机器学习引论

彭玺

pengxi@scu.edu.cn

www.pengxi.me

提纲

一．Review

二．Spectral clustering

- Manifold learning
- Graph cut

三．Summary

提纲

一．Review

二．Spectral clustering

- Manifold learning
- Graph cut

三．Summary

一、Review

k-means：学习k个means（均值），where each mean corresponds to a cluster center. In other words, k-means achieves clustering by learning/finding k cluster centers.

- Given a set of data points, group them into multiple clusters so that:
  - points within each cluster are similar to each other $\min \sum_{j} \sum_{\mathbf{x}_i \in \mathrm{C}_j} \|\mathbf{x}_i - \mathbf{u}_j\|_2^2$
  - points from different clusters are dissimilar $\max \sum_{i} \sum_{j} \|\mathbf{u}_i - \mathbf{u}_j\|_2^2$

i.e. the cluster assignment (label) and centers are unknown.

一、Review

k-means：学习k个means（均值），where each mean corresponds to a cluster center. In other words, k-means achieves clustering by learning/finding k cluster centers.

- Given a set of data points, group them into multiple clusters so that:
  - points within each cluster are similar to each other $\min \sum_{j} \sum_{\mathbf{x}_i \in \mathrm{C}_j} \|\mathbf{x}_i - \mathbf{u}_j\|_2^2$
  - points from different clusters are dissimilar $\max \sum_{i} \sum_{j} \|\mathbf{u}_i - \mathbf{u}_j\|_2^2$

Solution：Iteratively learning clustering assignment and cluster centers so that

一、Review

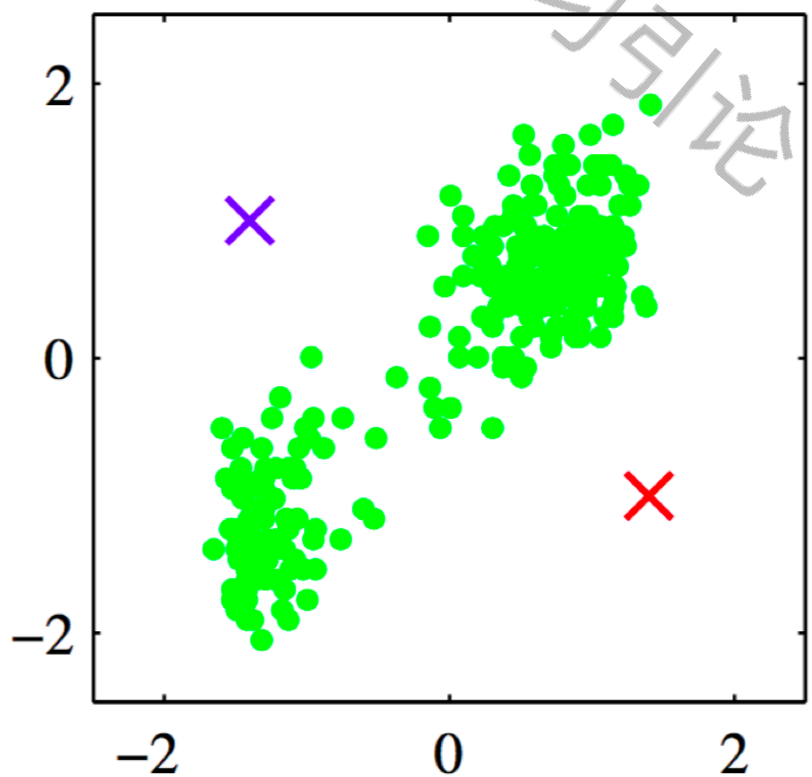
k-means：k个means（均值），clearly, each mean corresponds to a cluster center. In other words, k-means achieves clustering by learning/finding k cluster centers.

Solution: Iteratively learning clustering assignment and cluster centers so that

$$\min \sum_{j} \sum_{\mathbf{x}_i \in C_j} \|\mathbf{x}_i - \mathbf{u}_j\|_2^2 \quad \max \sum_{i} \sum_{j} \|\mathbf{u}_i - \mathbf{u}_j\|_2^2$$

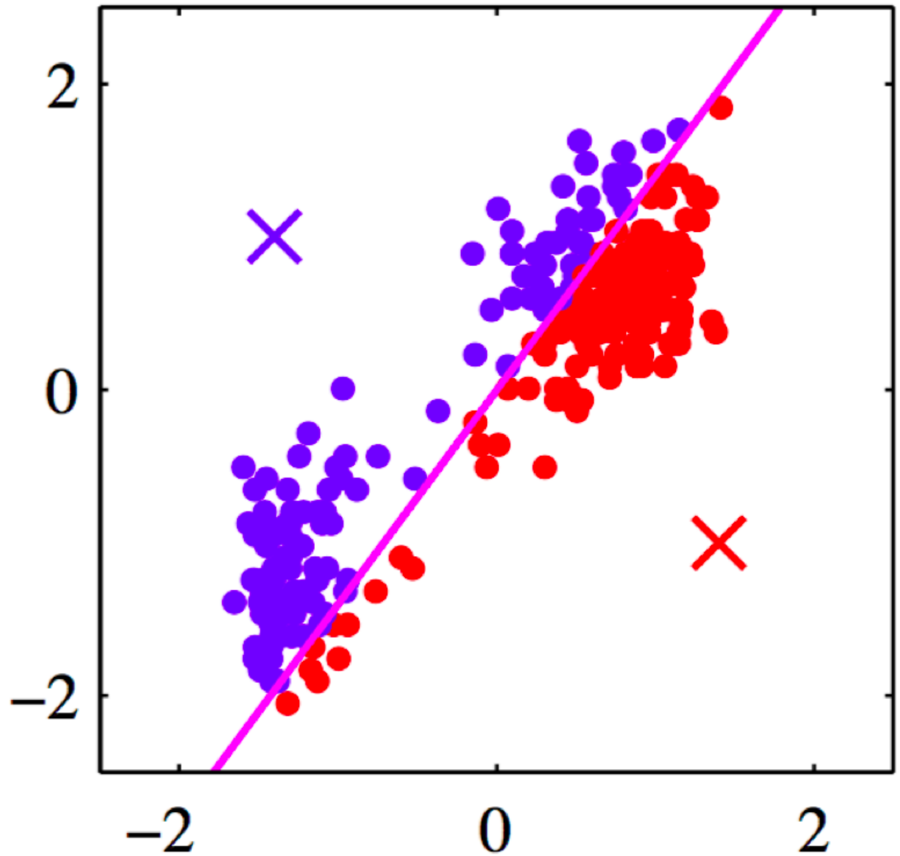
Example:

Iter1: randomly choose two points as cluster centers



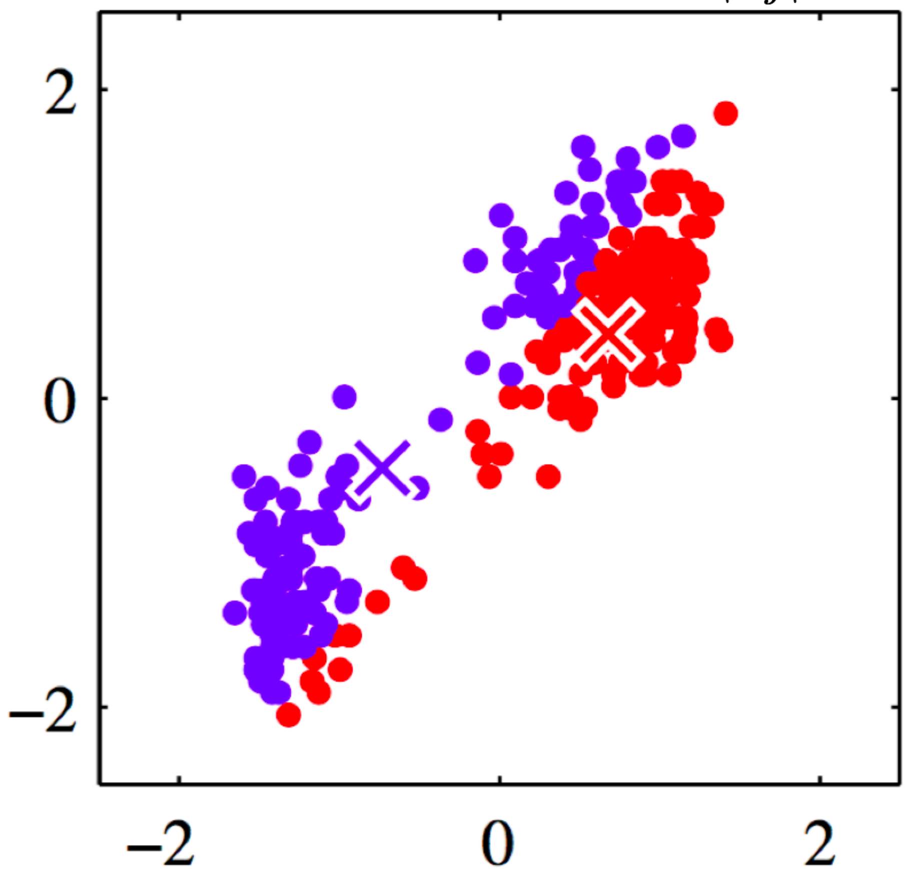
一、Review

Iter1: Assign each point to closest center.



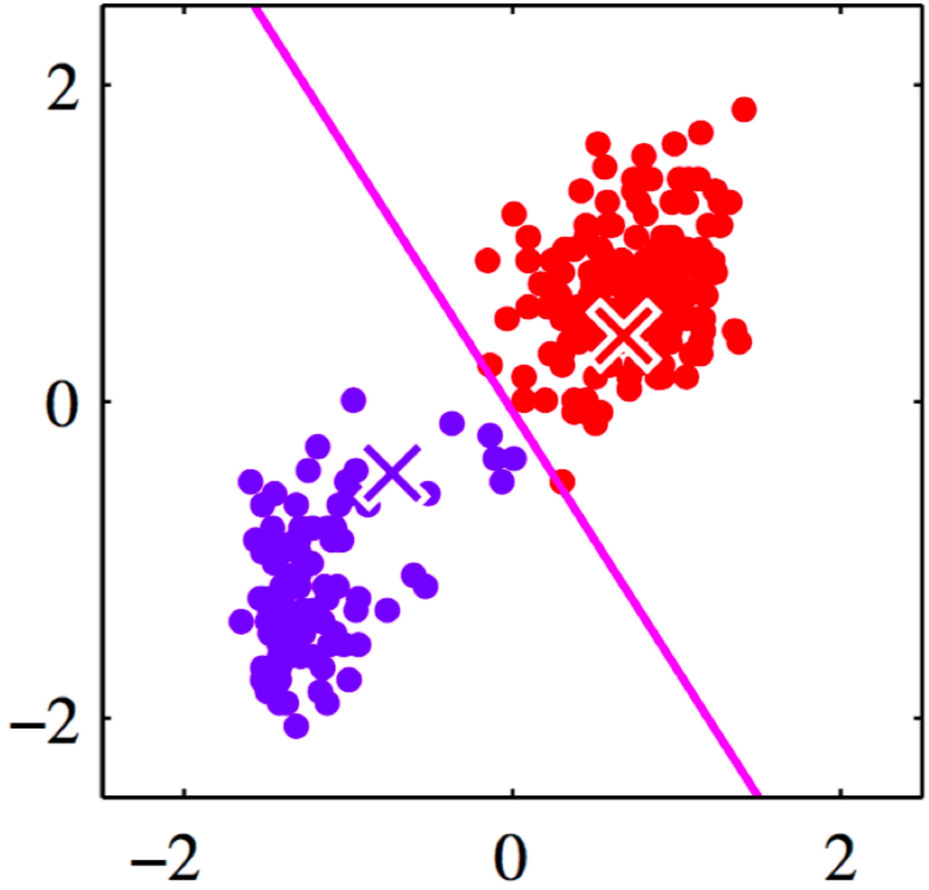
一、Review

Iter2: Compute new class centers by $\mu_j = \frac{\sum_{\mathbf{x}_i \in \mathcal{C}_j} \mathbf{x}_i}{|\mathcal{C}_j|}$



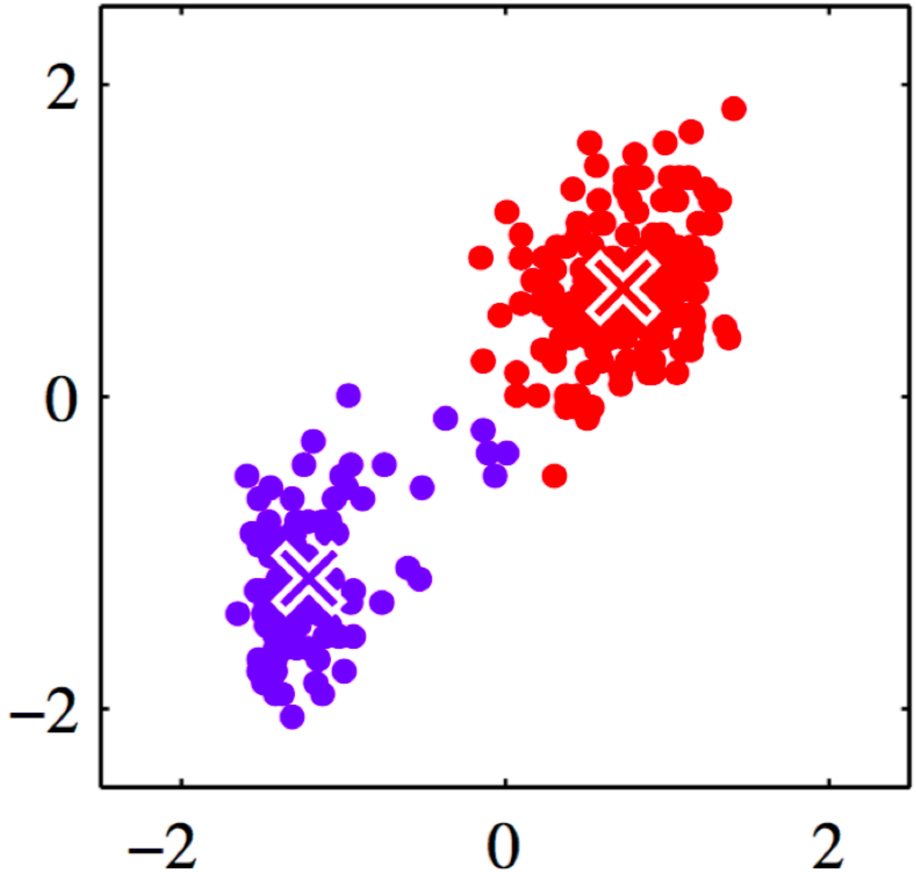
一、Review

Iter2: Assign points to closest center.



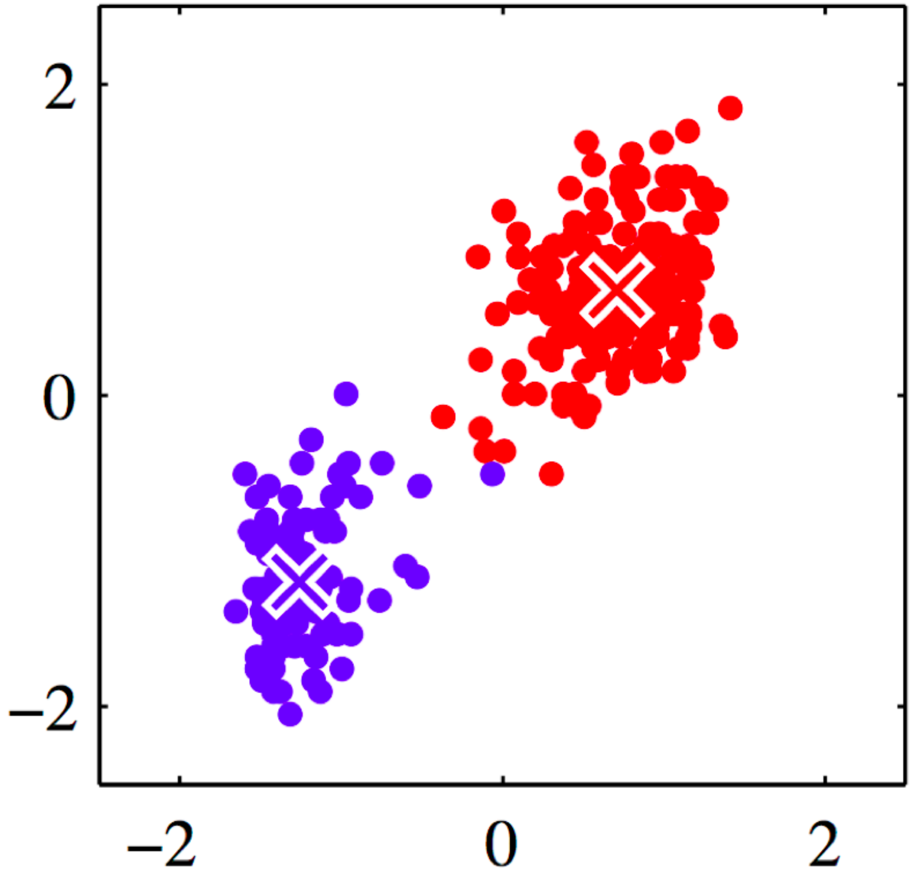
一、Review

Iter3: Compute cluster centers by $\mu_j = \frac{\sum_{\mathbf{x}_i \in \mathcal{C}_j} \mathbf{x}_i}{|\mathcal{C}_j|}$



一、Review

Iterate until convergence (reach to the max iteration number or the loss is smaller than a given threshold).



一、Review

- Dataset $\mathcal{D} = \{x_1, \ldots, x_n\} \in \mathbf{R}^d$
- Goal (version 1): Partition data into k clusters.
- Goal (version 2): Partition $\mathbf{R}^d$ into k regions.
- Let $\mu_1, \ldots, \mu_k$ denote cluster centers.
- For each x_i, use a **one-hot encoding** to designate membership:

$$r_i = (0, 0, \ldots, 0, 0, 1, 0, 0) \in \mathbf{R}^k$$

- Let

$$r_{ic} = 1(x_i \text{ assigned to cluster } c).$$

- Then

$$r_i = (r_{i1}, r_{i2}, \ldots, r_{ik}).$$

一、Review

- We will use an **alternating minimization** algorithm:
  1. Choose initial cluster centers $\mu = (\mu_1, \ldots, \mu_k)$.
     - e.g. choose k randomly chosen data points
  2. Repeat
     1. For given cluster centers, find optimal cluster assignments:

$$r_{ic}^{\text{new}} = 1(c = \arg\min_{j} \|x_i - \mu_j\|^2)$$

     2. Given cluster assignments, find optimal cluster centers:

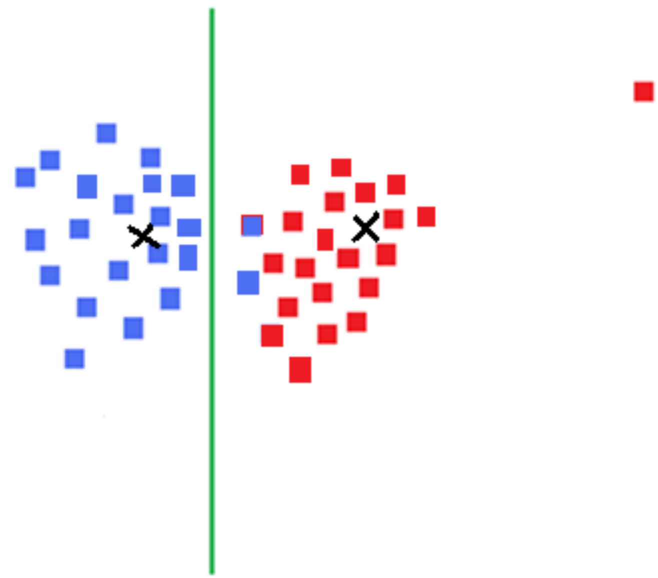
$$\mu_c^{\text{new}} = \arg\min_{m \in \mathbf{R}^d} \sum_{\{i|r_{ic}=1\}} \|x_i - \mu_c\|^2$$

一、Review

- Disadvantages
  - Dependent on initialization
    - Select random seeds with at least D_{min}
    - Or, run the algorithm many times

一、Review

- Disadvantages
  - Dependent on initialization
  - Sensitive to outliers



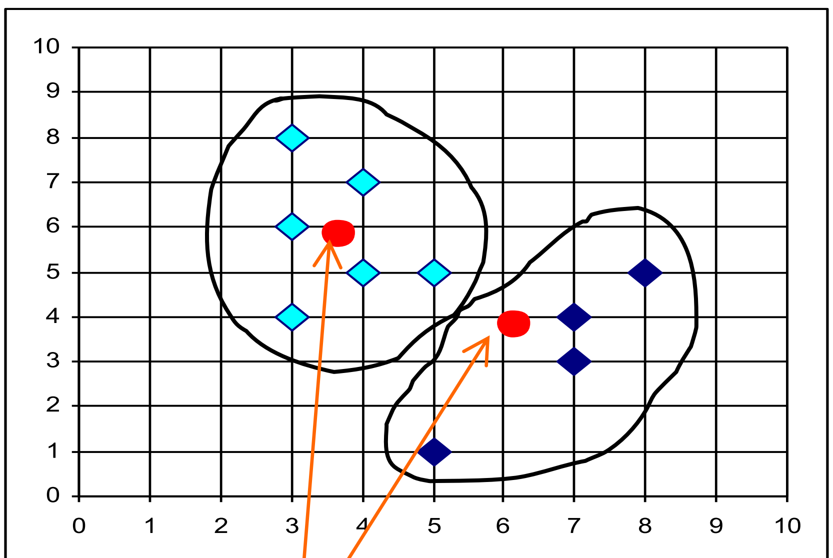
一、Review

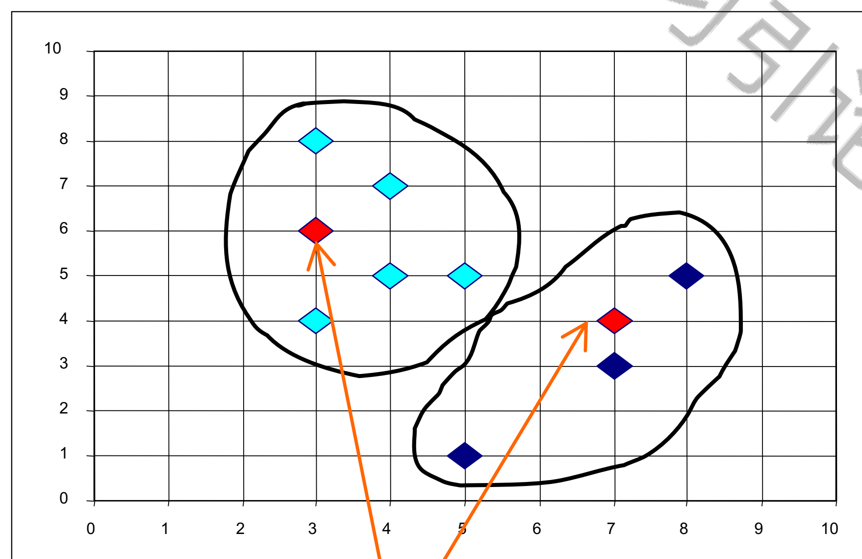
- *k*-means (MacQueen'67): Each cluster is represented by center of cluster
  - Sensitive to noise/outlier

- *k*-medoids (Kaufman & Rousseeuw'87): Each cluster is represented by one of the objects (medoid) in cluster
  - Robust to noise/outlier
  - keep the physical meaning of the dataset
  - Higher computational cost than *k*-means

一、Review

- ***k*-medoids: Find *k representative objects*, called *medoids***





k-means

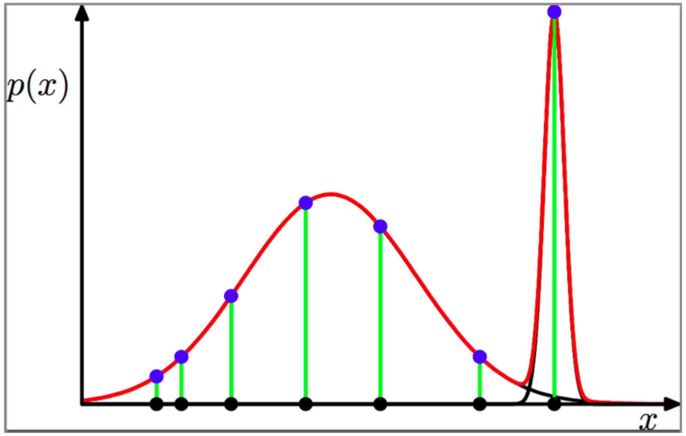
质心

k-medoids

中值，需要遍历整个数据集以得到medoid

一、Review

Universal Approximation: any distribution could be represented by a MOG, namely, any data set is a MOG and each cluster corresponds to a Gaussian distribution.



1-dimensional

$$\mathcal{N}(x|\mu,\sigma^2)=\frac{1}{2\pi\sigma^2}\exp(-\frac{1}{2\sigma^2}(x-\mu)^2)$$

Multivariate Gaussian

$$\mathcal{N}(\mathbf{x}|\mu,\mathbf{\Sigma})=\frac{1}{(2\pi)^{D/2}}\frac{1}{|\mathbf{\Sigma}|^{1/2}}\exp(-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mu))$$

一、Review

Definition

A probability density $p(x)$ represents a **mixture distribution** or **mixture model**, if we can write it as a **convex combination** of probability densities. That is,

$$p(x) = \sum_{i=1}^{k} w_i p_i(x),$$

where $w_i \geqslant 0$, $\sum_{i=1}^{k} w_i = 1$, and each p_i is a probability density.

- In our Gaussian mixture model, X has a **mixture distribution**.
- More constructively, let S be a set of probability distributions:
  1. Choose a distribution randomly from S.
  2. Sample X from the chosen distribution.
- Then X has a mixture distribution.

一、Review

$$\text{Cluster probabilities:} \quad \pi = (\pi_1, \ldots, \pi_k)$$
$$\text{Cluster means:} \quad \mu = (\mu_1, \ldots, \mu_k)$$
$$\text{Cluster covariance matrices:} \quad \Sigma = (\Sigma_1, \ldots \Sigma_k)$$

- The model likelihood for $\mathcal{D} = \{x_1, \ldots, x_n\}$ is

$$L(\pi, \mu, \Sigma) = \prod_{i=1}^{n} p(x_i) = \prod_{i=1}^{n} \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z).$$

Since we only observe X, we need the marginal distribution

$$\boxed{\begin{aligned} p(x) &= \sum_{z=1}^{k} p(x, z) \\ &= \sum_{z=1}^{k} \pi_z \mathcal{N}(x \mid \mu_z, \Sigma_z) \end{aligned}}$$

- As usual, we'll take our objective function to be the log of this:

$$J(\pi, \mu, \Sigma) = \sum_{i=1}^{n} \log \left\{ \sum_{z=1}^{k} \pi_z \mathcal{N}(x_i \mid \mu_z, \Sigma_z) \right\}$$

$$\mathcal{N}(\mathbf{x}|\mu,\mathbf{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\mathbf{\Sigma}|^{1/2}}\exp(-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mu))$$

一、Review

- Let's start by considering the MLE for the Gaussian model.
- For data $\mathcal{D} = \{x_1, \ldots, x_n\}$, the log likelihood is given by

$$\sum_{i=1}^{n} \log \mathcal{N}(x_i \mid \mu, \Sigma) = -\frac{nd}{2}\log(2\pi) - \frac{n}{2}\log|\Sigma| - \frac{1}{2}\sum_{i=1}^{n}(x_i-\mu)'\Sigma^{-1}(x_i-\mu).$$

- With some calculus, we find that the MLE parameters are

$$\mu_{\text{MLE}} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\Sigma_{\text{MLE}} = \frac{1}{n}\sum_{i=1}^{n}(x_i-\mu_{\text{MLE}})(x_i-\mu_{\text{MLE}})^T$$

- For GMM, If we knew the cluster assignment z_i for each x_i,
  - we could compute the MLEs for each cluster.

一、Review

- Denote the probability that observed value x_i comes from cluster j by

$$\gamma_i^j = \mathbb{P}(Z = j \mid X = x_i).$$

- The **responsibility** that cluster j takes for observation x_i.
- Computationally,

$$\begin{aligned}\gamma_i^j &= \mathbb{P}(Z = j \mid X = x_i).\\ &= p(Z = j, X = x_i)/p(x)\\ &= \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{c=1}^{k} \pi_c \mathcal{N}(x_i \mid \mu_c, \Sigma_c)}\end{aligned}$$

- The vector $(\gamma_i^1, \ldots, \gamma_i^k)$ is exactly the **soft assignment** for x_i.
- Let $n_c = \sum_{i=1}^{n} \gamma_i^c$ be the number of points "soft assigned" to cluster c.

一、Review

1. Initialize parameters μ, Σ, π.
2. "E step". Evaluate the responsibilities using current parameters:

$$\gamma_i^j = \frac{\pi_j \mathcal{N}(x_i \mid \mu_j, \Sigma_j)}{\sum_{c=1}^{k} \pi_c \mathcal{N}(x_i \mid \mu_c, \Sigma_c)},$$

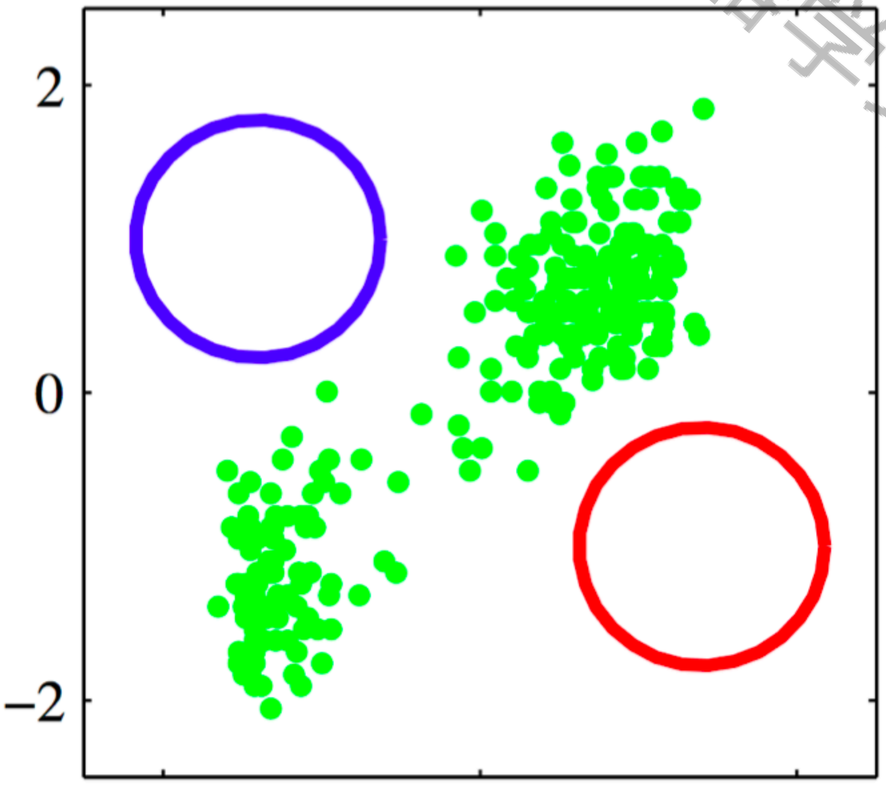
   for $i = 1, \ldots, n$ and $j = 1, \ldots, k$.
3. "M step". Re-estimate the parameters using responsibilities:

$$\begin{aligned}
\mu_c^{\text{new}} &= \frac{1}{n_c} \sum_{i=1}^{n} \gamma_i^c x_i \\
\Sigma_c^{\text{new}} &= \frac{1}{n_c} \sum_{i=1}^{n} \gamma_i^c (x_i - \mu_{\text{MLE}})(x_i - \mu_{\text{MLE}})^T \\
\pi_c^{\text{new}} &= \frac{n_c}{n},
\end{aligned}$$

4. Repeat from Step 2, until log-likelihood converges.

一、Review

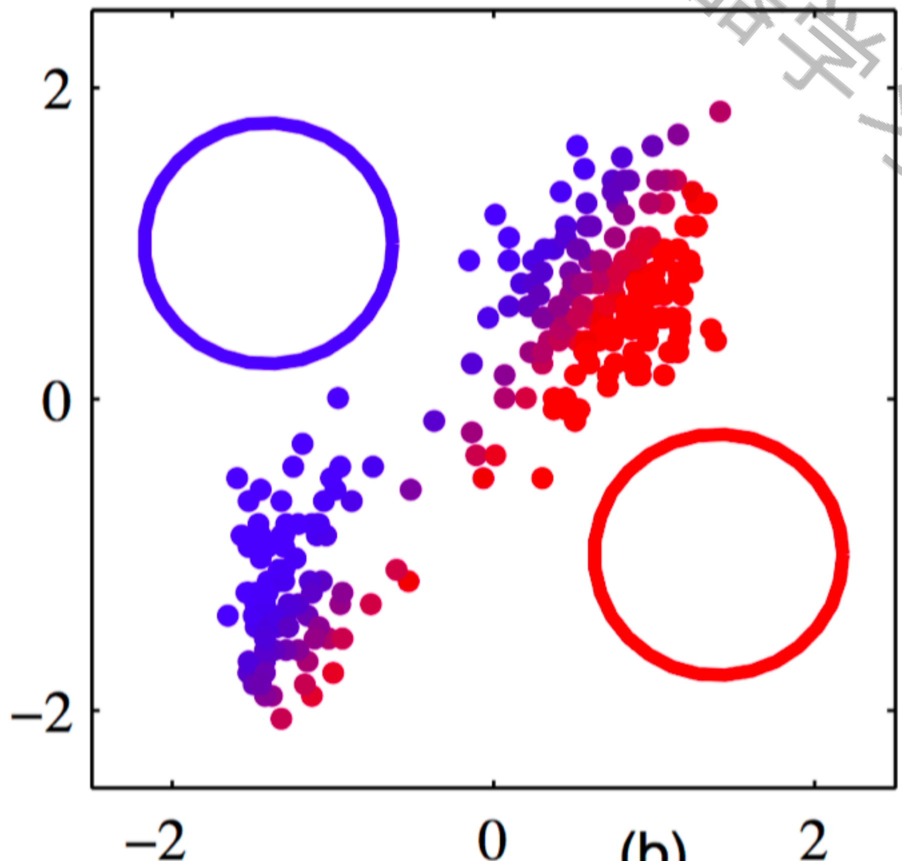
- Initialization



(a)

一、Review

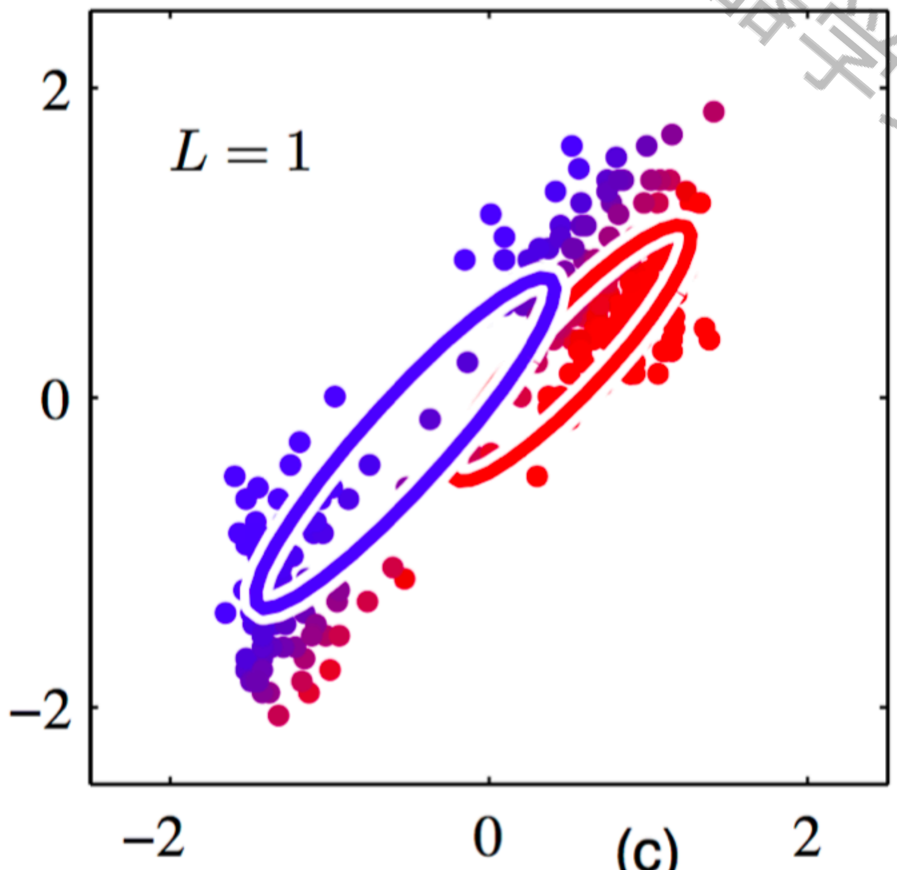
- First soft assignment:



(b)

一、Review

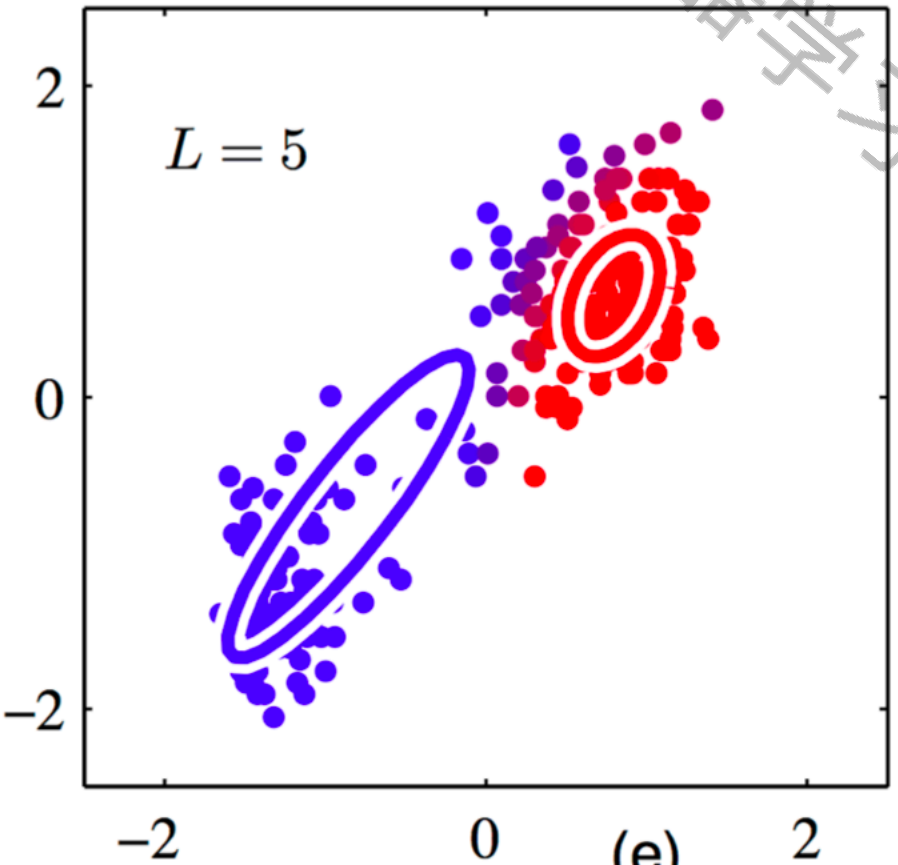
- First soft assignment:



(c)

一、Review

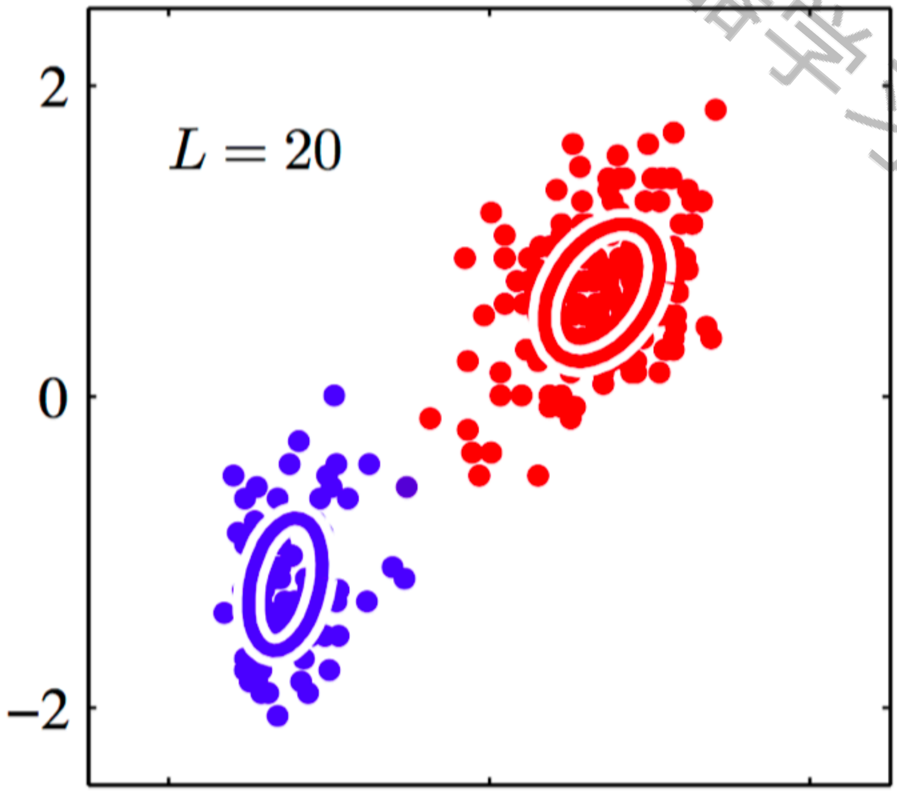
- After 5 rounds of EM:



(e)

一、Review

- After 20 rounds of EM:



(f)

一、Review

k-means vs. MOG

- EM for GMM seems a little like *k*-means.
- In fact, there is a precise correspondence.
- First, fix each cluster covariance matrix to be $\sigma^2 I$.
- As we take $\sigma^2 \to 0$, the update equations converge to doing *k*-means.
- If you do a quick experiment yourself, you'll find
  - Soft assignments converge to hard assignments.
  - Has to do with the tail behavior (exponential decay) of Gaussian.

提纲

一．Review

二．Spectral clustering

- Manifold learning
- Graph cut

三．Summary

二、Spectral Clustering via Manifold Learning

1. Calculate the similarity among data points, e.g., $S_{ij} = exp^{-\frac{d(x_i,x_j)}{\sigma}}$.
2. Form a non-negative affinity matrix $W = |S| + |S^T|$.
3. Construct a Lapacian matrix $L = I - D^{-1/2}WD^{-1/2}$, $D = [d_{ij}]$ is a diagonal matrix with $d_{ij} = \sum_j w_{ij}$.
4. Construct a matrix $C \in R^{n \times k}$ which consists of the k eigenvectors of the L, corresponding to its k smallest eigenvalues.
5. Infer the segmentations of the data by conducting k-means algorithm onto C.

二、Spectral Clustering via Manifold Learning

1. Calculate the similarity among data points, e.g., $S_{ij} = exp^{-\frac{d(x_i,x_j)}{\sigma}}$.
2. Form a non-negative affinity matrix $W = |S| + |S^T|$.
3. ? Construct a Lapacian matrix $L = I - D^{-1/2}WD^{-1/2}$, $D = [d_{ij}]$ is a diagonal matrix with $d_{ij} = \sum_j w_{ij}$.
4. Construct a matrix $C \in R^{n \times k}$ which consists of the k eigenvectors of the L, corresponding to its k smallest eigenvalues.
5. Infer the segmentations of the data by conducting k-means algorithm onto C.

二、Spectral Clustering via Manifold Learning

Spectral Clustering = Dimension Reduction (LE) + k-means

Non-linear dimension reduction via Manifold learning: Embed the similarity graph achieved from the original space into a low-dimensional one.

1. Sam T. Roweis and Lawrence K. Saul, Nonlinear Dimensionality Reduction by Locally Linear Embedding, Science, 2000. Google citation: 5535;

2. Mikhail Belkin. Partha Niyogi, Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering, NIPS2001. Google citation: 1322;
3. Mikhail Belkin. Partha Niyogi, Laplacian Eigenmaps for Dimensionality Reduction and Data Representation, Neural Computation, 2003. Google citation: 2484;

二、Spectral Clustering via Manifold Learning

Intrinsic measurement:

- LLE: learning a similarity graph with representation coefficients;
- LE: Euclidean distance with Heat Kernel derives from differential geometry.

Embedding function

- LLE: reconstruction error;
- LE: Laplace-Beltrami operator.

二、Spectral Clustering via Manifold Learning

Intrinsic measurement:

- LLE: learning a similarity graph with representation coefficients;
- LE: Euclidean distance with Heat Kernel derives from differential geometry.

Embedding function

- LLE: reconstruction error;
- LE: Laplace-Beltrami operator.

二、Spectral Clustering via Manifold Learning

- Similarity graph:

$$w_{ij} = \begin{cases} exp^{-\frac{d_{ij}^2}{\sigma}}, p_{ij} < \epsilon \\ x, \qquad p_{ij} \geq \epsilon \end{cases}$$

$$p_{ij} = \left\| x_i - x_j \right\|_2$$

- Embedding function:

$$\sum_{i,j} \left(y_i - y_j \right)^2 w_{ij}$$

$$\text{s.t.}\, YDY^T = I$$

二、Spectral Clustering via Manifold Learning

- Similarity graph:

$$w_{ij} = \begin{cases} exp^{-\frac{d_{ij}^2}{\sigma}}, p_{ij} < \epsilon \\ x, \qquad p_{ij} \geq \epsilon \end{cases}$$

$$p_{ij} = \left\|x_i - x_j\right\|_2$$

- Embedding function:

$$\sum_{i,j}(y_i - y_j)^2 w_{ij}$$
$$\text{s.t. } YDY^T = I$$

$$\sum_{i,j}(y_i - y_j)^2 w_{ij}$$

$$= \sum_{i,j}(y_i^2 - 2y_iy_j + y_j^2)w_{ij}$$

$$= \sum_i y_i^2 d_i - 2\sum_{i,j} y_iy_jw_{ij} + \sum_j y_j^2 d_j$$

$$= 2Y(D - W)Y^T$$

$$d_i = \sum_j w_{ij}$$

$$D = diag(d_i)$$

Then,

$$\min trace(Y(D - W)Y^T) + \lambda trace(I - YDY^T)$$

It gives

$$(D - W)Y^T = \lambda DY^T$$

二、Spectral Clustering via Manifold Learning

$$(D-W)Y^T=\lambda DY^T$$

Clearly, the desired low-dimensional representation $Y \in R^{d\times n}$ is the d smallest eigenvectors of $D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}}$ by $D^{-\frac{1}{2}}$.

Spectral clustering performs Laplacian Eigenmap to get k-dimensional representation, and then conducts k-means to get the results.

二、Spectral Clustering via Manifold Learning

$$(D - W)Y^T = \lambda D Y^T$$

Clearly, the desired low-dimensional representation $Y \in R^{d \times n}$ is the d smallest eigenvectors of $D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$.

Spectral clustering performs Laplacian Eigenmap to get k-dimensional representation, and then conducts k-means to get the results.

Spectral clustering

1. Calculate the similarity among data points, e.g., $S_{ij} = exp^{-\frac{d(x_i, x_j)}{\sigma}}$.
2. Form a non-negative affinity matrix $W = (|S| + |S^T|)/2$.
3. Construct a Lapacian matrix $L = I - D^{-1/2} W D^{-1/2}$, $D = [d_{ij}]$ is a diagonal matrix with $d_{ij} = \sum_j w_{ij}$.
4. Construct a matrix $C \in R^{n \times k}$ which consists of the k eigenvectors of the L, corresponding to its k smallest eigenvalues.
5. Infer the segmentations of the data by conducting k-mean algorithm onto C.

提纲

一．Review

二．Spectral clustering

- Manifold learning
- Graph cut

三．Summary

二、Spectral Clustering via graph cut

Group the data into multiple clusters by maximizing inter-cluster dissimilarity and intra-cluster similarity, in mathematics,

$$\max \frac{\sum_{x_i \in c, x_j \in \bar{c}} d(x_i, x_j)}{\sum_{x_i \in c, x_j \in c} d(x_i, x_j)}$$

- Inter-cluster dissimilarity:

$$\sum_{x_i \in c, x_j \in \bar{c}} d(x_i, x_j)$$

- Intra-cluster dissimilarity:

$$\sum_{x_i \in c, x_j \in c} d(x_i, x_j)$$

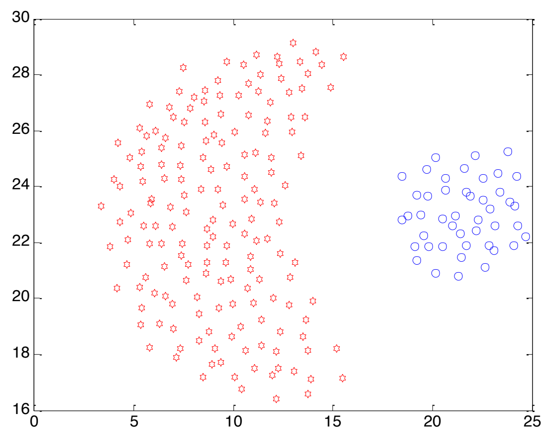
二、Spectral Clustering via graph cut

Normalized cuts and **image segmentation**

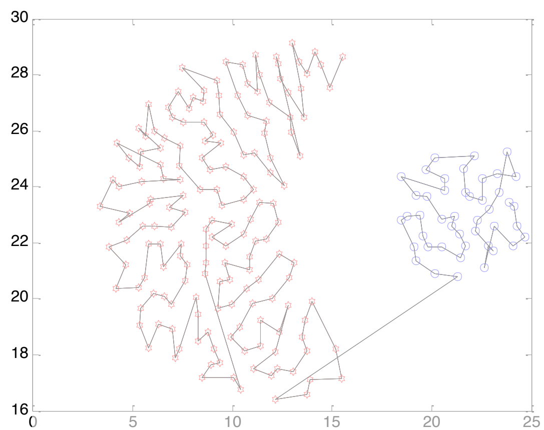
J Shi, J Malik - IEEE Transactions on pattern analysis and …, 2000 - ieeexplore.ieee.org

We propose a novel approach for solving the perceptual grouping problem in vision. Rather than focusing on local features and their consistencies in the image data, our approach aims at extracting the global impression of an image. We treat image segmentation as a graph …

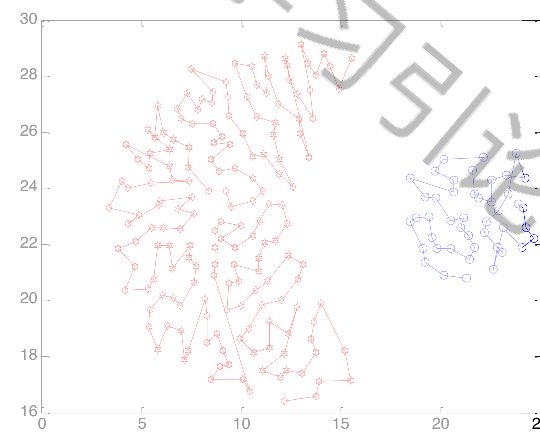
★ 99 Cited by 14675 Related articles All 80 versions »



The original data

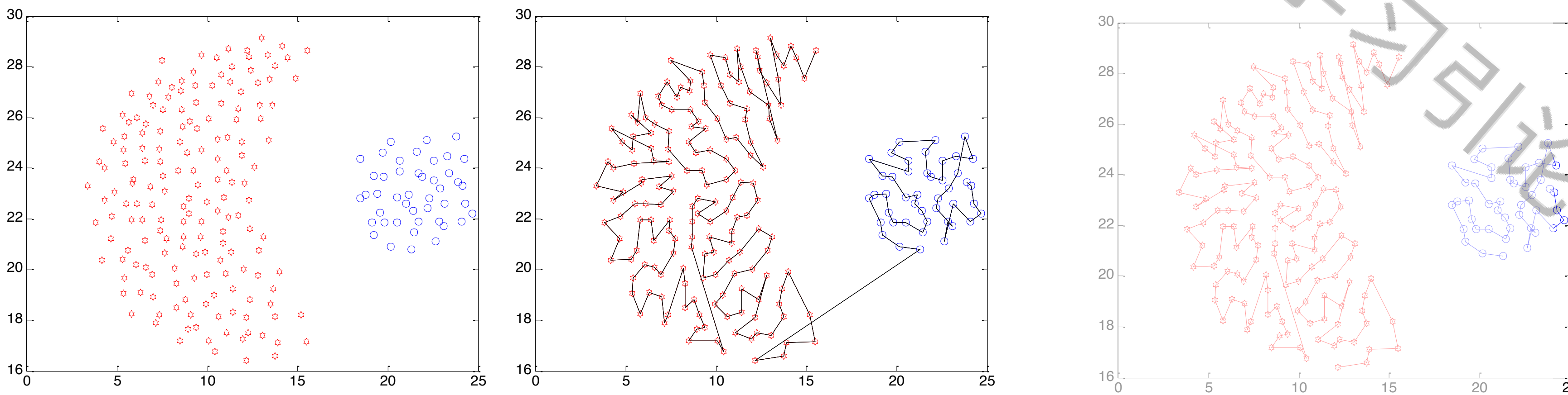


The similarity graph



After removing the edges connected two different components

二、Spectral Clustering via graph cut



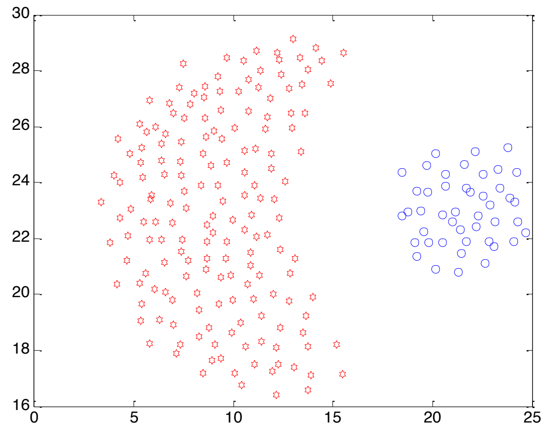
The original data

The similarity graph

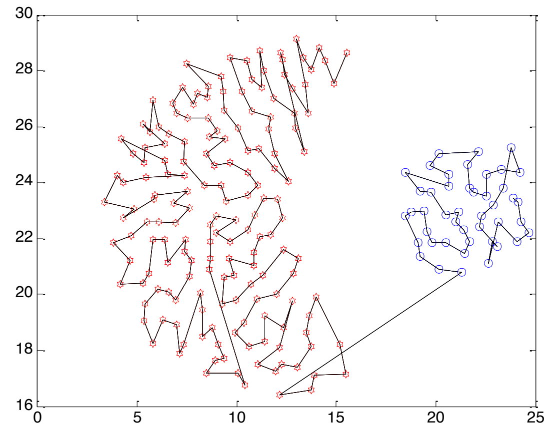
After removing the edges connected two different components

二、Spectral Clustering via graph cut

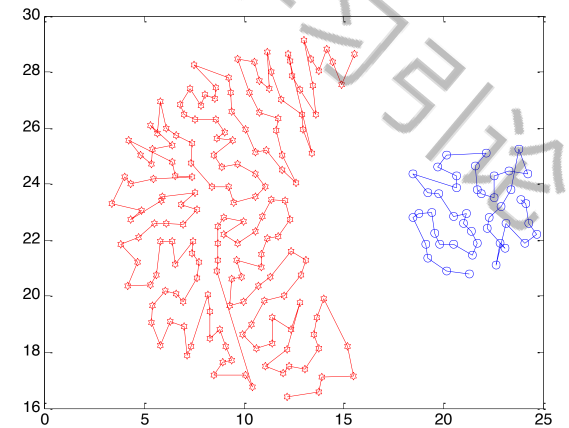
Graph Partitioning = Clustering



The original data



The similarity graph

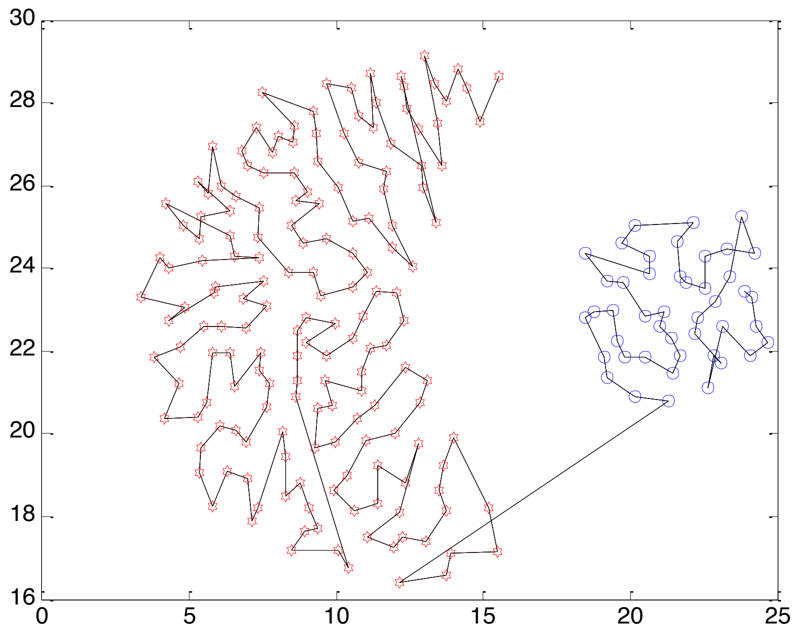


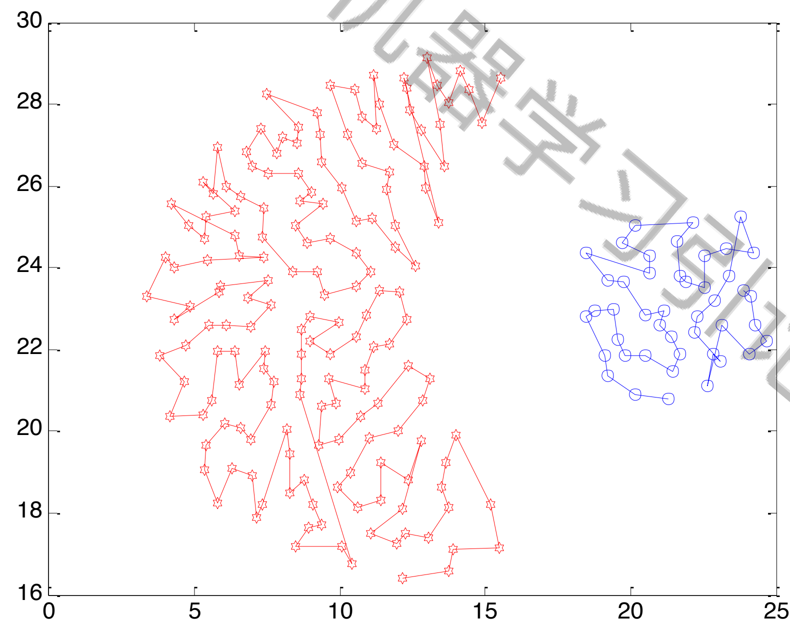
After removing the edges connected two different components

$$\min cut(A, B)$$

Where $cut(A, B)$ is the sum of similarity between clusters A and B, i.e., it is the Inter-cluster similarity.

二、Spectral Clustering via graph cut





$$cut(V,V) = \mathrm{cut}(A,A) + cut(A,B) + cut(B,B)$$

$$\mathrm{cut}(A,A) + cut(B,B)$$

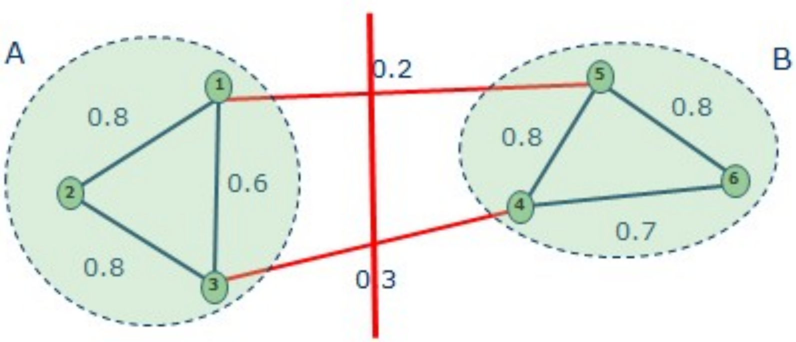
二、Spectral Clustering via graph cut

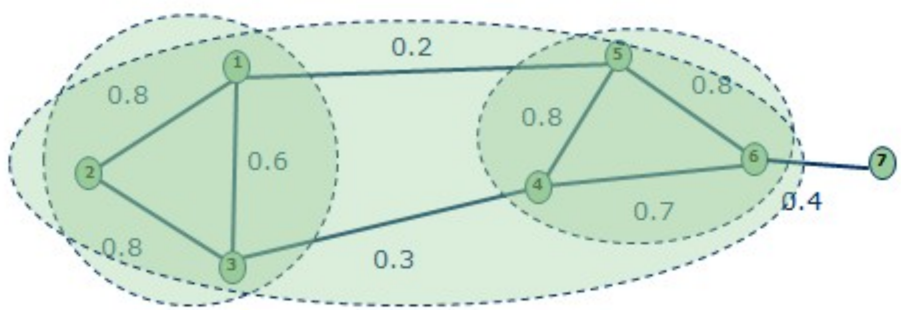
nCut aims to solve:

$$\min \frac{cut(A,B)}{cut(A,V)} + \frac{cut(B,A)}{cut(B,V)}$$

Which could minimize between-class similarity without maximizing intra-class similarity

Minimum cut: $cut(A,B)$





二、Spectral Clustering via graph cut

Minimizing the inter-class similarity = maximizing the intra-class similarity.

$$\mathrm{E} = \frac{cut(A,B)}{cut(A,V)} + \frac{cut(B,A)}{cut(B,V)}$$

$$= \frac{cut(A,V) - cut(A,A)}{cut(A,V)} + \frac{cut(B,V) - cut(B,B)}{cut(B,V)}$$

$$= 2 - \left(\frac{cut(A,A)}{cut(A,V)} + \frac{cut(B,B)}{cut(B,V)}\right)$$

二、Spectral Clustering via graph cut

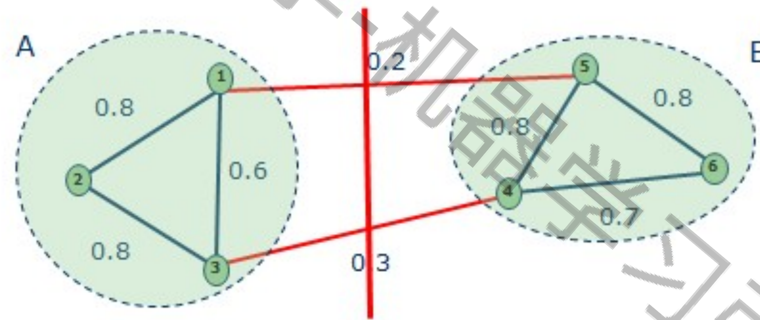
$$E = \frac{cut(A,B)}{cut(A,V)} + \frac{cut(B,A)}{cut(B,V)}$$

$$cut(A,B) = \sum_{x_i=1,x_j=-1} -w_{ij}x_ix_j$$

$$cut(B,A) = \sum_{x_i=-1,x_j=1} -w_{ij}x_ix_j$$

$$cut(A,V) = \sum_{x_i=1} d_i$$

$$cut(B,V) = \sum_{x_i=-1} d_i$$



$cut(A,B)$=0.2+0.3
$cut(B,A)$=0.2+0.3
$cut(A,V)$=4.9
$cut(B,V)$=5.1

After some algebraic manipulations,
Then,

$$E = \frac{y^T(D-W)y}{y^TDy}$$

二、Spectral Clustering via graph cut

$$y^* = argmin_y \frac{y^T(D-W)y}{y^T D y}$$

Where W is a symmetric matrix whose entry is the similarity between two points, $D = diag(d_i)$, $d_i = \sum_{j=1}^{n} w_{ij}$, and $y \in R^n$ is a column vector indicts the labels of data points.

$$y_i = \begin{cases} 1, x_i \in A \\ -1, x_i \in B \end{cases}$$

It is a NP-hard problem!

二、Spectral Clustering via graph cut

$$\min \frac{y^T(D-W)y}{y^T D y}$$

Two steps to get a approximate solution of nCut

Solve the above generalized eigen problem;

Binarize the achieved solution.

Step 1:

Using Lagrange multiplier method, then

$$\min y^T(D-W)y - \lambda y^T D y$$

i.e.,

$$(D-W)y = \lambda D y$$

Clearly, the 2nd smallest eigenvector of $L = D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}}$ is the solution of the above problem, since the smallest eigenvector is **1** having no discrimination.

Step 2:

Binarize y using k-means or bipartition

提纲

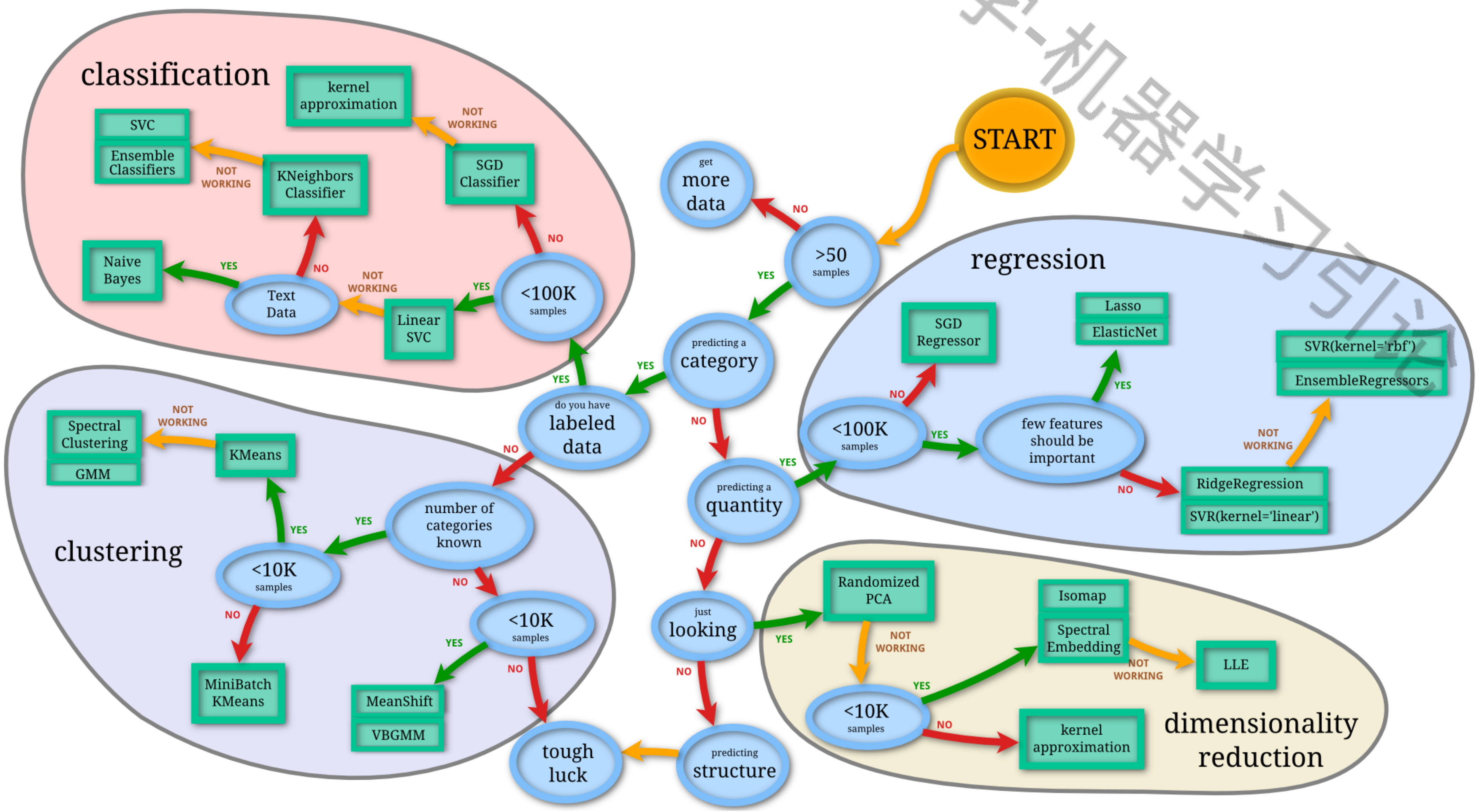
一．Review

二．Spectral clustering

- Manifold learning
- Graph cut

三．Summary

三、Summary



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

三、Summary

- S1-S2：人工智能及机器学习相关概念及发展史
- S3：数学基础
- S4：最近邻居分类器、分类算法的性能度量、模型选择、数据预处理
- S5：神经元、感知机、最大边界分类器
- S6：核、非线性SVM
- S7：降维的基本概念及PCA
- S8-S9：典型相关分析及线性判别分析
- S10-S11：局部线性嵌入及拉普拉斯特征映射
- S12：子空间学习
- S13：聚类的基本概念、聚类性能指标、层次聚类、基于密度的聚类
- S14：基于划分方法的聚类、基于概率分布的聚类
- S15：谱聚类

Final Test (40%)

For a given data set (mnist test partition), achieving

1. a classification accuracy over 80% using the methods introduced in this course. Report the corresponding F-measure.
2. alternatively, a clustering accuracy over 58% using the methods introduced in this course. Report the corresponding NMI.

Requirements:

- Give the design details and explain why it as does
- Report the mean and std score
- Report the tuned parameters
- Report the hardware and used time cost

Q&A

非常感谢大家!