Foundations of Clustering

Debarghya Ghoshdastidar

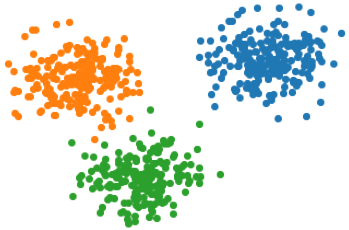
Theoretical Foundations of Artificial Intelligence

Department of Informatics

Technical University of Munich

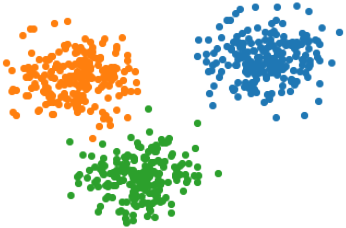
What is clustering?

Whats are clusters?



Closely packed points

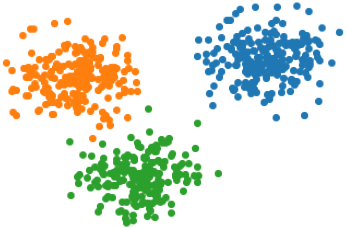
Whats are clusters?



Closely packed points

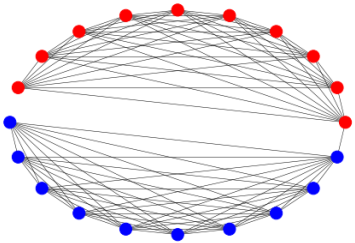
Points in same pattern

Whats are clusters?



Closely packed points

Points in same pattern



Similar / connected entities

. . . and others

Definition of cluster analysis

- Cambridge dictionary
  A way of studying or examining large amounts of data to find groups that are **more like each other** than they are like the data in other group

Definition of cluster analysis

- Cambridge dictionary
  A way of studying or examining large amounts of data to find groups that are **more like each other** than they are like the data in other group

- Wikipedia
  The notion of a “cluster” cannot be precisely defined, which is one of the reasons why there are so many clustering algorithms

Definition of cluster analysis

- Cambridge dictionary
  A way of studying or examining large amounts of data to find groups that are **more like each other** than they are like the data in other group

- Wikipedia
  The notion of a "cluster" cannot be precisely defined, which is one of the reasons why there are so many clustering algorithms

- *Clustering: Science or Art?* [Luxburg et al 2012]
  Clustering should not be treated as an application-independent mathematical problem, but should always be studied in the context of its end-use

Clustering vs classification / estimation

- Formal objective

  Classification: Yes, reduce number of errors

  Clustering: None

Clustering vs classification / estimation

- Formal objective

  Classification: Yes, reduce number of errors

  Clustering: None

- Solved using optimisation

  Classification: Mostly, via training

  Clustering: Not clear for most algorithms / heuristics

Goal of this lecture

- Few popular clustering approaches
  - k-means and more (centroid based clustering)
  - Linkage methods (hierarchical clustering)
  - GMM, DBSCAN (density based clustering)
  - Deep networks
- What do these algorithms solve (formally)?
- How do we measure the goodness of clustering?

Practical guides

- Software documentation
- Blogs (easier than papers, reliability issues)
- Tutorial videos

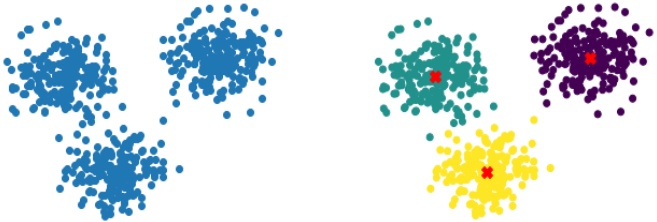
Practical guides

- Software documentation
  - Python: sklearn clustering
  - HDBSCAN documentation. Comparing Python Clustering Algorithms
- Blogs (easier than papers, reliability issues)
  - towards data science
  - G. Seif. The 5 Clustering Algorithms Data Scientists Need to Know
- Tutorial videos

Centroid based clustering

Basic setup

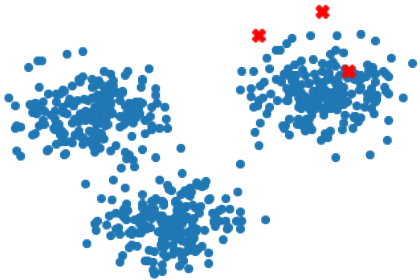
- Given n points, and number of clusters k
- Goal: Data compression
  - Find k centers that best represent the n points
  - Associate each point with nearest center



k-means algorithm

[Lloyd 1982]

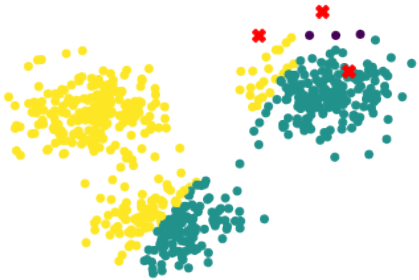
1. Choose any k locations as potential centers



k-means algorithm

[Lloyd 1982]

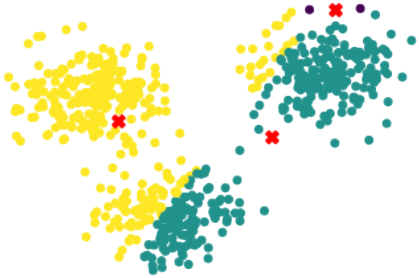
1. Choose any k locations as potential centers
2. Associate every data point to nearest center



k-means algorithm

[Lloyd 1982]

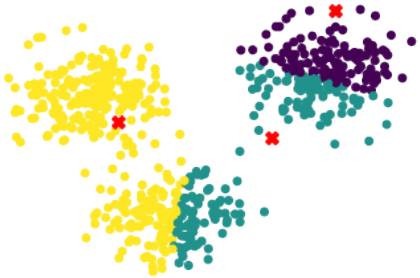
1. Choose any k locations as potential centers
2. Associate every data point to nearest center
3. Update centers to be means of clusters



k-means algorithm

[Lloyd 1982]

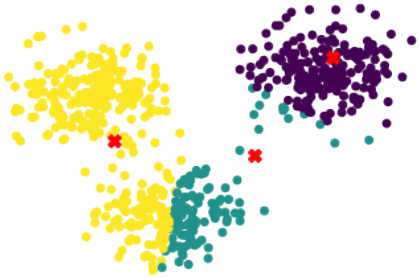
1. Choose any k locations as potential centers
2. Associate every data point to nearest center
3. Update centers to be means of clusters
4. Reiterate steps 2 and 3 till convergence



k-means algorithm

[Lloyd 1982]

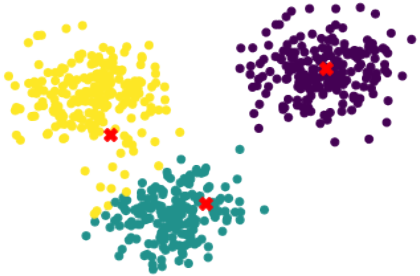
1. Choose any k locations as potential centers
2. Associate every data point to nearest center
3. Update centers to be means of clusters
4. Reiterate steps 2 and 3 till convergence



k-means algorithm

[Lloyd 1982]

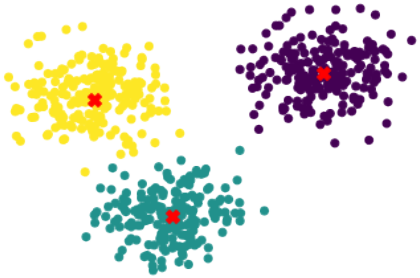
1. Choose any k locations as potential centers
2. Associate every data point to nearest center
3. Update centers to be means of clusters
4. Reiterate steps 2 and 3 till convergence



k-means algorithm

[Lloyd 1982]

1. Choose any k locations as potential centers
2. Associate every data point to nearest center
3. Update centers to be means of clusters
4. Reiterate steps 2 and 3 till convergence



Questions

- Do the iterations converge?
- If yes, how long can it take to converge?

- What formal problem does k-means solve?
- Is the solution always ‘good’?

k-means problem

[Lloyd 1982]

- Given $\mathcal{X} = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^p$
- For any k centers $c_1, \ldots, c_k \in \mathbb{R}^p$

$$\text{k-means cost:} \quad f(c_1, \ldots, c_k) = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - c_j\|^2$$

where $C_j = \{x \in \mathcal{X} \ : \ c_j \text{ is closest center for } x\}$

$\|x_i - c_j\| =$ Euclidean distance

k-means problem

[Lloyd 1982]

- Given $\mathcal{X} = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^p$
- For any k centers $c_1, \ldots, c_k \in \mathbb{R}^p$

$$\text{k-means cost:} \quad f(c_1, \ldots, c_k) = \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - c_j\|^2$$

where $C_j = \{x \in \mathcal{X} \ : \ c_j \text{ is closest center for } x\}$

$\|x_i - c_j\| =$ Euclidean distance

$$\text{k-means problem:} \quad \underset{c_1,\ldots,c_k}{\text{minimise}} \ f(c_1, \ldots, c_k)$$

Convergence of k-means iterations

- Each iteration of k-means reduces the k-means cost
- Iterations converge to a local minimum

Convergence of k-means iterations

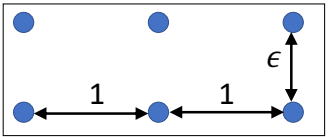
- Each iteration of k-means reduces the k-means cost
- Iterations converge to a local minimum
- Can we converge to the global minimum (or close to it)?
  - Depends on the initial centers
  - Can be arbitrarily bad

Convergence of k-means iterations

- Each iteration of k-means reduces the k-means cost
- Iterations converge to a local minimum
- Can we converge to the global minimum (or close to it)?
  - Depends on the initial centers
  - Can be arbitrarily bad
- How long does it take to converge?
  - No non-trivial bound on number of iterations

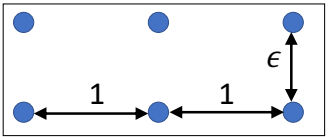
Exercise: Sub-optimality of Lloyd's algorithm

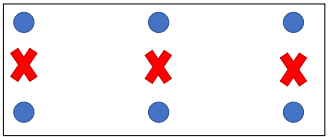
- Consider 6 points in $\mathbb{R}^2$ ($\epsilon \ll 1$)



Exercise: Sub-optimality of Lloyd's algorithm

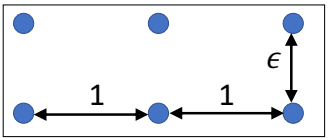
- Consider 6 points in $\mathbb{R}^2$ ($\epsilon \ll 1$)
- Optimal centers have k-means cost: $f_{opt} = 1.5\epsilon^2$



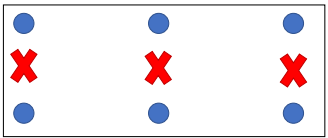


Exercise: Sub-optimality of Lloyd's algorithm

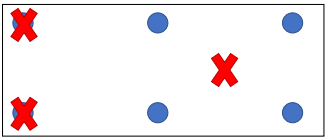
- Consider 6 points in $\mathbb{R}^2$ ($\epsilon \ll 1$)
- Optimal centers have k-means cost: $f_{opt} = 1.5\epsilon^2$
- No updates if we initialise with configuration on right
  - Cost $f = 2\sqrt{1+\epsilon^2} \quad \gg f_{opt}$



Six points



Optimal centers



Local optimum

k-means++ [Arthur & Vassilvitskii 2007]

- Careful choice of centers (seeding)
- Define clusters given by chosen centers

k-means++ [Arthur & Vassilvitskii 2007]

- Careful choice of centers (seeding)
- Define clusters given by chosen centers
- Merits:
  - Not iterative; completes in $O(kn)$-runtime
  - Theoretical guarantee (not arbitrarily worse than f_{opt})

k-means++ [Arthur & Vassilvitskii 2007]

- Careful choice of centers (seeding)
- Define clusters given by chosen centers
- Merits:
  - Not iterative; completes in $O(kn)$-runtime
  - Theoretical guarantee (not arbitrarily worse than f_{opt})
- Standard implementations of k-means
  - Run k-means++, follows by few iterations of k-means

k-means++ algorithm

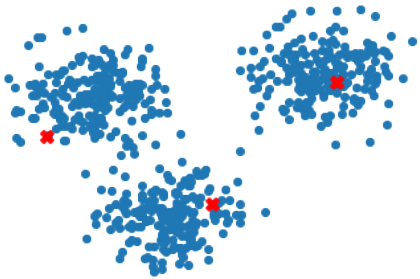
1. Pick $x \in \mathcal{X}$ uniformly at random and set $c_1 = x$

k-means++ algorithm

1. Pick $x \in \mathcal{X}$ uniformly at random and set $c_1 = x$
2. For $j = 2, \ldots, k$
   1. Define $w(x) = \min_{r \in \{1,\ldots,j-1\}} \|x - c_r\|^2$ for all $x \in \mathcal{X}$
   2. Sample $x \in \mathcal{X}$ according to probability $\propto w(x)$
   3. Set $c_j = x$

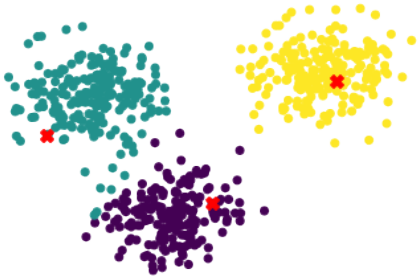
k-means++ algorithm

1. Pick $x \in \mathcal{X}$ uniformly at random and set $c_1 = x$
2. For $j = 2, \ldots, k$
   1. Define $w(x) = \min_{r \in \{1,\ldots,j-1\}} \|x - c_r\|^2$ for all $x \in \mathcal{X}$
   2. Sample $x \in \mathcal{X}$ according to probability $\propto w(x)$
   3. Set $c_j = x$



k-means++ algorithm

1. Pick $x \in \mathcal{X}$ uniformly at random and set $c_1 = x$
2. For $j = 2, \ldots, k$
   1. Define $w(x) = \min_{r \in \{1,\ldots,j-1\}} \|x - c_r\|^2$ for all $x \in \mathcal{X}$
   2. Sample $x \in \mathcal{X}$ according to probability $\propto w(x)$
   3. Set $c_j = x$
3. Define $C_j = \{x \in \mathcal{X} \ : \ c_j \text{ is closest center for } x\}$



Theory for k-means

- Worst-case approximation guarantee (definition):

  An algorithm is b-factor approximation if, for any data, solution $c_1, \ldots, c_k$ satisfies

$$f(c_1, \ldots, c_k) \leq b \cdot f_{opt}$$

Theory for k-means

- Worst-case approximation guarantee (definition):

  An algorithm is b-factor approximation if, for any data, solution $c_1, \ldots, c_k$ satisfies

$$f(c_1, \ldots, c_k) \;\leq\; b \cdot f_{opt}$$

- Guarantee for k-means++: [Arthur & Vassilvitskii 2007]

$$f(c_1, \ldots, c_k) \;\leq\; 8(\log k + 2) \cdot f_{opt}$$

  averaged over randomness in algorithm

Theory for k-means

- Worst-case approximation guarantee (definition):

  An algorithm is b-factor approximation if, for any data, solution $c_1, \ldots, c_k$ satisfies

  $$f(c_1, \ldots, c_k) \;\leq\; b \cdot f_{opt}$$

- Guarantee for k-means++: [Arthur & Vassilvitskii 2007]

  $$f(c_1, \ldots, c_k) \;\leq\; 8(\log k + 2) \cdot f_{opt}$$

  averaged over randomness in algorithm

- Impossibility result: [Lee et at 2017]

  NP-Hard to find a worst-case approximation $b \leq 1.0013$

Clustering in metric spaces

$$\text{k-means problem:} \quad \underset{c_1,\ldots,c_k}{\text{minimise}} \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - c_j\|^2$$

- Means can only be defined in vector spaces
  - It minimises squared distance to all points only in Hilbert space

Clustering in metric spaces

$$\text{k-means problem:} \quad \underset{c_1,\dots,c_k}{\text{minimise}} \sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - c_j\|^2$$

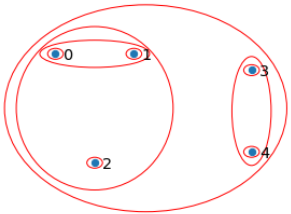
- Means can only be defined in vector spaces
  - It minimises squared distance to all points only in Hilbert space
- General metric space
  - Restrict centers to be points in data set $\mathcal{X}$, and replace Euclidean distance by metric d

$$\text{k-medoid problem:} \quad \underset{c_1,\dots,c_k \in \mathcal{X}}{\text{minimise}} \sum_{j=1}^{k} \sum_{x_i \in C_j} d(x_i, c_j)$$

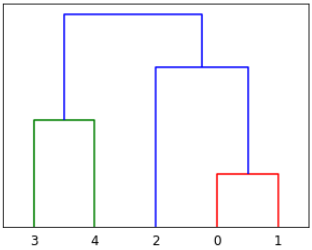
Hierarchical clustering

Hierarchical clustering

- Group data $\mathcal{X}$ at different levels of granularity
  - Hierarchy of clusters
  - One can derive k large clusters or many small clusters
- Dendrogram: Binary tree depicting hierarchy of clusters



Clusters at different levels



Dendrogram / Tree

Average linkage

- Average linkage between two clusters C, C'

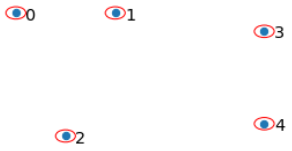
$$d_{avg}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{x \in C, x' \in C'} d(x, x') \qquad d = \text{distance metric}$$

Average linkage

- Average linkage between two clusters C, C'

$$d_{avg}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{x \in C, x' \in C'} d(x, x') \qquad d = \text{distance metric}$$

1. Start with m singleton clusters, $C_i = \{x_i\}$



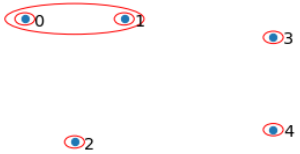
Average linkage

- Average linkage between two clusters C, C'

$$d_{avg}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{x \in C, x' \in C'} d(x, x') \qquad d = \text{distance metric}$$

1. Start with m singleton clusters, $C_i = \{x_i\}$

2. Merge clusters C_i, C_j that have smallest $d_{avg}(C_i, C_j)$



Average linkage

- Average linkage between two clusters C, C'

$$d_{avg}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{x \in C, x' \in C'} d(x, x') \qquad d = \text{distance metric}$$

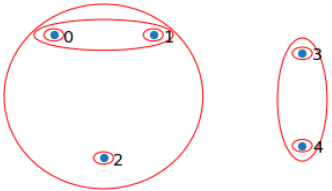
1. Start with m singleton clusters, $C_i = \{x_i\}$
2. Merge clusters C_i, C_j that have smallest $d_{avg}(C_i, C_j)$
3. Repeat step-2 till all clusters merged

Average linkage

- Average linkage between two clusters C, C'

$$d_{avg}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{x \in C, x' \in C'} d(x, x') \qquad d = \text{distance metric}$$

1. Start with m singleton clusters, $C_i = \{x_i\}$
2. Merge clusters C_i, C_j that have smallest $d_{avg}(C_i, C_j)$
3. Repeat step-2 till all clusters merged

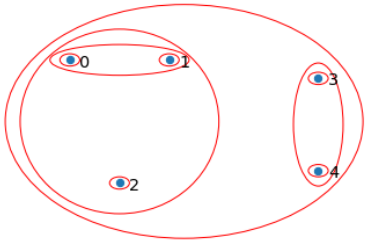


Average linkage

- Average linkage between two clusters C, C'

$$d_{avg}(C, C') = \frac{1}{|C| \cdot |C'|} \sum_{x \in C, x' \in C'} d(x, x') \qquad d = \text{distance metric}$$

1. Start with m singleton clusters, $C_i = \{x_i\}$
2. Merge clusters C_i, C_j that have smallest $d_{avg}(C_i, C_j)$
3. Repeat step-2 till all clusters merged



Agglomerative vs divisive clustering

- Agglomerative clustering
  - Initialisation: Each cluster contains a single data
  - Recursion: Merge most similar clusters at each level
  - Example: Average linkage

Agglomerative vs divisive clustering

- Agglomerative clustering
  - Initialisation: Each cluster contains a single data
  - Recursion: Merge most similar clusters at each level
  - Example: Average linkage
- Divisive clustering
  - Initialisation: Entire set is a single cluster
  - Recursion: Split each cluster into smaller clusters
  - Example: Recursive k-means

Analysing hierarchical clustering

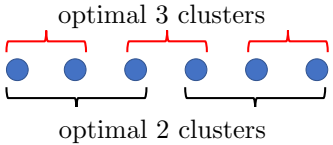
- How can we measure goodness of obtained tree / dendrogram?
- Is hierarchical clustering an optimisation problem?

Analysing hierarchical clustering

- How can we measure goodness of obtained tree / dendrogram?
- Is hierarchical clustering an optimisation problem?
- Approach 1:
  - Measure goodness / cost of induced k-way clustering for every $k \geq 2$

Analysing hierarchical clustering

- How can we measure goodness of obtained tree / dendrogram?
- Is hierarchical clustering an optimisation problem?
- Approach 1:
  - Measure goodness / cost of induced k-way clustering for every $k \geq 2$
  - Example: No tree optimal for both $k = 2, 3$

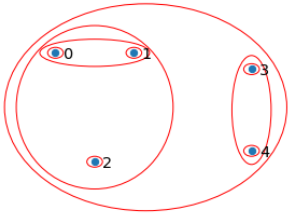


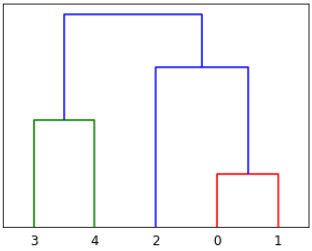
Analysing hierarchical clustering

- How can we measure goodness of obtained tree / dendrogram?
- Is hierarchical clustering an optimisation problem?
- Approach 2:
  - Define new cost / value function for a dendrogram
  - Formulate hierarchical clustering as optimisation over all dendrograms

Cost / value of dendrogram

[Dasgupta 2016; Cohen-Addad et al 2018]

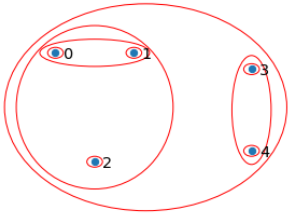


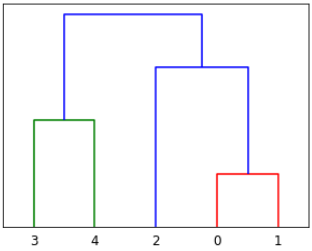


- T = binary tree / dendrogram
  - value(T) = measure of goodness of clusters in the tree

Cost / value of dendrogram

[Dasgupta 2016; Cohen-Addad et al 2018]





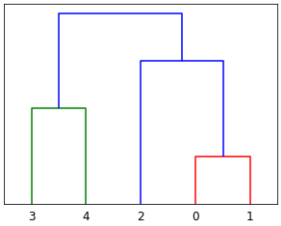
- T = binary tree / dendrogram
  - value(T) = measure of goodness of clusters in the tree
- N = node in tree = corresponding cluster
  - N_1, N_2 = children of N

Value function of dendrogram

[Cohen-Addad et al 2018]

- Distance between two nodes

$$d(N_1, N_2) = \sum_{x \in N_1} \sum_{x' \in N_2} d(x, x')$$

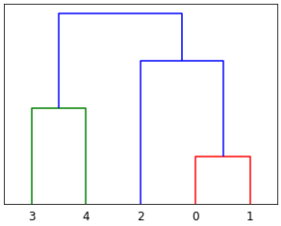


Value function of dendrogram

[Cohen-Addad et al 2018]

- Distance between two nodes

$$d(N_1, N_2) = \sum_{x \in N_1} \sum_{x' \in N_2} d(x, x')$$



- Value function for T

$$\begin{aligned} \text{value}(T) &= \sum_{N \in T} d(N_1, N_2) \cdot |N| \\ &= \sum_{x \neq x'} d(x, x') \cdot |\text{lca}(x, x')| \end{aligned}$$

- $\text{lca}(x, x')$ = smallest cluster / node containing both x, x' (least common ancestor)

Hierarchical clustering as optimisation

- High value(T) if

$$\begin{aligned} \text{high } d(x, x') &\implies x, x' \text{ merged higher in } T \\ \text{closer } x, x' &\implies x, x' \text{ merged towards bottom of } T \end{aligned}$$

Hierarchical clustering as optimisation

- High value(T) if

$$\begin{aligned} \text{high } d(x, x') &\implies x, x' \text{ merged higher in } T \\ \text{closer } x, x' &\implies x, x' \text{ merged towards bottom of } T \end{aligned}$$

- Formal hierarchical clustering problem

$$\underset{T = \text{binary tree}}{\text{maximise}} \quad \text{value}(T)$$

Hierarchical clustering as optimisation

- High value(T) if

$$\begin{aligned} \text{high } d(x, x') &\implies x, x' \text{ merged higher in } T \\ \text{closer } x, x' &\implies x, x' \text{ merged towards bottom of } T \end{aligned}$$

- Formal hierarchical clustering problem

$$\underset{T = \text{binary tree}}{\text{maximise}} \ \text{value}(T)$$

- Approximation guarantee for average linkage

$$\text{value}\left(\widehat{T}_{avg-linkage}\right) \ \geq \ \frac{1}{2} \cdot \max_{T} \ \text{value}(T)$$

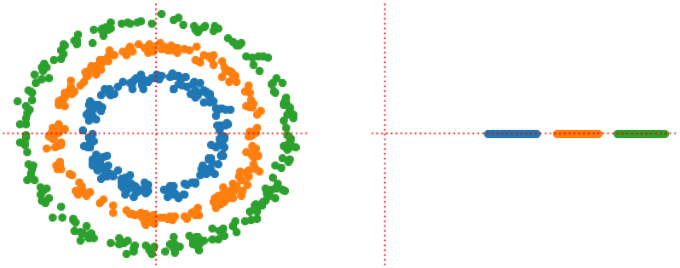
Deep clustering

Motivation

- Drawback of k-means:
  - k-means can find only non-overlapping spherical clusters

Motivation

- Drawback of k-means:
  - k-means can find only non-overlapping spherical clusters
- Sometimes we can transform the data to get this



x $f(x) = \|x\|$

Few characteristics of deep learning

- Learn a complex function $x \mapsto f(x)$ suitable for tasks

Few characteristics of deep learning

- Learn a complex function $x \mapsto f(x)$ suitable for tasks
- Representation learning
  - Learn representation $f(x)$ in unsupervised manner (autoencoder)
  - Can also learn a generative model for x

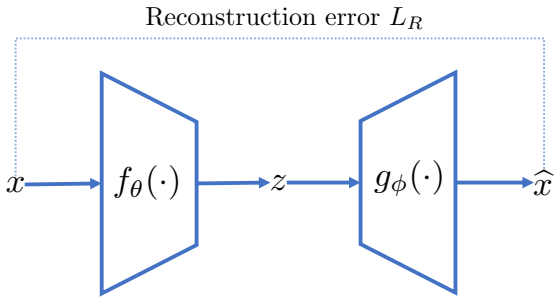
Few characteristics of deep learning

- Learn a complex function $x \mapsto f(x)$ suitable for tasks
- Representation learning
  - Learn representation $f(x)$ in unsupervised manner (autoencoder)
  - Can also learn a generative model for x
- Every problem boils down to a large optimisation

Clustering with Autoencoder

[Xie et al 2016; Dizaji et al 2017]

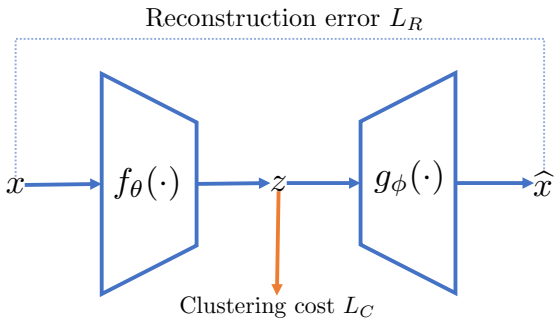
- Autoencoder finds low dimensional representation by minimising reconstruction error



Clustering with Autoencoder

[Xie et al 2016; Dizaji et al 2017]

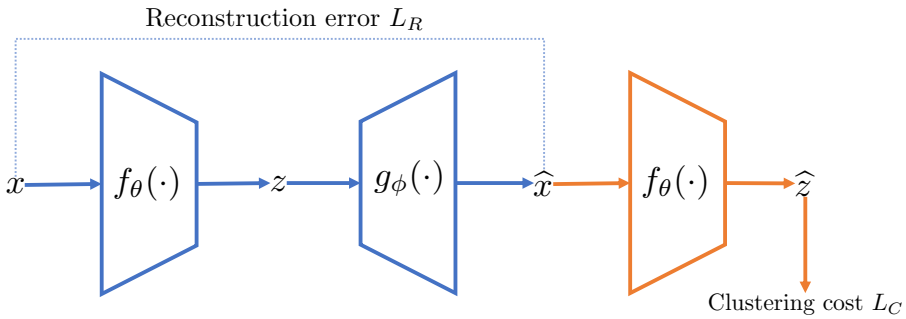
- Autoencoder finds low dimensional representation by minimising reconstruction error
- Cluster the transformed data by minimising clustering cost [Xie et al 2016]



Clustering with Autoencoder

[Xie et al 2016; Dizaji et al 2017]

- Autoencoder finds low dimensional representation by minimising reconstruction error
- Cluster the transformed data by minimising clustering cost [Dizaji et al 2017]



Clustering with Autoencoder [Xie et al 2016; Dizaji et al 2017]

- Autoencoder finds low dimensional representation by minimising reconstruction error
- Cluster the transformed data by minimising clustering cost
- Different ways to minimise both L_R and L_C

Clustering with Autoencoder [Xie et al 2016; Dizaji et al 2017]

- Autoencoder finds low dimensional representation by minimising reconstruction error
- Cluster the transformed data by minimising clustering cost
- Different ways to minimise both L_R and L_C
- No theoretical guarantee
- Nice overview: P. Dahal. Deep clustering

Density based clustering

Statistical view of clustering

- Previous perspective:
  - Observed data are some fixed entities
  - Clustering = optimisation problems

Statistical view of clustering

- Previous perspective:
  - Observed data are some fixed entities
  - Clustering = optimisation problems
- Statistical view:
  - Observations are manifestations of hidden laws of nature

... often corrupted by noise

Statistical view of clustering

- Previous perspective:
  - Observed data are some fixed entities
  - Clustering = optimisation problems
- Statistical view:
  - Observations are manifestations of hidden laws of nature

    ... often corrupted by noise
  - Data = independent samples from some probabilistic model

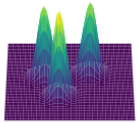
Statistical view of clustering

- Previous perspective:
  - Observed data are some fixed entities
  - Clustering = optimisation problems
- Statistical view:
  - Observations are manifestations of hidden laws of nature
    ... often corrupted by noise
  - Data = independent samples from some probabilistic model
    ... model has cluster structure (mixture model)
  - Clustering = (part of) model estimation

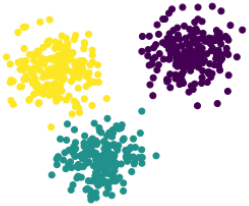
Gaussian mixture model

Data

Fitted GMM



Clusters

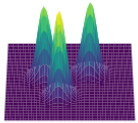


- Assume data = samples from mixture of d-variate Gaussians

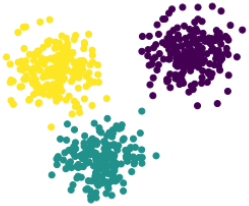
Gaussian mixture model

Data

Fitted GMM



Clusters

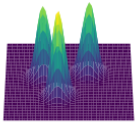


- Assume data = samples from mixture of d-variate Gaussians
- Fit GMM model: $\mathcal{D}_x = w_1\mathcal{N}_d(\mu_1, \Sigma_1) + w_2\mathcal{N}_d(\mu_2, \Sigma_2) + \ldots + w_k\mathcal{N}_d(\mu_k, \Sigma_k)$

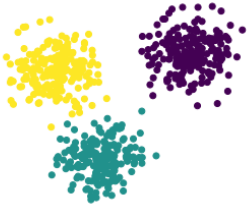
Gaussian mixture model

Data

Fitted GMM



Clusters



- Assume data = samples from mixture of d-variate Gaussians
- Fit GMM model: $\mathcal{D}_x = w_1\mathcal{N}_d(\mu_1, \Sigma_1) + w_2\mathcal{N}_d(\mu_2, \Sigma_2) + \ldots + w_k\mathcal{N}_d(\mu_k, \Sigma_k)$
- Typically solved via Expectation Maximisation (EM)

GMM estimation via k-means

- Data $= x_1, \ldots, x_n$
- Define $Z \in \{0,1\}^{n \times k}$: $Z_{ij} = 1$ if and only if x_i assigned to cluster j

GMM estimation via k-means

- Data $= x_1, \ldots, x_n$
- Define $Z \in \{0,1\}^{n \times k}$: $\quad Z_{ij} = 1$ if and only if x_i assigned to cluster j
- Lloyd's k-means: Iterate between
  1. (for every i) Z-update : $\quad Z_{ij} = 1$ if x_i is closest to μ_j, and 0 for other j
  2. (for every j) μ-update: $\quad \mu_j = \dfrac{\sum_{i=1}^{n} Z_{ij} x_i}{\sum_{i=1}^{n} Z_{ij}}$

GMM estimation via k-means

- Data $= x_1, \ldots, x_n$
- Define $Z \in \{0,1\}^{n \times k}$: $\quad Z_{ij} = 1$ if and only if x_i assigned to cluster j
- GMM estimation via k-means: Iterate between
  1. (for every i) Z-update : $\quad Z_{ij} = 1$ if x_i is closest to μ_j, and 0 for other j
  2. (for every j) μ-update: $\quad \mu_j = \dfrac{\sum_{i=1}^{n} Z_{ij} x_i}{\sum_{i=1}^{n} Z_{ij}}$

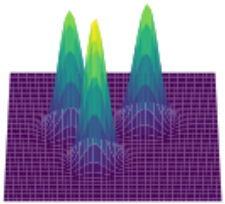
Σ-update: $\quad \Sigma_j = \dfrac{\sum_{i=1}^{n} Z_{ij}(x_i - \mu_j)(x_i - \mu_j)^T}{\sum_{i=1}^{n} Z_{ij}}$ $\qquad$ w-update: $\quad w_j = \dfrac{\sum_{i=1}^{n} Z_{ij}}{n}$

EM as soft k-means

- Expectation-maximisation (EM): [Dempster et al 1977]
  - Iterative solution for maximum likelihood estimation (MLE) of model parameters

EM as soft k-means

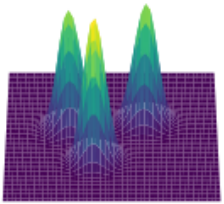
- Expectation-maximisation (EM): [Dempster et al 1977]
  - Iterative solution for maximum likelihood estimation (MLE) of model parameters
- GMM has soft cluster assignments



EM as soft k-means

- Expectation-maximisation (EM): [Dempster et al 1977]
  - Iterative solution for maximum likelihood estimation (MLE) of model parameters
- GMM has soft cluster assignments

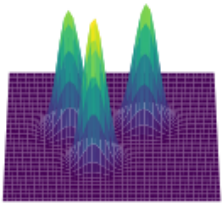
$$Z_{ij} = \text{Prob}\big(x_i \sim \mathcal{N}_d(\mu_j, \Sigma_j) \;\big|\; w, \mu, \Sigma\big)$$



EM as soft k-means

- Expectation-maximisation (EM): [Dempster et al 1977]
  - Iterative solution for maximum likelihood estimation (MLE) of model parameters
- GMM has soft cluster assignments

$$Z_{ij} = \text{Prob}\big(x_i \sim \mathcal{N}_d(\mu_j, \Sigma_j) \;\big|\; w, \mu, \Sigma\big)$$

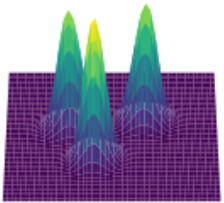


- Z-update in EM: $$Z_{ij} = \frac{w_j \exp\left(-(x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j)\right)}{\sum_{l=1}^{k} w_j \exp\left(-(x_i - \mu_l)^T \Sigma_l^{-1} (x_i - \mu_l)\right)}$$

EM as soft k-means

- Expectation-maximisation (EM): [Dempster et al 1977]
  - Iterative solution for maximum likelihood estimation (MLE) of model parameters
- GMM has soft cluster assignments

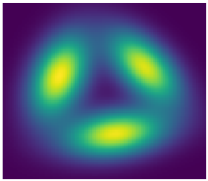
$$Z_{ij} = \text{Prob}\big(x_i \sim \mathcal{N}_d(\mu_j, \Sigma_j) \;\big|\; w, \mu, \Sigma\big)$$



- Z-update in EM: $$Z_{ij} = \frac{w_j \exp\left(-(x_i - \mu_j)^T \Sigma_j^{-1} (x_i - \mu_j)\right)}{\sum_{l=1}^{k} w_j \exp\left(-(x_i - \mu_l)^T \Sigma_l^{-1} (x_i - \mu_l)\right)}$$
- w, μ and Σ updates in EM: Same as previous slide

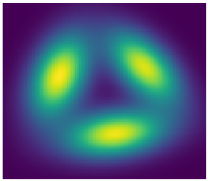
Limitations of GMM (and k-means)

- Finds only elliptical clusters



Limitations of GMM (and k-means)

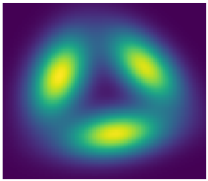
- Finds only elliptical clusters



- Cannot deal with outliers

Limitations of GMM (and k-means)

- Finds only elliptical clusters



- Cannot deal with outliers

- Need to know number of clusters

Nonparametric clustering

- Nonparametric = No parametric assumption on data or clusters
- Examples: [?] GMM [?] k-means [?] average linkage

Nonparametric clustering

- Nonparametric = No parametric assumption on data or clusters
- Examples: [×] GMM [!!] k-means [✓] average linkage

Nonparametric clustering

- Nonparametric = No parametric assumption on data or clusters
- Examples: [×] GMM [!!] k-means [✓] average linkage
- Nonparametric density based clustering:
  - Data = independent samples from a probabilistic model
  - No parametric assumption on model, not even k

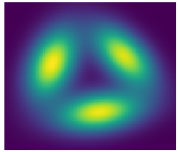
Nonparametric clustering

- Nonparametric = No parametric assumption on data or clusters
- Examples: [×] GMM [!!] k-means [✓] average linkage
- Nonparametric density based clustering:
  - Data = independent samples from a probabilistic model
  - No parametric assumption on model, not even k
  - Example: DBSCAN [Ester et al 1996]
    Density-based Spatial Clustering of Applications with Noise

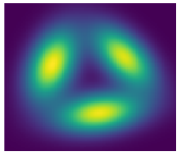
Philosophy behind DBSCAN

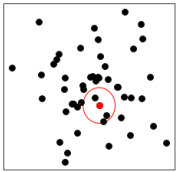
- Clusters are high density regions

  . . . and points in low density region are outliers



Philosophy behind DBSCAN





- Clusters are high density regions

  ... and points in low density region are outliers

- Estimate the density locally, around a point
  - Exercise: Consider ball $B(x, \epsilon)$ centred at x and radius ϵ

$$x \sim \mathcal{D} \quad \Longrightarrow \quad \text{pdf}_{\mathcal{D}}(x) \approx \frac{\#\text{points in } B(x,\epsilon)}{\text{total } \#\text{points}} \cdot \frac{1}{\text{volume}(B(x,\epsilon))}$$

Philosophy behind DBSCAN

- Clusters are high density regions

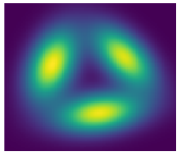
  ... and points in low density region are outliers

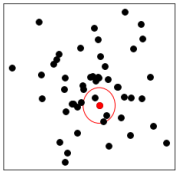
- Estimate the density locally, around a point
  - Exercise: Consider ball $B(x, \epsilon)$ centred at x and radius ϵ

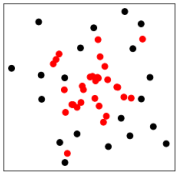
$$x \sim \mathcal{D} \quad \Longrightarrow \quad \text{pdf}_{\mathcal{D}}(x) \approx \frac{\#\text{points in } B(x,\epsilon)}{\text{total } \#\text{points}} \cdot \frac{1}{\text{volume}(B(x,\epsilon))}$$

- Can threshold $\text{pdf}_{\mathcal{D}}(x)$ to decide if point inside cluster or outlier

  ... or threshold based on #points in $B(x, \epsilon)$







DBSCAN

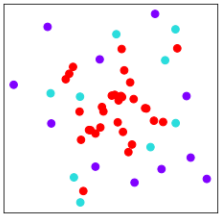
[Ester et al 1996]

- Hyperparameters: Radius $= \epsilon$, threshold $= \texttt{minPts}$

DBSCAN

[Ester et al 1996]

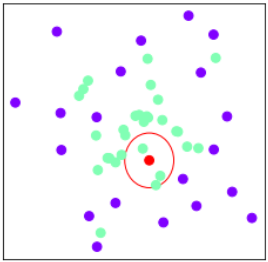
- Hyperparameters: Radius $= \epsilon$, threshold $=$ `minPts`
- Types of points
  - Core point: #points in $B(x, \epsilon) \geq$ `minPts`
  - Border point: #points in $B(x, \epsilon) <$ `minPts`, but lies in ball of core point
  - Outlier: neither core or border point



DBSCAN

[Ester et al 1996]

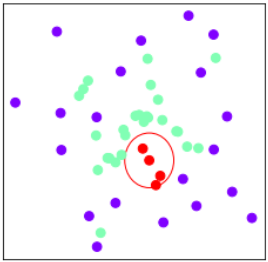
1. Select any **core point** x that is not clustered yet



DBSCAN

[Ester et al 1996]

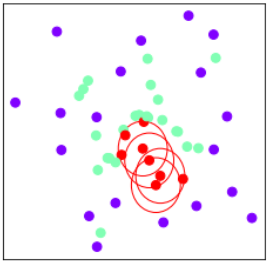
1. Select any **core point** x that is not clustered yet
2. Assign x to a new cluster
   i. Assign all points in $B(x, \epsilon)$ to same cluster



DBSCAN

[Ester et al 1996]

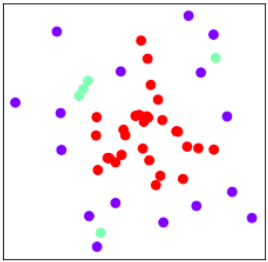
1. Select any **core point** x that is not clustered yet
2. Assign x to a new cluster
   i. Assign all points in $B(x, \epsilon)$ to same cluster
   ii. If $y \in B(x, \epsilon)$ is **core point**
      - Go to step-i with $B(y, \epsilon)$



DBSCAN

[Ester et al 1996]

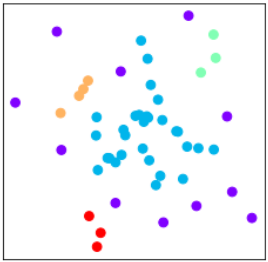
1. Select any **core point** x that is not clustered yet
2. Assign x to a new cluster
   i. Assign all points in $B(x, \epsilon)$ to same cluster
   ii. If $y \in B(x, \epsilon)$ is **core point**
      - Go to step-i with $B(y, \epsilon)$
   iii. Propagate labels (i - ii) till we cannot reach any more core point



DBSCAN

[Ester et al 1996]

1. Select any **core point** x that is not clustered yet
2. Assign x to a new cluster
   i. Assign all points in $B(x, \epsilon)$ to same cluster
   ii. If $y \in B(x, \epsilon)$ is **core point**
      - Go to step-i with $B(y, \epsilon)$
   iii. Propagate labels (i - ii) till we cannot reach any more core point
3. Repeat steps 1-2 till there is no more unclustered core point



Theoretical analyses

- GMM is theoriticians' favourite model [Dasgupta 1999; Vankadara & Ghoshdastidar 2020]
  - Used to analyse many clustering algorithms
  - Sample data from GMM, and prove algorithms can correctly cluster them

Theoretical analyses

- GMM is theoriticians' favourite model [Dasgupta 1999; Vankadara & Ghoshdastidar 2020]
  - Used to analyse many clustering algorithms
  - Sample data from GMM, and prove algorithms can correctly cluster them
- Less understanding of clustering mixture of nonparametric distributions

Theoretical analyses

- GMM is theoriticians' favourite model [Dasgupta 1999; Vankadara & Ghoshdastidar 2020]
  - Used to analyse many clustering algorithms
  - Sample data from GMM, and prove algorithms can correctly cluster them
- Less understanding of clustering mixture of nonparametric distributions
- Huge literature on performance / convergence of EM algorithm

Theoretical analyses

- GMM is theoriticians' favourite model [Dasgupta 1999; Vankadara & Ghoshdastidar 2020]
  - Used to analyse many clustering algorithms
  - Sample data from GMM, and prove algorithms can correctly cluster them
- Less understanding of clustering mixture of nonparametric distributions
- Huge literature on performance / convergence of EM algorithm
- DBSCAN finds level sets of probability distributions [Sriperumbudur & Steinwart 2012]

Semidefinite programming and similarity based clustering

Reformulating k-means problem

- Co-occurrence matrix $M \in \mathbb{R}^{n \times n}$ for clusters $C_1, \ldots, C_k$

$$M_{ij} = \begin{cases} \dfrac{1}{|C_\ell|} & \text{if both } x_i, x_j \in C_\ell \\ 0 & \text{if } x_i \text{ and } x_j \text{ in different clusters} \end{cases}$$

Reformulating k-means problem

- Co-occurrence matrix $M \in \mathbb{R}^{n \times n}$ for clusters $C_1, \ldots, C_k$

$$M_{ij} = \begin{cases} \dfrac{1}{|C_\ell|} & \text{if both } x_i, x_j \in C_\ell \\ 0 & \text{if } x_i \text{ and } x_j \text{ in different clusters} \end{cases}$$

- Define $X \in \mathbb{R}^{n \times p}$ where rows are data points $x_1, \ldots, x_n$

- Rewriting k-means cost (exercise)

$$\sum_{j=1}^{k} \sum_{x_i \in C_j} \|x_i - c_j\|^2 \quad = \quad \|X - MX\|_F^2 \qquad\qquad \|A\|_F^2 := \sum_{i,j} A_{ij}^2$$

Reformulating k-means problem

- Equivalent k-means optimisation

$$\underset{M \in \mathbb{R}^{n \times n}}{\text{maximise}} \quad \text{trace}\left(XX^T M\right) \qquad \text{trace}(A) = \sum_i A_{ii}$$

$$\text{s.t.} \quad M \text{ is co-occurrence for some } C_1, \ldots, C_k$$

Reformulating k-means problem

- Equivalent k-means optimisation

$$\underset{M \in \mathbb{R}^{n \times n}}{\text{maximise}} \quad \text{trace}\left(XX^T M\right) \qquad \text{trace}(A) = \sum_i A_{ii}$$

$$\text{s.t.} \quad M \text{ is co-occurrence for some } C_1, \ldots, C_k$$

- Above is still combinatorial problem (computationally expensive)
- Relaxation: Replace constraint by simpler condition satisfied by co-occurrence matrix

Reformulating k-means problem

- Equivalent k-means optimisation

$$\underset{M \in \mathbb{R}^{n \times n}}{\text{maximise}} \quad \text{trace}\left(XX^T M\right) \qquad \text{trace}(A) = \sum_i A_{ii}$$

$$\text{s.t.} \quad M \text{ is co-occurrence for some } C_1, \ldots, C_k$$

- Above is still combinatorial problem (computationally expensive)

- Relaxation: Replace constraint by simpler condition satisfied by co-occurrence matrix

(i) M has non-negative entries

(ii) M is positive semi-definite

(iii) $\text{trace}(M) = k$

(iv) M has row sum 1

Semidefinite programming (SDP)

- Optimisation over positive semidefinite (psd) matrices
- Objective and other constraints linear

Semidefinite programming (SDP)

- Optimisation over positive semidefinite (psd) matrices
- Objective and other constraints linear
- Convex optimisation problem:
  - Many practical software, including few specific for SDP

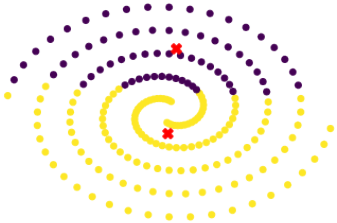
Semidefinite programming (SDP)

- Optimisation over positive semidefinite (psd) matrices
- Objective and other constraints linear
- Convex optimisation problem:
  - Many practical software, including few specific for SDP
- SDP relaxation of k-means: [Peng & Wei, 2007]

$$\begin{aligned} \underset{M \text{ is psd}}{\text{maximise}} \quad & \operatorname{trace}\left(XX^T M\right) \\ \text{s.t.} \quad & M_{ij} \geq 0, \qquad \textstyle\sum_j M_{ij} = 1, \qquad \operatorname{trace}(M) = k \end{aligned}$$

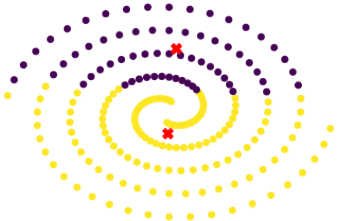
Drawbacks of k-means problem

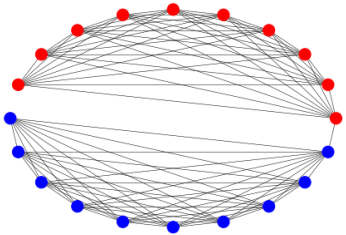
- k-means produces linear cluster boundaries
  - Can only find convex non-overlapping clusters



Drawbacks of k-means problem

- k-means produces linear cluster boundaries
  - Can only find convex non-overlapping clusters
- k-means requires data representation $x_1, \ldots, x_n$
  - What happens if we can only observe similarity between two items?





Similarity (kernel) matrix

- $S = n \times n$ symmetric matrix

$$S_{ij} = \text{ similarity (kernel) function between } x_i \text{ and } x_j$$

- S could be computed from data or directly observed
- Examples:
  - Gaussian kernel: $S_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{\sigma^2}\right)$
  - Graph: $S =$ adjacency matrix of a similarity graph

Similarity based clustering

- Given $S \in \mathbb{R}^{n \times n}$, find

$$\text{k-clustering of } n \text{ items} \qquad \text{or} \qquad \text{co-occurence matrix } M$$

Similarity based clustering

- Given $S \in \mathbb{R}^{n \times n}$, find

$$\text{k-clustering of } n \text{ items} \qquad \text{or} \qquad \text{co-occurence matrix } M$$

- k-means uses a linear similarity

$$S_{ij} = x_i^T x_j \qquad \text{or} \qquad S = XX^T$$

Similarity based clustering

- Given $S \in \mathbb{R}^{n \times n}$, find

$$\text{k-clustering of } n \text{ items} \qquad \text{or} \qquad \text{co-occurence matrix } M$$

- k-means uses a linear similarity

$$S_{ij} = x_i^T x_j \qquad \text{or} \qquad S = XX^T$$

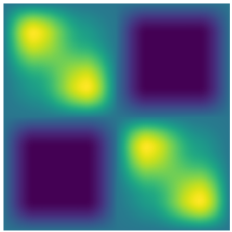
- Similarity / kernel SDP: [Vankadara & Ghoshdastidar 2020]

$$\begin{aligned} \underset{M \text{ is psd}}{\text{maximise}} \quad & \text{trace}\,(SM) \\ \text{s.t.} \quad & M_{ij} \geq 0, \quad \textstyle\sum_j M_{ij} = 1, \quad \text{trace}(M) = k \end{aligned}$$

Example

- S = similarity based on mutual 2-nearest neighbours (2-NN)

$$S_{ij} = \begin{cases} 1 & \text{if } i = j \\ 1 & \text{if } x_i \text{ is 2-NN of } x_j \text{ and } x_j \text{ is one of 2-NN of } x_i \\ 0 & \text{otherwise} \end{cases}$$



Co-occurrence matrix M

Obtained clusters

Remarks

- Getting clusters from M
  - Use clustering (example: direct k-means on rows of M)

Remarks

- Getting clusters from M
  - Use clustering (example: direct k-means on rows of M)
- Unknown number of clusters [Yan et al 2018, Perrot et al 2020]

λ-SDP: $\underset{M \text{ is psd}}{\text{maximise}} \ \text{trace}\,(SM) - \lambda \cdot \text{trace}(M)$ $\lambda =$ hyperparameter

$$\text{s.t.} \quad M_{ij} \geq 0, \qquad \sum_j M_{ij} = 1$$

Remarks

- Getting clusters from M
  - Use clustering (example: direct k-means on rows of M)
- Unknown number of clusters [Yan et al 2018, Perrot et al 2020]

λ-SDP: $\underset{M \text{ is psd}}{\text{maximise}} \ \text{trace}\,(SM) - \lambda \cdot \text{trace}(M)$ $\lambda =$ hyperparameter

s.t. $M_{ij} \geq 0, \quad \sum_j M_{ij} = 1$

- Theoretical guarantees for similarity SDP
  - Graph / Ordinal data clustering [Yan et al 2018, Perrot et al 2020]

How good is the clustering?

Clustering metrics

- Implicit goodness of clustering

- Comparison with ground truth

Clustering metrics

- Implicit goodness of clustering
  - Define measure of goodness of clusters
  - Tied to our belief of how good clusters should look like
  - Example: k-means cost, silhouette score
- Comparison with ground truth

Clustering metrics

- Implicit goodness of clustering
  - Define measure of goodness of clusters
  - Tied to our belief of how good clusters should look like
  - Example: k-means cost, silhouette score
- Comparison with ground truth
  - Measure difference between two clustering / co-occurrences
  - Example: Classification error (up to permutation of labels)
    Adjusted Rand index (ARI)
    Normalsed mutual information (NMI)

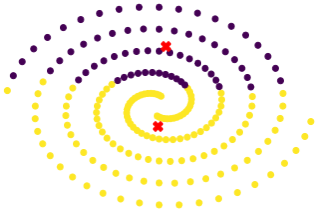
Clustering stability [Ben-David et al 2006]

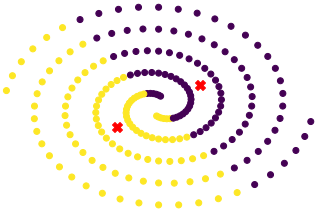
- If data is perturbed slightly, clustering should not change a lot
- Stability: Distance between clusterings of datasets sampled from same distrbution

Clustering stability

[Ben-David et al 2006]

- If data is perturbed slightly, clustering should not change a lot
- Stability: Distance between clusterings of datasets sampled from same distrbution





Different runs of k-means++ give different clusterings

Similarity SDP returns stable clustering

Finding good and stable clusters

- How do we know if algorithm returns (good) stable clusters?

Finding good and stable clusters

- How do we know if algorithm returns (good) stable clusters?
- Theoretical analysis of algorithms:
  - Data has good clustering. Find this true clustering [Balcan et al 2013]
  - Data sampled from mixture models [Dasgupta 1999]

Finding good and stable clusters

- How do we know if algorithm returns (good) stable clusters?
- Theoretical analysis of algorithms:
  - Data has good clustering. Find this true clustering [Balcan et al 2013]
  - Data sampled from mixture models [Dasgupta 1999]
- Measuring stability:
  - Verify if obtained clustering is good and stable [Meila 2018]

Verifying 'goodness' via optimisation [Meila 2018]

- Clustering via minimising cost (k-means)

$$\text{cost}(M) \;=\; \text{cost achieved by co-occurrence } M$$

- Distance between clusterings $\;=\; \|M - M'\|_F$

Verifying 'goodness' via optimisation [Meila 2018]

- Clustering via minimising cost (k-means)

$$\text{cost}(M) \;=\; \text{cost achieved by co-occurrence } M$$

- Distance between clusterings $=$ $\|M - M'\|_F$

- Measure instability of solution M:

$$\epsilon(M) = \max_{\text{co-occurrence } M'} \left\{ \|M - M'\|_F \;:\; \text{cost}(M') \leq \text{cost}(M) \right\}$$

  - How far are other clusterings which are as good as M?
  - For some algorithms, this can be solved via SDP

References

1. D. Arthur, S. Vassilvitskii (2007). k-means++: The advantages of careful seeding. *SODA*

2. M.-F. Balcan, A. Blum, A. Gupta (2013). Clustering under approximation stability. *Journal of the ACM*

3. S. Ben-David, U. von Luxburg, D. Pál (2006). A Sober Look at Clustering Stability. *COLT*

4. V. Cohen-Addad, V. Kanade, F. Mallmann-Trenn, C. Mathieu (2018). Hierarchical clustering: objective functions and algorithms. *SODA*

5. S. Dasgupta (1999). Learning mixtures of Gaussians. *FOCS*

6. S. Dasgupta (2016). A cost function for similarity-based hierarchical clustering. *STOC*

7. A. P. Dempster, N. M. Laird, D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*

8. K. G. Dizaji, A. Herandi, C. Deng, W. Cai, H. Huang (2017). Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization. *ICCV*

9. M. Ester, H.-P. Kriegel, J. Sander, X. Xu (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD*

10. E. Lee, M. Schmidt, J. Wright (2017). Improved and simplified inapproximability for k-means. *Information Processing Letters*

1. S. P. Lloyd (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*

2. U. v. Luxburg, R. C. Williamson, I. Guyon (2012). Clustering: Science or art? *Workshop on Unsupervised and Transfer Learning*

3. M. Meila (2018). How to tell when a clustering is (approximately) correct using convex relaxations. *Neurips*

4. J. Peng, Y. Wei (2007). Approximating k-means-type clustering via semidefinite programming. *SIAM Journal on Optimization*

5. M. Perrot. P. Esser, D. Ghoshdastidar (2020). Near-optimal comparison based clustering. *Neurips*

16. B. Sriperumbudur, I. Steinwart (2012). Consistency and Rates for Clustering with DBSCAN. *AISTATS*

17. L. Vankadara, D. Ghoshdastidar (2020). On the optimality of kernels for high-dimensional clustering. *AISTATS*

18. J. Xie, R. Girshick, A. Farhadi (2016). Unsupervised deep embedding for clustering analysis. *ICML*

19. B. Yan, P. Sarkar, X. Cheng (2018). Provable estimation of the number of blocks in block models. *AISTATS*