

# 20

## K-Means Clustering

# *K*均值聚类

簇内距离和最小，迭代求解



几何是万物美的本原。

*Geometry is the archetype of the beauty of the world.*

—— 约翰内斯·开普勒 (Johannes Kepler) | 德国天文学家、数学家 | 1571 ~ 1630



- ▶ `numpy.cov()` 计算协方差矩阵
- ▶ `pandas.DataFrame.cov()` 计算数据帧协方差矩阵
- ▶ `scipy.spatial.Voronoi` 函数获得沃罗诺伊图相关数据
- ▶ `scipy.spatial.voronoi_plot_2d` 函数绘制沃罗诺伊图
- ▶ `sklearn.cluster.KMeans()` *K* 均值聚类算法函数; `model.fit()` 拟合数据, `model.predict()` 预测聚类标签, `model.cluster_centers_` 输出簇质心位置, `model.inertia_` 输出簇 SSE 之和
- ▶ `sklearn.metrics.silhouette_score` 计算轮廓系数
- ▶ `yellowbrick.cluster.SilhouetteVisualizer` 函数绘制轮廓图

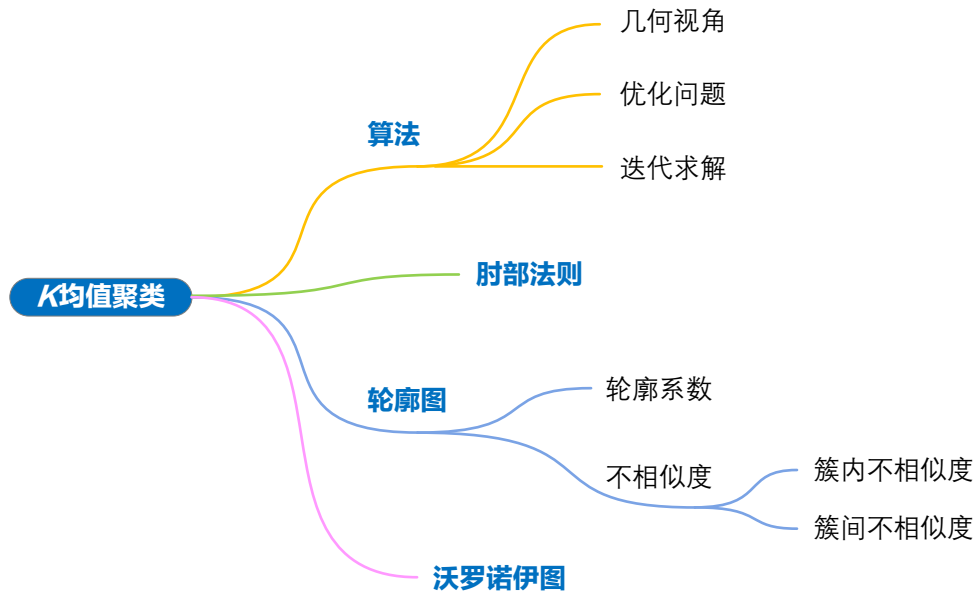
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)



## 20.1 $K$ 均值聚类

$K$ 均值聚类 ( $K$ -means clustering) 的  $K$  不同于  $k$  近邻中的  $k$ 。

▲ 注意，本书第 2 章介绍的  $k$  近邻算法 ( $k$ -Nearest Neighbors,  $k$ -NN) 是有监督学习分类算法，样本数据有标签，它的  $k$  是指设定的近邻数量。

而  $K$  均值聚类则是无监督学习聚类算法，样本数据无标签， $K$  是指将给定样本集  $\Omega$  划分成  $K$  簇  $C = \{C_1, C_2, \dots, C_K\}$ 。

### 原理

图 1 所示为  $K$  均值算法原理图。 $K$  均值聚类的每一簇样本数据用簇质心 (cluster centroid) 来描述。比如，二聚类问题有两个簇质心  $\mu_1$  和  $\mu_2$ 。

如果以欧氏距离为距离度量，距离质心  $\mu_1$  更近的点，被划分为  $C_1$  簇；而距离质心  $\mu_2$  更近的点，被划分为  $C_2$  簇。

比如，图 1 中  $A$  点明显距离  $\mu_1$  更近， $A$  点被划分为  $C_1$  簇， $C$  点距离  $\mu_2$  更近，因此  $C$  点划分到  $C_2$  簇。 $B$  点距离  $\mu_1$  和  $\mu_2$  相等，因此  $B$  点位于决策边界。很明显，决策边界为  $\mu_1$  和  $\mu_2$  中垂线 (perpendicular bisector)。



建议大家回顾《矩阵力量》第 19 章讲解的有关中垂线内容。

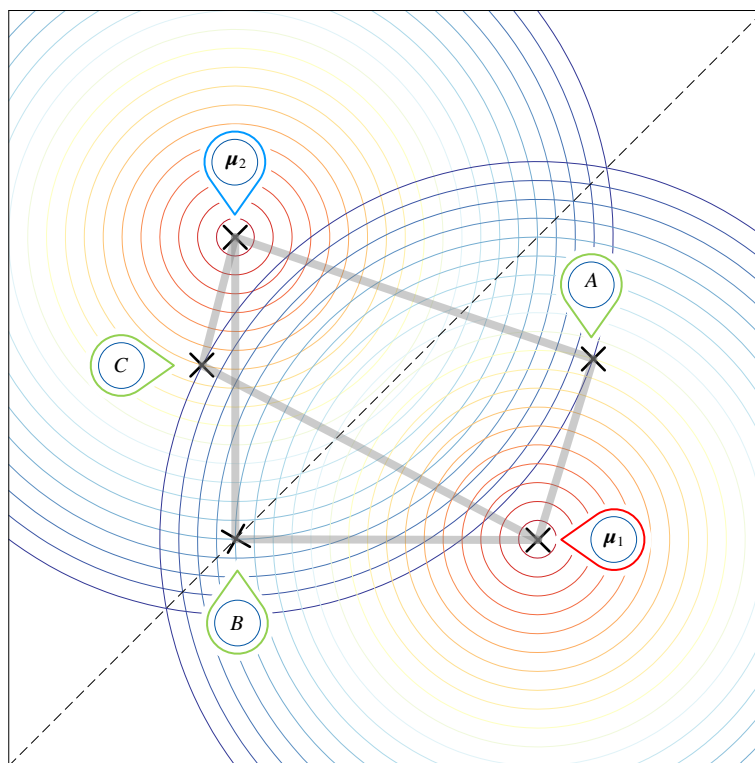


图 1.  $K$  均值算法原理

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

由于采用欧氏距离，图 1 中簇质心  $\mu_1$  和  $\mu_2$  等高线为两组同心圆；同心圆颜色相同，代表距离簇质心  $\mu_1$  和  $\mu_2$  距离相同。因此，同色同心圆的交点位于决策边界上。

## 20.2 优化问题

$K$  均值聚类算法的优化目标是，将所有给定样本点划分  $K$  簇，并使得簇内距离平方和最小。采用最简单的欧拉距离，以上优化目标记做：

$$\arg \min_C \sum_{k=1}^K \sum_{x \in C_k} \|x - \mu_k\|^2 \quad (1)$$

其中， $x$  为样本数据任意一点，形式为列向量； $\mu_k$  为任意一簇  $C_k$  样本数据的质心。

实际上，(1) 中簇内距离平方和相当于**残差平方和** (Sum of Squared Error, SSE)。SSE 度量样本数据的聚集程度；为了方便读者理解，下一节专门讲解质心、协方差矩阵、残差平方和等描述簇数据的数学工具。

任意一点  $x$  和质心  $\mu_k$  欧氏距离平方，可以通过下式计算得到：

$$d^2 = \text{dist}(x, \mu_k)^2 = \|x - \mu_k\|^2 = (x - \mu_k)^T (x - \mu_k) \quad (2)$$

其中，

$$x = [x_1 \ x_2 \ \cdots \ x_D]^T, \quad \mu_k = [\mu_{k,1} \ \mu_{k,2} \ \cdots \ \mu_{k,D}]^T \quad (3)$$

### 两特征聚类

当特征数为  $D = 2$  时，欧氏距离平方和展开为下式：

$$\begin{aligned} d^2 &= (x - \mu_k)^T (x - \mu_k) \\ &= (x_1 - \mu_{k,1})^2 + (x_2 - \mu_{k,2})^2 \end{aligned} \quad (4)$$

丛书反复介绍，当  $d$  取某一定值时，(4) 所示解析式是以  $(\mu_{k,1}, \mu_{k,2})$  为圆心的正圆， $d$  为半径。如图 1 所示， $d$  取不同值时，(4) 的几何表达为以  $\mu_1$  和  $\mu_2$  为中心得到两组同心正圆。

对于二分类问题，决策边界满足：

$$(x - \mu_1)^T (x - \mu_1) = (x - \mu_2)^T (x - \mu_2) \quad (5)$$

整理得到决策边界解析式

$$(\mu_1 - \mu_2)^T \left( x - \frac{(\mu_1 + \mu_2)}{2} \right) = 0 \quad (6)$$

发现  $K$  均值聚类决策边界为一超平面，超平面通过  $\mu_1$  和  $\mu_2$  中点，并垂直于  $\mu_1$  和  $\mu_2$  连线，即垂直于  $(\mu_2 - \mu_1)$ 。

### 三聚类

图2所示为三聚类问题簇数据和质心位置。根据这三个质心位置，可以绘制两两质心的中垂线。决策边界在这三条中垂线上。图2实际上便是**沃罗诺伊图** (Voronoi diagram)。本章最后一节介绍沃罗诺伊图。

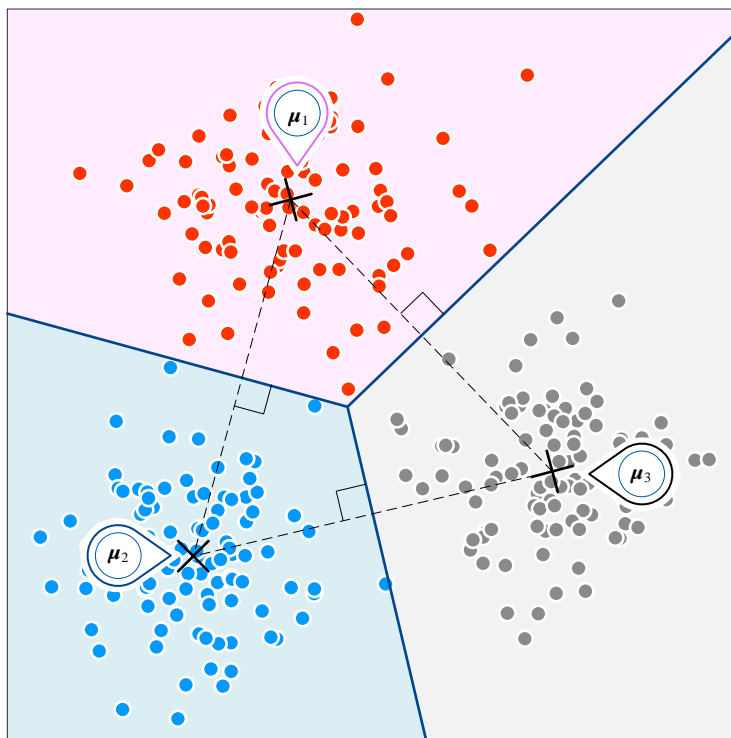


图 2. 三聚类问题簇质心、决策边界和区域划分

图3中， $z$ 轴高度代表三聚类预测标签。

容易发现，在确定决策边界位置上， $K$ 均值聚类原理和本书第2章介绍的**最近质心分类器** (Nearest Centroid Classifier) 很相似。

不同的是， $K$ 均值聚类算法采用迭代方式找到簇质心位置，这是下一节要介绍的内容。

采用欧氏距离的  $K$  均值聚类相当于**高斯混合模型** (Gaussian Mixture Model, GMM) 的一个特例。这一点，下一章会详细介绍。

目前 Scikit-learn 中  $K$  均值聚类算法距离度量仅支持欧氏距离；MATLAB 中  $K$  均值聚类算法函数还支持城市街区距离、余弦距离、相关系数距离等。

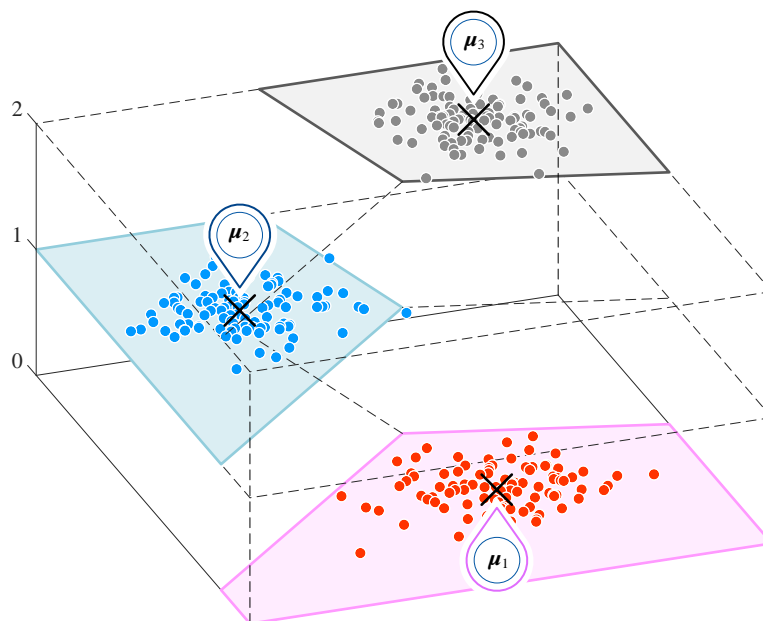


图 3. 三聚类预测标签

## 20.3 迭代过程

本节以二聚类为例介绍  $K$  均值聚类流程图。

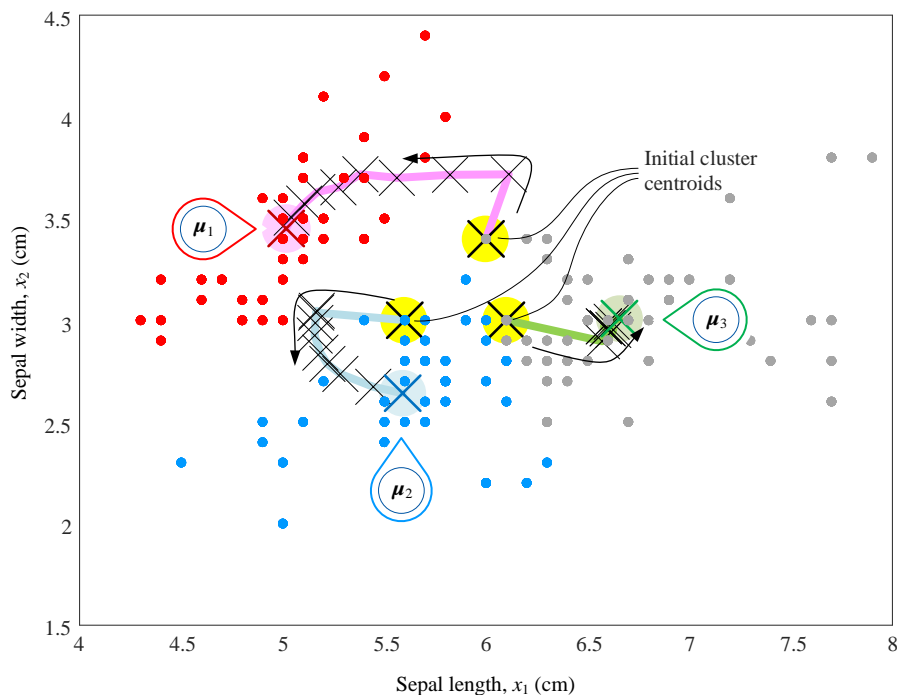
### 流程

流程输入为样本数据和聚类簇数 (比如 2)。然后, 从样本中随机选取 2 个数据作为初始簇质心  $\mu_1$  和  $\mu_2$ 。然后进入如下迭代循环:

- ◀ 计算每一个样本和均值向量  $\mu_1$  和  $\mu_2$  距离;
- ◀ 比较每个样本和  $\mu_1$  和  $\mu_2$  距离, 确定簇划分;
- ◀ 根据当前簇, 计算并更新均值向量  $\mu_1$  和  $\mu_2$

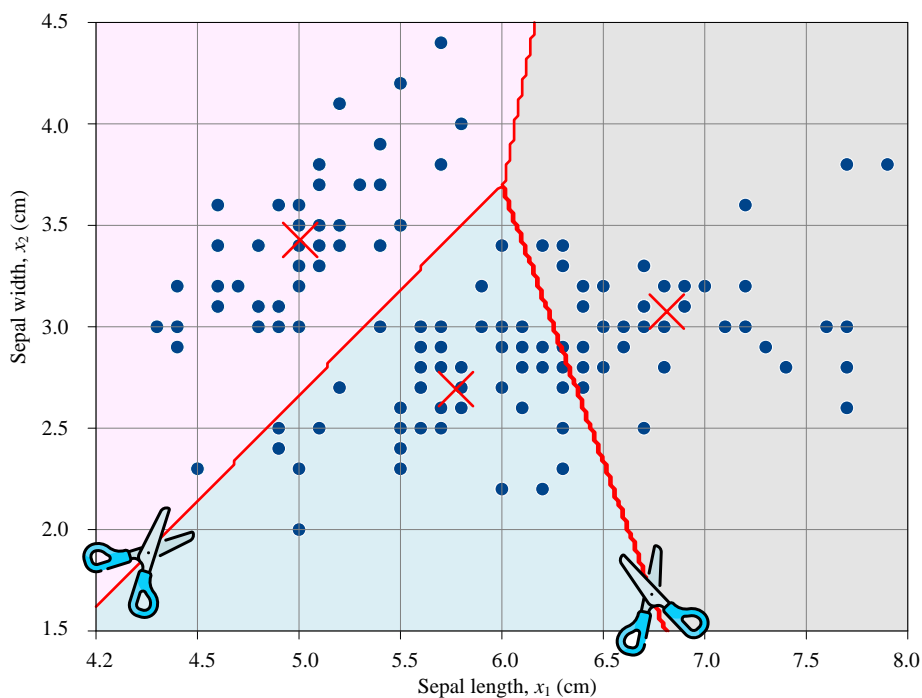
直到均值向量  $\mu_1$  和  $\mu_2$  满足迭代停止条件, 得到簇划分。

图 4 所示为一鸢尾花数据为例  $K$  均值算法迭代过程。随机选取三个样本点 (黄色高亮) 作为初始簇质心  $\mu_1$ 、 $\mu_2$  和  $\mu_3$ , 经过 10 次迭代, 簇质心位置不断连续变化, 最终收敛。

图 4.  $K$  均值算法迭代过程，鸢尾花数据为例

### 鸢尾花数据聚类

图 5 所示为  $K$  均值算法聚类鸢尾花数据。Scikit-learn 工具包用来  $K$  均值聚类算法函数为 `sklearn.cluster.KMeans()`。利用 `model.fit()` 拟合数据后，`model.predict()` 预测聚类标签，`model.cluster_centers_`，输出簇质心位置。

图 5.  $K$  均值算法聚类鸢尾花数据

代码 Bk7\_Ch20\_01.ipynb 绘制图 5。下面聊聊其中关键语句。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

a 用 `sklearn.datasets.load_iris()` 导入数据。

b 只用鸢尾花数据的前两个特征（花萼长度、花萼宽度）训练聚类。请大家尝试使用鸢尾花其他特征组合完成聚类。

c 用 `sklearn.cluster.KMeans()` 创建 KMeans 对象。

`n_clusters=3` 指定了要将数据分成的簇的数量。请大家尝试其他簇数，比如 2、4 等等，并比较结果。

`n_init='auto'` 指定了初始化中心点的次数。KMeans 算法的结果可能受到初始中心点位置的影响，因此可以尝试多次不同的初始化以找到更好的聚类结果。'auto' 表示算法会自动选择一个合适的初始化次数，当前默认 10 次。

d 调用 KMeans 对象 `kmeans`，用 `fit()` 方法对训练数据进行拟合。

e 使用 KMeans 模型对网格数据进行聚类预测。

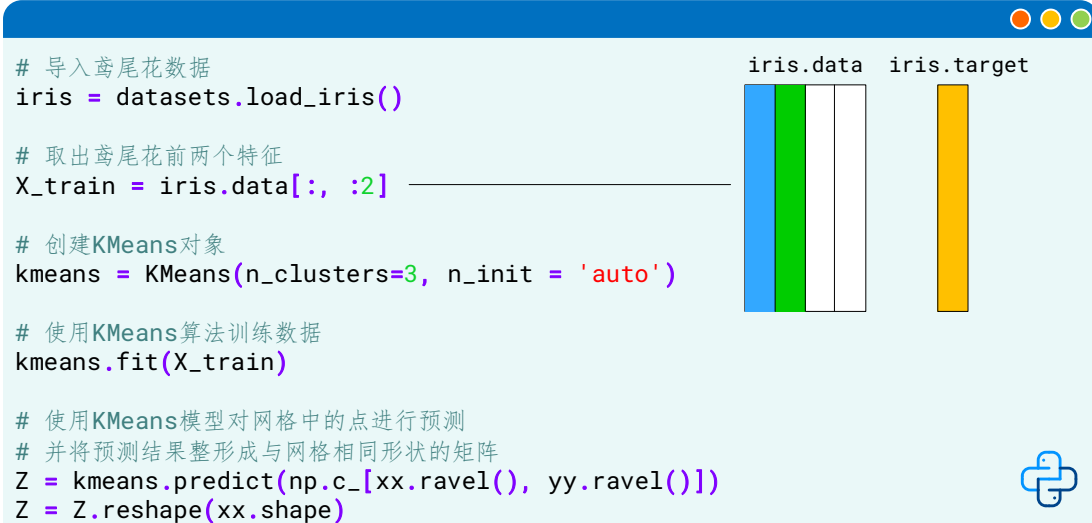
其中，`ravel()` 是 `numpy` 中的方法，它将多维数组转换为一维。`np.c_[xx.ravel(), yy.ravel()]` 按列并列拼接两个一维数组，形成一个包含网格中所有点坐标的二维数组。

《编程不难》介绍过，对于已经训练好的聚类模型，如果模型可以将全新的数据点分配到确定的簇中，这类聚类算法叫做归纳聚类（inductive clustering）。

不具备这种能力的聚类算法叫做非归纳聚类（non-inductive clustering）。非归纳聚类只能对训练数据进行聚类，而不能将新数据点添加到已有的模型中进行预测。

显然，KMeans 是一种归纳聚类算法。

f 将聚类预测结果规整成和网格数据相同形状的矩阵。



```

# 导入鸢尾花数据
a iris = datasets.load_iris()

# 取出鸢尾花前两个特征
b X_train = iris.data[:, :2]

# 创建KMeans对象
c kmeans = KMeans(n_clusters=3, n_init = 'auto')

# 使用KMeans算法训练数据
d kmeans.fit(X_train)

# 使用KMeans模型对网格中的点进行预测
# 并将预测结果整形成与网格相同形状的矩阵
e Z = kmeans.predict(np.c_[xx.ravel(), yy.ravel()])
f Z = Z.reshape(xx.shape)
  
```

代码 1. 用 `sklearn.cluster.KMeans()` 完成聚类 | Bk7\_Ch20\_01.ipynb

## 20.4 肘部法则：选定聚类簇值

**手肘法则** (elbow method) 可以用来判断合适的聚类簇值  $K$ 。手肘法则的关键指标是误差平方和 SSE：

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微课视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)



$$\text{SSE}(X|K) = \sum_{k=1}^K \text{SSE}(C_k) = \sum_{k=1}^K \sum_{x \in C_k} \|x - \mu_k\|^2 \quad (7)$$

SSE 也叫**惯性量** (inertia)。

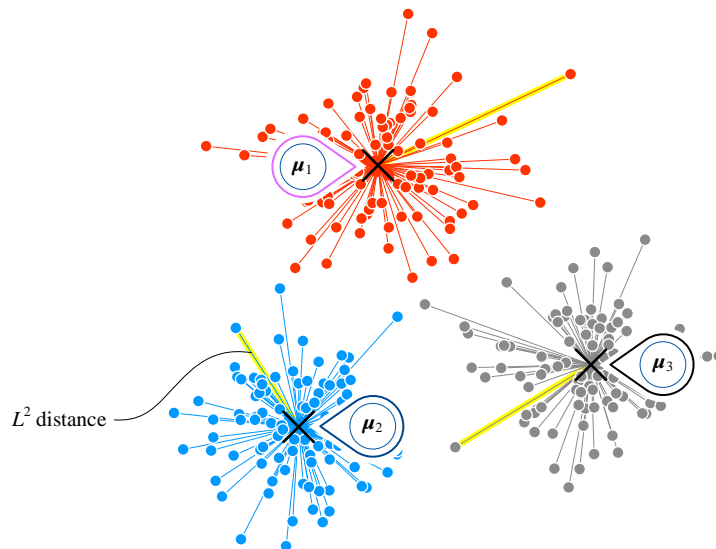


图 6. 样本数据和各簇质心  $\mu_1$ 、 $\mu_2$  和  $\mu_3$  之间的距离

随着聚类簇数  $K$  不断增大，均值聚类算法对样本数据划分会逐渐变得更加精细；因此，随着  $K$  不断增大，每个簇的聚合程度会逐渐提高，SSE 会逐渐变小。

极端情况，当  $K = n$ ，也就是每个样本数据自成一簇， $\text{SSE} = 0$ 。

$K$  不断增大，SSE 不断减小的过程如图 7 所示。观察此图发现一个有意思的现象，当  $K$  小于“合适”聚类数时， $K$  增大时，会导致 SSE 大幅下降；但是， $K$  大于“合适”聚类数时， $K$  再增大，SSE 下降幅度不断变缓。

这就是为什么图 7 呈现出“手肘”形状，也便是手肘法则的名称来由。理想的聚类簇数  $K$  便是“肘”拐点的位置。 $K$  均值聚类算法函数输出值 `model.inertia_` 便是当前 SSE 值。

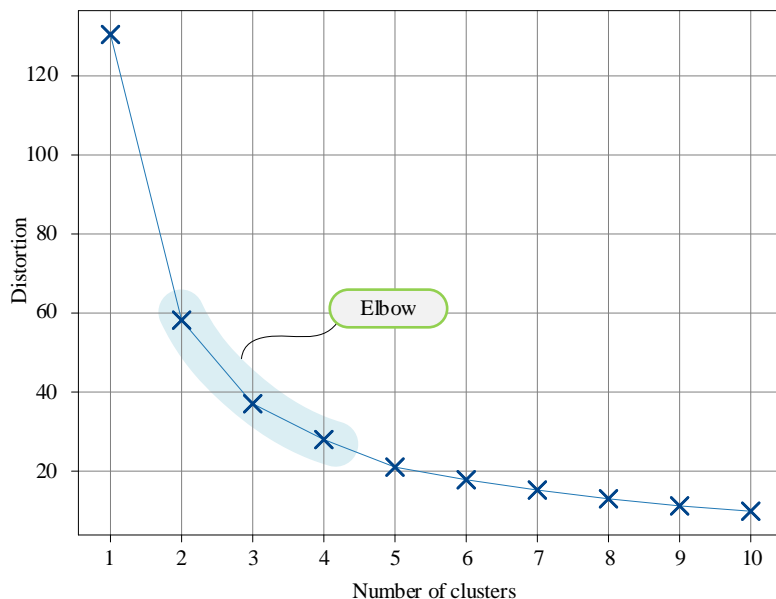


图 7. K 均值算法聚类鸢尾花数据



代码 Bk7\_Ch20\_02.ipynb 绘制图 7。请大家自行学习这段代码。

## 20.5 轮廓图：选定聚类簇值

**轮廓图** (silhouette plot) 也常用来选定聚类簇值  $K$ 。

轮廓图上每一条线代表的是**轮廓系数** (silhouette coefficient),  $s_i$ , 可以通过下式计算获得：

$$s_i = \frac{b_i - a_i}{\max\{a_i, b_i\}} \quad (8)$$

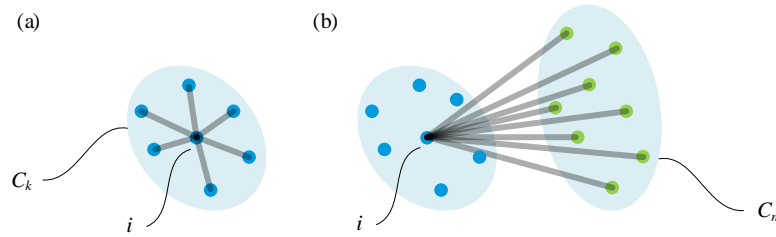
其中,  $a_i$  为簇内不相似度,  $b_i$  为簇间不相似度。

### 簇内不相似度

如图 8 (a) 所示, 簇内不相似度  $a_i$  代表样本  $i$  ( $i \in C_k$ ) 到同簇其他样本  $j$  ( $j \in C_k, i \neq j$ ) 距离平均值：

$$a_i = \frac{1}{\text{count}(C_k) - 1} \sum_{j \in C_k, i \neq j} d_{i,j} \quad (9)$$

其中,  $d_{i,j}$  为样本  $i$  和  $j$  之间距离。 $a_i$  越小, 说明样本  $i$  越应该被划分到  $C_k$  簇。

图 8.  $a_i$  为簇内不相似度，和  $b_i$  为簇间不相似度

### 簇间不相似度

如图 8 (b) 所示，簇间不相似度  $b_i$  代表样本  $i$  ( $i \in C_k$ ) 到其他簇 ( $C_m$ ) 样本  $j$  ( $j \in C_m, C_m \neq C_k$ ) 距离平均值的最小值：

$$b_i = \min \frac{1}{\text{count}(C_m)} \sum_{j \in C_m} d_{i,j} \quad (10)$$

$b_i$  越大，说明样本  $i$  越不应该被划分到其他簇。

⚠ 注意，当簇数超过 2 时， $b_i$  需要在不同簇之间取最小值。

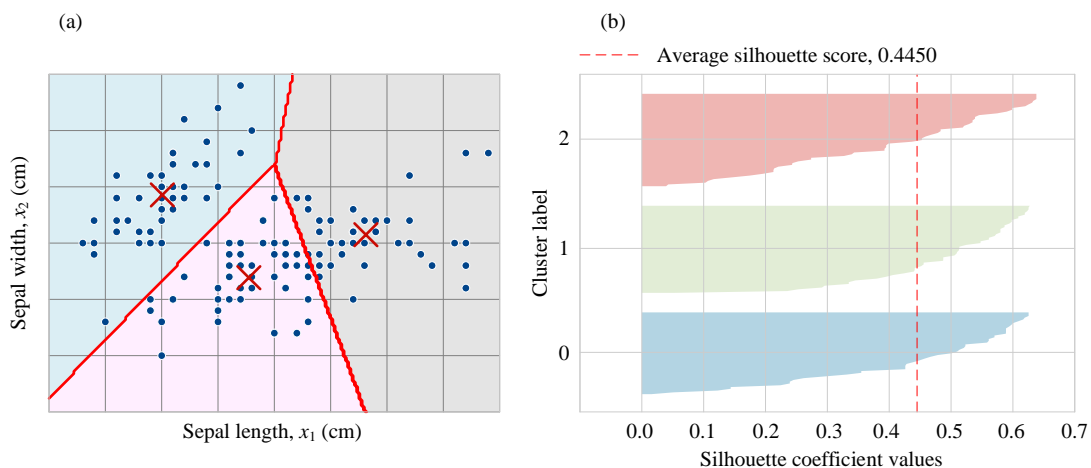
### 以鸢尾花数据为例

轮廓系数  $s_i$  的取值在  $[-1, 1]$  区间。 $s_i$  越趋向于 1，说明样本  $i$  分类越正确； $s_i$  越趋向于 -1，说明样本  $i$  分类越错误。当  $s_i$  在 0 附近时，样本  $i$  靠近聚类边界。

图 9、图 10 和图 11 所示为  $K$  分别取 3、4 和 5 时，聚类边界和轮廓图。理想的聚类结果是，簇内尽量紧密，簇间尽量远离。轮廓系数平均值越高，说明分类越合理。比较图 9、图 10 和图 11， $K = 3$  时，轮廓系数较高，并且轮廓图簇宽度均匀。轮廓图结合肘部法则判断聚类簇数更合适。

计算轮廓系数的函数为 `sklearn.metrics.silhouette_score`。

`yellowbrick.cluster.SilhouetteVisualizer` 函数绘制轮廓图。

图 9.  $K$  均值算法聚类鸢尾花数据和轮廓图， $K = 3$ 

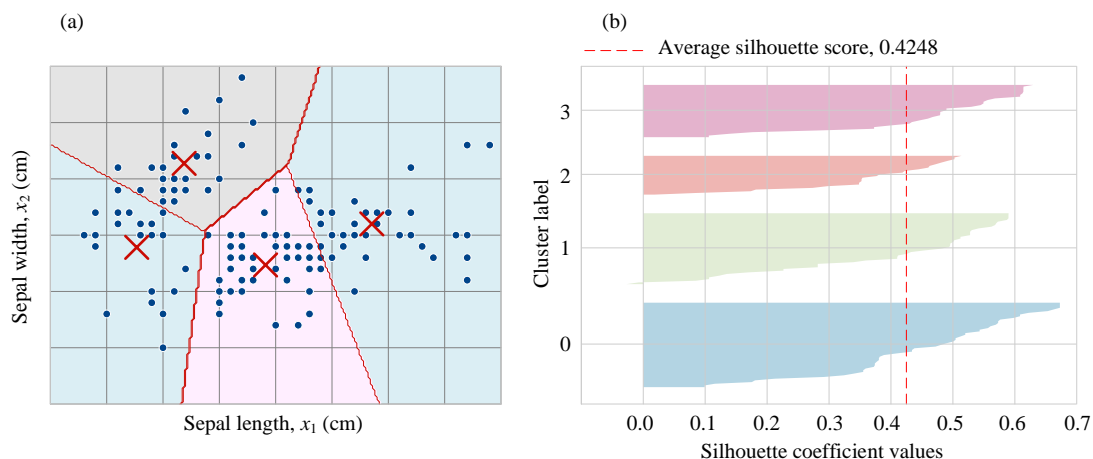
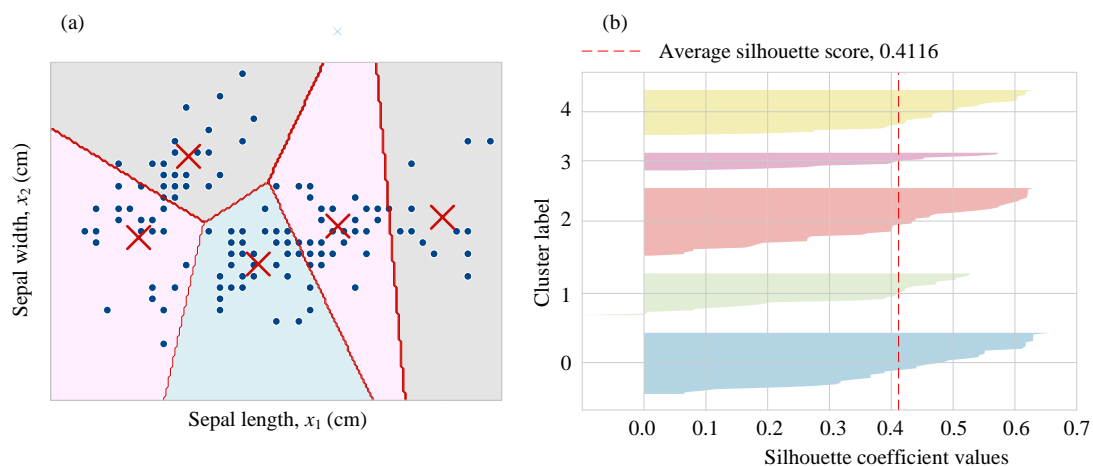
本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有，请勿商用，引用请注明出处。

代码及 PDF 文件下载：<https://github.com/Visualize-ML>

本书配套微视频均发布在 B 站——生姜 DrGinger：<https://space.bilibili.com/513194466>

欢迎大家批评指教，本书专属邮箱：[jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

图 10.  $K$  均值算法聚类鸢尾花数据和轮廓图,  $K = 4$ 图 11.  $K$  均值算法聚类鸢尾花数据和轮廓图,  $K = 5$ 

代码 Bk7\_Ch20\_03.ipynb 绘制图 9、图 10 和图 11。请大家先用 `pip install yellowbrick` 安装 yellowbrick。

## 20.6 沃罗诺伊图

**沃罗诺伊图** (Voronoi diagram), 是由俄国数学家格奥尔吉·沃罗诺伊 (Georgy Voronoy) 发明的空间分割算法。本章介绍的  $K$  均值聚类, 本书前文介绍的**最近质心分类器** (Nearest Centroid Classifier), 实际上都依赖沃罗诺伊图确定决策边界。

图 12 所示为平面 4 点构造的沃罗诺伊图。距离较近的两点连线, 绘制中垂线; 若干中垂线便是分割平面区域的边界线。

配套代码中先用 `scipy.spatial.Voronoi` 函数获得沃罗诺伊图相关数据。

本 PDF 文件为作者草稿, 发布目的为方便读者在移动终端学习, 终稿内容以清华大学出版社纸质出版物为准。

版权归清华大学出版社所有, 请勿商用, 引用请注明出处。

代码及 PDF 文件下载: <https://github.com/Visualize-ML>

本书配套微视频均发布在 B 站——生姜 DrGinger: <https://space.bilibili.com/513194466>

欢迎大家批评指教, 本书专属邮箱: [jiang.visualize.ml@gmail.com](mailto:jiang.visualize.ml@gmail.com)

然后用 `scipy.spatial.voronoi_plot_2d` 函数绘制沃罗诺伊图。图 13 所示为随机生成平面 30 个点，以及它们构造的沃罗诺伊图。

$K$  均值聚类，相当于在利用圆圈（欧氏距离）描述每个簇质心；而实际上，描述簇数据更好的形状可能是正椭圆，甚至旋转椭圆。这就是下一章高斯混合模型 GMM 可以解决的问题。

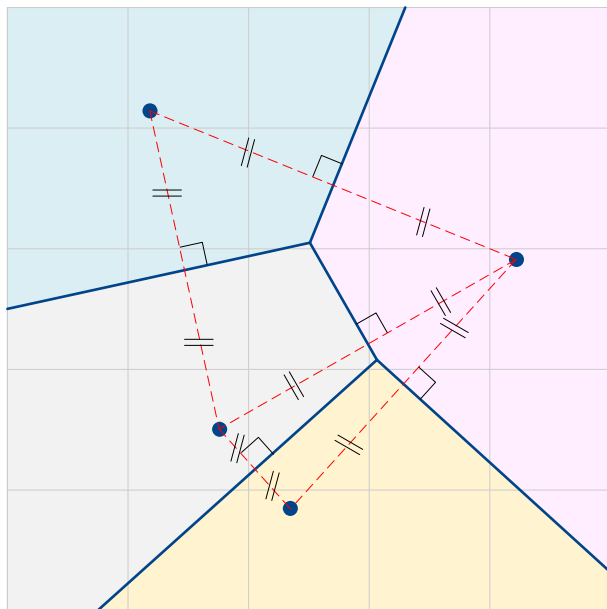


图 12.4 点平面沃罗诺伊图

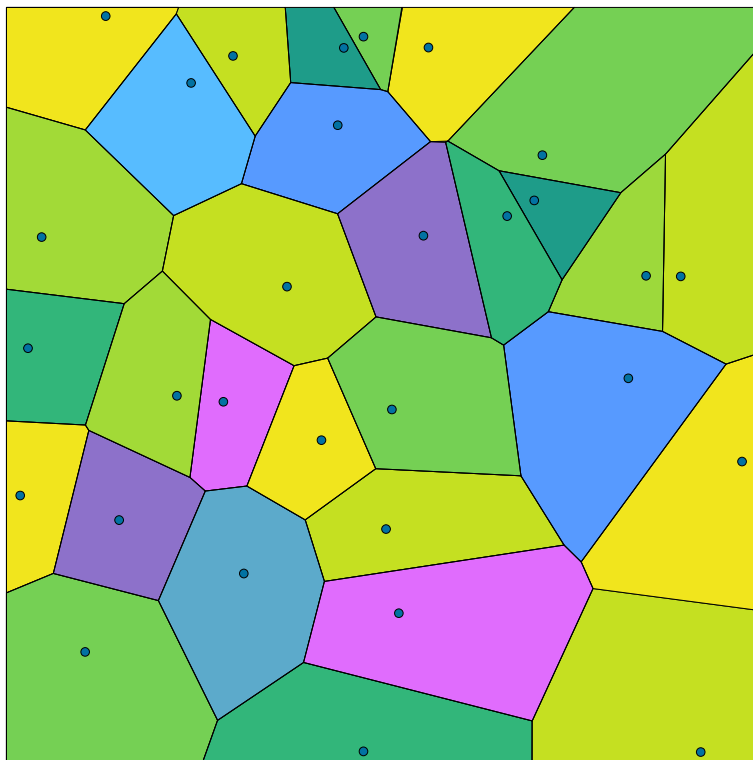


图 13.30 点平面沃罗诺伊图



代码 Bk7\_Ch20\_04.ipynb 绘制图 13。



$K$  均值聚类是一种无监督的机器学习技术，用于将数据集分为  $K$  个不同的簇。该算法首先需要随机初始化  $K$  个聚类中心，然后根据数据点和聚类中心的距离将数据点划分到最近的簇中。接着更新聚类中心，并重复以上步骤，直到聚类中心不再发生变化或达到预设的迭代次数。

该算法的优化问题是最小化数据点与其所属聚类中心之间的距离和，可以使用梯度下降等方法来求解。肘部法则是一种确定最佳  $K$  值的方法，它基于聚类中心数量  $K$  与聚类误差平方和之间的关系。当  $K$  值增大时，SSE 逐渐减小，但减小速度会逐渐变慢，当  $K$  达到某个值时，SSE 的下降速度会急剧减缓，这个  $K$  值对应的点就是肘部。轮廓图是一种衡量聚类结果质量的方法，它基于数据点与其所属簇的紧密度和分离度之间的平衡。



$K$ -means 聚类结果的簇质心并不是从样本数据点挑选出来的；如果从样本数据点所在位置挑选合适的位置作为簇质心的话，这种方法叫做  $k$  中心聚类 ( $k$ -medoids clustering)。请大家参考下例，这个例子还使用不同距离度量。

[https://scikit-learn-extra.readthedocs.io/en/latest/auto\\_examples/cluster/plot\\_kmedoids\\_digits.html](https://scikit-learn-extra.readthedocs.io/en/latest/auto_examples/cluster/plot_kmedoids_digits.html)