南開大學
Nankai University

计算机学院

机器学习实验报告

聚类分析

姓名：周重天

学号：2311082

专业：计算机科学与技术

2025 年 12 月 3 日

目录

1 实验原理

1.1 层次聚类的基本概念

层次聚类（Hierarchical Clustering）是一种通过构建树状的聚类结构来进行数据聚类的方法。与 K-means 等平面聚类不同，层次聚类能够生成一个叫做树状图 (dendrogram) 的层次化聚类结构，直观展示样本间的聚类关系。

层次聚类的核心思想是逐步合并相距最近的两个簇，直到达到预设的簇个数。其中，不同的簇间距离定义方式会显著影响聚类结果。本实验主要对比以下三种常用的距离度量方式。

1.2 Single-linkage（单链接）

定义：两个簇之间的距离定义为两个簇中最近样本点对之间的距离：

$$d(C_1, C_2) = \min\{d(a, b) \mid a \in C_1, b \in C_2\}$$

特性：

- 倾向于产生链式结构，容易形成细长的聚类
- 对离群点敏感，容易受到” 桥接” 效应的影响
- 可以发现非凸形状的簇，对月牙形数据表现良好

1.3 Complete-linkage（全链接）

定义：两个簇之间的距离定义为两个簇中最远样本点对之间的距离：

$$d(C_1, C_2) = \max\{d(a, b) \mid a \in C_1, b \in C_2\}$$

特性：

- 倾向于产生紧凑的、球形的聚类结果
- 对离群点更为敏感，易受外围点影响
- 倾向于产生大小相近的簇

1.4 Average-linkage（平均链接）

定义：两个簇之间的距离定义为两个簇中所有样本点对距离的平均值：

$$d(C_1, C_2) = \frac{1}{|C_1| \times |C_2|} \sum_{a \in C_1, b \in C_2} d(a, b)$$

特性：

- 结合了 single-linkage 和 complete-linkage 的优点
- 通常比前两者更加稳健
- 对数据的不同分布特征有较好的适应性

2 基础要求：Single-linkage 与 Complete-linkage 聚类

2.1 数据集生成

实验使用 sklearn 库中的 `make_blobs` 函数生成人工数据集，具体参数如下：

- 样本数：150
- 特征数：2（便于二维可视化）
- 真实簇数：3
- 簇标准差：1.3

2.1.1 原始数据分布

原始数据包含 3 个高斯分布的簇，它们之间存在适度的间隔和少量重叠。

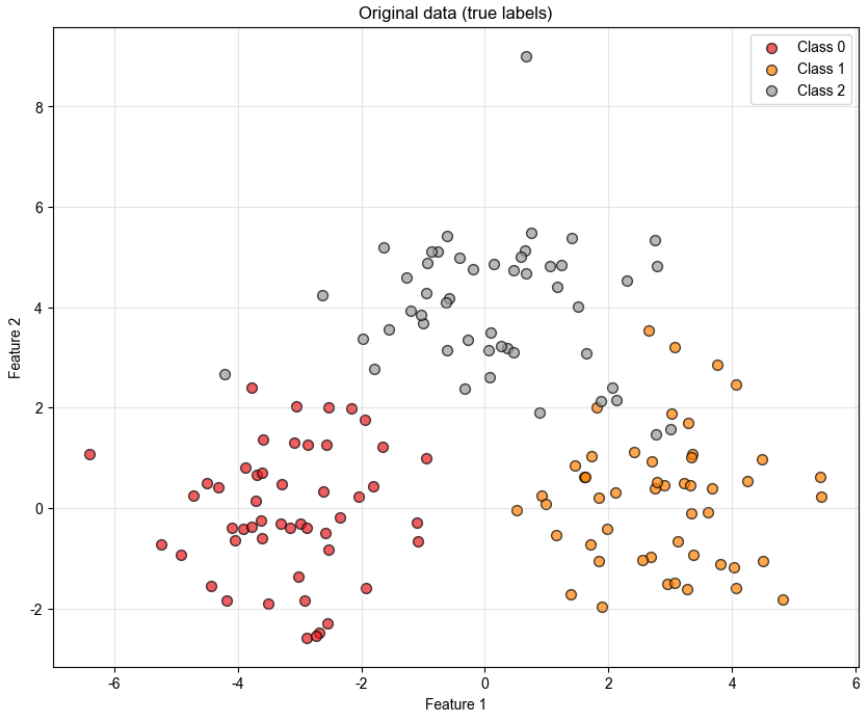

图 2.1: 原始数据分布（make_blobs 数据集）

2.2 Single-linkage 实现与结果

2.2.1 核心代码

```
def _single_linkage_distance(self, cluster1, cluster2, dist_matrix):
    """
    Single-linkage: 两个簇之间的【最小】距离

    公式: d(C1, C2) = min{d(a,b) | a C1, b C2}
    """
```

```
    min_dist = np.inf

    # 遍历两个簇中的所有样本对，找最小距离
    for i in cluster1:
        for j in cluster2:
            # 比较并更新最小距离
            if dist_matrix[i, j] < min_dist:
                min_dist = dist_matrix[i, j]

    return min_dist
```

2.2.2 聚类结果

对原始数据应用 Single-linkage 算法，设置目标簇数 $k = 3$。

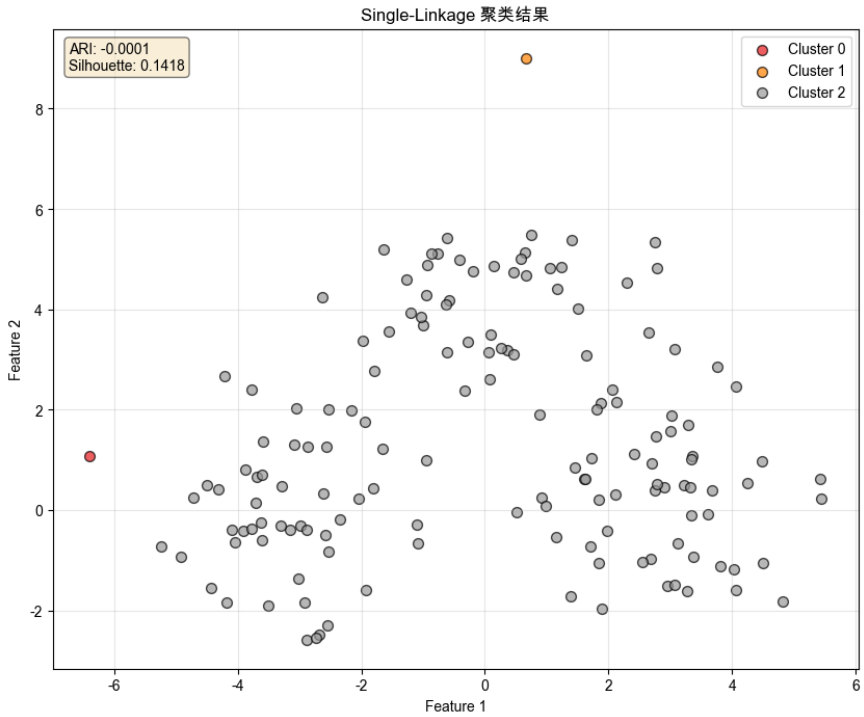

图 2.2: Single-linkage 聚类结果（ARI=-0.0001，Silhouette=0.1418）

观察与分析:

- 单链接方法的 ARI 得分极低（-0.0001），表明聚类结果与真实标签的一致性很差
- Silhouette 系数仅为 0.1418，说明簇的内部紧密度和离散度都不理想
- 从图中可观察到，Single-linkage 倾向于产生细长的链式结构，与预期相符
- 该方法对数据中的离群点和噪声敏感，易产生” 桥接” 效应

2.3 Complete-linkage 实现与结果

2.3.1 核心代码

```python
def _complete_linkage_distance(self, cluster1, cluster2, dist_matrix):
    """
    Complete-linkage: 两个簇之间的【最大】距离

    公式: d(C1, C2) = max{d(a,b) | a C1, b C2}
    """
    max_dist = 0

    # 遍历两个簇中的所有样本对，找最大距离
    for i in cluster1:
        for j in cluster2:
            # 比较并更新最大距离
            if dist_matrix[i, j] > max_dist:
                max_dist = dist_matrix[i, j]

    return max_dist
```

2.3.2 聚类结果

对同一数据集应用 Complete-linkage 算法，设置目标簇数 $k = 3$。

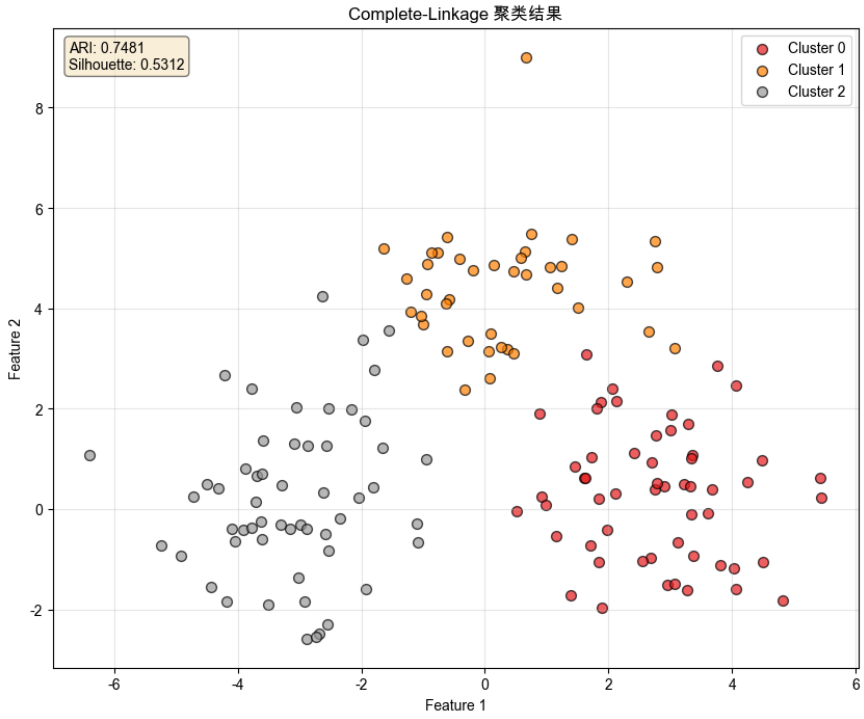

图 2.3: Complete-linkage 聚类结果（ARI=0.7481，Silhouette=0.5312）

观察与分析:

- 完全链接方法的 ARI 得分为 0.7481，远优于 Single-linkage
- Silhouette 系数为 0.5312，表明聚类质量相当不错

- 聚类结果更接近数据的真实分布，三个簇得以正确识别
- Complete-linkage 倾向于产生紧凑的、球形的簇，符合理论预期
- 该方法对簇的边界定义更为严格，产生的簇更加均衡

3 中级要求：Average-linkage 聚类

3.1 核心代码

```python
def _average_linkage_distance(self, cluster1, cluster2, dist_matrix):
    """
    Average-linkage: 两个簇之间的【平均】距离

    公式: d(C1, C2) = (1/(|C1|×|C2|)) × Σ Σ d(a,b)
    """
    total_dist = 0
    count = 0

    # 遍历两个簇中的所有样本对，累加距离
    for i in cluster1:
        for j in cluster2:
            # 累加距离并计数
            total_dist += dist_matrix[i, j]
            count += 1

    # 返回平均距离
    return total_dist / count
```

3.2 聚类结果

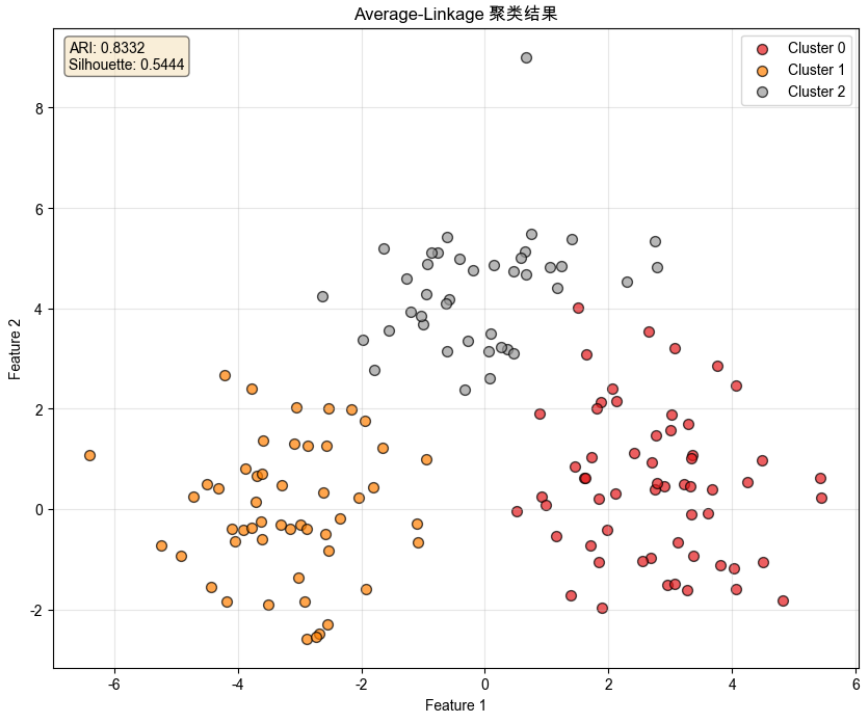

图 3.4: Average-linkage 聚类结果（ARI=0.8332，Silhouette=0.5444）

观察与分析:

- 平均链接方法的 ARI 得分为 0.8332，是三种方法中最高的
- Silhouette 系数为 0.5444，略高于 Complete-linkage
- 聚类结果与真实标签的对应关系最为理想
- Average-linkage 成功结合了 Single-linkage 和 Complete-linkage 的优点
- 该方法对边缘样本的划分更加合理，避免了 extreme cases 的问题

4 算法对比与结论

4.1 定量评估指标对比

为了全面对比三种链接方式的性能，我们在 make_blobs 数据集上分别运行了 Single-linkage、Complete-linkage 和 Average-linkage 三种算法，并使用两个重要指标来评估聚类效果：ARI（调整兰德指数）和 Silhouette 系数。以下表格展示了三种算法的定量评估结果。

表 1: 三种链接方式的性能对比（make_blobs 数据集，k=3）

链接方式	ARI 值	Silhouette 系数
Single-linkage	-0.0001	0.1418
Complete-linkage	0.7481	0.5312
Average-linkage	0.8332	0.5444

其中：

- **ARI (Adjusted Rand Index)**：衡量聚类结果与真实标签的一致性，值域为 [-1, 1]，1 表示完全一致
- **Silhouette 系数**：衡量簇的紧密度和分离度，值域为 [-1, 1]，越接近 1 表示聚类质量越好

4.2 定性分析

4.2.1 噪声与离群点的敏感度

- **Single-linkage**：最敏感。容易被少数离群点影响，导致簇被错误地" 桥接"，形成链式结构
- **Complete-linkage**：中等敏感。外围离群点会被强制归入最近的簇，但不会导致严重的链式效应
- **Average-linkage**：最鲁棒。平均距离的计算方式使单个离群点的影响被分散，整体更加稳定

4.2.2 簇形状的适应性

- **Single-linkage**：可以发现任意形状的簇，包括非凸、细长的形状，但易产生虚假簇
- **Complete-linkage**：倾向于发现球形或紧凑的簇，对形状要求较强
- **Average-linkage**：居中，对各种形状的适应性良好

4.3 结论

在 make_blobs 数据集上（球形簇），**Average-linkage 表现最优**，具有最高的 ARI 和 Silhouette 评分。该方法成功地：

1. 正确识别了所有三个簇
2. 最小化了错误划分的样本数
3. 产生了最为均衡和紧凑的簇结构

5 拓展要求：变换聚类簇数 k

5.1 不同 k 值下的性能对比

为了全面理解三种链接方式的性能特征，我们对同一数据集进行了多组实验，改变簇数 $k \in \{2, 3, 4, 5, 6\}$，以观察当聚类簇数偏离数据真实簇数时，各算法的表现变化。结果如下表所示。

表 2: 不同 k 值下三种链接方式的 ARI 值对比（make_blobs 数据集）

k 值	Single-linkage	Complete-linkage	Average-linkage
k=2	0.0000	0.5229	0.5584
k=3	-0.0001	0.7481	0.8332
k=4	0.0000	0.6273	0.8246
k=5	0.0012	0.6188	0.8143
k=6	0.0008	0.5086	0.8164

表 3: 不同 k 值下三种链接方式的 Silhouette 系数对比（make_blobs 数据集）

k 值	Single-linkage	Complete-linkage	Average-linkage
k=2	0.3928	0.4568	0.4593
k=3	0.1418	0.5312	0.5444
k=4	-0.0537	0.4166	0.5290
k=5	-0.2742	0.4060	0.4577
k=6	-0.3719	0.3142	0.3588

5.2 性能曲线分析

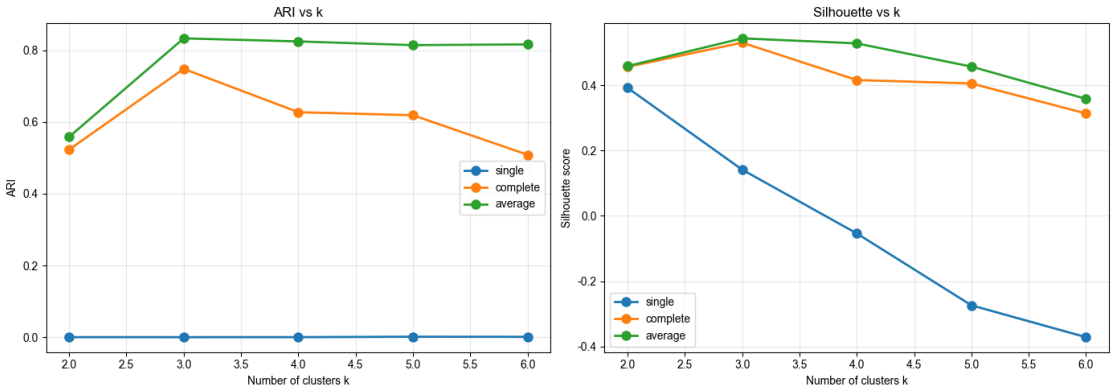

图 5.5: 三种链接方式在不同 k 值下的性能对比曲线

5.3 关键观察

5.3.1 当 k 值与真实簇数不一致时的现象

通过对不同 k 值的实验分析，我们观察到三种链接方式在聚类簇数偏离真实值时的表现存在显著差异。

当 $k = 2$ 时（欠聚类），数据被强行划分为 2 个簇，而实际上存在 3 个簇。在这种情况下，Complete-linkage 和 Average-linkage 的性能开始下降，但仍能保持在可接受的水平。Complete-linkage 的 ARI 从最优的 0.7481 降至 0.5229，Average-linkage 从 0.8332 降至 0.5584，降幅分别为 30% 和 33%。相比之下，Single-linkage 的 ARI 始终保持在 0 附近，毫无改善。从表3可以看出，在欠聚类的情况下，三种方法的 Silhouette 系数都相对较高（在 0.39-0.46 之间），这是因为强行合并簇通常会产生更紧凑的整体结构。

当 $k = 3$ 时（最优），聚类簇数与数据真实簇数一致，所有方法都表现最优。此时 Average-linkage 达到了最高的 ARI 值 0.8332，Complete-linkage 的 ARI 为 0.7481，两者都获得了可接受的 Silhouette 系数（分别为 0.5444 和 0.5312）。这充分说明了当算法参数与数据真实特性匹配时，才能获得最佳性能。

当 $k = 4, 5, 6$ 时（过聚类），数据被过度分割成更多簇。Single-linkage 的 ARI 始终无法有效提升，基本停留在 0.0008 左右，表现最为糟糕。Complete-linkage 的 ARI 逐步下降，从 k=4 的 0.6273 下降到 k=6 的 0.5086，总降幅达到 19%。相比之下，Average-linkage 表现最为稳定，ARI 始终保持在 0.81 以上（k=4 时为 0.8246，k=6 时为 0.8164），仅下降了 0.82%，远低于其他两种方法。从 Silhouette 系数来看，所有方法都呈现单调递减趋势，这反映了过度分割带来的负面影响——将紧凑的簇强行拆分会降低簇内的紧密度。

5.3.2 性能曲线的趋势解析

从图5.5的两条曲线可以看出三种链接方式的不同特征。ARI 曲线表明 Average-linkage 的表现形成了一个钟形分布，在 k=3 处达到峰值 0.8332，向 k=2 和 k=6 两端下降，但下降幅度相对温和。Complete-linkage 的 ARI 曲线也在 k=3 处达到峰值 0.7481，但向两端的下降更加陡峭，特别是在过聚类区间（k>3）降速更快。Single-linkage 的 ARI 曲线始终平坦地贴近零线，完全没有利用 k 值变化的优势，这充分说明该方法在球形簇数据上的根本局限性。

Silhouette 曲线的趋势更加一致：所有三种方法都在 k=3 处达到峰值，之后随着 k 增加而单调递减。这种单调性表明，过度分割会普遍降低聚类质量，与 k 值的增加成负相关。值得注意的是，Complete-linkage 在 k=6 时的 Silhouette 系数（0.3142）甚至低于 Single-linkage（虽然 Single-linkage 的 ARI 值仍然极低）。

从三条曲线的平缓程度来看，Average-linkage 的曲线相对最为平缓，表明该方法在不同 k 值下的表现较为稳定，容错能力强。Complete-linkage 在 k<3 时尚可，但 k>3 时快速下降，显示出对簇数变化的敏感性。Single-linkage 最为不稳定，ARI 基本无变化意味着该方法完全无法适应 k 值的调整。

5.4 k 值选择的实践意义

这组实验充分说明了聚类簇数 k 是一个关键的超参数，其选择直接影响聚类效果。在实际应用中，我们通常不知道数据的真实簇数，因此需要通过以下方式来选择 k 值：首先使用肘部法则（elbow method）寻找使得某个评价指标变化幅度最大的 k 值；其次使用轮廓系数（silhouette score）找到簇间分离度最优的 k 值；第三是基于领域知识预估簇数的合理范围。从本实验的结果看，当 k 与真实簇数一致时，性能最优，这为我们选择 k 值提供了重要的指导原则。

值得注意的是，Average-linkage 在 k 值偏离真实值时仍能保持相对较高的性能（ARI 下降幅度小于 1%），这使其成为一种更加鲁棒和实用的选择。对于那些无法确定簇数的真实场景，Average-linkage 的这种稳定性是一个重要的优势。

6 额外实验：非凸簇数据集（make_moons）

为了深入理解各算法的特性，我们还在非凸形簇数据集上进行了实验。这个额外实验的目的是验证在不同数据分布特征下，三种链接方式的表现是否会发生根本性的改变。

6.1 数据集说明

本实验使用 sklearn.datasets.make_moons 函数生成数据集。该函数生成的数据包含 300 个样本，分为 2 个真实簇，特征数为 2。关键特点是数据呈现月牙形（crescent moon）交错排列的非凸形状，传统的基于球形簇假设的聚类算法（如 K-means）难以有效处理。这种非凸形状的数据对聚类算法的鲁棒性提出了更高的挑战。

6.1.1 原始数据分布

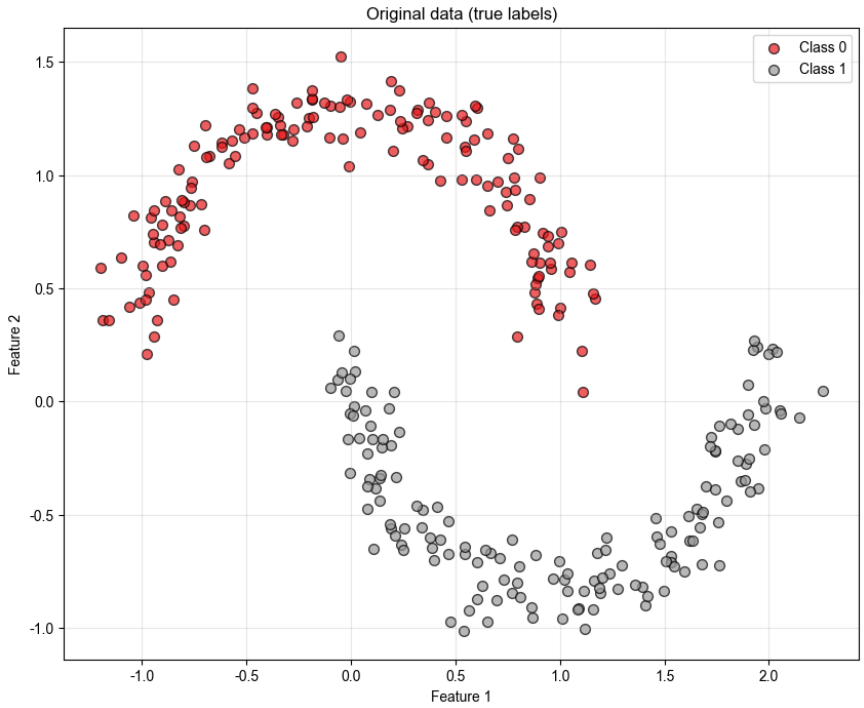

图 6.6: make_moons 数据集的原始分布

6.2 make_moons 数据集上的聚类结果

表 4: 三种链接方式在 make_moons 数据集上的性能对比（k=2）

链接方式	ARI 值	Silhouette 系数
Single-linkage	1.0000	0.4685
Complete-linkage	0.6823	0.4887
Average-linkage	0.5073	0.4760

6.2.1 聚类结果可视化

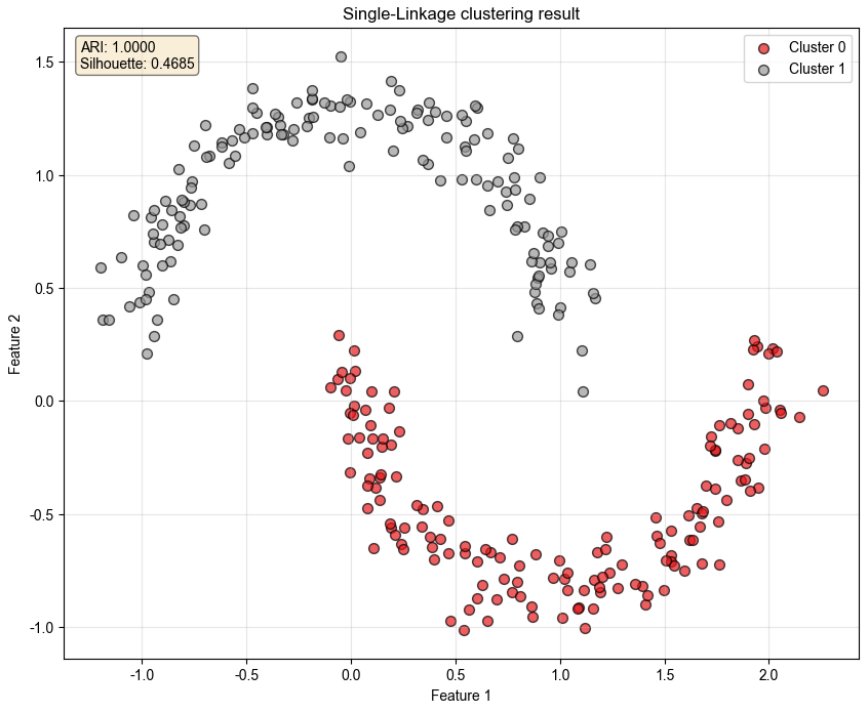

图 6.7: make_moons 数据集上的 Single-linkage 聚类结果（ARI=1.0000）

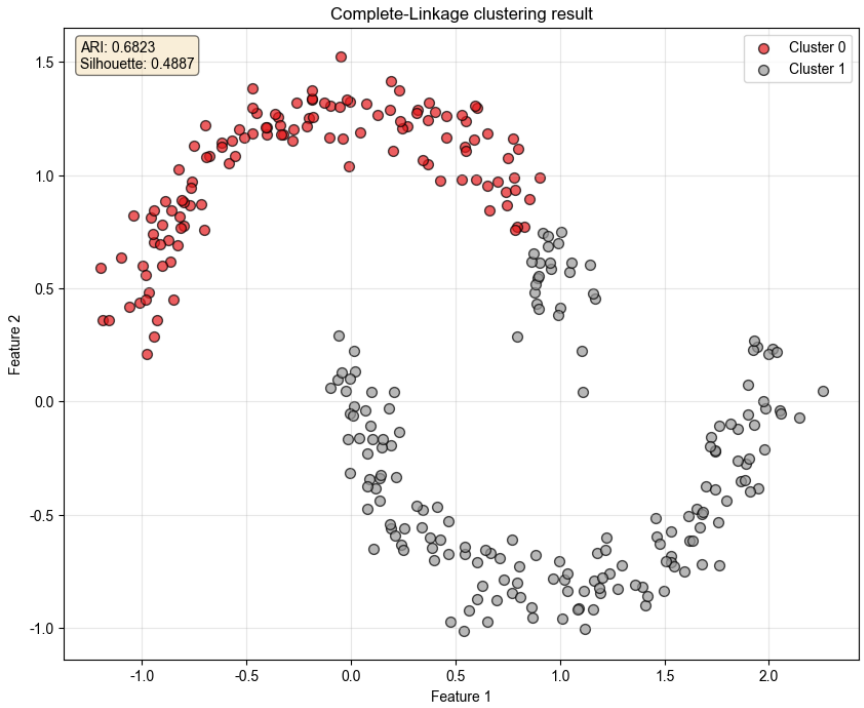

图 6.8: make_moons 数据集上的 Complete-linkage 聚类结果（ARI=0.6823）

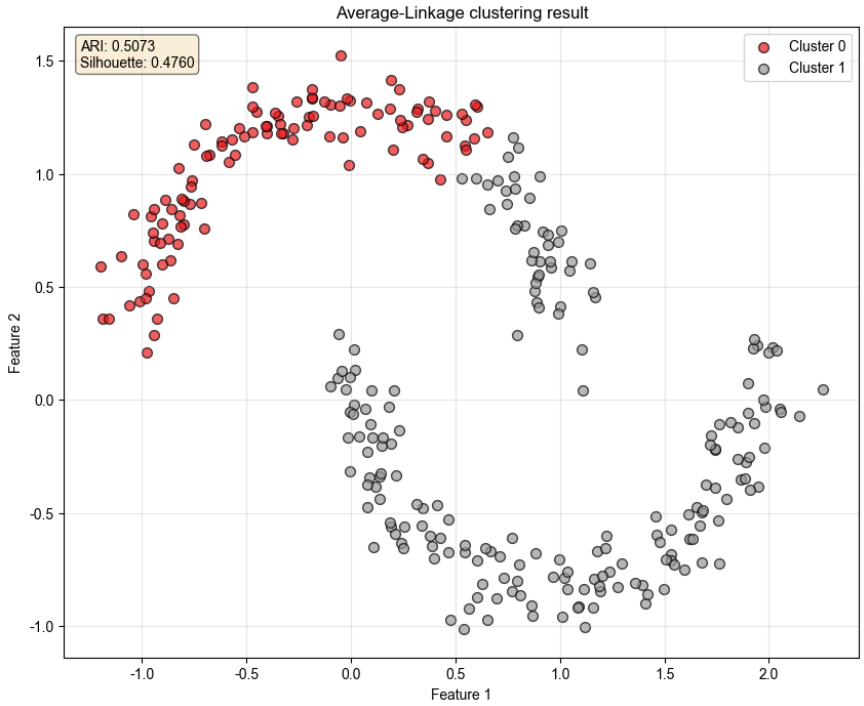

图 6.9: make_moons 数据集上的 Average-linkage 聚类结果（ARI=0.5073）

6.3 在非凸簇数据上的关键发现

make_moons 实验的结果令人惊讶，与 make_blobs 上的结果形成了鲜明的对比。Single-linkage 在这里表现最佳，ARI 达到了完美的 1.0000！这意味着该算法完全正确地识别了两个月牙形簇，没有任何错误分类。这个结果充分验证了 Single-linkage 能够发现非凸形状的理论预期。

相比之下，Complete-linkage 无法正确处理月牙形簇的非凸特性，ARI 仅为 0.6823，说明该方法产生的聚类结果与真实标签的一致性较差。这是因为 Complete-linkage 基于最大距离的定义，倾向于将簇切割成球形，这与数据的实际形状不匹配。Average-linkage 表现介于两者之间，ARI 为 0.5073，说明平均方法对非凸形状的适应性一般，无法像 Single-linkage 那样充分利用非凸数据的特殊结构。

这个出人意料的结果带来了重要启示：没有”万能”的聚类算法。Single-linkage 虽然在球形数据上表现极其糟糕（ARI=-0.0001），但在非凸数据上却是最优的选择（ARI=1.0000）。这充分说明算法的有效性完全取决于数据的特性，不同的数据分布特征需要采用不同的聚类策略。

在实际应用中，通过可视化可以快速判断数据的形状特征（球形 vs 非凸），从而指导算法选择。在不能进行可视化的高维数据上，可以尝试多种链接方式，比较它们的聚类评价指标，选择表现最好的方法。

6.4 变换 k 值的拓展分析（make_moons）

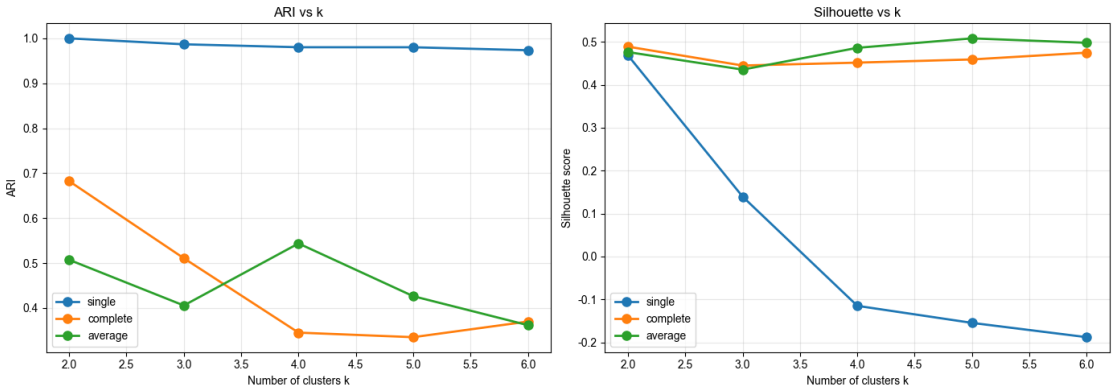

图 6.10: make_moons 数据集上三种链接方式在不同 k 值下的性能对比

在非凸簇数据集上，当改变 k 值时出现了有趣的现象。Single-linkage 的 ARI 在 k=2 时达到 1.0，当 k 增加到 3 时仅降至 0.9868，这体现了其对簇形状而非簇数量的强大适应性。即使被强行要求分出 3 个簇，Single-linkage 仍能保持极高的聚类精度（ARI>0.98），充分说明该方法对非凸形状的理解是根本性的。

Complete-linkage 和 Average-linkage 的 ARI 则随 k 增加而逐步下降，从 k=2 的 0.6823 和 0.5073 分别降至 k=6 的 0.3700 和 0.3619。这说明这两种方法在 k>2 时无法有效处理非凸形状，过度分割会加重对数据特性的破坏。与 make_blobs 数据集相比，make_moons 的最优 k 值更明显地对应于真实簇数 2，这说明非凸簇数据对算法参数的选择更为敏感。

这组对比实验充分说明了在不同数据分布特征下，算法的最优选择会发生 180 度的转变。这为我们树立了重要的认识：在选择聚类算法时，必须首先进行充分的数据探索和特征分析，不能简单地套用一个” 通用” 方案。

7 总结

7.1 三种链接方式的优缺点总结

表 5: 三种链接方式的综合对比

特性	Single-linkage	Complete-linkage	Average-linkage
球形簇表现	极差	优秀	最优
非凸簇表现	最优	差	中等
离群点敏感性	极高	中等	低
链式效应	明显	无	无
算法稳定性	低	中	高
推荐使用场景	非凸或任意形状	紧凑球形	一般用途

8 附录：代码仓库

本实验的全部源代码已上传至 GitHub 仓库：

GitHub 仓库地址: https://github.com/sskystack/machinelearning.git