机器学习讲义(L19-B) K-Means 算法

授课教师：王贝伦 / 助教：张嘉琦，黄旭，谈笑，徐浩卿

1 K-Means 算法

在前面的讲义内容中，我们已经提到 K-Means 算法属于聚类算法中的分割演算法（Partitioning Algorithms），我们提到其通常以随机的一次划分开始，但随机的划分不一定是合理的，我们需要在后续的迭代过程中使之逐渐合理化。

基于这种思想而提出的 K-Means 算法易于理解，算法简单。算法将样本集合划分为 K 个子集，构成 K 个类，将 n 个样本分到 K 个类中，每个样本到其所属类的中心的距离最小。每个样本只能属于一个类，所以 K-Means 聚类是硬聚类。

K-Means 聚类算法的思想很简单，就是通过损失函数的最小化来选取最优的划分。在 K-Means 聚类算法中我们采用欧式距离平方作为样本之间的距离衡量尺度。我们假设划分的类集合为 $\boldsymbol{C} = \{C_1, C_2, ..., C_K\}$，每个类的中心点表示为 $\mu_k, k = 1, 2, ..., K$ 损失函数的定义为样本与其所属类的中心之间的距离的总和，即

$$L(C) = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2 \tag{1}$$

K-Means 聚类算法就是求解最优化问题：

$$C^* = \underset{C}{\operatorname{argmin}}\, L(C) = \underset{C}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - \mu_k\|^2 \tag{2}$$

1.1 K-Means 聚类算法过程详解

K-Means 聚类算法是一个迭代的过程，每次迭代主要包括两个步骤。首先选择 K 个类的中心，将样本逐个指派到与其最近的中心的类中，得到一个聚类结果；然后更新每个类的样本的均值，作为类的新的中心；重复以上步骤，直到收敛为止。下面我们以一个例子详细解释一下这个过程。

步骤一：随机选取 K 个样本点作为初始聚类中心，我们这里取 $K = 3$；

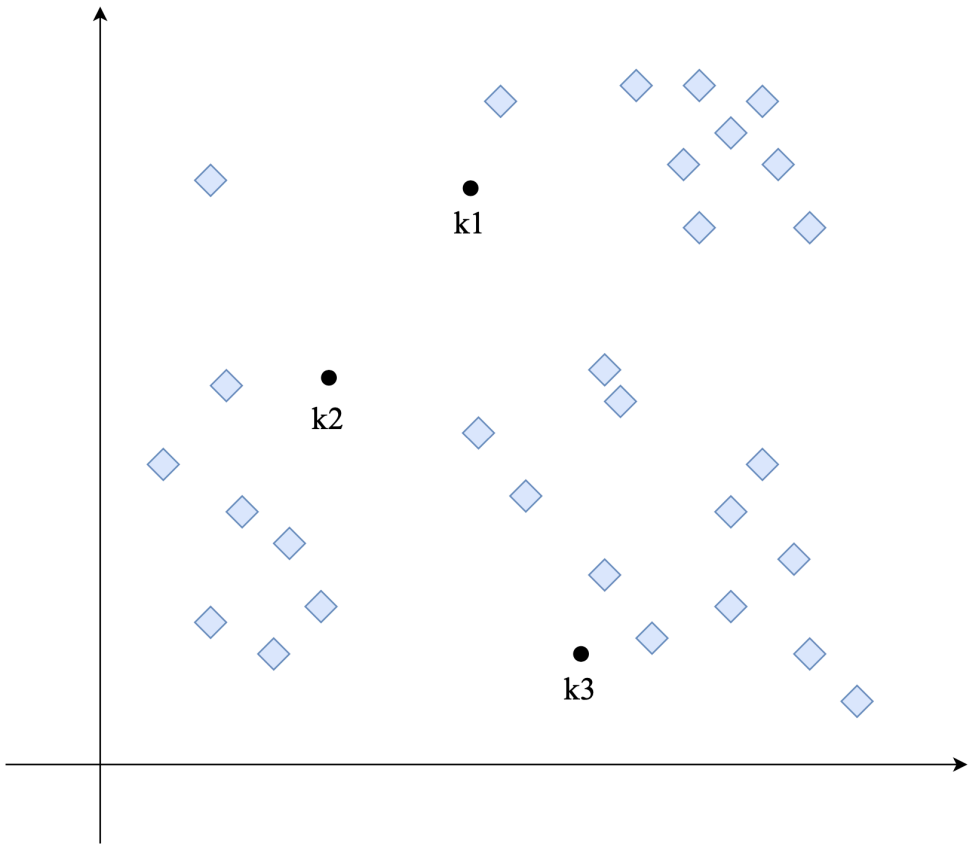

Figure 1: 步骤一

步骤二：计算每个样本到这 3 个样本点中心的距离，将每个样本指派到与其最近的中心的类中，构成聚类结果，如下图，与三个点最近的聚类分别用粉绿黄三种颜色表示。

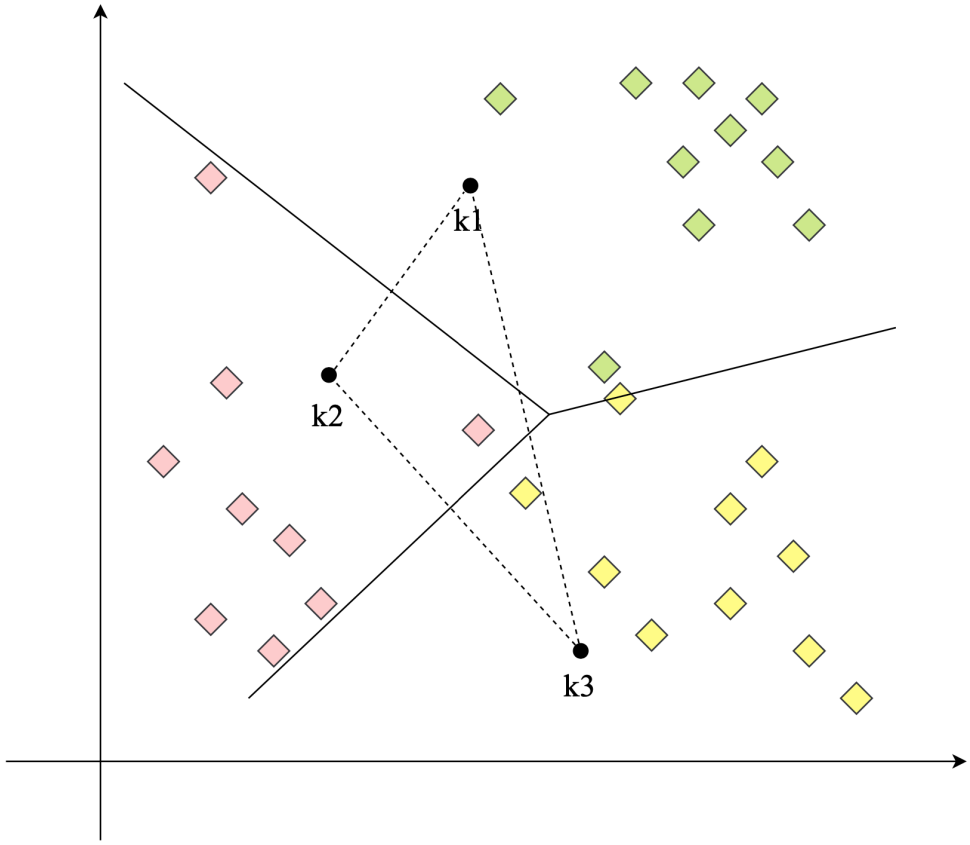

Figure 2: 步骤二

步骤三：对这粉绿黄三个聚类求每个类中样本的均值，作为新的类中心。

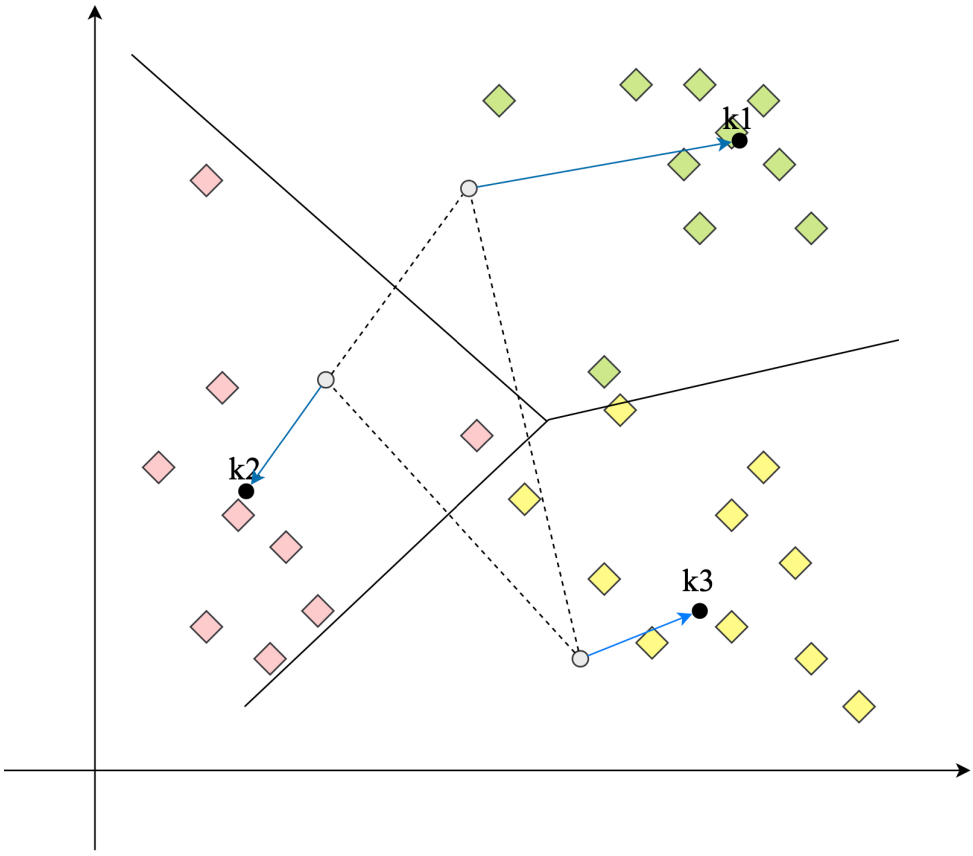

Figure 3: 步骤三

步骤四：重复步骤二和步骤三，直至类中心几乎不再变化或变化很小。

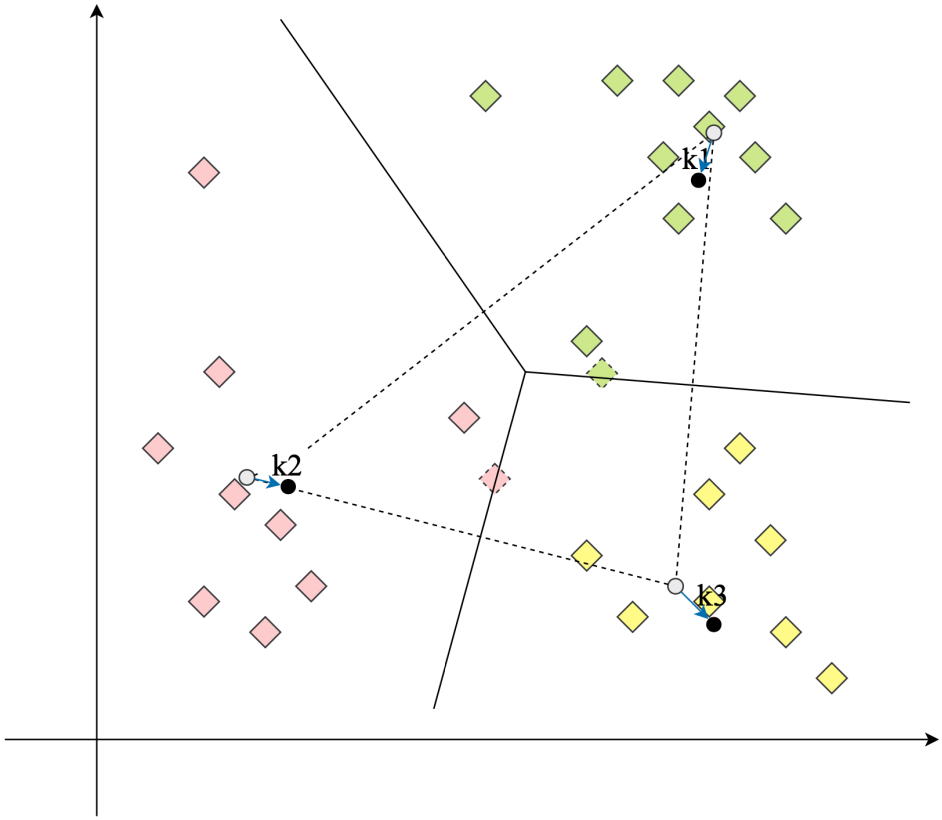

Figure 4: 步骤四

1.2 时间复杂度

K-Means 聚类算法本质上是一个组合优化问题。将 n 个样本分到 k 类，所有可能分法的数目是：

$$S(n,k)=\frac{1}{k!}\sum_{l=1}^{k}(-1)^{k-l}C_k^l k^n \tag{3}$$

这个数字是指数级的。**事实上，K-Means 聚类算法的最优解求解问题是 NP 难问题**。现实中采用迭代的方法求解。
接下来我们细化到每一步，首先我们设问题背景为 p 维空间，总共有 n 个样本。

Table 1: 时间复杂度

步骤	时间复杂度
步骤二	$O(Knp)$
步骤三	$O(np)$
迭代上述两步 l 次	$O(lKnp)$

从上表可看出，每次迭代我们都需要重新计算中心点到所有样本之间的欧氏距离，时间复杂度很高。同样，在本章后面内容中，我们将学习 K 值的选取，学习 K 值的时间开销同样非常大。另一方面，如果初始点没有选好，可能无法进行迭代得到有效的聚类结果。

1.3 算法伪代码

Algorithm 1: K-Means 聚类算法

1 **输入：** n **个样本的集合** X**。初始化。令** $t=0$**，随机选择** K **个点作为初始聚类中心** $m^{(0)}=(m_1^{(0)},m_2^{(0)},...,m_K^{(0)})$**；**
2 **对样本进行聚类。对固定的类中心** $m^{(t)}=(m_1^{(t)},m_2^{(t)},...,m_K^{(t)})$**，计算每个样本到类中心的距离，将每个样本指派到与其最近的中心的类中，构成聚类结果** C^t**。；**
3 **计算新的类中心。对聚类结果** $C^{(t)}$**，计算当前各个类中的样本的均值，作为新的类中心** $m^{(t+1)}=(m_1^{(t+1)},m_2^{(t+1)},...,m_K^{(t+1)})$**。；**
4 **如果迭代收敛或者符合停止条件，输出** $C^*=C^{(t)}$**。否则，令** $t=t+1$**，返回第二步。输出：样本集合的聚类** C^***。**

2 优化算法

K-means 算法非常简单直观，但却很有效。前面我们描述了 K-mean 算法的聚类过程，在这一节我们用数学语言来解释 K-mean 算法背后隐含的优化过程，以及为什么 K-means 可以获得比较好的聚类效果。

我们在前面解释过，一个好的聚类结果应该是：**类内距离很小，类间距离很大**。K-means 算法的本质其实就是在最小化类内的距离。具体来说，假设我们想要划分出 K 类，用 $\mathcal{C}_k$ 表示第 k 类中的所有样本，μ_k 表示第 k 类中的数据的平均值（即第 k 类的中心），K-means 的优化目标为

$$\underset{\mu_k, M}{\operatorname{argmin}} \sum_{k=1}^{K} \sum_{x_i \in \mathcal{C}_k} M_{ik} \parallel x_i - \mu_k \parallel_2^2, \tag{4}$$

其中 $M_{ik} \in \{0, 1\}$ 表示样本 x_i 是否属于第 k 类，即当 $x_i \in \mathcal{C}_k$ 时 $M_{ik} = 1$，否则为 0。因为一个样本只能属于一类，对于每一个 x_i，我们有 $\sum_{k=1}^{K} M_{ik} = 1$。现在我们来求解公式4，这里我们有损失函数 $\mathcal{L} = M_{ik} \parallel x_i - \mu_k \parallel_2^2$，所以

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \mu_k} &= 0 \\ \mu_k &= \frac{\sum_{i=1}^{n} M_{ik} x_i}{\sum_{i=1}^{n} M_{ik}} \end{aligned} \tag{5}$$

显然，μ_k 的最优解是被分类为第 k 累的样本的平均值。计算得到了 μ_k，我们再来计算 M_{ik}，即怎样给样本划分类别。为了完成公式4的优化目标，直觉上来说我们计算 M_{ik} 的方法应该是

$$M_{ik} = \begin{cases} 1, & k = \underset{k}{\operatorname{argmin}} \parallel x_i - \mu_k \parallel_2^2 \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

即将样本 x_i 划分到距离最近的类中。可以看到上述的优化方法就是 K-means 所使用的聚类方法。

3 K-means 收敛性

接下来我们说明 K-means 算法的收敛性。K-means 算法的过程是在最小化类内距离（公式4）。随着迭代，我们将样本分类到最近的类中并更新每一类的中心位置，因此随着迭代，类内距离逐渐下降。所以 K-means 算法会慢慢收敛。K-means 本质上是期望最大算法（Expectation Maximization，EM）的一种特殊情况，EM 算法被证明是收敛的，所以 K-means 算法最终会收敛。我们会在下一节中详细解释 EM 算法。

4 模型选择

在 K-means 算法中，我们需要预先设定我们需要划分出的类别数 K。因为我们并不知道真正的 K 值是什么，所以只能去根据不同方法猜测。虽然没有一种可以完美选择 K 的方法，但是有一些启发性方法在实际应用中已经被证明是有效的。

最简单的方法就是我们用很多 K 值都试一下，根据损失函数值来选择最好的 K 值。我们以图5中的数据点为例，假设数据本身属于两类。我们需要根据无标签的数据来对他们进行聚类。现在我们选择 $K = 1, 2, 3$ 分别完成聚类任务，并按照公式4计算类内距离。

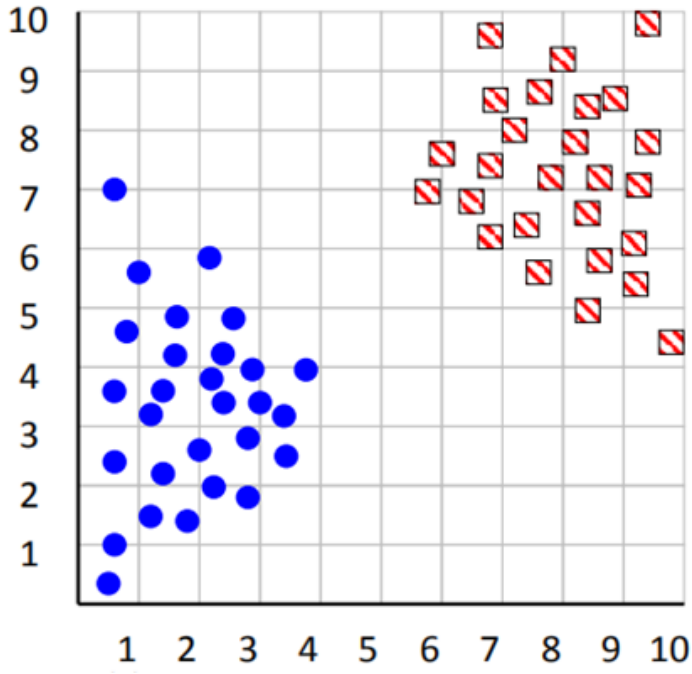

Figure 5: 样本示意图。

图6为设定 $K = 1$ 时的聚类结果，类内距离为 873；图7为设定 $K = 2$ 时的聚类结果，类内距离为 173.1；图8为设定 $K = 3$ 时的聚类结果，类内距离为 133.6。

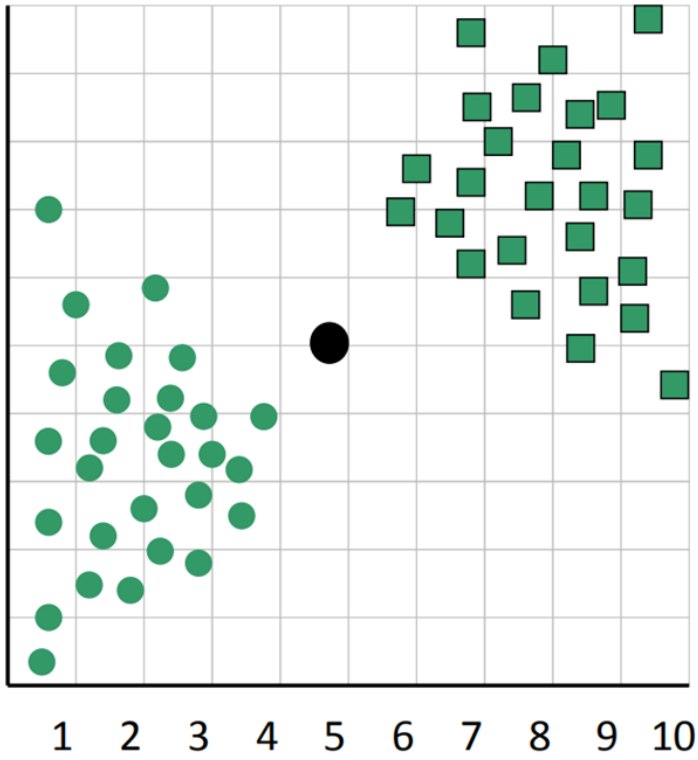

Figure 6: $K = 1$ 时的聚类结果。

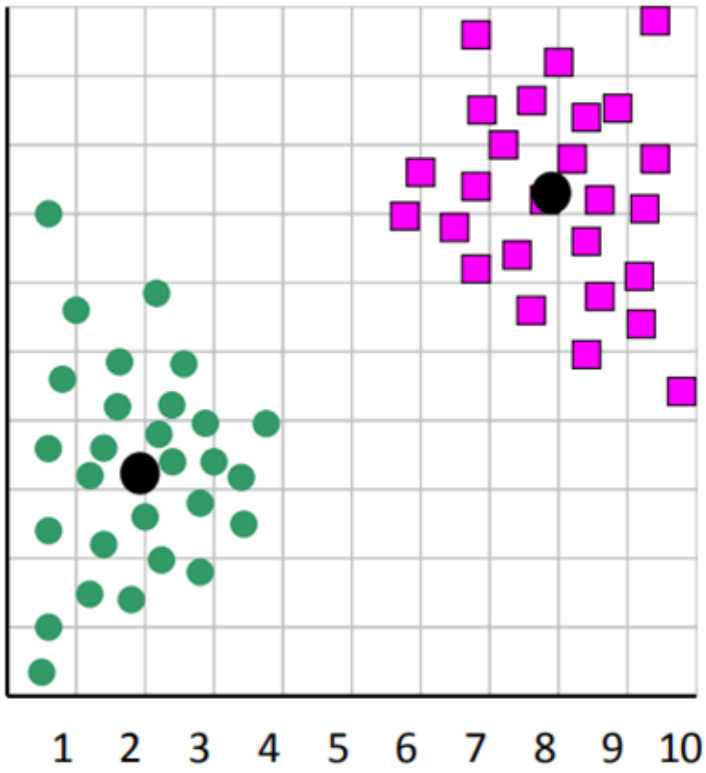

Figure 7: $K = 2$ 时的聚类结果。

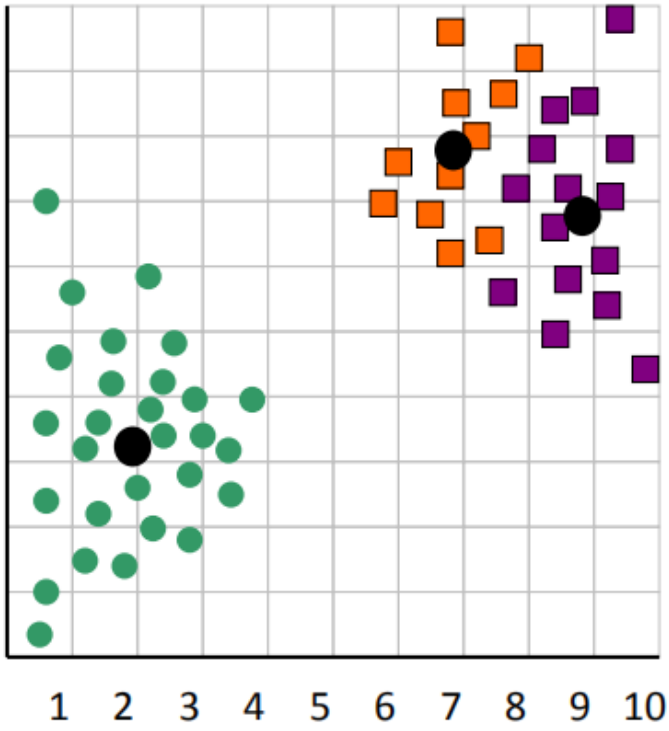

Figure 8: $K = 3$ 时的聚类结果。

如果我们把不同 K 值聚类出的结果的类内距离绘制成折线图（图9），可以看到当 $K > 2$ 的时候类内距离会很小。$K > 2$ 虽然会使得类内距离减小，但是同样会增加运算时间，所以选择 $K = 2$ 或 $K = 3$ 是比较好的选择。

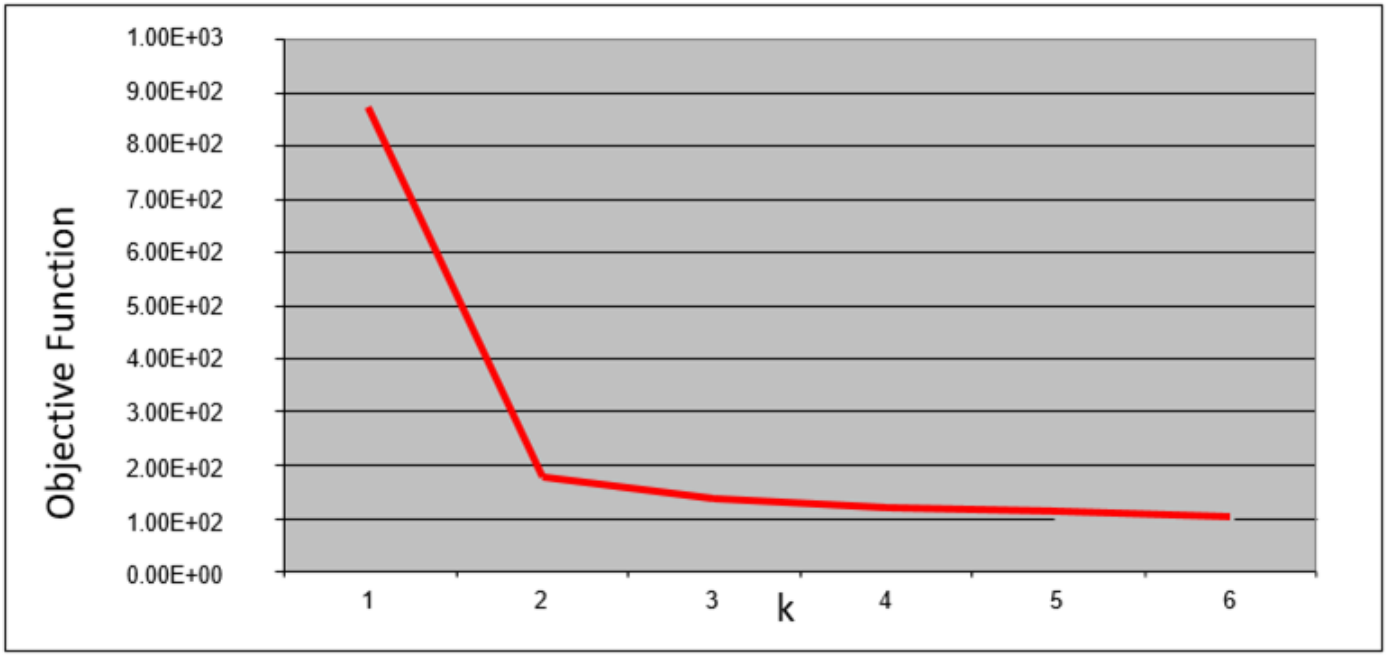

Figure 9: 不同 K 值聚类结果对应类内距离值。

除了上述的选择方法，我们还会用其他的指标来选择 K 值，比如**轮廓系数（silhouette coefficient）**。轮廓系数综合考虑类内和类间距离，计算方法为

$$S = \frac{b - a}{\max(a, b)} \tag{7}$$

其中 a 为所有类的类内平均距离，b 为所有类的类间平均距离。轮廓系数越大说明聚类结果越好。

引用

[1] Silhouette coefficient：https://en.wikipedia.org/wiki/Silhouette_(clustering)

[2] K-means Clustering: https://en.wikipedia.org/wiki/K-means_clustering

[3 Hang Li 'Statistical Learning'