A PAC-Theory of Clustering with Advice

Hassan Ashtiani

School of Computer Science
University of Waterloo

Postgraduate Affiliate
Vector Institute

May 2018

UNIVERSITY OF WATERLOO

Clustering

- Clustering is the task of automatically partitioning a set of instances into "meaningful" subsets.
- An essential tool for exploratory data analysis.
  - Marketing
  - Social Sciences
  - Data Management Systems
  - Computer Vision
  - Medicine
  - Urban Planning
  - ...
- Tons of algorithmic choices with conflicting outcomes
  - Clustering algorithms
  - Parameters, distances, preprocessing techniques

Clustering

- Clustering is the task of automatically partitioning a set of instances into "meaningful" subsets.
- An essential tool for exploratory data analysis.
  - Marketing
  - Social Sciences
  - Data Management Systems
  - Computer Vision
  - Medicine
  - Urban Planning
  - ...
- Tons of algorithmic choices with conflicting outcomes
  - Clustering algorithms
  - Parameters, distances, preprocessing techniques

Clustering is generally an under-specified task.

Market Segmentation

Clustering the consumers based on

- Demographic (age, sex, address, etc.)
- Behavior (rating, feedback, etc.)

Understand the market, send the right message to the right consumer.

Market Segmentation

Just feed the data to an off-the-shelf clustering algorithm (?!)

Market Segmentation

Just feed the data to an off-the-shelf clustering algorithm (?!)

- Usually impossible!
- The desired clustering is task-specific! (for movies or furnitures?)

Market Segmentation

Just feed the data to an off-the-shelf clustering algorithm (?!)

- Usually impossible!
- The desired clustering is task-specific! (for movies or furnitures?)

Model selection requires exploiting domain knowledge.

Outline

- Model Selection for Clustering
  - An expert provides some advice about the ground truth.
  - “Advice complexity” of clustering?

Outline

- Model Selection for Clustering
  - An expert provides some advice about the ground truth.
  - "Advice complexity" of clustering?

- Efficient Clustering with Advice
  - Optimizing K-means, K-median, K-center, ... objective functions is NP-hard
  - Can using advice have computational benefits?

Outline

- Model Selection for Clustering
  - An expert provides some advice about the ground truth.
  - "Advice complexity" of clustering?
- Efficient Clustering with Advice
  - Optimizing K-means, K-median, K-center, ... objective functions is NP-hard
  - Can using advice have computational benefits?
- Learning Mixture Models

Incorporating Domain Knowledge

- Trial-and-error/Intuitions

Incorporating Domain Knowledge

- Trial-and-error/Intuitions
- Off-line Models
  - Constrained Clustering (Wagstaff et. al (2000))
  - Demonstration-based Clustering (Ashtiani, Ben-David (2015))
- Interactive Clustering
  - Merge/split Model (Balcan and Blum (2008))
  - Same-cluster Queries (Ashtiani, Kushagra, Ben-David (2016))

Incorporating Domain Knowledge

- Trial-and-error/Intuitions
- Off-line Models
  - Constrained Clustering (Wagstaff et. al (2000))
  - **Demonstration-based Clustering (Ashtiani, Ben-David (2015))**
- Interactive Clustering
  - Merge/split Model (Balcan and Blum (2008))
  - Same-cluster Queries (Ashtiani, Kushagra, Ben-David (2016))

Demonstration-based Clustering

Goal: learn a mapping under which k-means outputs an approximation to the *ground truth*.

Protocol

- Take a small subset of data.
- Have a domain expert cluster that subset.
- Learn a "transformation" consistent with that clustering.
- Cluster the rest of data!

Demonstration-based Clustering

Goal: learn a mapping under which k-means outputs an approximation to the *ground truth*.

Protocol

- Take a small subset of data.
- Have a domain expert cluster that subset.
- Learn a "transformation" consistent with that clustering.
- Cluster the rest of data!

Looking for an agnostic guarantee.

$$d(C^*, \hat{C}) \leq \inf_{f \in \mathcal{F}} d(C^*, C^f) + \epsilon$$

Demonstration-based Clustering

Goal: learn a mapping under which k-means outputs an approximation to the *ground truth*.

Protocol

- Take a small subset of data.
- Have a domain expert cluster that subset.
- Learn a "transformation" consistent with that clustering.
- Cluster the rest of data!

Looking for an agnostic guarantee.

$$d(C^*, \hat{C}) \leq \inf_{f \in \mathcal{F}} d(C^*, C^f) + \epsilon$$

What is the distance?

$$d(C^*, \hat{C}) = \min_{\sigma \in \pi^k} \frac{1}{|X|} \sum_{i=1}^{k} |C_i^* \Delta \hat{C}_{\sigma(i)}|$$

Demonstration-based Clustering: Results

Advice complexity of end-to-end representation learning?

- Let $m_{\mathcal{F}}$ be the advice complexity w.r.t. class $\mathcal{F}$ of mappings.

Demonstration-based Clustering: Results

Advice complexity of end-to-end representation learning?

- Let $m_{\mathcal{F}}$ be the advice complexity w.r.t. class $\mathcal{F}$ of mappings.

Theorem

Under appropriate uniqueness conditions we have

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \widetilde{O}\left(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}\right)$$

.

- Pseudo-dimension measures the capacity of $\mathcal{F}$

Demonstration-based Clustering: Results

Advice complexity of end-to-end representation learning?

- Let $m_{\mathcal{F}}$ be the advice complexity w.r.t. class $\mathcal{F}$ of mappings.

Theorem

Under appropriate uniqueness conditions we have

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \widetilde{O}\left(\frac{k + Pdim(\mathcal{F}) + \log(\frac{1}{\delta})}{\epsilon^2}\right)$$

.

- Pseudo-dimension measures the capacity of $\mathcal{F}$

Corollary

Let $\mathcal{F}$ be a set of *linear* mappings from $\mathbb{R}^{d_1}$ to $\mathbb{R}^{d_2}$. Then

$$m_{\mathcal{F}}(\epsilon, \delta) \leq \widetilde{O}\left(\frac{k + d_1 d_2 + \log(\frac{1}{\delta})}{\epsilon^2}\right)$$

Incorporating Domain Knowledge

- Trial-and-error/Intuitions
- Off-line Models
  - Constrained Clustering (Wagstaff et. al (2000))
  - Demonstration-based Clustering (Ashtiani, Ben-David (2015))
- Interactive Clustering
  - Merge/split Model (Balcan and Blum (2008))
  - **Same-cluster Queries (Ashtiani, Kushagra, Ben-David (2016))**

Clustering with Queries

- Learner interacts with an expert/oracle to get advice.
- Same-Cluster Query
  - Do x_1 and x_2 belong to the same cluster?
- A natural/user-friendly form of query
  - Record De-Duplication
  - Assisted Troubleshooting
  - ...

Key Takeaways

Interactive clustering in the form of same-cluster queries can help us in

- Dealing with under-specificity
- Reducing computational complexity (?!!)

Key Takeaways

Interactive clustering in the form of same-cluster queries can help us in

- Dealing with under-specificity
- Reducing computational complexity (?!!)

The use of a few queries can make an otherwise NP-hard clustering problem tractable!

Problem Setting

- Input is $X = \{x\}_{i=1}^{n} \subset \mathbb{R}^d$.
- Learner asks same-cluster queries from the oracle.
- Goal is to recover the target clustering $C^* = (C_1^*, \ldots, C_k^*)$.

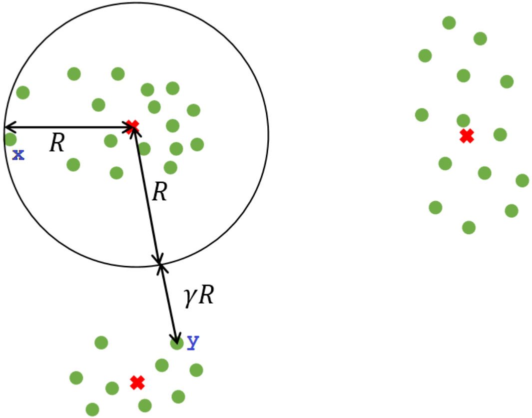
Problem Setting

- Input is $X = \{x\}_{i=1}^n \subset \mathbb{R}^d$.
- Learner asks same-cluster queries from the oracle.
- Goal is to recover the target clustering $C^* = (C_1^*, \ldots, C_k^*)$.

Still need more structure/inductive-bias ...

- "No-free-lunch" in clustering!
  - Need to ask $\Omega(n)$ queries.
- Target C^* is "nice".

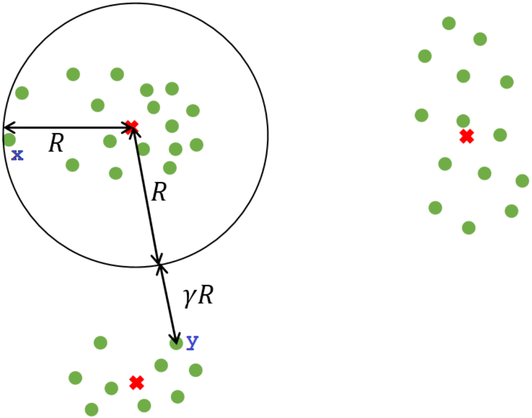
γ-Margin Property



$\mathcal{C} = \{C_1, \ldots, C_k\}$ with centers $\{\mu_1, \mu_2, \ldots, \mu_k\}$ satisfies the γ-margin property if for all $x \in C_i$ and $y \in X \setminus C_i$,

$$(1+\gamma)d(x, \mu_i) < d(y, \mu_i)$$

γ-Margin Property



- Query complexity?
- Computational complexity?

Positive Result (with Queries)

Theorem

There is an algorithm that finds C^* for any $\gamma > 0$ (with constant probability) which

- Runs in $\widetilde{O}(knd + k^2)$.
- Asks $O(k^2 \log k + k \log n)$ queries.

- Works for any "nice" target.
- No need to know k.
- Query complexity is
  - Dimension-independent!
  - Only logarithmic in n.

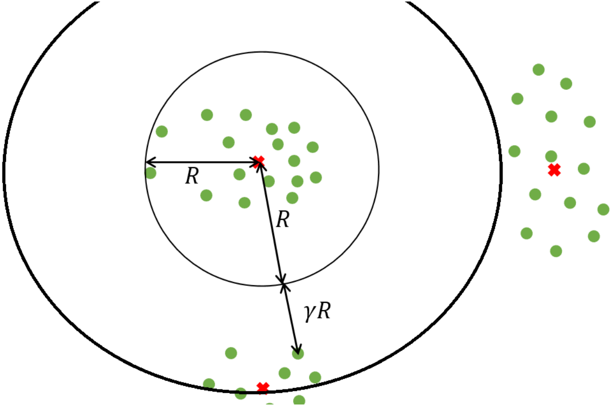
Computational Complexity

- Special case: oracle's clustering is the solution of K-means.
  - $\min_{\{\mu\}_{j=1}^{k}} \sum_{x \in X} \min_j \|x - \mu_j\|_2^2$.
  - Additional structure: γ-margin property.

Theorem

Solving Euclidean K-means clustering with no queries is NP-hard if $\gamma \leq 0.84$.

Computational Complexity



Without queries:

- NP-hard for realistic values of γ.
- Tractable only for unrealistically large values of γ.

Surprising Conclusion

For the realistic situation $0 \leq \gamma \leq 0.84$

Clustering is NP-hard without queries

BUT

tractable with a small number of queries

Hardness Result

Euclidean k-means is NP-hard even when the optimal solution satisfies the γ-margin property for $\gamma < 0.84$.

- True even with $O(\log n)$ same-cluster queries.
- Reduction from Exact Cover by 3-Sets.

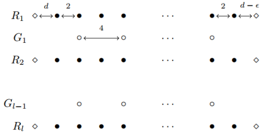


Figure 1: Geometry of $H_{l,m}$. This figure is similar to Fig. 1 in [Vat09]. Reading from letf to right, each row R_i consists of a diamond (s_i), $6m+1$ bullets

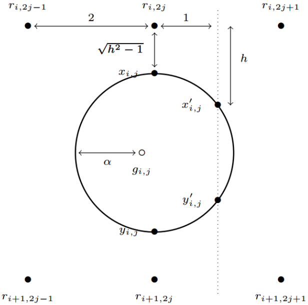


Figure 2: The locations of $x_{i,j}$, $x'_{i,j}$, $y_{i,j}$

Summary I

- Clustering is an under-specified problem.
- Domain knowledge can be conveyed using interaction.
- Same-cluster queries can also reduce computational complexity.
- Handling noisy oracles?
- Can we exploit same-cluster queries in other settings?

Summary I

- Clustering is an under-specified problem.
- Domain knowledge can be conveyed using interaction.
- Same-cluster queries can also reduce computational complexity.
- Handling noisy oracles?
- Can we exploit same-cluster queries in other settings?
  - Approximate k-means without γ-margin (Ailon et al. (2017))
  - Stochastic Block Model (Mazumdar, Saha (2017))
  - Correlation Clustering (Ailon et al. (2017))
  - Noisy Oracles (Kim and Ghosh (2017))
  - Mixture models? (rest of the talk!)

Density Estimation

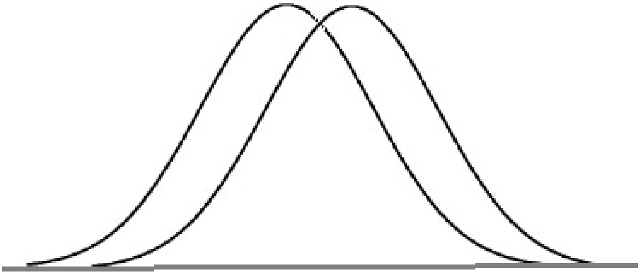
Density Estimation:

- Given an i.i.d. sample from an unknown density g^*, find a density $\hat{g}$ that is ϵ-close to g^* in **total variation distance**.

Density Estimation

Density Estimation:

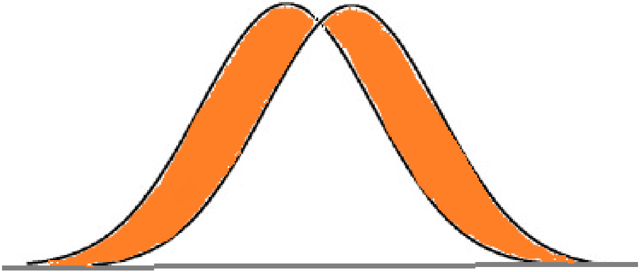
- Given an i.i.d. sample from an unknown density g^*, find a density $\hat{g}$ that is ϵ-close to g^* in **total variation distance**.



Density Estimation

Density Estimation:

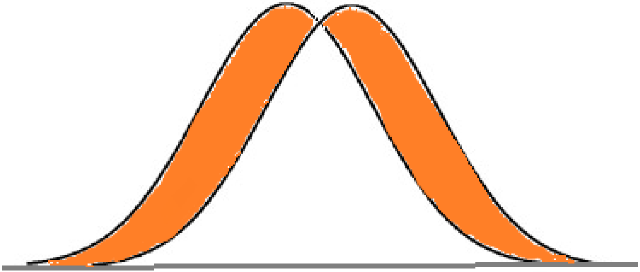
- Given an i.i.d. sample from an unknown density g^*, find a density $\hat{g}$ that is ϵ-close to g^* in **total variation distance**.



Density Estimation

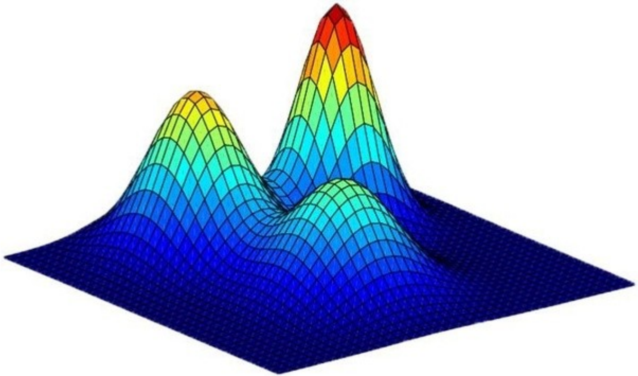
Density Estimation:

- Given an i.i.d. sample from an unknown density g^*, find a density $\hat{g}$ that is ϵ-close to g^* in **total variation distance**.



E.g., learning a Gaussian requires $\Theta\left(d^2/\epsilon^2\right) \approx (\#\text{params}/\epsilon^2)$ samples.

Mixture Models



How many samples is needed to learn a mixture of k Gaussian distributions?

Motivation

Mixture Models

- Have been studied for over a century!
- Are rich!
- Are building blocks of more complex models!

Motivation

Mixture Models

- Have been studied for over a century!
- Are rich!
- Are building blocks of more complex models!

Fundamental Open Problems:

- Sample complexity of learning mixtures of k Gaussian distributions over $\mathbb{R}^d$?
- Sample complexity of other mixture classes?

Motivation

Mixture Models

- Have been studied for over a century!
- Are rich!
- Are building blocks of more complex models!

Fundamental Open Problems:

- Sample complexity of learning mixtures of k Gaussian distributions over $\mathbb{R}^d$?
- Sample complexity of other mixture classes?

In practice, the data is never generated *exactly* from a GMM

- Looking for an agnostic (robust) guarantee!

Mixture Learning Theorem

- Let $m_{\mathcal{F}}$ be the sample complexity of learning $\mathcal{F}$.

Mixture Learning with Queries

For any natural class $\mathcal{F}$, the class of k-mixtures of $\mathcal{F}$ can be learned with $\widetilde{O}\left(k.m_{\mathcal{F}}\right)$ queries and $\widetilde{O}\left(k.m_{\mathcal{F}}/\epsilon^2\right)$ samples.

Mixture Learning Theorem

- One can remove the queries with a simulation trick!
  - Simulate all possible outcomes of queries.
  - Create a set of candidate pdfs.
  - Choose the best one based on a fresh new sample.

Mixture Learning Theorem

- One can remove the queries with a simulation trick!
  - Simulate all possible outcomes of queries.
  - Create a set of candidate pdfs.
  - Choose the best one based on a fresh new sample.

Mixture Learning (Ashtiani, Ben-David, Mehrabian (2018))

For any natural class $\mathcal{F}$, the class of k-mixtures of $\mathcal{F}$ can be learned with $\widetilde{O}\left(k.m_{\mathcal{F}}/\epsilon^2\right)$ samples.

- So an increase by a factor of at most k/ϵ^2.
- Generic but surprisingly tight!
- Robust (agnostic)!

Mixture Learning Theorem: Applications

Learning Mixture of Gaussians

Mixtures of k Gaussians in $\mathbb{R}^d$ can be learned with $\widetilde{O}(kd^2/\epsilon^4)$ samples.

- Improvement over previous known upper bounds
  - $\widetilde{O}(d^2k^3/\epsilon^4)$ (Diakonikolas et al. (2017))
  - $\widetilde{O}(d^4k^4/\epsilon^2)$ (Karpinski and Macintyre (1997))

Mixture Learning Theorem: Applications

Learning Mixture of Axis-aligned Gaussians

The class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ can be learned with $\widetilde{O}(kd/\epsilon^4)$ samples.

- Improvement over previous known upper bounds
  - $\widetilde{O}(dk^9/\epsilon^4)$ (Suresh et al. (2014))
  - $\widetilde{O}((d^2k^4 + d^3k^3)/\epsilon^2)$ (Karpinski and Macintyre (1997))

Mixture Learning Theorem: Applications

Learning Mixture of Axis-aligned Gaussians

The class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ can be learned with $\widetilde{O}(kd/\epsilon^4)$ samples.

- Improvement over previous known upper bounds
  - $\widetilde{O}(dk^9/\epsilon^4)$ (Suresh et al. (2014))
  - $\widetilde{O}((d^2k^4 + d^3k^3)/\epsilon^2)$ (Karpinski and Macintyre (1997))

Learning Mixture of Log-concave Distributions

The class of mixtures of k log-concave distributions in $\mathbb{R}^d$ can be learned with $\widetilde{O}(kd^{(d+5)/2}\epsilon^{-(d+9)/2})$ samples.

Summary II

- If a class is learnable, so is its mixture.
- Sample complexity is increased by a factor of $\widetilde{O}(k/\epsilon^2)$.
- Surprisingly sharp, e.g. for GMMs.

Summary II

- If a class is learnable, so is its mixture.
- Sample complexity is increased by a factor of $\widetilde{O}(k/\epsilon^2)$.
- Surprisingly sharp, e.g. for GMMs.

Is it tight for GMMs?

- $\widetilde{O}(kd^2/\epsilon^4)$ is still loose in terms of ϵ!

How about Axis-aligned GMMs?

- $\widetilde{O}(kd/\epsilon^4)$ is loose in terms of ϵ!

Summary II

- If a class is learnable, so is its mixture.
- Sample complexity is increased by a factor of $\widetilde{O}(k/\epsilon^2)$.
- Surprisingly sharp, e.g. for GMMs.

Is it tight for GMMs?

- $\widetilde{O}(kd^2/\epsilon^4)$ is still loose in terms of ϵ!

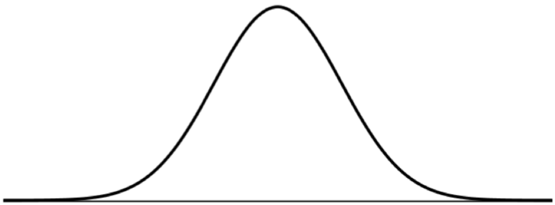
How about Axis-aligned GMMs?

- $\widetilde{O}(kd/\epsilon^4)$ is loose in terms of ϵ!

Settling the sample complexity of learning GMMs?

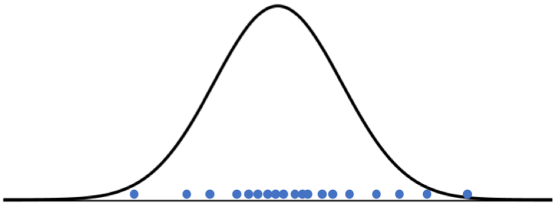
- Idea: compression schemes!
- $\widetilde{\Theta}(kd/\epsilon^2)$ for axis-aligned GMMs (Ashtiani, Ben-David, Mehrabian)
- $\widetilde{\Theta}(kd^2/\epsilon^2)$ for general GMMs (Ashtiani, Ben-David, Harvey, Liaw, Mehrabian, Plan)

Density Estimation via Compression



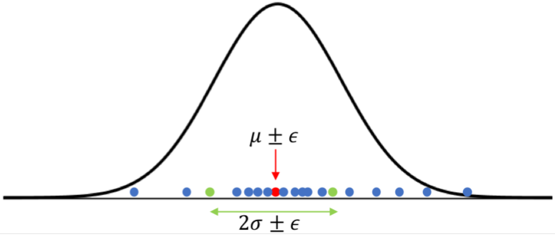
1-dimensional Gaussians with unknown mean and variance

Density Estimation via Compression



Generate $1/\epsilon$ i.i.d. samples.

Density Estimation via Compression



1-dimensional Gaussians admit $(3, 1/\epsilon)$ compression!

Distribution Learning via Compression

Compression Implies Learnability

If $\mathcal{F}$ admits (t, m) compression, then $\mathcal{F}$ can be learned using

$$\widetilde{O}\left(m(\frac{\epsilon}{6}) + \frac{t(\epsilon)}{\epsilon^2} \right) \textit{ samples.}$$

Distribution Learning via Compression

Compression Implies Learnability

If $\mathcal{F}$ admits (t, m) compression, then $\mathcal{F}$ can be learned using

$$\widetilde{O}\left(m(\frac{\epsilon}{6}) + \frac{t(\epsilon)}{\epsilon^2}\right) \textit{ samples.}$$

Compressing Product Distributions

If $\mathcal{F}$ admits $(t(\epsilon), m(\epsilon))$ compression, then $\mathcal{F}^d$ admits $(dt(\epsilon/d), m(\epsilon/d) \log 3d)$ compression.

Distribution Learning via Compression

Compression Implies Learnability

If $\mathcal{F}$ admits (t, m) compression, then $\mathcal{F}$ can be learned using

$$\widetilde{O}\left(m(\frac{\epsilon}{6}) + \frac{t(\epsilon)}{\epsilon^2}\right) \text{ samples.}$$

Compressing Product Distributions

If $\mathcal{F}$ admits $(t(\epsilon), m(\epsilon))$ compression, then $\mathcal{F}^d$ admits $(dt(\epsilon/d), m(\epsilon/d)\log 3d)$ compression.

Compressing Mixtures

Under natural conditions, if $\mathcal{F}$ admits (t, m) compression, then k-mix$(\mathcal{F})$ admits $(kt + k\log(k/\epsilon), m(\epsilon)k\log k)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^d$ admit $(d, (d \log d)/\epsilon)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^d$ admit $(d, (d \log d)/\epsilon)$ compression.

Compressing mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^d$ admit $(kd + k\log(k/\epsilon), (kd \log k \log d)/\epsilon)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^d$ admit $(d, (d \log d)/\epsilon)$ compression.

Compressing mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^d$ admit $(kd + k\log(k/\epsilon), (kd \log k \log d)/\epsilon)$ compression.

Learning mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^d$ can be learned using $\widetilde{O}(kd/\epsilon^2)$ samples.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^d$ admit $(d, (d \log d)/\epsilon)$ compression.

Compressing mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^d$ admit $(kd + k \log(k/\epsilon), (kd \log k \log d)/\epsilon)$ compression.

Learning mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^d$ can be learned using $\widetilde{O}(kd/\epsilon^2)$ samples.

- The first known tight result (up to logarithmic factors).

Summary III

- Distribution compression implies learnability.
- Compression has nice closure properties.
- Mixtures of axis-aligned Gaussians can be effectively compressed.

Summary III

- Distribution compression implies learnability.
- Compression has nice closure properties.
- Mixtures of axis-aligned Gaussians can be effectively compressed.

Also,

- Mixtures of general Gaussians can be effectively compressed.

Summary III

- Distribution compression implies learnability.
- Compression has nice closure properties.
- Mixtures of axis-aligned Gaussians can be effectively compressed.

Also,

- Mixtures of general Gaussians can be effectively compressed.
- Robust learning using "robust compression"!

Summary III

- Distribution compression implies learnability.
- Compression has nice closure properties.
- Mixtures of axis-aligned Gaussians can be effectively compressed.

Also,

- Mixtures of general Gaussians can be effectively compressed.
- Robust learning using "robust compression"!

Open problems

- Compression with respect to other distances?
- Compressing deep generative models?

Thank You!

Bigger Picture

Filling the gap between supervised and unsupervised learning!

- Supervised learning
  - Not many labeled instances?
  - Noisy/malicous instances/labels?
  - Discrepancy between the train and test distributions?
  - Using unlabeled data?
- Unsupervised learning
  - Unsupervised representation learning?
  - Interactive clustering?
  - Computationally efficient clustering?
  - Sample-efficient density estimation?
- Reinforcement Learning
  - Non-i.i.d. samples.
  - Weaker form of supervision (i.e., reward function)
  - Delayed feedback.

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Representation/parametrization dependent
  - Identifiability problems, separability assumptions

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Representation/parametrization dependent
  - Identifiability problems, separability assumptions
- Density Estimation
  - The output of the algorithm is a pdf, $\hat{g}$
  - The goal is to have $d(g, \hat{g}) < \epsilon$
  - Various choices for dissimilarity

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Representation/parametrization dependent
  - Identifiability problems, separability assumptions
- Density Estimation
  - The output of the algorithm is a pdf, $\hat{g}$
  - The goal is to have $d(g, \hat{g}) < \epsilon$
  - Various choices for dissimilarity
- Dissimilarity Measures
  - Total variation distance
  - KL-divergence
  - Hellinger distance
  - ...

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Representation/parametrization dependent
  - Identifiability problems, separability assumptions
- Density Estimation
  - The output of the algorithm is a pdf, $\hat{g}$
  - The goal is to have $d(g, \hat{g}) < \epsilon$
  - Various choices for dissimilarity
- Dissimilarity Measures
  - Total variation distance
  - KL-divergence
  - Hellinger distance
  - ...
- Our focus: density estimation w.r.t. the total variation distance
  - $\|g - \hat{g}\|_{TV} := \sup_{A \subset \mathbb{R}^d} |g(A) - \hat{g}(A)|$
  - $\|g - \hat{g}\|_{TV} = \frac{1}{2}\|g - \hat{g}\|_1 = \frac{1}{2}\int_z |g - \hat{g}| dz$

Mixture Learning Theorem

Assume that $\mathcal{F}$ can be learned with $m_{\mathcal{F}}(\epsilon, \delta) = \lambda(\mathcal{F}, \delta)/\epsilon^{\alpha}$ samples for some $\alpha \geq 1$ and some function $\lambda(\mathcal{F}, \delta) = \Omega(\ln(1/\delta))$. Then the class k-mix($\mathcal{F}$) can be learned with

$$O\left(\frac{\lambda(\mathcal{F}, \delta/3k) k \log k}{\epsilon^{\alpha+2}}\right) = O\left(\frac{k \log k \cdot m_{\mathcal{F}}(\epsilon, \delta/3k)}{\epsilon^{2}}\right)$$

samples.

Furthermore, if the base learner is robust, then the mixture learner will be robust as well.

VC-Bound

Minimum Distance Estimator (e.g., Devroye and Lugosi (2001))

If VC-dim(SUBLEVEL($\Delta\mathcal{F}$)) $\leq v$, then $m^3_{\mathcal{F}}(\epsilon, \delta) = O((v + \log \frac{1}{\delta})/\epsilon^2)$.

- VC-dim(SUBLEVEL(Δk-mix($\mathcal{F}$))) $\leq k$.VC-dim(SUBLEVEL($\Delta\mathcal{F}$))?

VC-Bound

Minimum Distance Estimator (e.g., Devroye and Lugosi (2001))

If VC-dim(SUBLEVEL($\Delta\mathcal{F}$)) $\leq v$, then $m^3_{\mathcal{F}}(\epsilon, \delta) = O((v + \log \frac{1}{\delta})/\epsilon^2)$.

- VC-dim(SUBLEVEL(Δk-mix($\mathcal{F}$))) $\leq k$.VC-dim(SUBLEVEL($\Delta\mathcal{F}$))?
- Bounding the VC-dimension is not easy even for GMMs over $\mathbb{R}^n$.
- This upper bound is in general loose.

Density Estimation via Compression

Distribution Decoder

A *distribution decoder* for $\mathcal{F}$ is a function $\mathcal{J}$ that takes a finite sequence of elements of the domain, and outputs a member of $\mathcal{F}$.

Distribution Compression Schemes

$\mathcal{F}$ admits (d, m) compression if there exists a decoder $\mathcal{J}$ for $\mathcal{F}$ such that for any $g \in \mathcal{F}$, if $S \sim g^{m(\epsilon)}$, then with probability at least $2/3$, there exists a sequence L of at most $d(\epsilon)$ elements of S such that $\|\mathcal{J}(L) - g\|_1 \leq \epsilon$.

Distribution Learning via Compression

Compression Implies Learnability

If $\mathcal{F}$ admits (d, m) compression, then $\mathcal{F}$ can be learned using

$$O\left(m(\frac{\epsilon}{6})\log\frac{1}{\delta} + \frac{d(\epsilon)\log(m(\frac{\epsilon}{6})\log(1/\delta)) + \log(1/\delta)}{\epsilon^2}\right)$$
$$= \widetilde{O}\left(m(\frac{\epsilon}{6}) + \frac{d(\epsilon)}{\epsilon^2}\right) \text{ samples.}$$

Distribution Learning via Compression

Compression Implies Learnability

If $\mathcal{F}$ admits (d, m) compression, then $\mathcal{F}$ can be learned using

$$O\left(m(\frac{\epsilon}{6})\log\frac{1}{\delta} + \frac{d(\epsilon)\log(m(\frac{\epsilon}{6})\log(1/\delta)) + \log(1/\delta)}{\epsilon^2}\right)$$
$$= \widetilde{O}\left(m(\frac{\epsilon}{6}) + \frac{d(\epsilon)}{\epsilon^2}\right) \textit{ samples.}$$

Compressing Product Distributions

If $\mathcal{F}$ admits $(d(\epsilon), m(\epsilon))$ compression, then $\mathcal{F}^n$ admits $(nd(\epsilon/n), m(\epsilon/n)\log 3n)$ compression.

Distribution Learning via Compression

Compression Implies Learnability

If $\mathcal{F}$ admits (d, m) compression, then $\mathcal{F}$ can be learned using

$$O\left(m(\frac{\epsilon}{6})\log\frac{1}{\delta} + \frac{d(\epsilon)\log(m(\frac{\epsilon}{6})\log(1/\delta)) + \log(1/\delta)}{\epsilon^2}\right)$$
$$= \widetilde{O}\left(m(\frac{\epsilon}{6}) + \frac{d(\epsilon)}{\epsilon^2}\right) \text{ samples.}$$

Compressing Product Distributions

If $\mathcal{F}$ admits $(d(\epsilon), m(\epsilon))$ compression, then $\mathcal{F}^n$ admits $(nd(\epsilon/n), m(\epsilon/n)\log 3n)$ compression.

Compressing Mixtures

Under natural conditions, if $\mathcal{F}$ admits (d, m) compression, then k-mix$(\mathcal{F})$ admits $(kd + k\log(k/\epsilon), m(\epsilon)k\log k)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^n$ admit $(n, (n \log n)/\epsilon)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^n$ admit $(n, (n \log n)/\epsilon)$ compression.

Compressing mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^n$ admit $(kn + k\log(k/\epsilon), (kn \log k \log n)/\epsilon)$ compression.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^n$ admit $(n, (n \log n)/\epsilon)$ compression.

Compressing mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^n$ admit $(kn + k\log(k/\epsilon), (kn \log k \log n)/\epsilon)$ compression.

Learning mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^n$ can be learned using $\widetilde{O}(kn/\epsilon^2)$ samples.

Learning Mixtures of Axis-Aligned Gaussians

Compressing 1-dimensional Gaussians

1-dimensional Gaussians admit $(O(1), 1/\epsilon)$ compression.

Compressing axis-aligned Gaussians

Axis-aligned Gaussians over $\mathbb{R}^n$ admit $(n, (n \log n)/\epsilon)$ compression.

Compressing mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^n$ admit $(kn + k \log(k/\epsilon), (kn \log k \log n)/\epsilon)$ compression.

Learning mixtures of axis-aligned Gaussians

Mixtures of k axis-aligned Gaussians over $\mathbb{R}^n$ can be learned using $\widetilde{O}(kn/\epsilon^2)$ samples.

- The first known tight result (up to logarithmic factors).

Mixture Learning Algorithm

- Create a set of candidate pdfs based on the sample.
- Choose the best one based on a fresh new sample.

Mixture Learning Algorithm

- Create a set of candidate pdfs based on the sample.
- Choose the best one based on a fresh new sample.

Learning Finite Classes (e.g., Devroye and Lugosi (2001))

If $|\mathcal{F}| \leq M$, then there is a polynomial time algorithm that learns $\mathcal{F}$ with $O(\log(M/\delta)/\epsilon^2)$ samples.

Mixture Learning Algorithm

- Create a set of candidate pdfs based on the sample.
- Choose the best one based on a fresh new sample.

Learning Finite Classes (e.g., Devroye and Lugosi (2001))

If $|\mathcal{F}| \leq M$, then there is a polynomial time algorithm that learns $\mathcal{F}$ with $O(\log(M/\delta)/\epsilon^2)$ samples.

Input: k, ϵ, δ and an iid sample S

0. Let $\widehat{W}$ be an (ϵ/k)-cover for Δ_k in ℓ_∞ distance.
1. $\mathcal{C} = \emptyset$. (set of candidate distributions)
2. For each $(\widehat{w}_1, \ldots, \widehat{w}_k) \in \widehat{W}$ do:
   3. For each possible partition of S into $A_1, A_2, ..., A_{k+1}$:
      4. Provide A_i to the $\mathcal{F}$-learner, and let $\widehat{G}_i$ be its output.
      5. Add the candidate distribution $\sum_{i\in[k]} \widehat{w}_i \widehat{G}_i$ to $\mathcal{C}$.
6. Apply the algorithm for finite classes to $\mathcal{C}$ and output its result.

Mixture Learning Theorem: Proof Idea

- Prove that there is a “good” candidate in $\mathcal{C}$.
- Show that $\log |\mathcal{C}|$ is small.

Mixture Learning Theorem: Proof Idea

- Prove that there is a "good" candidate in $\mathcal{C}$.
- Show that $\log |\mathcal{C}|$ is small.
- Naive analysis
  - Ignore components whose weights smaller than ϵ/k.

Mixture Learning Theorem: Proof Idea

- Prove that there is a "good" candidate in $\mathcal{C}$.
- Show that $\log |\mathcal{C}|$ is small.
- Naive analysis
  - Ignore components whose weights smaller than ϵ / k.
  - We need m points from each of the "non-negligible" components.

Mixture Learning Theorem: Proof Idea

- Prove that there is a "good" candidate in $\mathcal{C}$.
- Show that $\log |\mathcal{C}|$ is small.
- Naive analysis
  - Ignore components whose weights smaller than ϵ/k.
  - We need m points from each of the "non-negligible" components.
  - If we generate $m\frac{k}{\epsilon}\log\frac{k}{\epsilon} \approx mk/\epsilon$ samples, we will get m from all non-negligible components.

Mixture Learning Theorem: Proof Idea

- Prove that there is a "good" candidate in $\mathcal{C}$.
- Show that $\log |\mathcal{C}|$ is small.
- Naive analysis
  - Ignore components whose weights smaller than ϵ/k.
  - We need m points from each of the "non-negligible" components.
  - If we generate $m\frac{k}{\epsilon} \log \frac{k}{\epsilon} \approx mk/\epsilon$ samples, we will get m from all non-negligible components.
  - We will have $\approx k^{mk/\epsilon}$ candidates.

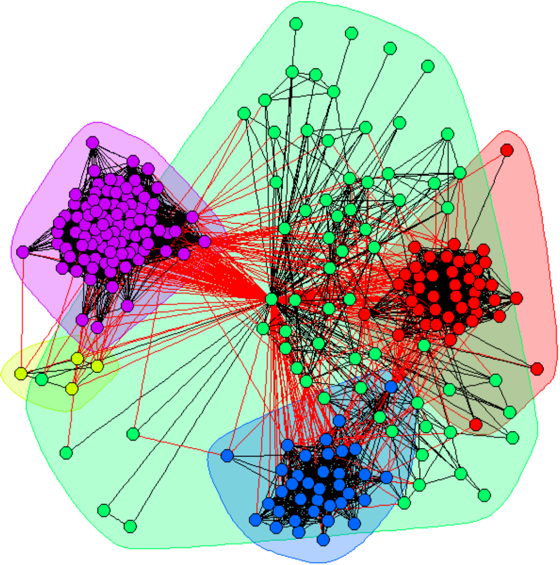
Mixture Learning Theorem: Proof Idea

- Prove that there is a "good" candidate in $\mathcal{C}$.
- Show that $\log |\mathcal{C}|$ is small.
- Naive analysis
  - Ignore components whose weights smaller than ϵ/k.
  - We need m points from each of the "non-negligible" components.
  - If we generate $m\frac{k}{\epsilon}\log\frac{k}{\epsilon} \approx mk/\epsilon$ samples, we will get m from all non-negligible components.
  - We will have $\approx k^{mk/\epsilon}$ candidates.
  - Final sample complexity: $\frac{\log|\mathcal{C}|}{\epsilon^2} = \widetilde{O}\left(\frac{km}{\epsilon^3}\right)$

Mixture Learning Theorem: Proof Idea

- Prove that there is a "good" candidate in $\mathcal{C}$.
- Show that $\log |\mathcal{C}|$ is small.
- Naive analysis
  - Ignore components whose weights smaller than ϵ/k.
  - We need m points from each of the "non-negligible" components.
  - If we generate $m\frac{k}{\epsilon}\log\frac{k}{\epsilon} \approx mk/\epsilon$ samples, we will get m from all non-negligible components.
  - We will have $\approx k^{mk/\epsilon}$ candidates.
  - Final sample complexity: $\frac{\log|\mathcal{C}|}{\epsilon^2} = \widetilde{O}\left(\frac{km}{\epsilon^3}\right)$
  - More careful analysis gives $\widetilde{O}\left(\frac{km}{\epsilon^2}\right)$

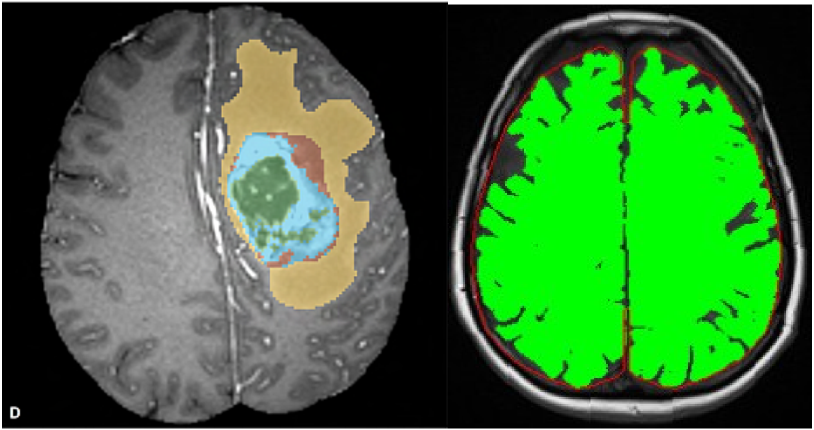
Community Detection



Record Deduplication

- The *ground truth* seems obvious.
- What clustering method should we use?

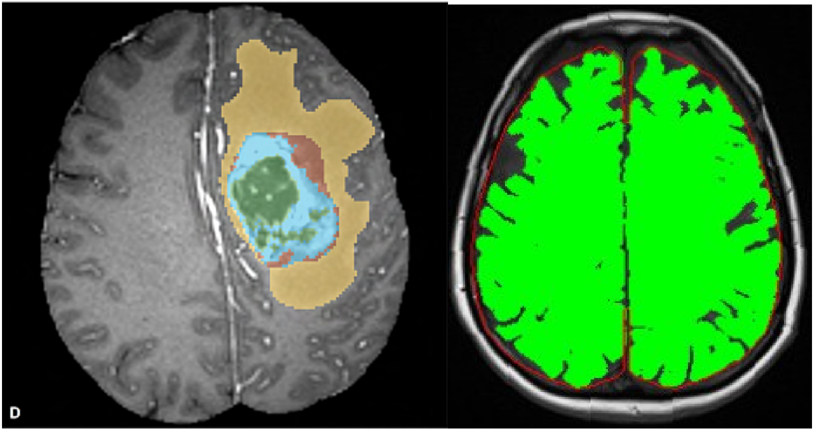
Brain Segmentation



Clustering various regions in the brain, used for

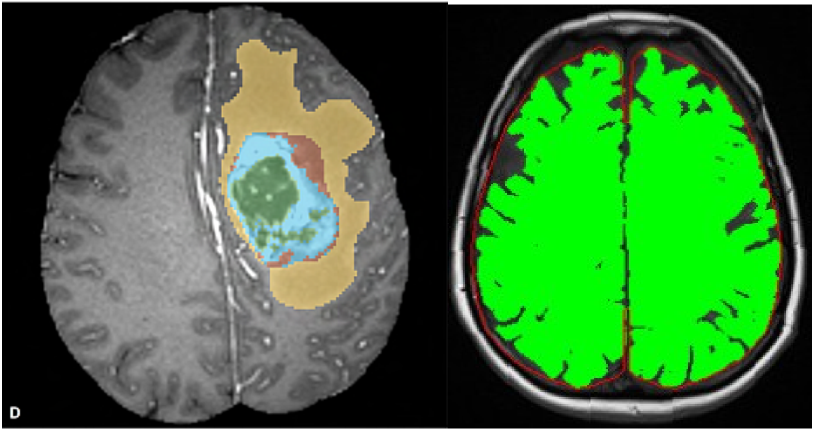
- Visualizing and analyzing the brain structure
- Locating tumors
- Planing for surgery

Brain Segmentation



- No "single right" segmentation.
- Segmentation's validity is task-specific.
- Labeled data is scarce and expensive.

Brain Segmentation



- No "single right" segmentation.
- Segmentation's validity is task-specific.
- Labeled data is scarce and expensive.

A semi-supervised clustering framework?

Sketch of the Proof

1. Bound $Pdim(\mathcal{F})$
2. Bound $\mathcal{N}(\mathcal{F}, d_{L_1}^{X}, \epsilon)$ based on $Pdim(\mathcal{F})$ and ϵ
3. Bound $\mathcal{N}(\mathcal{F}, \Delta_X, \epsilon)$ based on $\mathcal{N}(\mathcal{F}, d_{L_1}^{X}, \epsilon)$
4. Bound the $m_{UC}^{\mathcal{F}}(\epsilon, \delta)$ based on δ and $\mathcal{N}(\mathcal{F}, \Delta_X, \epsilon)$
5. Bound $m^{\mathcal{F}}(\epsilon, \delta)$ based on $m_{UC}^{\mathcal{F}}(\epsilon, \delta)$

Uniqueness of Solution Assumption

- k-means' solution may not be unique for some mappings

Uniqueness of Solution Assumption

- k-means' solution may not be unique for some mappings
- Such mappings should not be selected!

Uniqueness of Solution Assumption

- k-means' solution may not be unique for some mappings
- Such mappings should not be selected!
- We should compare the the output of the algorithm only to those mappings in $\mathcal{F}$ that have unique solutions

Uniqueness of Solution Assumption

- k-means' solution may not be unique for some mappings
- Such mappings should not be selected!
- We should compare the the output of the algorithm only to those mappings in $\mathcal{F}$ that have unique solutions

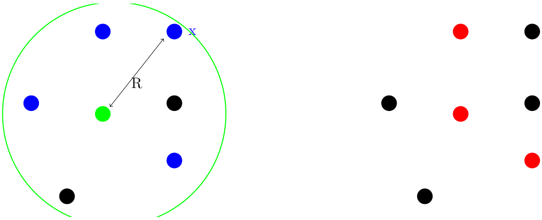
- (η, ϵ)-*Uniqueness*: Every η-optimal solution to k-means' cost is ϵ-close to the optimal solution

Uniqueness of Solution Assumption

- *k*-means' solution may not be unique for some mappings
- Such mappings should not be selected!
- We should compare the the output of the algorithm only to those mappings in $\mathcal{F}$ that have unique solutions

- (η, ϵ)-*Uniqueness*: Every η-optimal solution to *k*-means' cost is ϵ-close to the optimal solution
- For simplifying the presentation of the results, we assume the class $\mathcal{F}$ includes only the mappings under which the solution is unique.

Algorithm



Algorithm's Idea

1. Estimate a center.
   - Query uniformly till we have "enough" points from one cluster.
2. Prune points belonging to that cluster.
   - Binary search to find the "effective radius".
3. Repeat for the other clusters.

No need to know k in advance.