Lecture 11: Clustering Part III

Statistical Methods for Data Science

Yinan Yu

Department of Computer Science and Engineering

December 10, 2020

Today

1. Clustering models
   - Centroid clustering
   - Distribution clustering
   - Hierarchical clustering
   - Density clustering
2. Cluster validation
3. Summary

Learning outcome

- Be able to explain the EM algorithm for one dimensional GMM
- Be able to explain the difference between K-means and the EM algorithm in terms of parameter estimation

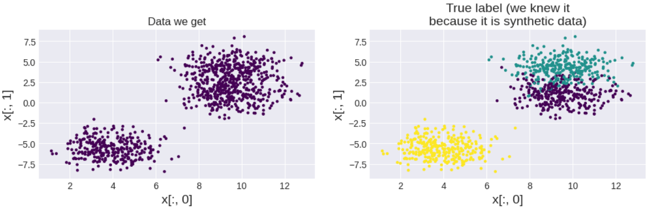
Today

1. Clustering models
   - Centroid clustering
   - Distribution clustering
   - Hierarchical clustering
   - Density clustering
2. Cluster validation
3. Summary

Centroid clustering

Recall: K-means

- **Data** x: d dimensional feature vector $\boldsymbol{x}$



- **Target** y:

$$y = \arg \min_{k \in \{1,\cdots,K\}} dist(\boldsymbol{x}, \boldsymbol{\mu}_k)$$

  where $dist(\cdot,\cdot)$ is a distance measure; in this course, we use the Euclidean distance (cf. lecture 9)
- **Parameters**: K centroids $\boldsymbol{\mu}_k$
- **Hyperparameters**: K
- **Parameter estimation**: an iterative method to update the centroids until convergence
- It is a **hard clustering** technique - one data point is assigned to only one cluster

K-means: pros and cons

- Pros:
  - Convergence guaranteed
  - Easy to implement
  - Scale to large data sets
- Cons:
  - Need to choose the hyperparameter K manually
  - Dependence on random initial values
  - Do not work well on very high dimensional data
  - Not robust to outliers

Distribution clustering

Recall: Gaussian Mixture Models (GMM) - overview

- **Data $\boldsymbol{x}$**: a d dimensional feature vector $\boldsymbol{x} = [x_1, \cdots, x_d]$ with PDF $f(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k f(\boldsymbol{x} \mid k)$
- **Target y**: y is a set of K posterior probabilities; for $k = 1, \cdots, K$

$$\overbrace{P(k \mid \boldsymbol{x})}^{\textit{posterior}} = \frac{\overbrace{P(k)}^{\textit{prior}} \quad \overbrace{f(\boldsymbol{x} \mid k)}^{\substack{\textit{likelihood of } k \\ \textit{given data}}}}{\underbrace{\sum_{c=1}^{K} P(c) f(\boldsymbol{x} \mid c)}_{\substack{\textit{likelihood of the mixture} \\ \textit{distribution given data}}}}$$

It is **soft clustering** - $\boldsymbol{x}$ is assigned to **all clusters** with a probability - the posterior $P(k \mid \boldsymbol{x})$; **alternatively**, y can be defined as the cluster index with the highest posterior probability, i.e.

$$y = \arg \max_{k \in \{1, \cdots, K\}} P(k \mid \boldsymbol{x}) = \arg \max_{k \in \{1, \cdots, K\}} P(k) f(\boldsymbol{x} \mid k)$$

- **Parameter**: the parameters of the mixture distribution $f(\boldsymbol{x})$
  - The parameters for each Gaussian likelihood $f(\boldsymbol{x} \mid k)$
  - The prior $P(k)$, typically denoted as π_k

Recall: two challenges for parameter estimation

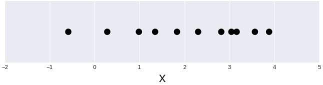
Compared to the Gaussian naive Bayes classifier parameter estimation

- Cluster labels unknown
- High dimensional features not independent - joint probability distribution

Let's address the first issue by looking at one dimensional Gaussian Mixture models

Recall: parameter estimation for 1-d GMM

- **Data**: $x_1, \cdots, x_N$



- **Random variable**: $X_1, \cdots, X_N$ i.i.d. with PDF

$$f(x) = \sum_{k=1}^{K} \pi_k f(x \mid k)$$

The joint probability distribution of all data points is defined as

$$f_{X_1,\cdots,X_N}(x_1, \cdots, x_N) \overset{i.i.d.}{=} \prod_{i=1}^{N} f(x_i) = \prod_{i=1}^{N} \sum_{k=1}^{K} \pi_k f(x_i \mid k) \tag{1}$$

This is the **likelihood** of the **mixture distribution** given data $x_1, \cdots, x_N$.

- **Parameter of interest**: π_k, μ_k, σ_k for all $k = 1, \cdots, K$
- **Parameter estimation method**: maximum likelihood estimation

Recall: parameter estimation for 1-d GMM (cont.)

- The **log likelihood** (cf. Eq. (1) on page 11) is defined as:

$$\begin{aligned} Q(\theta) &= \log L\big(\theta \mid x_1, \cdots, x_N\big) = \log f_{X_1,\cdots,X_N}(x_1, \cdots, x_N) \\ &= \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \pi_k f(x_i \mid k) \right) \end{aligned} \tag{2}$$

  where $\theta = (\mu_1, \cdots, \mu_K, \sigma_1, \cdots, \sigma_K, \pi_1, \cdots, \pi_K)$

- The parameters are estimated by **maximizing the log likelihood**

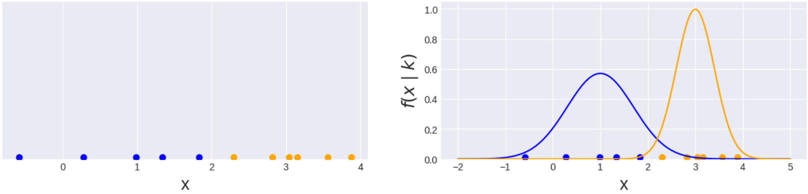
$$\hat{\theta} = \arg\max_{\theta} Q(\theta)$$

- There is no closed-form solution due to the summation inside the log!
- We need to apply an iterative method to find the solution - the **EM algorithm**

Intuition behind EM

Scenario 1: if we knew the label of each data point, the task would be to estimate the parameters

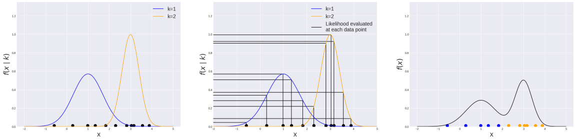
- $\pi_k = P(k) = \frac{N_k}{N}$, where N_k is the count of data points that belong to cluster k, i.e. $\pi_1 = P(\text{blue}) = \frac{5}{11}$ and $\pi_2 = P(\text{orange}) = \frac{6}{11}$
- $\mu_k = \frac{1}{N_k} \sum_{i=1}^{N_k} x_i$
- $\sigma_k^2 = \frac{1}{N_k} \sum_{i=1}^{N_k} (x_i - \mu_k)^2$



Scenario 1 ≈ **maximization** step in the EM algorithm

Intuition behind EM (cont.)

Scenario 2: if we knew the two priors $P(1)$, $P(2)$ and the two Gaussian distributions $f(x \mid 1)$, $f(x \mid 2)$, the task would be to compute the posterior probability $P(k \mid x)$ for $k \in \{\text{orange}, \text{blue}\}$; then we can assign each data point x to a cluster y by the maximum a posteriori estimation $y = \begin{cases} \text{blue}, & P(\text{orange} \mid x) < P(\text{blue} \mid x) \\ \text{orange}, & P(\text{orange} \mid x) \geq P(\text{blue} \mid x) \end{cases}$



- **Prior**: $\pi_1 = P(z_1) = P(blue)$, $\pi_2 = P(z_2) = P(orange)$
- **Likelihood**: $f(x \mid \text{blue}) = \text{blue}$, $f(x \mid \text{orange}) = \text{orange}$
- **Posterior**:

$$P(\text{blue} \mid x) = \frac{P(\text{blue}) f(x \mid \text{blue})}{P(\text{blue}) f(x \mid \text{blue}) + P(\text{orange}) f(x \mid \text{orange})}$$

,

$$P(\text{orange} \mid x) = \frac{P(\text{orange}) f(x \mid \text{orange})}{P(\text{blue}) f(x \mid \text{blue}) + P(\text{orange}) f(x \mid \text{orange})}$$

Scenario 2 $\approx$ **expectation** step in the EM algorithm

Intuition behind EM (cont.)

- In reality, we don't know any of these!
- The idea here is that we alternate scenario 1 and 2 **iteratively** until convergence
- This is essentially how the **Expectation-Maximization (EM) algorithm** works
    - **E-step (expectation)**: estimate the posterior for all data points given each cluster (scenario 2)
    - **M-step (maximization)**: estimate the parameters for each cluster (scenario 1)
- Disclaimer: the purpose of the scenario 1 and 2 is to build up some intuitions only; for details of the EM algorithm, please refer to the description in the next few slides

The EM algorithm: two main steps

Two main steps in the EM algorithm

- **E-step (expectation)**: compute the **posterior probability** of the cluster for each data point x_i

$$\gamma_{ik} = P(k \mid x_i) = \frac{\pi_k f(x_i \mid k)}{\sum_{c=1}^{K} \pi_c f(x_i \mid c)}, \quad \text{for all } k = 1, \cdots, K$$

  This posterior is also called the **responsibility**, denoted as γ_{ik}

- **M-step (maximization)**: estimate the parameters
  - $\pi_k = P(k) = \frac{N_k}{N}$, where $N_k = \sum_{i=1}^{N} \gamma_{ik}$ - **soft clustering**
  - $\mu_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \gamma_{ik} x_i$
  - $\sigma_k^2 = \frac{1}{N_k} \sum_{i=1}^{N_k} \gamma_{ik} (x_i - \mu_k)^2$

  for each cluster $k = 1, \cdots, K$

K-means as a special case of the EM algorithm

Expectation Step

- EM: soft clustering - posterior

$$\gamma_{ik} = P(k \mid x_i)$$

Why soft? - It is a probability, i.e. $\gamma_{ik} \in [0, 1]$

- K-means: hard clustering - equivalent to

$$\gamma_{ik} = \begin{cases} 1, & \text{if the centroid of cluster } k \text{ is the closest to } x_i \\ 0, & \text{otherwise} \end{cases}$$

Why hard? - It is a binary decision, i.e. $\gamma_{ik} \in \{0, 1\}$

Maximization Step

- EM: need to estimate μ_k, σ_k, where

$$\mu_k = \frac{1}{N_k}\sum_{i=1}^{K}\gamma_{ik}x_i, \quad \sigma_k^2 = \frac{1}{N_k}\sum_{i=1}^{N_k}\gamma_{ik}(x_i - \mu_k)^2, \quad N_k = \sum_{i=1}^{N}\gamma_{ik}, \quad \text{for } \gamma_{ik} \in [0, 1]$$

- K-means: only need to estimate μ_k, where

$$\mu_k = \frac{1}{N_k}\sum_{i=1}^{K}\gamma_{ik}x_i = \frac{1}{N_k}\sum_{x \in \text{cluster } k} x, \quad N_k = \sum_{i=1}^{N}\gamma_{ik}, \quad \text{for } \gamma_{ik} \in \{0, 1\}$$

The complete EM algorithm

- These two steps are the core of the EM algorithm
- Besides that, there are some extra steps involved
- The EM algorithm is an **iterative method**
- There are three important components in an iterative method:
    1) Initialization step
    2) Update the parameters in a while loop (the core, i.e. the maximization step and the expectation step)
    3) Stopping criteria

The complete EM algorithm (cont.)

- **1) Initialization step**: initialize π_k, μ_k, σ_k manually or using, e.g. the K-means algorithm μ_k, for all $i = 1, \cdots, N$, $k = 1, \cdots, K$
- **2) Update the parameters in a while loop**: repeat the expectation step and the maximization step until the stopping criteria is met; each repetition of this process is called **one iteration**
- **3) Stopping criteria**: something you check (using e.g. a conditional statement) inside the while loop; if the stopping criteria is true, the loop shall be escaped - then you are done! There are two alternative stopping criteria for the EM algorithm:
  - Has the objective function, i.e. the log likelihood (cf. Eq. (2)), changed since the last iteration?
  - Have any of the parameters, i.e. π_k, μ_k, σ_k, changed since the last iteration?

High dimensional GMM

- The second problem on page 10 is the high dimensional joint probability distribution of the correlated feature vectors
- In this course, we will not go into details of multivariate Gaussian distributions; nevertheless, the EM steps for $d > 1$ is presented as follows

Expectation Step

$$\gamma_{ik} = P(k \mid \boldsymbol{x}_i) = \frac{\pi_k f(\boldsymbol{x}_i \mid k)}{\sum_{c=1}^{K} \pi_c f(\boldsymbol{x}_i \mid c)}, \quad \text{for all } k = 1, \cdots, K$$

Maximization Step

- $\pi_k = P(k) = \frac{N_k}{N}$, where $N_k = \sum_{i=1}^{N} \gamma_{ik}$
- $\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{i=1}^{N_k} \gamma_{ik} \boldsymbol{x}_i$
- $\boldsymbol{\Sigma}_k = \frac{1}{N_k} \sum_{i=1}^{N} \gamma_{ik} (\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T$ - note that $\boldsymbol{x}_i$ is a column vector here

The covariance matrix $\boldsymbol{\Sigma}_k$ captures the dependence between features

Recap: GMM

- Gaussian Mixture Models (GMM) is a mixture distribution characterized by a mixture PDF

$$f(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k f(\boldsymbol{x} \mid k)$$

where each $f(\boldsymbol{x} \mid k)$ is a multivariate Gaussian PDF for each Gaussian component k **multivariate because $x \in \mathbb{R}^d$ is a d dimension feature vector**

  - $f(\boldsymbol{x})$ is the PDF of the mixture model with **parameters** $\theta = \{\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_K, \boldsymbol{\Sigma}_1, \cdots, \boldsymbol{\Sigma}_K\}$
  - $f(\boldsymbol{x})$ is the **likelihood** of θ given data $\boldsymbol{x}$
- Parameter estimation: maximum **likelihood** estimation - given $\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N$

$$\hat{\theta} = \arg\max_{\theta} \sum_{i=1}^{N} \log(f(\boldsymbol{x}_i))$$

  - No closed form solution - iterative algorithm for finding $\hat{\theta}$ - the EM algorithm
- Comparison to other techniques
  - Gaussian naive Bayes classifiers vs GMM
  - K-means vs the EM algorithm for parameter estimation

GMM: pros and cons

- Pros:
  - Relatively simple compared to other mixture models **we love Gaussians!**
  - Flexibility due to the soft clustering criterion
- Cons:
  - Might get stuck on local optimum of the objective function
  - Convergence can be slow
  - Covariance matrix estimate might lead to divergence in case of small data set
  - Need to choose the hyperparameter K manually
  - Gaussian mixture assumption might not be true

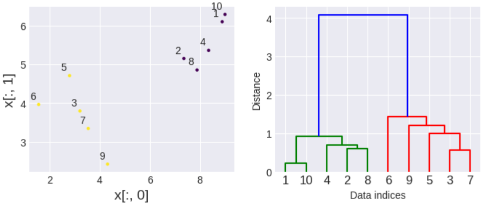
Hierarchical clustering

Agglomerative vs divisive

Two types of hierarchical clustering

- Agglomerative (bottom-up):
  - Start with each data point being its own cluster
  - Merge the closest clusters until there is only one cluster
- Divisive (top-down):
  - Start with one cluster
  - Split until each cluster contains only one data point

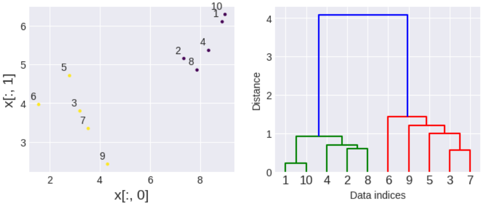
Agglomerative hierarchical clustering using a dendrogram



- Step 1: consider each data point as its own cluster; find the closest clusters - 1 and 10; group 1 and 10 into one cluster; the height of the dendrogram indicates the Euclidean distance - now we have 9 clusters in total
- Step 2: find the closest cluster - 3 and 7 - group 3 and 7 into one cluster
- Step 3: repeat until there is only one cluster left
- Step 4: draw a horizontal line to split data into different clusters

Divisive hierarchical clustering has a similar process (but top-down)

Distance between clusters - alternative linkages



- The height of the dendrogram shows the distance between two clusters
- We need to choose how to compute the distance on the cluster level when there are more than one point in each cluster
- There are different alternatives to defined the distance between two clusters, e.g. the distance between cluster {1, 10} and cluster {4, 2, 8}
  - **Single-linkage**: the distance between the closest pair, i.e. $dist(1, 4)$
  - **Complete-linkage**: the distance between the farthest pair, i.e $dist(10, 8)$
  - **Centroid**: the distance between two centroids, i.e.

$$dist(\text{centroid}(\{1{,}10\}), \text{centroid}(\{4,\ 2,\ 8\}))$$

Here, $dist(\cdot, \cdot)$ is the Euclidean distance

Hierarchical clustering: pros and cons

- Pros:
  - No need to choose K
  - Easy to implement
  - Might give meaningful taxonomies
- Cons:
  - Once two clusters are grouped, the action cannot be undone
  - Does not scale well with large data set
  - No well defined objective function

Density clustering

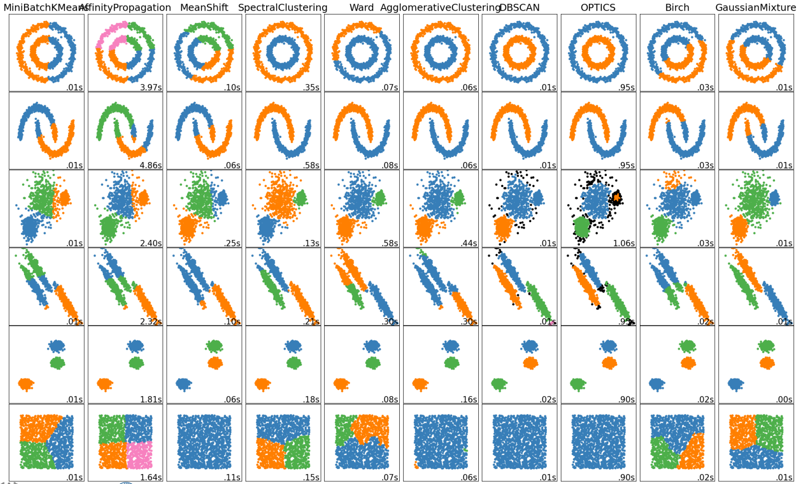
Introduction

- Idea: cluster data based on their closest points, i.e. the neighborhood
- Hyperparameter
  - Radius of the neighborhood ϵ
  - Minimum number of points n in its ϵ neighborhood

Density clustering: pros and cons

- Pros:
  - Handles clusters with arbitrary shapes
  - Handles noise explicitly
- Cons:
  - Sensitive to the sampling technique in the neighborhood
  - Need to choose the hyperparameters ϵ and n
  - Not optimal for clusters with varying density

Comparison (from Python scikit-learn)



Today

1. Clustering models
   - Centroid clustering
   - Distribution clustering
   - Hierarchical clustering
   - Density clustering
2. Cluster validation
3. Summary

Validation criteria

Cluster validation is to evaluate the quality of clusters; there are two types of criteria:

- Internal criteria: unsupervised; cluster labels are unknown
  - Algorithms that assume clusters with spherical shapes
    - Silhouette score
    - SSE
    - Many other indices, e.g. Davies–Bouldin index, Dunn index, etc
  - Algorithms that assume mixture distributions
    - AIC
    - BIC
  - Distance based criteria; more generic
    - Similarity matrix with data ordered by cluster indices
- External criteria: supervised; ground truth labels are given
  - Purity
  - F1-score
  - Entropy and mutual information

Today

1. Clustering models
   - Centroid clustering
   - Distribution clustering
   - Hierarchical clustering
   - Density clustering
2. Cluster validation
3. Summary

Summary

So far:

- Data types and data containers
- Descriptive data analysis: descriptive statistics, visualization
- Probability distributions, events, random variables, PMF, PDF, parameters
- CDF, Q-Q plot, how to compare two distributions (data vs theoretical, data vs data)
- Modeling
- Parameter estimation: maximum likelihood estimation (MLE) and maximum a posteriori estimation (MAP)
- Classification, multinomial naive Bayes classifier, Gaussian naive Bayes classifier
- Central limit theorem, interval estimation
- Hypothesis tests, comparison of two classifiers
- Clustering, cluster tendency, k-means
- SSE and Silhouette score for cluster evaluation, one dimensional Gaussian Mixture Models, AIC/BIC, the EM algorithm, clustering validation

Next:

- Hypothesis testing, review for exam

Before next lecture:

- Slides of lecture 8