Data Mining and Machine Learning

Guanxu Gleason WANG

University of Glasgow

版本：Past Paper of STAT5099 Data Mining and Machine Learning, UofG

更新：February 22, 2025

目录

1 k-Nearest Neighbour (k-NN) 分类

对于新的数据点，找到最近的 k 个点，并投票决定类别。

例 1.1 (2022-23 T1.(a)i.) 绘制此数据集上 1-nearest neighbour classifier 的决策边界, 假设使用欧几里得距离作为距离度量.

Obs.	X_1	X_2	Y
1	1	2.5	Triangle
2	2	2	Triangle
3	2.5	3	Triangle
4	3.5	1	Triangle
5	2	5	Circle
6	4	4	Circle
7	3	3	Circle
8	5	3	Circle

表 1: Two-dimensional dataset used in Question 1(a)

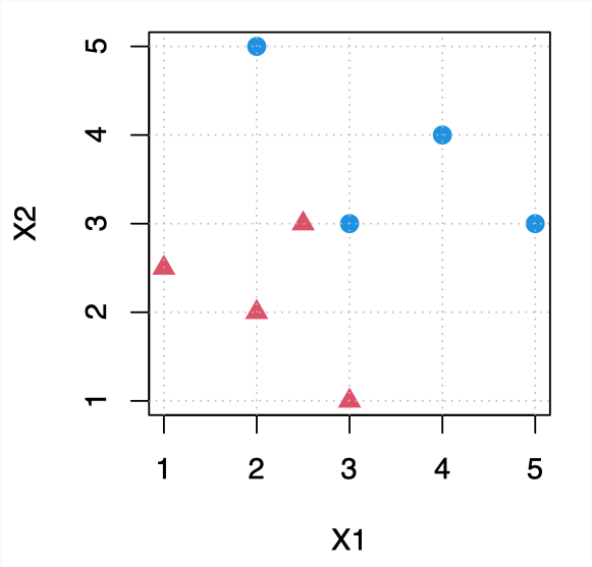


图 1: Scatterplot of the dataset

解. 1-nearest neighbour classifier: 利用最近的一个训练样本的类别来预测新数据点的类别.

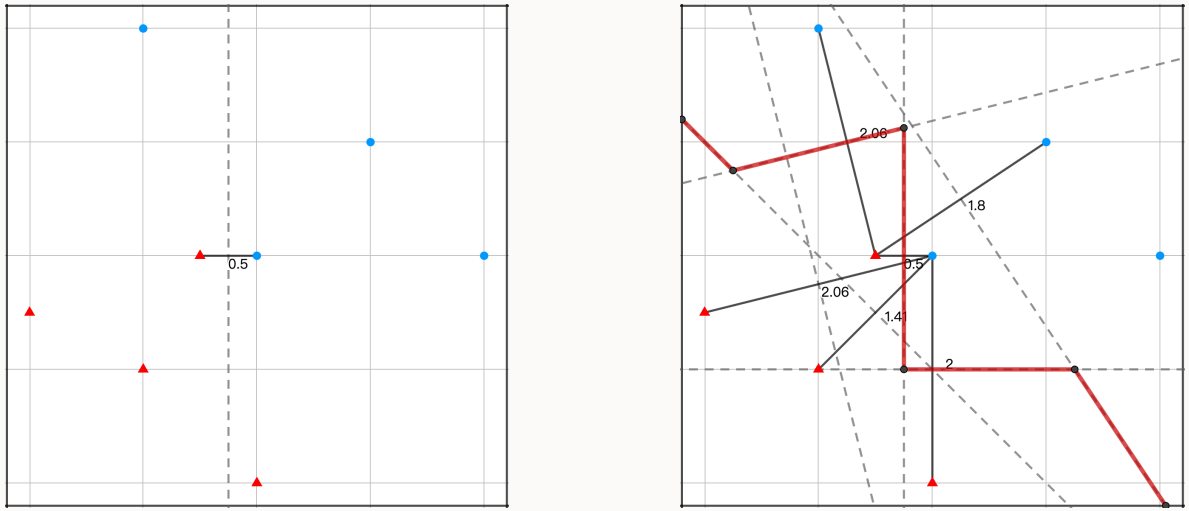
Euclidean distance (欧氏距离): 若两点 P, Q 坐标分别为 $(x_P, y_P), (x_Q, y_Q)$, 则 P, Q 两点的欧式距

离为:

$$d(P,Q) = \sqrt{(x_P - x_Q)^2 + (y_P - y_Q)^2}.$$

Step 1. 找到最近的两个不同分类点, 连线, 并画出其中垂线 (如图2a, 实线为最短两点连线, 虚线为中垂线). 若发现该中垂线无法将两个类别进行分割, 那么重复该步骤, 直到所有中垂线构成的边界可以将分类分割.

Step 2. 画出分割后的边界线 (如图2b红色折线).



(a) 例1.1 Step 1　　**(b)** 例1.1 Step 2

图 2: 例1.1解答图示

例 1.2 (2022-23 T1.(a)ii.) (续: 例1.1) 给定一个新的观测值 $X_{\text{new}} = (2, 3.9)$, 使用 1NN 为分类器, 假设使用曼哈顿距离作为距离度量, 预测其类别.

解. Manhanttan distance (曼哈顿距离): 若两点 P, Q 坐标分别为 $(x_P, y_P), (x_Q, y_Q)$, 则 P, Q 两点的曼哈顿距离为:

$$d(P, Q) = |x_P - x_Q| + |y_P - y_Q|.$$

不难发现, 两个离 $X_{\text{new}} = (2, 3.9)$ 最近的两个点分别为 Triangle: $X_3 = (2.5, 3)$, Circle: $X_5 = (2, 5)$. 他们的曼哈顿距离分别为:

$$\text{Triangle: } d(X_{\text{new}}, X_3) = |2 - 2.5| + |3.9 - 3| = 1.4,$$
$$\text{Circle: } d(X_{\text{new}}, X_5) = |2 - 2| + |3.9 - 5| = 1.1 < 1.4.$$

Hence the new observation should be assigned to class Circle.

例 1.3 Given a new observation $X_{\text{new}} = (X_{\text{new},1}, X_{\text{new},2}) = (3, 3)$, predict its class by using k-nearest neighbour as the classifier, assuming the **Manhattan distance** is used as the distance measure and $k = 3$.

Therefore, the 3NN are Obs. 4, 5 and 6. Using the majority vote, the predicted class is C.

因此, 3NN 是观测 4、5 和 6。使用多数投票, 预测的类别是 C。

例 1.4 When the value of k in the k-nearest neighbour classifier decreases from 3 to 1, how will this impact the bias and variance of the classifier? Explain why.

Obs.	X_1	X_2	Y	Manhattan distance
1	0	2	A	$\|3-0\|+\|3-2\|=4$
2	5	5	A	$\|3-5\|+\|3-5\|=4$
3	6	1	B	$\|3-6\|+\|3-1\|=5$
4	2	3	B	$\|3-2\|+\|3-3\|=\mathbf{1}$
5	4	4	C	$\|3-4\|+\|3-4\|=\mathbf{2}$
6	3	1	C	$\|3-3\|+\|3-1\|=\mathbf{2}$

解. As k decreases from 3 to 1, the bias is likely to decrease, but the variance is expected to increase. With a smaller k, the model becomes more complex, potentially fitting the training data more closely, thereby reducing bias. However, this increased flexibility may lead to higher variance, as the model becomes more sensitive to the noise in the training data, making it less generalisable to unseen data.

随着 k 从 3 减小到 1，偏差可能减小，但方差预计会增加。较小的 k 值会使模型变得更加复杂，可能更紧密地拟合训练数据，从而减少偏差。然而，这种增加的灵活性可能导致更高的方差，因为模型对训练数据中的噪声变得更加敏感，使其对未见数据的应用性降低。

2 Maximal Margin Classifier (最大间隔分类器)

最大间隔分类器通过在不同类别数据之间找到一条线性 决策边界, 使得边界与最近的数据点 (称为**支持向量 (support vector)**) 之间的间隔最大化.

例 2.1 (2022-23 T1.(a)iii.) 绘制最大间隔分类器的决策边界, 并圈出支持向量.

解.

显而易见, 最大间隔分类器的决策边界应该在如图2.1中红色直线所示的位置.
那么支持向量在图2.1中用粉色圆圈圈中的位置.

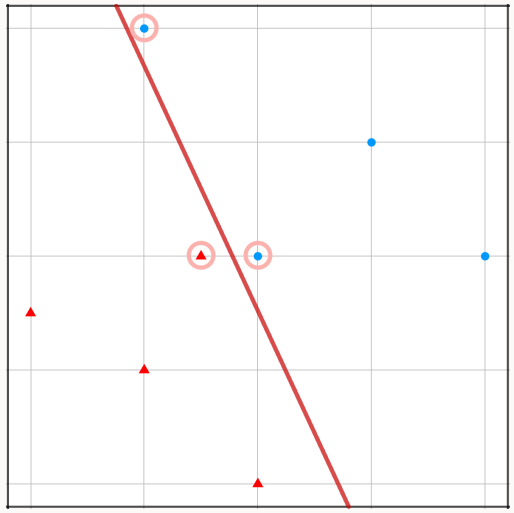


图 3: 最大间隔分类器和支持向量

Q1. 为什么不是图中间的两点? 因为如果用这两个点, 边界将变成二者连线的中垂线, 无法区分两个类别.

Q2. 为什么不用下面的三角形点? 如是, 边界无法将最上方蓝色圆形点正确分类.

Q3. 为什么不用这四个点? 在从 Q1 中的直线逆时针旋转的过程, 直到越过点 $(2,5)$ 才停止, 那么下面的三角形点并没有起到决策作用.

3 Soft-Margin Support Vector Classifier (软间隔支持向量分类器)

当数据集不是完全线性可分时, SVM 引入了软间隔, 通过允许一定的分类错误来提高模型的适应性.

软间隔引入了一个**松弛变量** ξ_i, 表示允许点违反间隔的程度. 那么优化目标将变为

$$\min \quad \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\xi_i,$$

其中 C 为**成本参数 (cost parameter)**, 用来平衡间隔大小和误分类惩罚.

当 C 很大时, 软间隔分类器会优先最小化误分类错误, 趋向于更严格的分类 (类似于最大间隔分类器). 因此, 在 $C \to \infty$ 时，加入软间隔的决策边界与最大间隔分类器的边界相同.

例 3.1 (2022-23 T1.(a)iv.) Now consider applying a **soft-margin support vector classifier** to this dataset **with a large positive cost parameter**. Will this classifier be the same as the maximal margin classifier in (iii)? If so, explain why; if not, state the difference in the decision boundary.

解. There are two acceptable answers for this question:

1. A large cost parameter means no violation of decision boundary and margin will be allowed. Therefore, the margin will remain the same/almost the same.
2. When the cost parameter is not large enough, the decision boundary will be less negative.

4 Support Vector Machine (SVM) (支持向量机)

- 如果数据集是线性可分的, 则可以通过硬间隔 SVM 实现 100% 分类准确率.
- 如果数据集是非线性可分的, 则

- 软间隔 SVM: 允许少量误分类，适合无法完全线性分离的情况.
- 核函数 (Kernel Function): 将数据映射到高维空间，在高维空间中找到线性可分的边界.

解. (2022-23 T1.(b)i.) Since the data are separable, it is possible to achieve 100% classification accuracy using a kernel SVM.

SVM with a **polynomial kernel** with degree of at least two and a positive coefficient, or an **RBF kernel** could perfectly classify the data.

Remark: It is important to specify the details of the polynomial kernel.

4.1 Radial Basis Function (RBF) Kernel (高斯核)

高斯核的数学公式为:

$$K(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|^2}{2\sigma^2}\right) \quad \Leftrightarrow \quad K(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2)$$

其中:

- x_i, x_j：输入数据点的特征向量。
- $\|x_i - x_j\|^2$：欧几里得距离的平方，表示两个点的相似度。
- σ：核宽度，控制高斯函数的“扩展范围”。
- $\gamma = \frac{1}{2\sigma^2}$：核参数，控制模型的灵活性。

作用： 将数据映射到无限维的特征空间，适合处理高度非线性的问题。

特点：

- 数据点之间的距离越近，核值 $K(x_i, x_j)$ 越接近 1.
- 数据点之间的距离越远，核值 $K(x_i, x_j)$ 越接近 0.

参数调整：

- 较大的 γ：模型更灵活，可能导致过拟合.
- 较小的 γ：模型更平滑，但可能欠拟合.

4.2 Polynomial Kernel (多项式核)

多项式核的数学公式为：

$$K(x_i, x_j) = (x_i \cdot x_j + c)^d$$

其中:

- x_i, x_j：输入数据点的特征向量。
- $x_i \cdot x_j$：内积，表示数据点之间的相似性。
- c (`coef`)：核中的自由参数，调整多项式函数的灵活性。
- d (`degree`)：多项式的次数，决定非线性映射的复杂度。

作用： 将数据映射到一个有限维的高阶多项式空间.

```
wine.svm <- svm(Type~., data=wine_1, type="C-classification", kernel="polynomial", degree=2, coef=0)
```

上述代码表述的为**二次多项式核**, 即 $K(x_i, x_j) = (x_i \cdot x_j)^2$.

The polynomial kernel with degree 1 is simply the linear kernel. Adding more degrees leads to a more flexible decision boundary. 一阶的多项式核就是线性核。增加更多阶数会导致决策边界更加灵活。

4.3 One-Versus-All (OVA) SVM

- 每个类别 i 训练一个 SVM 分类器，用于判断样本是否属于该类别。
- 对新样本，计算所有 SVM 的判别函数值，选择值最大的类别作为最终预测类别, 即

$$\widehat{y} = \arg\max_{g} \delta_g(x).$$

也就是说, SVM 的判别函数值最大的为预测类别.

5 Linear/Quadratic Discriminant Analysis (LDA/QDA) (线性/二次判别分析)

5.1 Linear Discriminant Analysis (LDA)

基本假设

- 数据的每个类别服从**多元正态分布**.
- 所有类别的协方差矩阵**相同**.
- 不同类别的均值向量 (μ_g) 可以不同.

目标函数

$$\delta_g(x) = x^T \Sigma^{-1} \mu_g - \frac{1}{2} \mu_g^T \Sigma^{-1} \mu_g + \ln(\pi_g)$$

其中:

- x: 输入特征向量.
- μ_g: 类别 g 的均值向量.
- Σ: 协方差矩阵（对所有类别相同）.
- π_g: 类别 g 的先验概率.

5.2 Quadratic Discriminant Analysis (QDA)

基本假设

- 数据的每个类别服从**多元正态分布**.

- 不同类别的协方差矩阵**可以不同**.

目标函数

$$\delta_g(x) = -\frac{1}{2}\ln|\Sigma_g| - \frac{1}{2}(x-\mu_g)^T\Sigma_g^{-1}(x-\mu_g) + \ln(\pi_g)$$

证明. (2022-23 T1.(b)iii.)

Assuming the mean vector, covariance matrix and prior probability of class g are given by μ_g, Σ_g and π_g respectively, and $g = 1, 2$.

$$\begin{aligned}
\widehat{g} &= \arg\max_g \Pr(G=g|x) \\
&\propto \arg\max_g \Pr(x|G=g)\Pr(G=g) && \text{Bayes theorem} \\
&= \arg\max_g \underbrace{\frac{1}{\sqrt{(2\pi)^d|\Sigma_g|}}\exp\left(-\frac{1}{2}(x-\mu_g)^T\Sigma_g^{-1}(x-\mu_g)\right)}_{\text{类别 } g \text{ 的样本服从多元高斯分布}} \underbrace{\pi_g}_{\text{prior probability}} && d=2 \\
&= \arg\max_g -\left(\log(2\pi) + \log|\Sigma_g|^{1/2}\right) - \frac{1}{2}(x-\mu_g)^T\Sigma_g^{-1}(x-\mu_g) + \log(\pi_g) && \text{take the log} \\
&= \arg\max_g -\frac{1}{2}\log|\Sigma_g| - \frac{1}{2}(x-\mu_g)^T\Sigma_g^{-1}(x-\mu_g) + \log(\pi_g) && \text{常数项不影响取值}
\end{aligned}$$

6 决策树

6.1 R 语言代码

```
tree1 <- rpart(Type ~ ., data = wine, control = rpart.control(minsplit = 2, cp = -1))
```

- 构建一个分类树 tree1.
- Type ~ . 表示目标变量为 Type，其余所有变量作为特征.
- data = wine 指定数据集为 wine.
- control = rpart.control(minsplit = 2, cp = -1):
  - minsplit = 2: 节点中至少有 2 个样本才能分裂.
  - cp = -1: 设置复杂度参数为负值, 这会强制构建出完全生长的树（即不提前剪枝）.

```
tree1.cp <- printcp(tree1)
```

- 显示决策树的复杂度参数表和交叉验证误差.
- 返回一个数据框，包含以下列:
  - CP: 复杂度参数 (Complexity Parameter).
  - nsplit: 树的分裂次数.
  - rel error: 相对误差.
  - xerror: 交叉验证误差.
  - xstd: 交叉验证误差的标准误差.

```r
cp <- tree1.cp[which.min(tree1.cp[, "xerror"]), "CP"]
```

- 提取最优剪枝参数 cp:
  - `which.min(tree1.cp[, "xerror"])` 找到交叉验证误差最小的行索引.
  - `tree1.cp[... , "CP"]` 返回对应的 CP 值.
- 最优 CP 用于剪枝以最小化模型的交叉验证误差.

```r
tree2 <- prune(tree1, cp)
```

- 使用提取的最优 `cp` 值对 `tree1` 进行剪枝。
- 剪枝后的树 `tree2` 保留了最优复杂度，避免了过拟合。

Listing 1: Summary

```r
tree1 <- rpart(Type ~ ., data = wine, control = rpart.control(minsplit = 2, cp = -1))
tree1.cp <- printcp(tree1)
cp <- tree1.cp[which.min(tree1.cp[, "xerror"]), "CP"]
tree2 <- prune(tree1, cp)
```

1. build a fully grown tree;
2. display the complexity parameter (CP) table;
3. select the optimal CP value from the tree with the minimum cross-validation error;
4. prune the tree using the optimal CP.

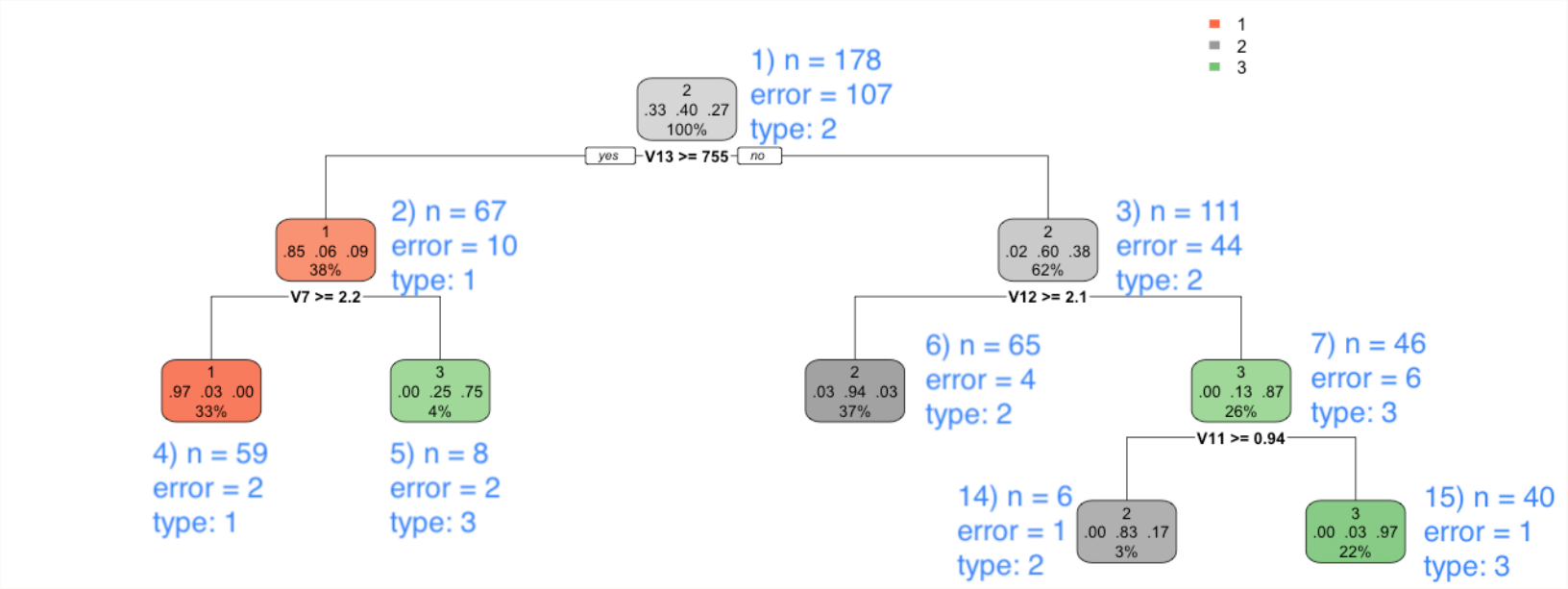
6.2 生成决策树

```
1) root 178 107 2 (0.33146067 0.39887640 0.26966292) # 样本数=178, 错误判别数=107, 分类类别=2
2) V13>=755 67 10 1 (0.85074627 0.05970149 0.08955224) # 括号里分别表示三个类别的分类概率
4) V7>=2.165 59 2 1 (0.96610169 0.03389831 0.00000000) * # "*"表示叶子结点
5) V7< 2.165 8 2 3 (0.00000000 0.25000000 0.75000000) *
3) V13< 755 111 44 2 (0.01801802 0.60360360 0.37837838)
6) V12>=2.115 65 4 2 (0.03076923 0.93846154 0.03076923) *
7) V12< 2.115 46 6 3 (0.00000000 0.13043478 0.86956522)
14) V11>=0.935 6 1 2 (0.00000000 0.83333333 0.16666667) *
15) V11< 0.935 40 1 3 (0.00000000 0.02500000 0.97500000) *
```

例 6.1 What would be your prediction, based on this tree, for the wine with Flavanoids (V7) of 3, Hue (V11) of 1, OD280/OD315 (V12) of 2 and Proline (V13) of 700? What is the probability of misclassification in this case?

由题可知: V7 = 3, V11 = 1, V12 = 2 and V13 = 700. 因此路径如下:

1. 由V13 = 700, 根节点选择 no, 向右到 node 3;
2. 由V12 = 2, 根节点选择 no, 向右到 node 7;
3. 由V11 = 1, 根节点选择 yes, 向左到 node 14. 在 node 14 中, 错判率为 $\frac{1}{6} = 0.1667$.



6.3 随机森林 (Random Forest)

1. repeatedly sampling with replacement from a sample;
2. fit a classification tree for each bootstrap sample by choosing the best split based on a random sample of features at each node;
3. predict new data using majority votes.

1. 从样本中重复有放回地抽样；
2. 通过在每个节点选择基于随机特征样本的最佳分割来为每个自助样本拟合分类树；
3. 使用多数投票预测新数据。

6.4 袋装树 (Bagging Tree)

袋装树与随机森林都是使用多个决策树进行预测, 但是袋装树的每个分裂节点使用**所有特征**, 而随机森林的每个分裂节点**随机选择部分特征**.

因此, 袋装树适用于减少过拟合，提高稳定性; 随机森林适用于高维数据，减少特征相关性.

Gini 指数定义为:

$$G = 1 - \sum_{i=1}^{k} p_i^2,$$

其中, p_i 指的是判别类别 i 的概率. 因此, 当数据中所有样本都属于同一类别时 (纯净), $G = 0$.

于是得到 Gini 指数的变化情况为

$$\Delta G = G_{\text{parent}} - (w_{\text{left}} G_{\text{left}} + w_{\text{right}} G_{\text{right}})$$

以及 Gini 指数的平均减少率为

$$\text{Average decrease in the Gini Index} = \frac{1}{T} \sum_{t=1}^{T} \sum_{\text{node } j} \Delta G_j.$$

直观解释为:

1. 如果某个特征在决策树中用于分裂时显著降低了基尼指数（即提升了数据的纯度），那么这个特征的重要性就高。
2. 如果某个特征很少被用于分裂，或者对数据纯度的影响很小，那么它的重要性就低。

The figure shows that V13 is the most important feature in bagging and V10 and V13 are the most important two features in random forest. V8 is the least important feature in both bagging and random forest.

图中显示，V13 是 bagging 中最重要的特征，V10 和 V13 是随机森林中最重要的两个特征。V8 是 bagging 和随机森林中最不重要的特征。

7 全连接神经网络 (Fully-Connected Neural Network, FCNN)

$$\phi_\theta(X_i) = \sigma_n(\cdots(\sigma_2(\sigma_1(X_i \cdot W_1 + b_1))W_2 + b_2)\cdots)W_{\text{out}} + b_{\text{out}}$$

其中, σ_i 为激活函数 (activation function). 神经网络参数的计算方法:

$$\begin{aligned}\#\text{参数} &= (\#\text{输入单元} \times \#\text{当前层神经元}(W_i)) + \#\text{当前层神经元}(b_i) \\ &= (\#\text{输入单元} + 1) \times \#\text{当前层神经元}\end{aligned}$$

例 7.1 The researcher then continued the analysis by fitting a fully-connected neural network to these data. ~~The input layer has 13 nodes.~~ The network has three hidden layers, where the first layer has 7 hidden nodes, the second layer has 5 hidden nodes, and the third layer has 3 hidden nodes. ~~The output layer has 3 nodes.~~ Calculate the number of parameters in this neural network.

解. 分层计算:

1. 输入层 (node=13) → 隐藏层 1 (node=7):
   - $\#$输入单元 $+ \#$偏置 $= 13 + 1 = 14$;
   - $\#$当前层参数 $= 14 \times 7 = 98$.
2. 隐藏层 1 (node=7) → 隐藏层 2 (node=5):
   - $\#$输入单元 $+ \#$偏置 $= 7 + 1 = 8$;
   - $\#$当前层参数 $= 8 \times 5 = 40$.
3. 隐藏层 2 (node=5) → 隐藏层 3 (node=3):
   - $\#$输入单元 $+ \#$偏置 $= 5 + 1 = 6$;
   - $\#$当前层参数 $= 6 \times 3 = 18$.
4. 隐藏层 3 (node=3) → 输出层 (node=3):
   - $\#$输入单元 $+ \#$偏置 $= 3 + 1 = 4$;
   - $\#$当前层参数 $= 4 \times 3 = 12$.

因此,

$$\#\text{参数} = 98 + 40 + 18 + 12 = 168.$$

在神经网络中，激活函数的主要作用是引入非线性，使神经网络能够学习复杂的模式和决策边界。

如果所有隐藏层都是线性函数，那么即使增加再多的层，整个神经网络仍然只是一个线性模型，本质上等价于单层线性回归或 Logistic 回归，无法学习复杂的非线性关系。

解. The activation function defines how the weighted sum of the input is transformed into an output for a node or nodes in a layer of the network. When the linear function is used in all hidden nodes, the network collapses to a network with no hidden layer at all and thus no longer a non-linear method.

激活函数定义了输入的加权和如何被转换成网络中某一层节点或节点的输出。当所有隐藏节点都使用线性函数时，网络会退化成没有任何隐藏层的网络，因此不再是非线性方法。

神经网络的训练过程中，如果训练集准确率 100%，但测试集准确率低于训练集，说明模型可能存在过拟合。过拟合的模型在训练数据上表现极佳，但在新数据上的泛化能力较差。要提高测试集准确率，需要降低过拟合，提高泛化能力，常见的方法包括：

1. 减少模型复杂度（减少隐藏层 (hidden layers) 或神经元 (hidden nodes) 数量）
2. 正则化（如 L1/L2 正则化、Dropout）
   - 原理：在每次训练过程中，随机丢弃一部分神经元（通常是 20%-50%）。
   - 作用：防止神经元过度依赖特定特征，提高模型的泛化能力。
3. 增加训练数据（数据增强）

4. 使用早停 (Early Stopping)
   - 原理：在训练过程中监控验证集误差，如果多次迭代误差没有下降，则停止训练。
   - 作用：避免模型在训练集上过度拟合，提高测试集表现。

8 主成分分析 (Principal Component Analysis, PCA)

The aim of PCA is to find a small number of uncorrelated linear combinations of the original variables which explain most of the variation in the data.

主成分分析（PCA）的目的是找到少数几个与原始变量不相关的线性组合，这些组合可以解释数据中的大部分变化。

This is achieved by finding a set of new axes, derived from the eigenvectors of the covariance or correlation matrix. The first axis lines up with the line/direction of most variability in our data, the second axis is orthogonal to the first new axis and follows the line of second most variability, the third is orthogonal to the first two new axes and follows the line of third most variability and so on.

这是通过找到一组新轴，这些轴来自协方差或相关矩阵的特征向量来实现的。第一条轴与数据中最大变异性所在的线/方向对齐，第二条轴与第一条新轴正交，并遵循第二大的变异性线，第三条轴与前两条新轴正交，并遵循第三大的变异性线，依此类推。

Check the variance of the variables. If the variables have very different variances, then it is advisable to

base the analysis on the sample correlation matrix.

检查变量的方差。如果变量具有非常不同的方差，那么基于样本相关矩阵进行分析是可取的。

例 8.1 下例说明不是所有数据都适配 PCA:

```
> cor(Concrete)
Cement Slag Fly ash Water SP Coarse aggr. Fine aggr.
Cement 1.000 -0.586 -0.279 0.178 0.317 -0.723 -0.338
Slag -0.586 1.000 -0.227 0.154 -0.154 0.526 -0.225
Fly ash -0.279 -0.227 1.000 -0.155 -0.307 -0.214 0.094
Water 0.178 0.154 -0.155 1.000 -0.540 -0.403 -0.396
SP 0.317 -0.154 -0.307 -0.540 1.000 0.106 0.160
Coarse aggr. -0.723 0.526 -0.214 -0.403 0.106 1.000 0.222
Fine aggr. -0.338 -0.225 0.094 -0.396 0.160 0.222 1.000
```

PCA might not reduce dimensionality very effectively as only Cement and Coarse aggregate shows a relatively strong correlation; the rest of correlation are quite weak.

主成分分析可能不会非常有效地降低维度，因为只有 Cement 和 Coarse aggr. 显示出相对较强的相关性 (abs>0.6)；其余的相关性都很弱。

累计方差比 (Cumulative Proportion of Variance)

$$\frac{\mathrm{Var}\left[X_1\right] + \mathrm{Var}\left[X_2\right] + \cdots + \mathrm{Var}\left[X_n\right]}{\#n}.$$

主成分得分 (Principal Score)

$$\sum_i \frac{x_i - \mu_i}{\sigma_i} \times \text{loading}_i$$

9 多维尺度分析 (Multidimensional Scaling, MDS)

9.1 应力函数 (Stress Function)

在 Metric MDS 中，映射的目标是最小化应力函数 (Stress Function)，即测量原始数据距离与映射后数据距离的差异。

Stress function of the usual metric MDS

$$\text{stress}_N(\mathbf{z}_1, \ldots, \mathbf{z}_n) = \left(\frac{\sum_{i<j} (\delta_{ij} - \|\mathbf{z}_i - \mathbf{z}_j\|)^2}{\sum_{i<j} \delta_{ij}^2} \right)^{1/2}$$

- 作用：度量降维后点之间的欧几里得距离 $\|\mathbf{z}_i - \mathbf{z}_j\|$ 与原始空间中的距离 δ_{ij} 之间的误差。
- 特点：均匀地考虑所有成对距离，不区分远距离和近距离。

Stress function of Sammon mapping

$$\text{stress}_{Sm}(\mathbf{z}_1, \ldots, \mathbf{z}_n) = \frac{1}{\sum_{l<k} \delta_{lk}} \sum_{i<j} \frac{(\delta_{ij} - \|\mathbf{z}_i - \mathbf{z}_j\|)^2}{\delta_{ij}}$$

- 作用：通过对小距离 δ_{ij} 赋予更大的权重，使得降维后局部结构尽可能保留。
- 特点：Sammon Stress 主要关注保留近邻关系，而不是整体数据结构。

The purpose of both stress functions are to find an optimal set of configuration points such that the (Euclidean) distance between configuration points can well approximate the dissimilarity information in the concrete dataset. Compared with the usual metric MDS, the stress function of Sammon mapping emphasises more on preserving smaller pairwise dissimilarities .

这两个应力函数的目的是找到一组最优的配置点，使得配置点之间的（欧几里得）距离能够很好地近似于具体数据集中的不相似信息。与通常的度量 MDS 相比，Sammon 映射的应力函数更强调保留较小的成对不相似性。

9.2 配置图 (Configuration Plot) 与 Shepard 图 (Shepard Diagram)

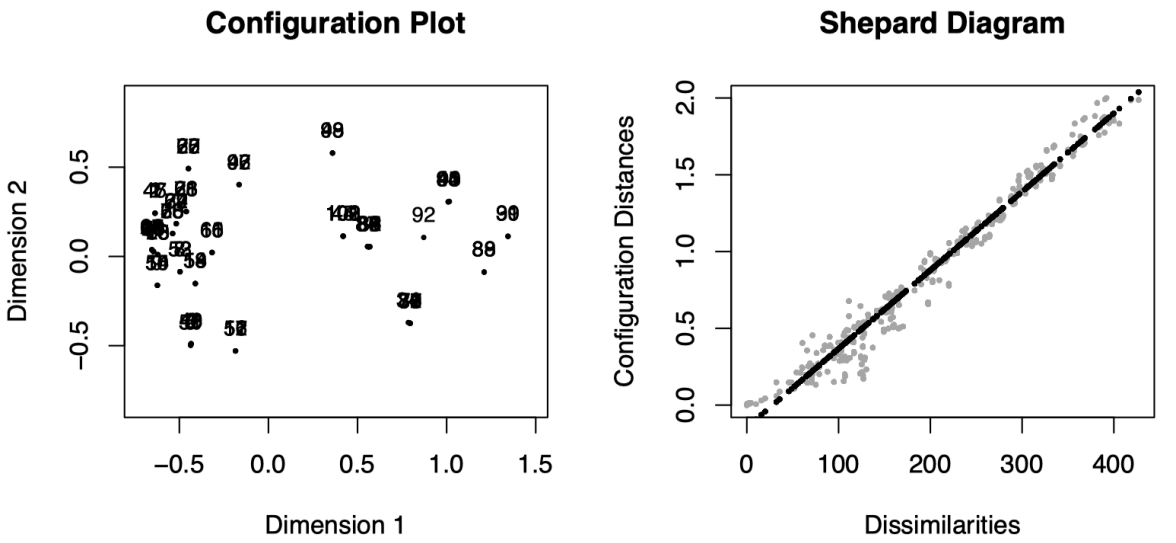
配置图 (Configuration Plot) MDS 的配置图是在低维空间 (通常是 2D 或 3D) 中可视化数据点的位置，目的是保留数据的相对距离关系。

- 如果数据点分布均匀，表示 MDS 很好地保留了原始空间的几何结构。
- 如果数据点之间的关系清晰（如形成簇或层级结构），表明 MDS 成功揭示了数据中的潜在模式。

Shepard 图 (Shepard Diagram) Shepard 图用于衡量 MDS 映射的质量，它显示了原始数据距离 d_{ij} 与 MDS 映射后的距离 $\widehat{d}_{ij}$ 之间的关系。

- 若点沿对角线 (y = x) 分布，说明 MDS 的映射效果很好，保持了数据的距离关系。
- 若点大范围偏离对角线，说明 MDS 没有很好地保留原始空间的结构，可能是维度不足或数据本身不适合 MDS。

例 9.1 配置图与 Shepard 图举例.



Comments about the observation points (including but not limiting to):

- Configuration points that are closer in the plot indicates higher similarity, and in some extreme cases, some observations are mapped to the same point in 2D.
  配置点在图中越接近，表示相似度越高，在某些极端情况下，一些观测值被映射到 2D 空间中的相同点。

- There may be two groups in the data.
  数据中可能有两组。

Comment about the performance of metric MDS:

Ideally, all points should adhere to a straight line in the Shepard diagram. As the data spreads a bit far, particularly for small dissimilarities, it indicates that embedding the data into 2D using metric MDS cannot fully capture the original distance information.

理想情况下，所有点都应遵循 Shepard 图中的直线。当数据分布得较远时，尤其是对于小差异，这表明使用度量多维尺度法将数据嵌入到二维中无法完全捕捉原始距离信息。

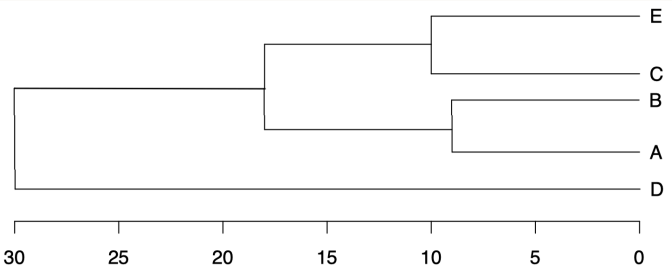
10 层次聚类 (hierarchical agglomerative clustering, HAC)

10.1 链接准则 (Linkage Criteria)

完全链接 (Complete Linkage) 计算两个簇之间所有样本点的**最大距离**:

$$d(A, B) = \max_{i \in A, j \in B} d(i, j)$$

- **生成的簇比较紧凑**（不易受到异常值影响）。
- 适用于聚类大小均匀的情况。
- 最牛逼的连接准则, 没有任何问题.



例 10.1 Draw the dendrogram to show the result of hierarchical agglomerative clustering using complete linkage. 绘制层次凝聚聚类树状图，使用完全链接。

	A	B	C	D	E
A	0	**9**	14	20	16
B	**9**	0	12	25	18
C	14	12	0	30	10
D	20	25	30	0	26
E	16	18	10	26	0

Iteration 1
Merge A and B at height 9

	A,B	C	D
C	14		
D	25	30	
E	18	**10**	26

Iteration 2
Merge C and E at height 10

	A,B	C,E
C,E	**18**	
D	25	30

- Iteration 4: merge A,B,C,E and D at height 30.

Iteration 3
Merge A,B and C,E at height 18

	A,B,C,E
D	30

单链接 (Single Linkage) 计算两个簇之间所有样本点的**最小距离**:

$$d(A, B) = \min_{i \in A, j \in B} d(i, j)$$

- 易受到链式效应 (chaining effect) 影响，即**会形成较长的、松散的簇**。
- 适用于数据间有连续变化的模式.

质心链接 (Centroid Linkage) 计算簇的质心 (centroid) 之间的距离:

$$d(A, B) = \|\mu_A - \mu_B\|$$

其中 μ_A 和 μ_B 是簇 A 和 B 的中心点。

- 适用于高维数据，但对异常值较敏感。
- 聚类之间的**间隔较大**，合并方式更平滑, 结构平稳, 但合并方式不同于完全链接.

- 若点沿对角线 (y = x) 分布，说明 MDS 的映射效果很好，保持了数据的距离关系。
- 若点大范围偏离对角线，说明 MDS 没有很好地保留原始空间的结构，可能是维度不足或数据本身不适合 MDS。

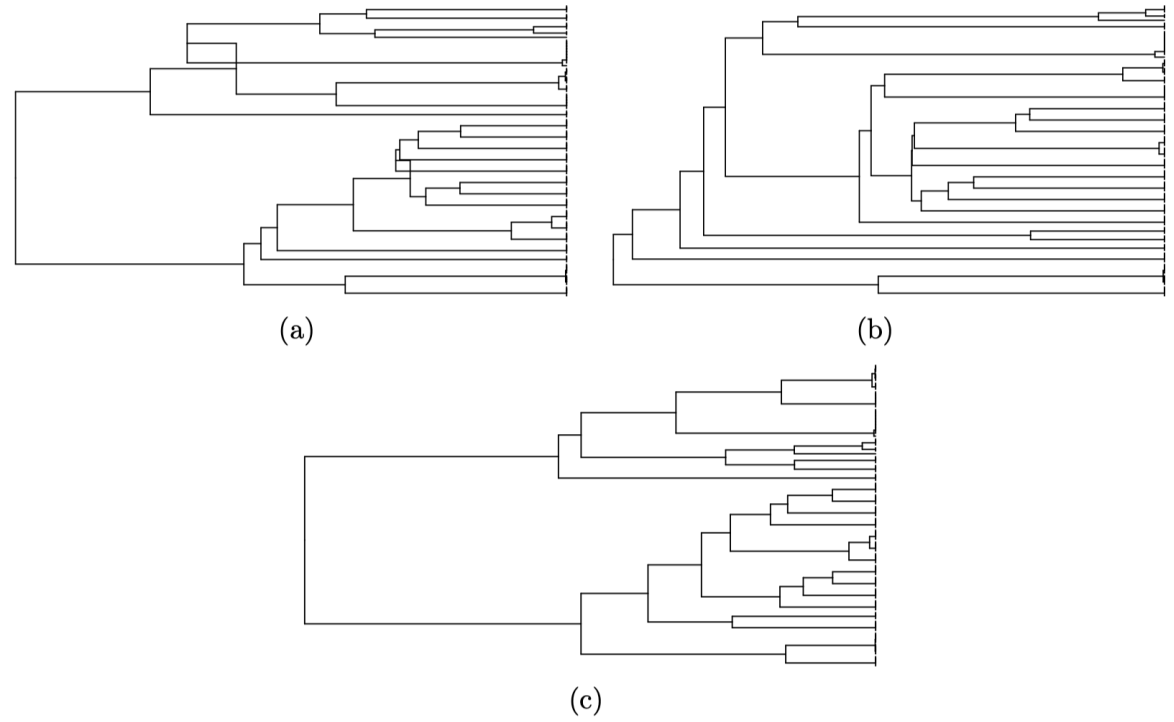
Limitation

- Single linkage suffers from chaining, where clusters may be merged together due to single elements being close to each other, even though many of the elements in each cluster may be very distant to each other.

单链接聚类存在链式效应，由于单个元素彼此靠近，即使每个簇中的许多元素彼此之间可能非常遥远，簇也可能被合并在一起。

- Centroid linkage suffers from the inversion issue, where similarity increases during the clustering.
  质心连接法存在反转问题，其中相似性在聚类过程中增加。

例 10.2 链接准则举例.



(a) Centroid linkage since it shows the inversion phenomenon[1].

[1]在树状图 (Dendrogram) 中，通常希望聚类距离应该是**单调递增**的，即在合并更多簇时，聚类之间的距离应该逐渐增加. 但是，如果某个聚类的合

质心连接，因为它显示了反转现象。

(b) Single linkage since it will face the chaining issue.

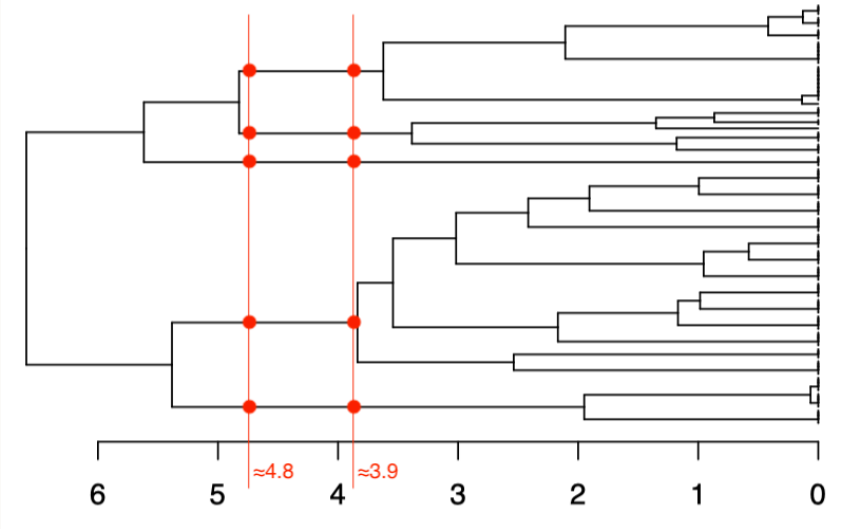
单链接，因为它将面临链式问题。

(c) Complete linkage does not have the above issues.

完全连锁不具有上述问题。

例 10.3 Decide at which height the dendrogram should be cut in order to generate five clusters.

Any value between 3.9 and 4.8 as any vertical line added in this range will cross the horizontal lines five times.



并距离反而小于之前的合并距离，就会出现 **inversion phenomenon**。

10.2 Silhouette 评分

通过比较不同簇的平均轮廓系数，可以评估哪个簇更紧密（紧密的簇应该有更高的 Silhouette 评分）。

$$k \text{ clusters } C_j$$

$$j : n_j \mid \text{ave}_{i \in C_j} s_i$$

- k clusters C_j: 分 k 个簇分别为 $C_1, \cdots, C_j, \cdots, C_k$.
- $j : n_j$: 第 j 个簇 (C_j) 内有 n 个样本 (数据点).
- $\text{ave}_{i \in C_j} s_i$: 该簇内所有数据点的平均轮廓系数, 即

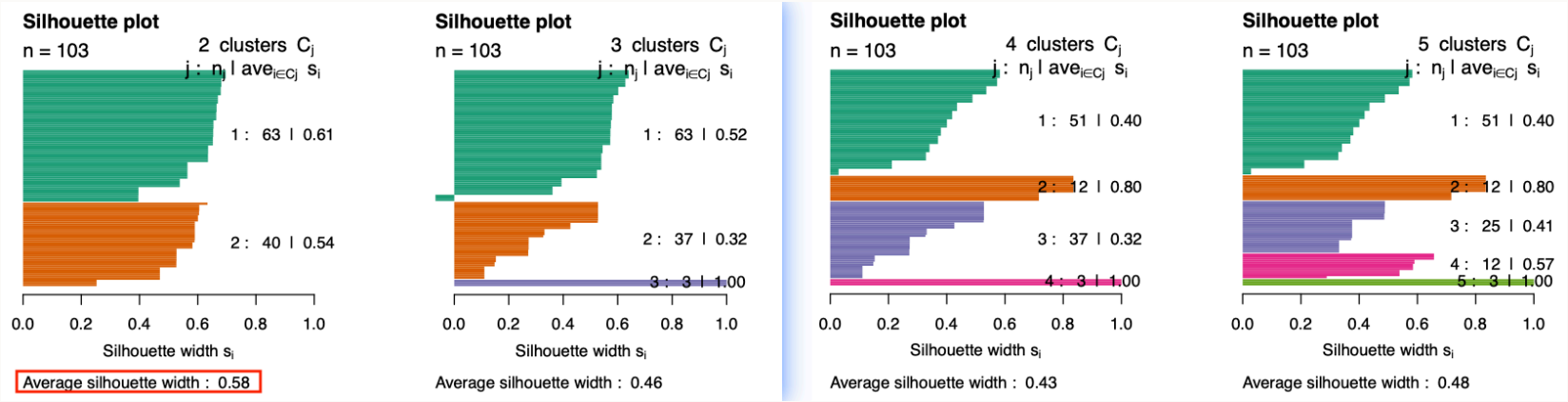
$$\frac{1}{n_j} \sum_{i \in C_j} s_i$$

例 10.4 Silhouette 评分举例.

Choose two clusters since **the average Silhouette width is the highest**.

There aren't any negative Silhouette score, suggesting no overlapping between clusters. However, the average Silhouette score is not very high, which may be a result of less compact clusters (large intra-cluster distances), less separable clusters (small inter-cluster distances), or both.

没有负 Silhouette 分数，表明聚类之间没有重叠。然而，平均 Silhouette 分数并不很高，这可能是由于聚类不够紧凑（较大的聚类内距离）、聚类区分度不高（较小的聚类间距离）或两者兼而有之的



结果。

替代办法 K-means or K-medroids. Both methods allow observations to move around different clusters, unlike permanently committed to a cluster in HAC.

K-means 和 K-medoids 这两种聚类方法允许样本点在不同的簇 (cluster) 之间移动，而不像层次聚类那样，一旦样本点被分配到某个簇，就无法再更改。

11 K-means 与 K-medroids

11.1 K-means

K-means 通过最小化簇内样本到簇中心的欧几里得距离，找到最优的簇划分。

算法步骤:

1. 初始化: 随机选择 K 个数据点作为初始簇中心.
2. 分配数据点: 计算每个数据点到所有簇中心的**欧几里得距离 (L2 Norm)**:

$$d(x_i, C_j) = ||x_i - C_j||^2 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

3. 更新簇中心: 计算每个簇的新中心，使用簇中所有点的均值:

$$C_j = \frac{1}{N_j} \sum_{i=1}^{N_j} x_i.$$

4. 重复步骤 2 和 3，直到簇中心不再发生变化（或达到最大迭代次数）。

11.2 K-medoids

K-medoids 是 K-means 的鲁棒版本，它选择数据点本身作为簇中心（medoid），并用**绝对距离**而非均值更新簇中心。

算法步骤:

1. 初始化: 随机选择 K 个数据点作为**簇中心 (medoids)**.
2. 分配数据点: 计算每个数据点到所有簇中心的**曼哈顿距离 (L1-norm)**:

$$d(x_i, C_j) = \sum |x_i - C_j| = |x_1 - x_2| + |y_1 - y_2|$$

3. 更新簇中心: 选择簇内数据点作为新的中心，使用簇内点到中心的距离总和最小:

$$C_j = \arg \min_{x \in \text{cluster}_j} \sum_{i=1}^{N_j} d(x_i, x)$$

4. 重复步骤 2 和 3，直到簇中心不再发生变化（或达到最大迭代次数）。

例 11.1 To start clustering, observation A is chosen to initialise cluster #1 and observation B is chosen to initialise cluster #2. Write down the cluster assignments immediately after this initialisation for observations C – F.

Based on the rule of assigning to the closest medoid,

A, C, D: cluster #1

B, E, F: cluster #2

After assigning observations to clusters in step (i), write down the new cluster centroid. 计算新的质心.

分别提取两个簇中的举例, 计算每个簇中所有点到簇内其他点的总距离.

	A	B	C	D	E	F
A	0.0	3.6	2.3	4.4	5.4	4.0
B	3.6	0.0	4.2	7.3	5.0	3.7
C	2.3	4.2	0.0	3.4	4.9	2.1
D	4.4	7.3	3.4	0.0	6.4	3.9
E	5.4	5.0	4.9	6.4	0.0	5.5
F	4.0	3.7	2.1	3.9	5.5	0.0

提取 A, B 两列
找到最小距离

	A (#1)	B (#2)	归类
A	**0.0**	3.6	#1
B	3.6	**0.0**	#2
C	**2.3**	4.2	#1
D	**4.4**	7.3	#1
E	5.4	**5.0**	#2
F	4.0	**3.7**	#2

	A	C	D	总距离
A	0.0	2.3	4.4	6.7
C	2.3	0.0	3.4	**5.7**
D	4.4	3.4	0.0	7.8

	B	E	F	总距离
B	0.0	5.0	3.7	**8.7**
E	5.0	0.0	5.5	10.5
F	3.7	5.5	0.0	9.2

$$\sum_{x\in\{A,C,D\}} d(A,x) = 2.3 + 4.4 = 6.7 \qquad \sum_{x\in\{B,E,F\}} d(B,x) = 5.0 + 3.7 = \mathbf{8.7}$$

$$\sum_{x\in\{A,C,D\}} d(C,x) = 2.3 + 3.4 = \mathbf{5.7} \qquad \sum_{x\in\{B,E,F\}} d(E,x) = 5.0 + 5.5 = 10.5$$

$$\sum_{x\in\{A,C,D\}} d(D,x) = 4.4 + 3.4 = 7.8 \qquad \sum_{x\in\{B,E,F\}} d(F,x) = 3.7 + 5.5 = 9.2$$

Based on the rule of choosing the cendroid with the lowest cost,

new centroid of cluster #1: C

new centroid of cluster #2: B

12 推荐系统 (Recommender Systems)

12.1 内容推荐系统 (Content-based) vs. 协同过滤 (Collaborative Filtering)

推荐系统主要分为：

1. 基于内容推荐 (Content-Based Recommendation):
   - 通过用户的历史偏好与物品的属性进行匹配来推荐新物品。
   - 例如，电影推荐系统会根据用户过去喜欢的电影的类型、导演等信息推荐新电影。
2. 协同过滤 (Collaborative Filtering):
   - **基于用户行为**而非物品内容。
   - 例如，如果两个用户的观影记录相似，则系统会推荐其中一人的喜爱电影给另一人。

同协同过滤相比, 内容推荐:

- Advantages:
  e.g. content-based recommender systems are less susceptible to the cold-start/new-item problem as in collaborative filtering; less susceptible to popularity bias.
  例如，基于内容的推荐系统相对于协同过滤来说，对冷启动/新项目问题不太敏感；对流行度偏差

也不太敏感。

- Limitations:

  e.g. content-based recommender systems might struggle to recommend items outside the user's established preferences.

  例如，基于内容的推荐系统可能难以向用户推荐超出其既定偏好的项目。

12.2 余弦相似度 (Cosine Similarity)

在基于内容的推荐系统中，我们通常使用余弦相似度 (Cosine Similarity) 来计算用户与产品的匹配程度:

$$\text{Cosine Similarity} = \frac{U \cdot P}{|U| \cdot |P|.}$$

例 12.1 下列给出了用户和产品的 ID 和特征向量:

User ID, Profile vector	Product ID, Profile vector
1, (1.7, 0.1, 5.2)	1, (6.9, 3.4, 3.7)
2, (2.1, 3.7, 0.6)	2, (8.3, 1.7, 0.1)
3, (0.1, 4.4, 2.1)	3, (0.8, 9.7, 4.8)
4, (1.1, 0.9, 3.6)	4, (1.2, 4.6, 5.3)

Based on this learned model, which product would you recommend most highly to User #2?

解.

$$U_2 \cdot P_1 = (2.1, 3.7, 0.6) \cdot (6.9, 3.4, 3.7) = 2.1 \times 6.9 + 3.7 \times 3.4 + 0.6 \times 3.7 = 29.29,$$
$$|U_2| = \sqrt{2.1^2 + 3.7^2 + 0.6^2} = 4.2965, \quad |P_1| = \sqrt{6.9^2 + 3.4^2 + 3.7^2} = 8.5358$$

Thus,

$$\cos\langle U_2, P_1\rangle = \frac{29.29}{4.2965 \times 8.5358} = 0.7987.$$

Similarly,

$$\cos\langle U_2, P_2\rangle = 0.6532, \quad \cos\langle U_2, P_3\rangle = \mathbf{0.8675}, \quad \cos\langle U_2, P_4\rangle = 0.7427.$$

Since product 3 has the largest cosine similarity, it is recommended to the user.