05_Clustering_and_Probabilistic_Model

1. Bayesian Model

Introduction of Bayesian Trend

As we know, in statistics inference, there are mainly three types of information that can be used:

- **overall information:** overall information is $p(x)$
- **sampling information:** information about sampling
- **prior information:** information about the problem

If a method only use the first two information, then it is a **frequentist**}method. If a method use all the three information, then it is a **Bayesian** method.

From **Frequentist** perspective, they regard the unknown parameter as a **"point" constant**. They always **use Maximum Likelihood Estimation (MLE)** as criterion.

From **Bayesian** perspective, they regard the unknown parameter as a **random variable**. They try to optimize the knowledge about the distribution of the random variable. They first have a prior knowledge and then sample and update the distribution based on the samples. It is always a interval estimation, by using **Maximum A Posterior Estimation (MAP)**, we can get a point estimation with Bayesian Method.

Bayesian Inference Framework

Idea

Bayesian Inference regard the unknown parameter as a **random variable**. It is a kind of **interval estimation**. It tries to get the distribution of unknown parameter θ from:

$$\Pr(\lambda) => \Pr(\lambda \mid \text{ observed data })$$

Process

- Get interval estimation from Bayesian Inference
- After get the distribution of the unknown parameter, we can generate a point estimation based on the distribution

Bayesian Polynomial Regression

Here, we will use Bayesian Linear Regression to illustrate the Bayesian Polynomial Regression.

Assume we have **prior knowledge**:

$$\begin{aligned} p(\mathbf{y} \mid \mathbf{w}) &= \mathcal{N}\left(\mathbf{y} \mid \mathbf{X}\mathbf{w}, \sigma^2 \mathbb{I}\right) \\ p(\mathbf{w}) &= \mathcal{N}(\mathbf{w} \mid \mathbf{0}, \alpha \mathbb{I}) \end{aligned}$$

Then after we obtained some data, we can do

$$\begin{aligned} p(\mathbf{w} \mid \mathbf{y}) &= \frac{p(\mathbf{y} \mid \mathbf{w}) p(\mathbf{w})}{p(\mathbf{y})} = \frac{\mathcal{N}\left(\mathbf{y} \mid \mathbf{X}\mathbf{w}, \sigma^2 \mathbb{I}\right) \mathcal{N}(\mathbf{w} \mid \mathbf{0}, \alpha \mathbb{I})}{Z} \\ \log p(\mathbf{w} \mid \mathbf{y}) &= C_1 - \frac{1}{\sigma^2}(\mathbf{y} - \mathbf{X}\mathbf{w})^T(\mathbf{y} - \mathbf{X}\mathbf{w}) + C_2 - \frac{1}{\alpha}\mathbf{w}^\top \mathbf{w} - Z \\ &= \frac{2}{\sigma^2}\mathbf{y}^\top \mathbf{X}\mathbf{w} - \frac{1}{\sigma^2}\mathbf{w}^\top \mathbf{X}^\top \mathbf{X}\mathbf{w} - \frac{1}{\alpha}\mathbf{w}^\top \mathbf{w} + C_1 + C_2 - Z - \frac{1}{\sigma^2}\mathbf{y}^\top \mathbf{y} \end{aligned}$$

Finally, we will get

$$\begin{aligned} p(\mathbf{w}) = \mathcal{N}(\mathbf{w} \mid \mathbf{0}, \alpha \mathbb{I}) \overset{\text{Observe Data}}{\longrightarrow} p(\mathbf{w} \mid \mathbf{y}) &= \mathcal{N}\left(\mathbf{w} \mid \mathbf{m}, \mathbf{S}^{-1}\right) \\ \mathbf{m} &= \frac{1}{\sigma^2}\left(\frac{1}{\sigma^2}\mathbf{X}^\top \mathbf{X} + \frac{1}{\alpha}\mathbb{I}\right)^{-1} \mathbf{X}^\top \mathbf{y} \\ \mathbf{S} &= \frac{1}{\sigma^2}\mathbf{x}^\top \mathbf{X} + \frac{1}{\alpha}\mathbb{I} \end{aligned}$$

Then we can also do **prediction**:

$$\begin{aligned} p\left(y^{\text{new}} \mid \mathbf{y}\right) &= \int p\left(y^{\text{new}} \mid \mathbf{w}\right) p(\mathbf{w} \mid \mathbf{y}) d\mathbf{w} \\ &= \mathcal{N}\left(y_{\text{new}} \mid \mathbf{x}_{\text{new}}^\top \frac{1}{\sigma^2}\left(\frac{1}{\sigma^2}\mathbf{X}^\top \mathbf{X} + \frac{1}{\alpha}\mathbb{I}\right)^{-1} \mathbf{X}^\top \mathbf{y}, \mathbf{x}_{\text{new}}^\top \left(\frac{1}{\sigma^2}\mathbf{X}^\top \mathbf{X} + \frac{1}{\alpha}\mathbb{I}\right)^{-1} \mathbf{x}_{\text{new}} + \sigma^2\right) \end{aligned}$$

Comparison and Gaussian Process

Bayesian Linear Regression is actually a subset of Gaussian Process. It is the gaussian process only use linear kernel.

Bayesian Network

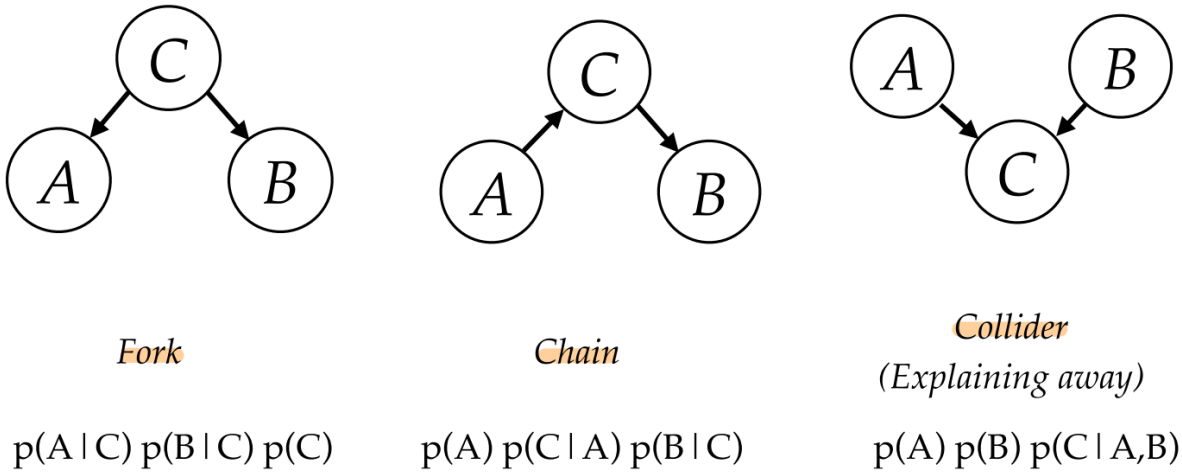
There are two classical **Probabilistic Graphical Model**, one is **Bayesian Network** another is **Markov Chain**. This section we will focus on **Bayesian Network**.

The Bayesian Network is used to modelling the joint distribution.

An important thing is to check c**onditionally independence** in a Bayesian Network. We can directly check it by calculation, but we will discuss a method totally based on the graph structure.

Independence in 3 Variables

There are three possibility for 3 variables: **fork, chain or collider**. They are shown in the Figure.



For the **Fork** Situation, we have the followign property

$$p(A,B,C) = p(A \mid C)p(B \mid C)p(C)$$
$$p(A,B) = \sum_C p(A \mid C)p(B \mid C)p(C) \neq p(A)p(B) \quad (A \not\perp B)$$
$$p(A,B \mid C) = \frac{p(A,B,C)}{p(C)} = p(A \mid C)p(B \mid C) \quad (A \perp B \mid C)$$

We can interpret as

- if we don't observe C, observing A gives information on C, which gives information about B.
- If we do observe C, all the information about C is already present, and observing A adds nothing to our knowledge of B

For the **Chain** Situation, we have the following property, also hold for C’s descendants

$$p(A)p(C \mid A)p(B \mid C) = p(C)p(A \mid C)p(B \mid C)$$
$$p(A, B) = p(A) \sum_C p(B \mid C)p(C \mid A) = p(A)p(B \mid A) \quad (A \not\perp B)$$
$$p(A, B \mid C) = \frac{p(A, B, C)}{p(C)} = p(A \mid C)p(B \mid C) \quad (A \perp B \mid C)$$

We can interpret as

- if we don't observe C, observing A gives information on C, which gives information about B.
- If we do observe C, all the information about B is already present.

For the **Collider** Situation, we have the following property (also hold for C's descendants):

$$p(A, B, C) = p(A)p(B)p(C \mid A, B)$$
$$p(A, B) = p(A)p(B) \sum_C p(C \mid A, B) = p(A)p(B) \quad (A \perp B)$$
$$p(A, B \mid C) = \frac{1}{p(C)} p(A)p(B)p(C \mid A, B) \neq p(A \mid C)p(B \mid C) \quad (A \not\perp B \mid C)$$

We can interpret it as

- If we don't observe C, knowing A does not tell me what information B provided to C.
- If we do observe C, knowing A gives me information on what B must have been to explain the value of C (explaining away)

D-separation

Assume A, B, C are (set of) variables in the graph,

Definition: Blocking

We consider a path between A and B blocked given C if

- a non-collider node on the path is in C
- there is a collider node on the path, and neither it or any of its descendants are in C

Note:

Here, when we talk about the path, we do not consider direction. When we talk about the node type, we consider the direction of the edges. When we talk about descendants, we are talk about the descendants with arrow point to and all further descendants.

Definition: D-Separation

if ever path between A and B is blocked, then $A \perp B \mid C$

Method: A simplified method to judge the d-separation in a graph

After knowing C

- First "block" all output of fork node in C
- Then "block" the collider node that is neither itself and its descendants are not in C

And then we can check whether path between A and B pass these blocked point.

Example

Examples can be seen from https://www.youtube.com/watch?v=8PfJHrydxV0

2. Clustering Tasks

Target and Usage

Grouping Observations based on **similarity / dissimilarity**.

Some typical use of clusterings can be

- Data reduction: selecting typical class examples
- Predicting Characteristics for new data

Intuition

There are two key-points in define clusterings: **shape** and **separation**.

From intuition, when clustering, we should make

- Groups are far apart
- In a group, samples are close together

In order for that, we need to define “far” and “close”

Distance Measure

Distance Measures

- Euclidean Distance Measure:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{l} (x_i - y_i)^2}$$

- City-block:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{l} |x_i - y_i|$$

- l_p-matrix:

$$d_p(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{l} |x_i - y_i|^p \right)^{1/p}$$

Similarity Measures

- Cosine Similarity

$$s_{\cos}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}^T \mathbf{y}}{\|\mathbf{x}\| \|\mathbf{y}\|}$$

- Pearson’s Correlation Coefficient

$$r_{\text{Pearson}}\left(\mathbf{x},\mathbf{y}\right)=\frac{\left(\mathbf{x}-\mu_x\right)^T\left(\mathbf{y}-\mu_y\right)}{\left\|\mathbf{x}-\mu_x\right\|\left\|\mathbf{y}-\mu_y\right\|}$$

Hierarchical Techniques

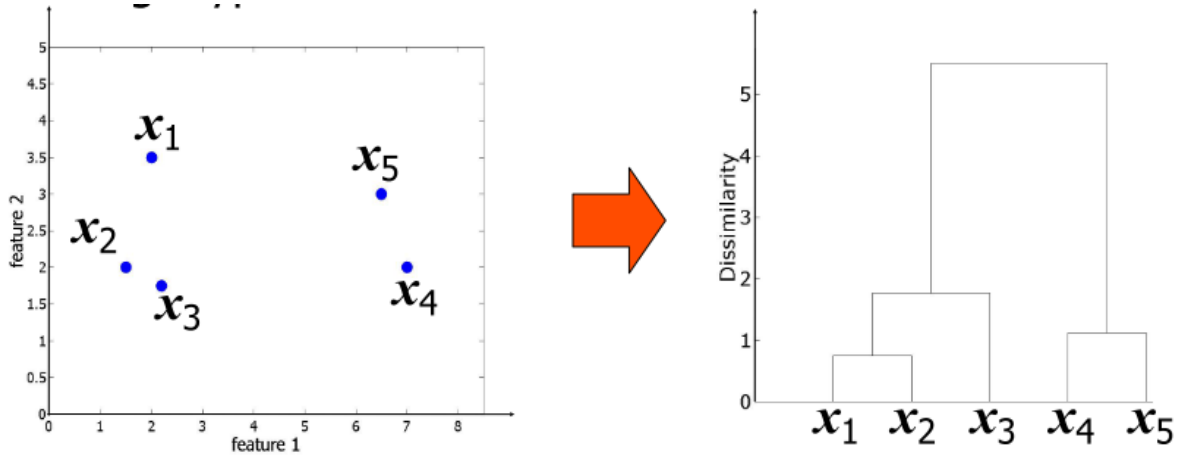
Agglomerative Hierarchical Clustering

The main idea of **Agglomerative Hierarchical Clustering** is starting from individual observations, produce sequence of clusterings of increasing size.

Method

- Determine distances between all clusters
- Merge clusters that are **closet**
- if $\#clusters > 1$, then repeat

Notes:
when using agglomerative hierarchical clusterings, we always output a **dendrogram**.



Different Merging Rules

- **single linkage**: use the distance between two nearest objects in the clusters as distance: optimistic

$$g(R,S)=\min_{i,j} d(x_i,x_j):x_i\in R,x_j\in S$$

- **Complete Linkage:** use the distance between two most remote objects in the clusters as distance: conservative

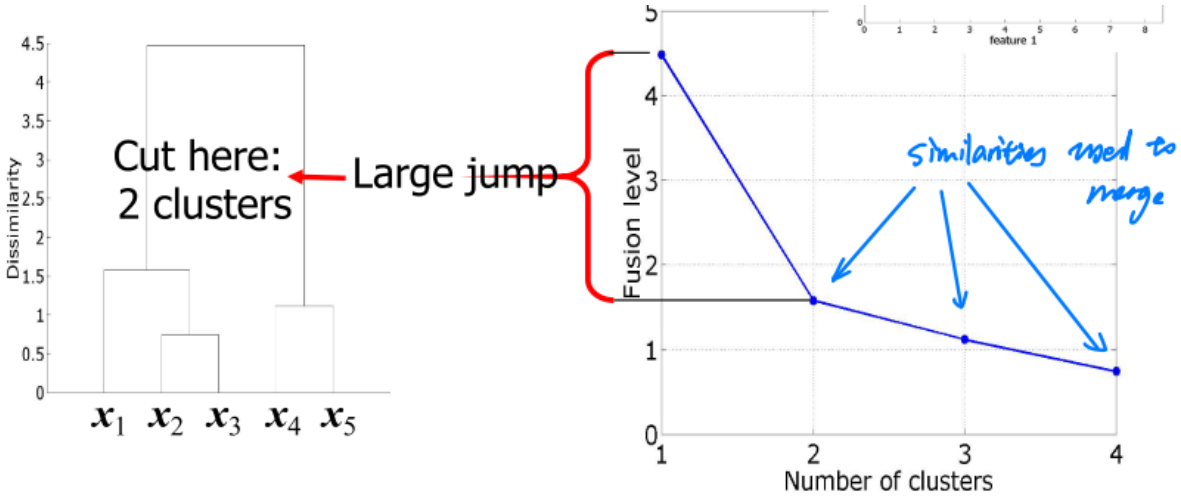
$$g(R,S)=\max_{i,j} d(x_i,x_j):x_i\in R,x_j\in S$$

- **average linkage**: use distance among cluster centers

$$g(R,S)=\frac{1}{|R||S|}\sum_{ij}\left\{d\left(\mathbf{x}_i,\mathbf{x}_j\right):\mathbf{x}_i\in R,\mathbf{x}_j\in S\right\}$$

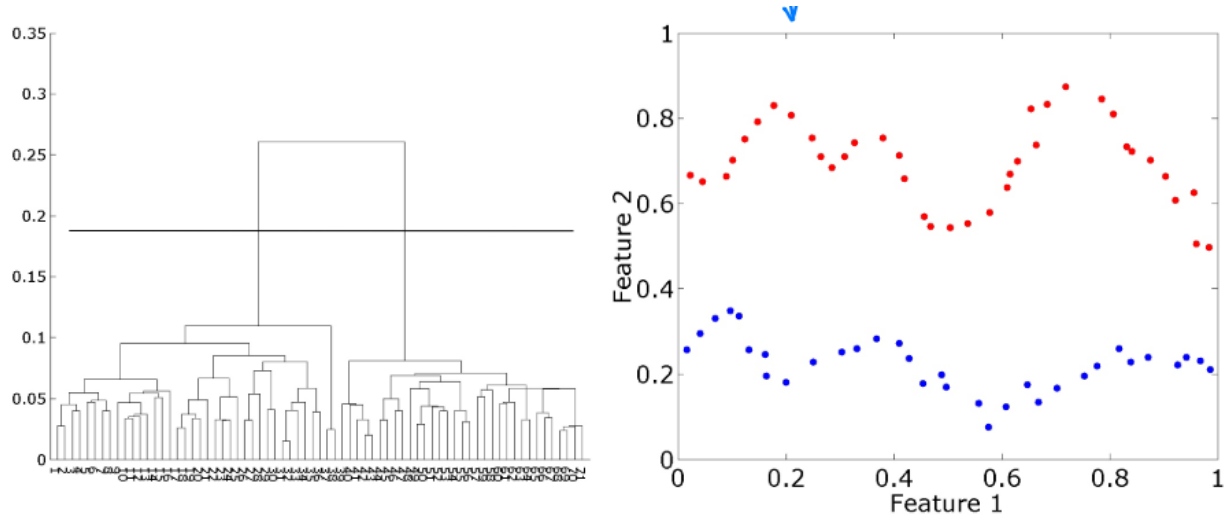
Fusion Graph: How many Clusters to Preserve

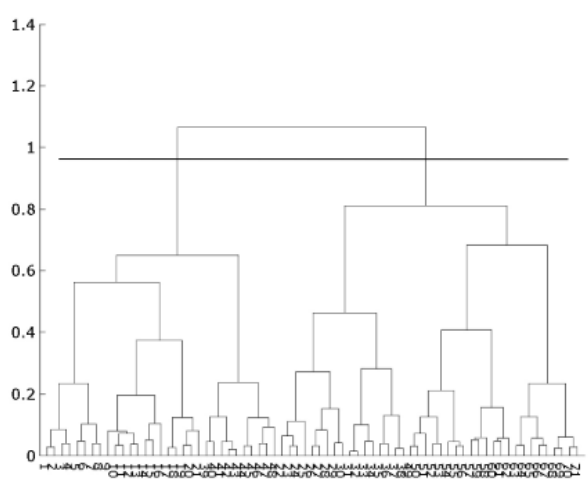
After generate a dendrogram, one of the most important thing is how we choose how many clusters we should preserve? An heuristic approach is **fusion level**

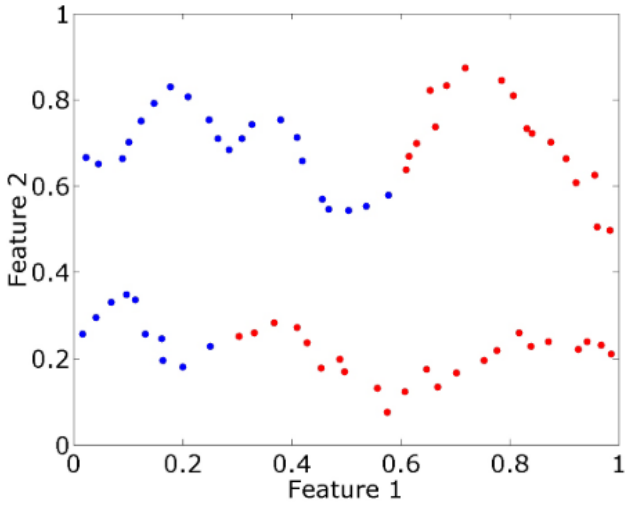


Some Examples

There are two results based on different distance measure of the line separation example:







An explanation of successful separation of single linkage is by use simple linkage, the clusterings are actually propagate to right.

Pros and Cons

Pros:

- Dendrogram gives overview all possible clusterings
- Linkage type allows to find clusters of varying shapes (convex and non-convex, e.g. the line separation)
- different disimilartiy measures can be used

Cons:

- computationally intensive: $O(n^2)$ in time and memory
- Clusterings limited to “hierarchical nestings”

Partitional Techniques

Probabilistic Mixture Model

- Each cluster is described by a **probability density**
- Total dataset is described by a **mixture of densities**
- **Clustering = maximizing the mixture fit**

Model

Probabilistic Mixture Model

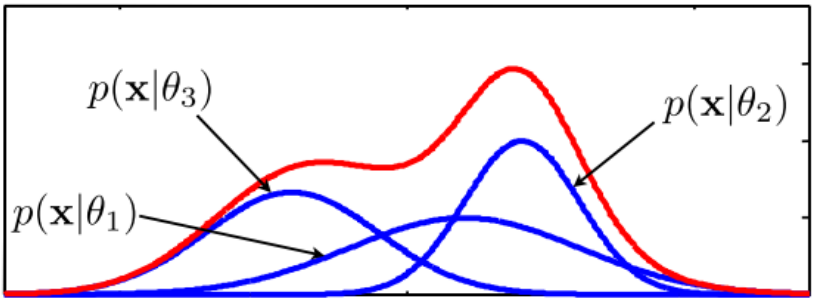
$$p(\mathbf{x} \mid \Theta) = \sum_{j=1}^{m} u_j p\left(\mathbf{x} \mid \theta_j\right)$$

with Mixing Proportions

$$u_j \geq 0, \quad \sum_{j=1}^{m} u_j = 1$$

Mixture of Gaussian Models

$$p(\mathbf{x} \mid \theta_j) = \frac{1}{\sqrt{2\pi^d \det(\Sigma_j)}} \exp\left(-\frac{1}{2}(\mathbf{x}-\mu_j)^T \Sigma_j^{-1}(\mathbf{x}-\mu_j)\right)$$



E-M Methods for Mixture Model

EM is expectation maximization

- E-step computes **cluster membership** $p(C_k|x;\theta)$ of each object based on current model
- M-step updates **maximum likelihood** estimates of parameters based on cluster membership
- Iterate E-M process

Proposition

EM Clustering

- assumes prior known number of clusters
- need to define cluster density
- Guarantees finding of local optimum
- depends on initialization

Generalized E-M Methods

Replace probability model by **arbitrary classifier**

- E-step : assign each observation x by classifier C to one of the classes
- M-step : use the labels to train new classifier C
- Stop when labels do not change

Sum-of-Squares Clustering

Solve an optimizing problem

$$\mathbf{S}_w = \sum_{i=1}^{m} \frac{n_i}{n} \Sigma_i \quad \text{as small as possible}$$

$$\mathbf{S}_B = \sum_{i=1}^{m} \frac{n_i}{n} (\mu_i - \mathbf{m})(\mu_i - \mathbf{m})^T \text{ as large as possible}$$

K-means

K-means Method

the k-means iteration

1. choose number of clusters (m)
2. position prototypes ($m_j, j = 1, \ldots, m$) randomly
3. assign samples to closest prototype (actually keeps the minimize property)
4. compute mean of samples assigned to same prototype and get new prototype position
5. repeat 3 and 4

Lose All Points Problem

Sometimes some cluster **may lose all samples**, i.e. no points is assigned to a cluster

Possible Solution: remove cluster or split largest cluster

Pros and Cons

K-means has pros and cons

Pros:

- very simple
- fast

Cons:

- finds only convex cluster
- Sensitive to initialization
- Can get stuck in local minimum

K means and EM Algorithm

If all cluster are spherical and the variance of each cluster is infinitely small, the EM algorithm then simplifies to the K-means algorithm

Graph Techniques

To be Continued

Summary

- Bayesian Method
    - Bayesian Inference Framework

    - Bayesian Polynomial Regression
    - Bayesian Network: Judge conditionally Independent
        - Three types of connection
        - D-separation
- Clustering
    - Distance Measure and Similarity Measure
    - Hierarchical Techniques
        - Agglomerative Hierarchical Clustering
            - Different Merging Rules
            - Fusion Graph
    - Partitional Techniques
        - Probabilistic Mixture Model
        - E-M Methods
        - Sum-of-Square Clustering
        - K-means