Subject: Stanford CS229 Machine Learning, Lecture 12, K-Means, GMM (non EM), Expectation Maximization

Date: from August 26, 2025 to August 28, 2025

Contents

Stanford CS229 Machine Learning, K-Means, GMM(non EM), Expectation Maximization, 2022, Lecture 12

Link on YouTube: Stanford CS229 Machine Learning, Introduction, 2022, Lecture 12

Introduction

Introduction

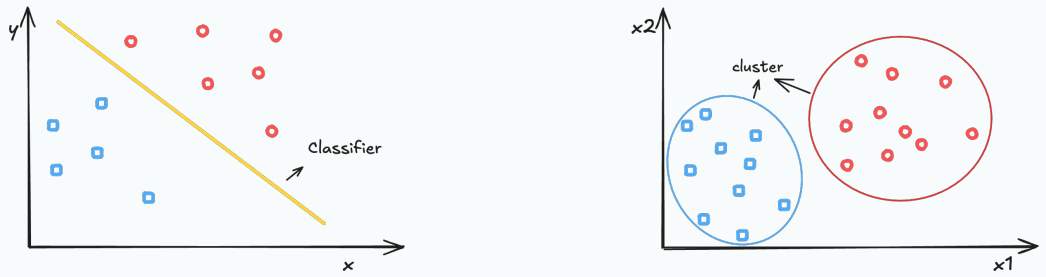
本节内容介绍了无监督学习方法，具体而言介绍了三种聚类算法：

1. K-means: 最经典的迭代型聚类算法，使用 hard assignment 进行数据点分配
2. Gaussian Mixture Model: 使用 soft assignment 进行分配，并通过 *E-M* 算法求解

K-Means

K-Means: Introduction

在前面所有课程中所介绍的机器学习算法均为有监督学习(supervised learning)，其一重要特征是数据集 $\{x_i, y_i\}$ 中存在标签(label) y_i. 而当标签并不存在时就需要使用无监督学习(unsupervised)方法(如图1b所示).



(a) Supervised learning with labels

(b) Unsupervised learning without labels

Figure 1: Comparison between supervised learning and unsupervised learning.

由于无法利用标签信息作为拟合目标，因此并没有所谓的"正确答案"[a]，故较监督学习实现更困难，为此需要更多的假设. 例如一些工作假设数据存在某种潜在结构. 因此对比监督学习而言，无监督学习 1. 需要更强的假设 2. 只能得出较弱的理论保证.

[a] 例如图 1b 中将数据点分为几类并无标准答案

K-Means: Setup

K-means 算法目的是将无标签的数据分为 k 类，将每一类称为一"聚类"(cluster)，例如图 1b 中有两个聚类.

Data:给定无标签数据集 $\{x^{(1)}, \ldots, x^{(n)}\}$，其中 $x^{(i)} \in \mathbb{R}^d$.

Do: 预先设定 k 个聚类，找到一种分配方式 C 将数据归类于相应的聚类中：

$$C^{(i)} = j : \text{point } x^{(i)} \to \text{cluster } j, i = 1, \ldots, n,\ j = 1 \ldots, k \tag{1}$$

K-Means: Algorithm

K-meas 算法是迭代型算法(见算法 1)，其流程如下：

1. 随机选取 k 个聚类中心点(cluster center point) $\mu^{(i)}$，即定义了 k 个 cluster
2. 根据距离的远近将所有的点分配给最近的聚类中心点，即形成了 k 个 cluster Ω_j
3. 对每个聚类中的点的坐标计算均值 $\mu^{(j)} = \frac{1}{|\Omega_j|} \sum_{i \in \Omega_j} x^{(i)}$，作为新的聚类中心点
4. 重复步骤 2-3 直至聚类不再变化

Algorithm 1 K-means Clustering Algorithm

Require: Dataset $X = \{x_1, x_2, \ldots, x_n\}$, number of clusters k

1: Randomly initialize k cluster centers $\mu^{(j)}, j = 1 \ldots, k$

2: **repeat**

3: **Cluster assignment:** Assign each data point to the nearest cluster center:

$$C^{(i)} = \arg\min_j \|\mu^{(j)} - x_i\|^2, j = 1 \ldots, k$$

4: **New center:** Update each cluster center as the mean of the points assigned to it:

$$\mu^{(j)} = \frac{1}{|\Omega_j|} \sum_{i \in \Omega_j} x^{(i)}, \Omega_j = \{i : C^{(i)} = j\}, j = 1 \ldots, k$$

5: **until** cluster assignments and cluster centers are no longer change.

Note 1. 定义算法损失函数为

$$J(C, \mu) = \sum_{j=1}^{k} \sum_{i \in \Omega_j} \|x^{(i)} - \mu^{C^{(i)}}\|^2 \tag{2}$$

则其单调递减，因此 *K-means* 算法一定会收敛. 并且抽象来看 *K-means* 算法是在对 $J(C, \mu)$ 做梯度下降. 需要额外注意的是：

1. 由于聚类问题本身的 *NP-Hard* 性质，*K-means* 并不能保证收敛至 *global minima*
2. 在计算新的聚类中心点时并不将现有的中心点纳入计算中，因为实际上此中心点是通过计算平均值得出的虚拟数据点.
3. *K-means* 容易受初始化*(initialization)* 的影响，目前最主流的 *K-means* 算法是由 *Stanford* 提出的 *K-means ++ [1]*，其已经成为 `scikit-learn` 中 *K-means* 的默认算法.

Note 2. 在 *K-means* 中聚类数 k 是提前给定的，实际上在没有相关信息*(domain knowledge)* 的前提下并没有所谓的最优 k. 此处我们更聚焦于能否给定想要的聚类数后通过算法自动得出一个令人满意的 *cluster assignment* 方案.

Note 3. 无监督学习方法也可以和监督学习一起使用. 例如当标签充满噪声以至于无法信任时，就可以先使用无监督学习对数据内部进行探索以获取更多信息以帮助进行监督学习.

Mixture of Gaussian

Mixture of Gaussian: Example

Motivation: Mixture of Gaussian 算法的一个简单例子来自于天文学中的光子(photon)检测问题：地球上放置探测器(light detector) 用以接受来自太空中的光子. 其源自普通行星(regular star) 和类星体(quasar)两类星体，其中普通行星的脉冲(pulse) 向各个方向均匀发散，而类星体的脉冲则相对集中. 探测器在接受到光子时无法知晓其来源，即无标签.

Given: 探测器上的光子分布位置.

Do: 将每个光子分配给一个来源，但由于无法完全确定，因此采用 **soft assignment** 方式进行分配，即用概率表示每个光子来自某一类星体的可能性，即：

$$P(z^{(i)} = j): \text{ probability that photon } i \text{ comes from source } j$$

其中 $z^{(i)}$ 实际为一随机变量，表示将点 i 分配给 cluster j.

Challenge: 1. 光子的来源可能有很多，但是此时我们假设已知有 k 个来源（与 K-means 假设相同)；2. 不同来源的光子呈现的强度(intensity) 和形状(shape) 不同.

Assumption: 假设每个来源的光子呈 Gaussian 分布 (μ_j, σ_j).

Note 4. 这里并不假设每个来源的光子的数量相同，因为有可能某个来源的光子集中且数量更多，例如占比 *90%*.

Mixture of Gaussian: $\mathbb{R}^1$ for Simplicity

以一维为例，如图2a 所示，若已知类别，只需要进行简单的 Gaussian fit 即可（例如 GDA），但目前的困难在于如图2b 所示，只能获取数据的分布但类别未知.

(a) Known clusters, then just fit Gaussian

(b) Unknown clusters

Figure 2: $\mathbb{R}^1$ for simplicity.

Given: 给定数据$x^{(1)}, \ldots, x^{(n)} \in \mathbb{R}$, 和类别数 (cluster number) $k \in \mathbb{R}^*$.

Do: 找到 soft assignment，即概率分布 $P(z^{(i)} = j), i = 1, \ldots, n,\ j = 1 \ldots, k$, 其中 $z^{(i)}$ 实际为一随机变量，表示将点 i 分配给 cluster j，并将其称为 latent.

Note 5. 实际上目前我们的任务可以认为是未知分布反推数据的生成过程，如果反过来思考，正向的数据过程应该是：每一个类别的数据分别满足 *Gaussian*：

$$x^{(i)}|(z^{(i)} = j) \sim \mathcal{N}(\mu_j, \sigma_j^2) \tag{3}$$

每个类别中数据量满足一定分布*(例如某一类数据点占 90%)*

$$z^{(i)} \sim \textit{multinational}(\Phi), \sum_{j=1}^{k} \phi_j = 1, \phi_j \geqslant 0, \tag{4}$$

在得到了每一类数据点如何分布、每一类数据点占比后根据 *Bayes* 进行数据点生成：

$$P(x^{(i)}, z^{(i)}) = P(x^{(i)}|z^{(i)}) \cdot P(z^{(i)}) \tag{5}$$

Mixture of Gaussian: Algorithm

Gaussian Mixture Model (GMM) 算法是著名的 *E-M* 算法(Expectation–Maximization algorithm) 的实例应用，其分为两步：

1. (E-step) 给定数据和当前 latent values (Φ, μ, σ)，预测 $z^{(i)}$，即

$$\begin{aligned} w_j^{(i)} &= P(z^{(i)} = j | x^{(i)}; \Phi, \mu, \sigma) = \frac{P(z^{(i)} = j, x^{(i)}; \Phi, \mu, \sigma)}{P(x^{(i)}; \Phi, \mu, \sigma)} \\ &= \frac{P(x^{(i)} | z^{(i)} = j; \Phi, \mu, \sigma) P(z^{(i)} = j)}{\sum_{l=1}^{k} P(x^{(i)} | z^{(i)} = l; \Phi, \mu, \sigma) P(z^{(i)} = l)} = \frac{\mathcal{N}(\mu_j, \sigma_j^2) \cdot \phi_j}{\sum_{l=1}^{k} \mathcal{N}(\mu_l, \sigma_l^2) \cdot \phi_l} \end{aligned} \tag{6}$$

2. (M-step) 给定当前 $P(z^{(i)=j})$ 估计值 w_j^i，使用 *MLE* 估计 observed parameters:

$$\phi_j = \frac{1}{n} \sum_{i=1}^{n} \approx \text{ fraction of points from cluster } j \tag{7}$$

Note 6. 这里 *observed parameters* 指目前可以观测到的，例如目前分配下每个聚类中的点数量，相较 *latent* $z^{(i)}$ 而言，其是可以观测到的.

Last update: December 15, 2025

References

[1] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.