Consistent Minimization of Clustering Objective Functions

Ulrike von Luxburg[1], Sébastien Bubeck[2], Stefanie Jegelka[1], Michael Kaufmann[3]

[1]Max Planck Institute for Biological Cybernetics, Tübingen, Germany; [2]INRIA Futurs Lille, France; [2]University of Tübingen, Germany

Discrete optimization approach to clustering

Given n data points and a clustering quality function Q_n (sum to cluster centers, graph cuts, ...)

Among all partitions of the data set, find the one with optimal quality value $Q_n(f)$

In practice: often NP hard ...

Clustering in a statistical setting

Data points have been sampled from some underlying space $\mathcal{X}$

Among all partitions of the underlying space, construct the one with optimal quality value $Q(f)$

Given a finite sample only: ϟϟϟ $f^* = \operatorname{argmin} Q(f)$ ϟϟϟ $\rightsquigarrow$ $f_n = \operatorname{argmin} Q_n(f)$

Need statistical consistency: $Q(f_n) \to Q(f^*)$

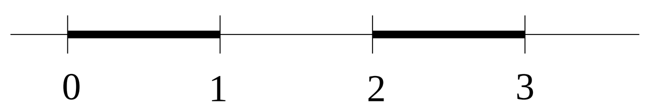
Optimal discrete solution $\Longrightarrow$ consistency? No!!!

Intuition based on statistical learning theory for classification:

- The class of "all possible partitions" is too large (K^n functions, is exponential in n)
- Consistency can only be guaranteed for "small" function classes (e.g., finite VC dim)
- Plausible: similar reasoning applies to clustering ...

Example for overfitting in clustering:

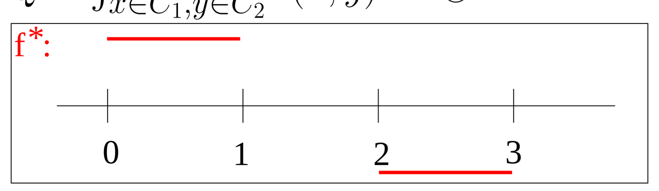
- Space $\mathcal{X} = [0, 1] \cup [2, 3]$ with uniform distribution
- Similarity function: $s(x, y) = 1$ if x, y in same interval, 0 otherwise



- Quality function: minimize between-cluster similarity:

Whole space:

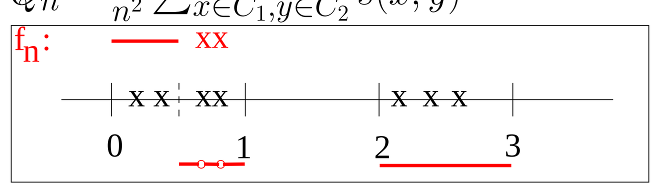
$Q = \int_{x \in C_1, y \in C_2} s(x, y) \, dP \otimes P$



$Q(f^*) = 0$

Finite sample case:

$Q_n = \frac{1}{n^2} \sum_{x \in C_1, y \in C_2} s(x, y)$



$Q_n(f_n) = 0$

$Q(f_n) = \int_{x \in C_1, y \in C_2} s(x, y) \, dP \otimes P$

$= \int_0^{1/2} \int_{1/2}^1 1 \, dP \otimes P = 1/16$

Thus $Q(f_n) \not\to Q(f^*)$, no consistency!

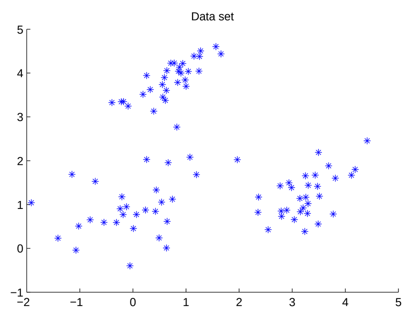
Optimal discrete solution $\Longrightarrow$ overfitting!!!
Need to optimize over "small" function class!!!

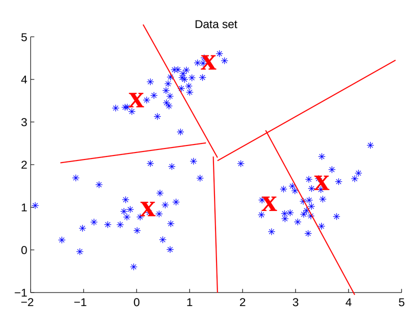
- $\mathcal{F}_n$ should be small enough to avoid overfitting
- $\mathcal{F}_n$ should be rich enough to approximate any partition of the underlying space (for large n)
- We need to be able to find the global minimizer of Q_n in $\mathcal{F}_n$.

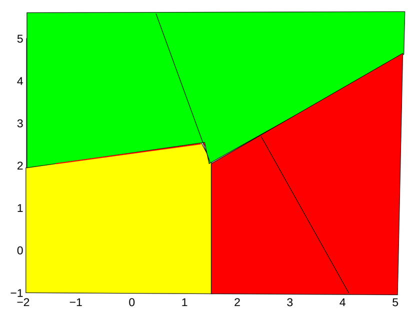
Idea: use functions which are constant on local neighborhoods

Nearest neighbor clustering (NNC)

- Subsample $m \approx \log(n)$ seed points from the data points
- Build the neighborhood cells $A_1, \dots, A_m$ by assigning all data points to their closest seed point
- $\mathcal{F}_n :=$ functions which are constant on all cells A_j
- $f_n := \operatorname{argmin}_{f \in \mathcal{F}_n} Q_n(f)$







When is nearest neighbor clustering consistent?

General setting:

- $\mathcal{F} := \{f : \mathbb{R}^d \to \{1, \dots, K\} \mid f$ continuous $\mathbb{P}$-a.e. and $A(f)$ is true$\}$
- $\mathcal{F}_n := \{f : \mathbb{R}^d \to \{1, \dots, K\} \mid f$ satisfies $f(x) = f(\mathrm{NN}_m(x))$, and $A_n(f)$ is true$\}$

(where $A(f)$ and $A_n(f)$ are predicates to define the classes)

- $f^* \in \operatorname{argmin}_{f \in \mathcal{F}} Q(f)$ and $f_n \in \operatorname{argmin}_{f \in \mathcal{F}_n} Q_n(f)$

Theorem (General consistency of nearest neighbor clustering) *Assume that:*

1. $Q_n(f)$ is a consistent estimator of $Q(f)$ which converges sufficiently fast:

$$\forall \varepsilon > 0, \quad K^m (2n)^{(d+1)m^2} \sup_{f \in \widetilde{\mathcal{F}}_n} \mathbb{P}(|Q_n(f) - Q(f)| > \varepsilon) \to 0.$$

2. $A_n(f)$ is an estimator of $A(f)$ which is "consistent" in the following way:

$$\mathbb{P}(A_n(\widetilde{f}^*) \text{ true}) \to 1 \qquad \text{and} \qquad \mathbb{P}(A(f_n) \text{ true}) \to 1.$$

3. Q is uniformly continuous with respect to the 0-1-distance L_n between $\mathcal{F}$ and $\mathcal{F}_n$:

$$\forall \varepsilon > 0 \; \exists \delta(\varepsilon) > 0 \; \forall f \in \mathcal{F} \; \forall g \in \mathcal{F}_n : \quad \mathrm{L}_n(f, g) \le \delta(\varepsilon) \implies |Q(f) - Q(g)| \le \varepsilon.$$

4. $m(n) \to \infty$.

Then nearest neighbor clustering is weakly consistent: $Q(f_n) \to Q(f^*)$ in probability.

Proof: Introduce functions

$$f_n^* \in \operatorname{argmin}_{f \in \mathcal{F}_n} Q(f) \qquad \text{and} \qquad \widetilde{f}^*(x) := f^*(\mathrm{NN}_m(x)).$$

Split in approximation and estimation error:

$$\mathbb{P}\big(Q(f_n) - Q(f^*) \ge \varepsilon\big) \le \mathbb{P}\big(Q(f_n) - Q(f_n^*) \ge \varepsilon/2\big) + \mathbb{P}\big(Q(f_n^*) - Q(f^*) \ge \varepsilon/2\big).$$

Estimation error:

- symmetrization by a ghost sample (attention, we do not assume $\mathbb{E}Q_n = Q$)
- use Assumption (1)

Approximation error: Split in cases "$A_n(\widetilde{f}^*)$ true" and "$A_n(\widetilde{f}^*)$ false"

$$\mathbb{P}\big(Q(f_n^*) - Q(f^*) \ge \varepsilon\big) \le \mathbb{P}(A_n(\widetilde{f}^*) \text{ false}) + \mathbb{P}\big(\widetilde{f}^* \in \mathcal{F}_n \text{ and } Q(\widetilde{f}^*) - Q(f^*) \ge \varepsilon\big)$$

First term $\to 0$ by Assumption (2)

Second term $\to 0$: show that under Assumption (4), the distance between $f(\cdot)$ and $f(\mathrm{NN}_m(\cdot))$ goes to 0 uniformly in f and use Assumption (3). □

Theorem (Consistency of NNC for common objective functions)
Use predicates specifying a minimal cluster size:

$$A(f) \text{ is true } :\iff \mathrm{vol}(f_k) > a \;\; \forall k = 1, \dots, K$$
$$A_n(f) \text{ is true } :\iff \mathrm{vol}_n(f_k) > a_n \;\; \forall k = 1, \dots, K$$

Assume that $a_n \to a$, $m(n) \to \infty$, $m^2 \log n / (n(a - a_n)^2) \to 0$.
Then nearest neighbor clustering is consistent for the following clustering objective functions:
cut, ratio cut, normalized cut, modularity, K-means objective function, ratio of between- and within-cluster similarity, ...

Experiments: Ncut and K-means objective functions

Setup of the experiments:

- Compare nearest neighbor clustering to spectral clustering and K-means algorithm
- Numeric data sets and graph-based data sets
- Several random restarts for all algorithms, results averaged over many train/test splits
- To compute "test quality", use greedy extension operator

Implementation of nearest neighbor clustering: using branch and bound

TO DO Steffi: can I get your figures 3.4.5 and 3.4.5 as pdf???

Results: First line: training quality, second line: test quality

Numeric	K-means obj.fct.		Ncut obj.fct	
data sets	K-means alg.	NNC	spectral cl.	NNC
breast-c.	6.95 ± 0.19	7.04 ± 0.21	0.11 ± 0.02	0.09 ± 0.02
	7.12 ± 0.20	7.12 ± 0.22	0.22 ± 0.07	0.21 ± 0.07
diabetis	6.62 ± 0.22	6.71 ± 0.22	0.03 ± 0.02	0.03 ± 0.02
	6.72 ± 0.22	6.72 ± 0.22	0.04 ± 0.03	0.05 ± 0.05
german	18.26 ± 0.27	18.56 ± 0.28	0.02 ± 0.02	0.02 ± 0.02
	18.35 ± 0.30	18.45 ± 0.32	0.04 ± 0.08	0.03 ± 0.03
heart	10.65 ± 0.46	10.77 ± 0.47	0.18 ± 0.03	0.17 ± 0.02
	10.75 ± 0.46	10.74 ± 0.46	0.28 ± 0.03	0.30 ± 0.07
splice	68.99 ± 0.24	69.89 ± 0.24	0.36 ± 0.10	0.44 ± 0.16
	69.03 ± 0.24	69.18 ± 0.25	0.58 ± 0.09	0.66 ± 0.18
bcw	3.97 ± 0.26	3.98 ± 0.26	0.02 ± 0.01	0.02 ± 0.01
	3.98 ± 0.26	3.98 ± 0.26	0.04 ± 0.01	0.08 ± 0.07
ionosph.	25.72 ± 1.63	25.77 ± 1.63	0.06 ± 0.03	0.04 ± 0.01
	25.76 ± 1.63	25.77 ± 1.63	0.12 ± 0.11	0.14 ± 0.12
pima	6.62 ± 0.22	6.73 ± 0.23	0.03 ± 0.03	0.03 ± 0.03
	6.73 ± 0.23	6.73 ± 0.23	0.05 ± 0.04	0.09 ± 0.13
cellcycle	0.78 ± 0.03	0.78 ± 0.03	0.12 ± 0.02	0.10 ± 0.01
	0.78 ± 0.03	0.78 ± 0.02	0.16 ± 0.02	0.15 ± 0.03

Network data	NNC	spectral cl.
ecoli.interact	0.06	0.06
ecoli.metabol	0.03	0.04
helico	0.16	0.16
beta3s	0.00	0.00
AS-19971108	0.02	0.02
AS-19980402	0.01	1.00
AS-19980703	0.02	0.02
AS-19981002	0.04	0.04
AS-19990114	0.08	0.05
AS-19990402	0.11	0.10
netscience	0.01	0.01
polblogs	0.11	0.11
power	0.00	0.00
email	0.27	0.27
yeastProtInt	0.04	0.06
protNW1	0.00	0.00
protNW2	0.08	1.00
protNW3	0.01	0.80
protNW4	0.03	0.76

- Training results: NNC can compete with K-means and spectral clustering ☺
- Test set results: not much better for NNC than for K-means and spectral clustering ☹
  Explanation: both K-means and spectral clustering also use small function classes ...

Conclusions:
To avoid overfitting in clustering: use a small function class
Do not attempt to solve the discrete problem exactly
One simple alternative: nearest neighbor clustering