λ is a factor depending on how much we're ready to pay for a mistake.

Illustration.



Here the two lines represent two legitimate solutions. We choose one of them by adjusting the λ parameter.
If $\lambda \simeq 0$, we stick as much as possible to the data. On the contrary, a parameter $\lambda \gg 1$ allows to generalize well.

Remark. One can rewrite $y_i\langle w, x_i\rangle \geq 1-\alpha_i, \alpha_i \in [0,+\infty[$ as $\alpha_i = \max(1 - y_i\langle w, x_i\rangle, 0) = (1 - y_i\langle w, x_i\rangle)_+$. Then,

$$\text{(OPT3)} = \min_{w\in\mathbb{R}^p} \lambda\frac{\|w\|^2}{2}+\frac{1}{n}\sum_{i=1}^{n}(1-y_i\langle w,x_i\rangle)_+$$

Writing $l(u) := (1-u)_+$, and denoting that $\forall u, l(u) \geq \mathbb{1}_{\{u<0\}}$, we get :

$$\min_{w\in\mathbb{R}^p}\frac{1}{n}\sum_{i=1}^{n} l(1-y_i\langle w,x_i\rangle)+\lambda\frac{\|w\|^2}{2} \geq \min_{w\in\mathbb{R}^p}\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}_{\{1-y_i\langle w,x_i\rangle<0\}}+\lambda\frac{\|w\|^2}{2}$$

So Soft-SVM appears a posteriori as a convexification of the ERM (empirical risk minimization) problem with quadratic penalization $\lambda\frac{\|w\|^2}{2}$.

IX.3 *Optimization of OPT3*

$$(\text{OPT3}) = \min_{w\in\mathbb{R}^p} \underbrace{\frac{1}{n}\sum_{i=1}^{n} l(1-y_i\langle w, x_i\rangle) + \lambda\frac{\|w\|^2}{2}}_{=:f(w)}$$

f is convex (since $(1-.)_+$ is convex and $y_i\langle ., x_i\rangle$ is linear), and even λ-strictly convex.

(f is λ-strictly convex if $\forall x \in \mathbb{R}^p, \exists g_x \in \mathbb{R}^p, \forall y \in \mathbb{R}^p, f(y) \geq f(x) + \langle g_x, y - x\rangle + \frac{\lambda}{2}\|y-x\|^2$.)

One can check the following proposition :

14 Proposition. a function f is λ-strictly convex $\Leftrightarrow x \mapsto f(x) - \frac{\lambda}{2}\|x\|^2$ is convex. If $\forall x, \|x_i\| \leq \mathrm{M}$, then on the set $\mathrm{B} := \{w \in \mathbb{R}^p : \|w\| \leq \mathrm{R}\}$, f is $\mathrm{L} := \mathrm{M} + \lambda\mathrm{R}$-

Lipschitz : $\forall w, w' \in \mathrm{B}, |f(y) - f(x)| \leq (\mathrm{M} + \lambda\mathrm{R})\|y - x\|$

Now let's focus on an algorithm for optimizing a L-Lipschitz λ-strictly convex function, whose main id

We set :

$$\begin{cases} w_0 = 0 \in \mathbb{R}^p \\ \forall t \geq 0, w_{t+1} = w_t - \gamma_t g_t \end{cases}$$

where g_t is the subgradient of f at w_t, and γ_t is a *learning rate* to be chosen correctly.

2 Theorem. if $\gamma_t = \frac{1}{\lambda(t+1)}$, then

$$f\left(\frac{1}{\mathrm{T}}\sum_{t=0}^{\mathrm{T}-1} w_t\right) \leq \min f + \frac{\mathrm{L}^2(1+\ln \mathrm{M})}{\lambda \mathrm{T}}$$

Remark. Be careful, the sequence itself does not necessarily converge, but the average does !

Proof. Let $w^* = \arg\min_{w\in\mathbb{R}^p} f$.

We know that $f(w^*) \geq f(w_t) + \langle g_t, w^* - w_t\rangle + \frac{\lambda}{2}\|w_t - w^*\|^2$. Then,

$$
\begin{aligned}
f(w_t) - f(w^*) &\leq \langle g_t, w_t - w^* \rangle - \frac{\lambda}{2}\|w_t - w^*\|^2 \\
&= \langle \frac{1}{\gamma_t}(w_t - w_{t+1}), w_t - w^* \rangle - \frac{\lambda}{2}\|w_t - w^*\|^2 \\
&= \frac{1}{2\gamma_t}\left(\|w_t - w_{t+1}\|^2 + \|w_t - w^*\|^2 - \|w_{t+1} - w^*\|^2\right) - \frac{\lambda}{2}\|w_t - w^*\|^2 \\
&= -\frac{(t+1)\lambda}{2}\|w_{t+1} - w^*\|^2 + \frac{t\lambda}{2}\|w_t - w^*\|^2 + \frac{(t+1)\lambda}{2}\|\gamma_t g_t\|^2 \\
&\leq -\frac{(t+1)\lambda}{2}\|w_{t+1} - w^*\|^2 + \frac{t\lambda}{2}\|w_t - w^*\|^2 + \frac{\mathrm{L}^2}{2(t+1)\lambda}
\end{aligned}
$$

This computation uses the fact that $\forall a, b, 2\langle a, b\rangle = \|a\|^2 + \|b\|^2 - \|a - b\|^2$ and $\|g_t\|^2 \leq \mathrm{L}^2$. One can show that $\|g_t\|^2 \leq \mathrm{L}^2$:

- the hypothesis f λ-strictly convex, using $x = w_t$ and y such that $y - x = g_t$, gives : $f(y) \geq f(x) + (1 + \frac{\lambda}{2})\|g_t\|^2$
- Then, $\|g_t\|^2 \leq \frac{1}{1+\frac{\lambda}{2}}\left(f(y) - f(x)\right)$
- finally, f L-Lipschitz gives : $\|g_t\|^2 \leq \frac{\mathrm{L}}{1+\frac{\lambda}{2}}\|g_t\|$ and we immediately get the result.

Then, summing this inequality $f(w_t) - f(w^*) \leq \langle g_t, w_t - w^*\rangle - \frac{\lambda}{2}\|w_t - w^*\|^2 \leq -\frac{(t+1)\lambda}{2}\|w_{t+1} - w^*\|^2 - \frac{t\lambda}{2}\|w_t - w^*\|^2 + \frac{\mathrm{L}^2}{2(t+1)\lambda}$ from 0 to $\mathrm{T} - 1$ gives :

$$
\begin{aligned}
\sum_{t=0}^{\mathrm{T}-1} f(w_t) - f(w^*) &\leq -\frac{\mathrm{T}\lambda}{2}\|w_{\mathrm{T}} - w^*\|^2 + \frac{0\lambda}{2}\|w_0 - w^*\|^2 + \frac{\mathrm{L}^2}{2\lambda}\sum_{t=0}^{\mathrm{T}-1}\frac{1}{t+1} \\
&\leq \frac{\mathrm{L}^2}{2\lambda}\left(1 + \ln(\mathrm{T})\right)
\end{aligned}
$$

Finally, by convexity,

$$
f\left(\frac{1}{\mathrm{T}}\sum_{t=0}^{\mathrm{T}-1} w_t\right) \leq \frac{1}{\mathrm{T}}\sum_{t=0}^{\mathrm{T}-1} f(w_t) \leq f(w^*) + \frac{\mathrm{L}^2(1 + \ln(\mathrm{T}))}{2\lambda\mathrm{T}}
$$

□

IX.4 *Dual formulation*

IX.5 *Kernel tricks*