一　简答题

1. 数据挖掘定义

从大量的、不完全的、有噪声的、模糊的数据中，提取隐含在其中的、人们事先不知道但是有用的信息和知识的过程。

2. 数据挖掘任务

(1) 分为描述性分析和预测性分析

(2) 预测型任务：分类、回归、离群点检测

(3) 描述型任务：聚类分析、关联分析、演化分析、序列模式挖掘

3. 数据挖掘流程

(1) 数据收集

(2) 数据预处理：数据清洗、数据集成、数据选择、数据转换

(3) 数据挖掘、模式评估、知识表示

4. 分类和聚类的区别

(1) 聚类是无指导的观察式学习，没有预先定义的类，数据集无类标号

(2) 分类是有指导的示例式学习，有预先定义的类，数据集有类标号

5. 分类和回归的区别

(1) 分类预测的输出为离散或标称的属性值

(2) 回归预测的输出为连续的属性值

6. 各数据挖掘任务的现实应用举例

7. 各章的算法举例

二　数据预处理

1. 频率、众数、百分位数、四分位数、均值、中列数、中位数、截断均值

2. 缺失值的处理方法

(1) 忽略元组（行）、忽略属性列

(2) 数据填充：全局常量、均值或众数、预测缺失值

3. 噪声数据的平滑方法

(1) 分箱：等深或等宽

(2) 回归拟合

(3) 离群点分析，剔除离群点

4. 分箱

(1) 等深（等频）：每个箱容量相等

(2) 等宽：每个箱区间范围相同

(3) 均值光滑：箱中的每个值被替换成箱中属性的均值

(4) 边界光滑：箱中的每个值被替换为最近的边界值

5. 数据规范化

(1) 最大-最小规范化

规范化到$[a,b]$：$\frac{x-\min}{\max-\min}\times(b-a)+a$

(2) z-score 规范化

① 计算均值EX、标准差σ

② $\sigma=\sqrt{\frac{1}{n}\sum(x-EX)^2}$，差方和的均值，开方

③ $z=\frac{x-EX}{\sigma}$

(3) 小数定标规范化

除以同一个 10 的次方，使得最大值的绝对值≤1

6. 相似性度量

(1) 闵可夫斯基、曼哈顿、欧式、切比雪夫

(2) 余弦相似度

$\cos(p,q)=\frac{p\bullet q}{\|p\|\times\|q\|}$，越大越相似

(3) 相关系数

$corr(p,q)=\frac{\sum(p_i-\overline{p})\times(q_i-\overline{q})}{\sqrt{\sum(p_i-\overline{p})^2}\times\sqrt{\sum(q_i-\overline{q})^2}}$，先减均值，再算余弦

(4) 简单匹配系数 SMC

$SMC = \frac{n_{11} + n_{00}}{n_{11} + n_{00} + n_{01} + n_{10}}$，相同的÷全部

(5) 杰卡德系数 Jaccard

$J = \frac{n_{11}}{n_{11} + n_{01} + n_{10}}$，剔除掉 00 后的 SMC

三　分类

1. 信息熵

(1) 划分前：$Entropy(S) = -\sum p_i \log_2 p_i$

(2) 划分后：$Entropy_A(S) = \sum \frac{|S_i|}{|S|} Entropy(S_i)$，按比例加权

(3) 信息增益：$Gain(S, A) = Entropy(S) - Entropy_A(S)$，前－后

(4) 分裂信息：$SplitE(S, A) = -\sum \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$

(5) 信息增益率：$GainRatio(S, A) = \frac{Gain(S, A)}{SplitE(S, A)}$

2. Gini 系数

(1) Gini 系数：$G(S) = 1 - \sum p_i^2$

(2) 差异性损失：$\Delta G(S, A) = G(S) - \left[\frac{|S_L|}{|S|} G(S_L) + \frac{|S_R|}{|S|} G(S_R) \right]$

3. ID3 算法建立决策树

(1) 使用信息增益，只能处理离散型数据

(2) 每次选择信息增益最大的属性进行划分

(3) 伪代码

```
函数：DT(S,F)
输入：训练集数据S，训练集数据属性集合F
输出：ID3决策树
(1)if 样本S全部属于同一个类别C then
(2)    创建一个叶结点，并标记类标号为C；
(3)    return；
(4)else
(5)    计算属性集F中每一个属性的信息增益，假定增益值最大的属性为A；
(6)    创建结点，取属性A为该结点的决策属性；
(7)    for 结点属性A的每个可能的取值V  do
(8)        为该结点添加一个新的分支，假设SV为属性A取值为V的样本子集；
(9)        if 样本SV全部属于同一个类别C then
(10)            为该分支添加一个叶结点，并标记类标号为C；
(11)       else
(12)            递归调用DT(SV, F-{A})，为该分支创建子树；
(13)       end if
(11)    end for
(12)end if
```

4. C4.5 算法建立决策树

(1) 使用信息增益率，作为属性选择标准

(2) 处理缺失数据

① 信息增益 Gain(S, A) 乘属性 A 不空值的比率

② 分裂信息 SplitE(S, A) 计算时，将缺失值也看作一个类别

(3) 处理连续型数据

① 先升序排序，将每对相邻值的中点看作可能的分裂点

② 选择划分后熵最小的作为最佳分裂点

③ $$Entropy_A(S)=\frac{|S_L|}{|S|}Entropy(S_L)+\frac{|S_R|}{|S|}Entropy(S_R)$$

(4) 伪代码

```
函数名：CDT(S,F)
输入：训练集数据S，训练集数据属性集合F
输出：一棵未剪枝的C4.5决策树
(1)if 样本S全部属于同一个类别C then
(2)    创建一个叶结点，并标记类标号为C；
(3)    return；
(4)else
(5)    计算属性集F中每一个属性的信息增益率，假定增益率值最大的属性为A；
(6)    创建结点，取属性A为该结点的决策属性；
(7)    for 结点属性A的每个可能的取值V  do
(8)        为该结点添加一个新的分支，假设SV为属性A取值为V的样本子集；
(9)        if 样本SV全部属于同一个类别C then
(10)           为该分支添加一个叶结点，并标记为类标号为C；
(11)       else
(12)           则递归调用CDT(SV ,F-{A})，为该分支创建子树；
(13)       end if
(14)   end for
(15)end if
```

5. CART 算法建立决策树

(1) 使用 Gini 系数的差异性损失作为属性选择标准

(2) 只能建立二叉树，当有多个取值或是连续型属性时，取差异性损失最大的作为划分点

(3) 伪代码类似 ID3 和 C4.5，将选择标准替换成差异性损失

6. 朴素贝叶斯分类算法

(1) 贝叶斯定理： $P(C_i \mid X) = \dfrac{P(X \mid C_i)P(C_i)}{P(X)}$

(2) 由于 $P(X)$ 对所有类别为常数，可以忽略不计算

(3) $P(X \mid C_i) = P(X_1 \mid C_i) \times P(X_2 \mid C_i) \times \cdots \times P(X_n \mid C_i)$

(4) Laplace 估计： $P(X_i \mid C_j) = \dfrac{n_C + l \times p}{n + l}$，避免了条件概率为 0 的结果

(5) 等价样本大小 l 和先验概率 p 由题目给定，或可取 $p = \dfrac{1}{l}$

(6) 伪代码

函数名：NaiveBayes

输入：类标号未知的样本X={x1,x2,…,xn}

输出：未知样本X所属类别号

(1) for j=1 to m

(2) 计算X属于每一个类别 C_j 的概率：

$$P(X \mid C_j) = \prod_{k=1}^{n} P(X_k \mid C_j) = P(X_1 \mid C_j) \times P(X_2 \mid C_j) \times \ldots \times P(X_n \mid C_j)$$

(3) 计算训练集中每个类别 C_j 的概率 $P(C_j)$；

(4) 计算概率值 $P = P(X \mid C_j) \times P(C_j)$；

(5) end for

(6) 选择计算概率值 P 最大的 C_j ($1 \le j \le m$) 作为类别输出。

---

7. k-最近邻分类算法

(1) 通过 k 个样本的类标号投票得出未知样本的类别

(2) k 应选择奇数：避免出现平局

(3) k 应至少为 c+1：保证某类有一个相对多数，c 为类别数

(4) 伪代码

算法名：KNN

输入：最近邻数目K ，训练集D，测试集Z

输出：对测试集Z中所有测试样本预测其类标号值

(1)for 每个测试样本 $z=(x',y')\in Z$ do

(2) 计算 z 和每个训练样本 $(x,y)\in D$ 之间的距离 $d(x',x)$

(3) 选择离 z 最近的 k-最近邻集合 $D_z\subseteq D$

(4) 返回 D_z 中样本的多数类的类标号

(5)end for

8. 集成分类方法

(1) 将多个分类方法聚集在一起，来提高分类准确率和模型稳定性

(2) 以投票策略集成多个基分类器的预测结果。伪代码见 P107。

(3) 装袋（自助投票）：根据均匀概率分布从数据中重复抽样（有放回）得到自助样本集。在每个抽样生成的自主样本集上训练一个基分类器，对训练过的 k 个基分类器投票，将测试样本指派到得票最高的类。伪代码见 P108。

(4) 提升：自适应地改变训练样本的分布，使得基分类器聚焦在那些难分类的样本上。开始时，所有样本都赋予相同的权值；每轮提升结束后，更新训练集样本的权值：增加被错误分类的样本的权值，减少被正确分类的样本的权值。

四 聚类

1. k-means 聚类算法

(1) 随机选择 k 个对象，每个对象代表一个簇的初始均值或中心；对剩余的每个对象，根据其与各簇中心的距离，指派到最近的簇，然后计算每个簇的新均值，得到更新后的簇中心；不断重复，直到簇中心不再变化。

(2) 通过误差平方和 SSE 作为度量聚类质量的目标函数。

(3) 伪代码

```
算法：k-means聚类
输入：数据集D，划分的簇个数k
输出：k个簇的集合
(1)从数据集D中任意选择k个对象作为初始簇中心；
(2) repeat
(3)     for 数据集D中每个对象P do
(4)         计算对象P到k个簇中心的距离
(5)         将对象P指派到与其最近的簇；
(6)     end for
(7)     计算每个簇中对象的均值，做为新的簇的中心；
(8) until  k个簇的簇中心不再发生变化
```

2. 距离定义

(1) 两个对象之间的距离

① 分类属性或二值属性：$\mathrm{dif}(p_i,q_i)=\begin{cases}1, & p_i\neq q_i\\ 0, & p_i=q_i\end{cases}$

② 连续属性或顺序属性：$\mathrm{dif}(p_i,q_i)=\left|p_i-q_i\right|$

(2) 对象和簇摘要之间的距离

① 分类属性：$\mathrm{dif}(p_i,C_i)=1-\dfrac{\mathrm{freq}_{C|D_i}(p_i)}{|C|}$

② 其中 freq 为 p_i 在簇摘要 C 的 D_i 属性分量中出现的频次

③ 数值属性：$\mathrm{dif}(p_i,C_i)=\left|p_i-C_i\right|$

④ 簇摘要 CSI 写法：C={8; red:3,green:4,blue:1; 24.5; 38.7; yes:7,no:1}

(3) 簇和簇之间的距离

① 分类属性：$\mathrm{dif}(C_i^{(1)},C_i^{(2)})=1-\dfrac{1}{|C_1|\times|C_2|}\sum_{p_i\in C_1}\mathrm{freq}_{C_1|D_i}(p_i)\times\mathrm{freq}_{C_2|D_i}(p_i)$

② 即 1 — 两个簇间共有类别的数量乘积 ÷ 总数量乘积，书 P146

3. k-summary 聚类算法

(1) 和 k-means 一样，但可处理混合属性数据集。用摘要距离替换对象距离。

(2) 伪代码

```
算法：k-summary聚类
输入：数据集D，划分簇的个数k
输出：k个簇的集合
(1)从数据集D中任意选择k个对象，并创建k个簇的摘要信息CSI；
(2) repeat
(3)     for 数据集D中每个对象P do
(4)         计算对象P到k个簇中心的距离；
(5)         将对象P指派到与其最近的簇；
(6)     end for
(7)     更新簇的摘要信息CSI；
(8) until  k个簇的摘要信息不再发生变化
```

4. 一趟聚类算法

(1) 基于最小距离原则的聚类算法 CABMDP

(2) 算法流程

① 初始时，簇集合为空，读入一个新的对象；

② 以这个对象构造一个新的簇；

③ 若已到数据库末尾，则转⑥，否则读入新对象，利用给定的距离定义，计算它与每个已有簇间的距离，并选择最小的距离；

④ 若最小距离超过给定的半径阈值 r，转②；

⑤ 否则将该对象并入具有最小距离的簇中并更新该簇的各分类属性值的统计频度及数值属性的质心，转③；

⑥ 结束。

(3) 逐个顺序读入，都超过阈值 r 则新开簇，否则并入最近簇，更新簇摘要

5. 基于密度的聚类算法 DBSCAN

(1) 可以识别具有任意形状和不同大小的簇，自动确定簇的数目

(2) 核心点、边界点、噪声点

(3) 对象半径 Eps 邻域

(4) 邻域包含的点的最小数目 MinPts（包括自身），用于确定是否核心对象

(5) 顺序扫描所有点，检查 Eps 邻域包含的点数是否达到 MinPts 个，达到则建立新簇，加入所有邻域点，并尝试继续扩展；其中邻域点达到 MinPts 个的点为核心点，其余为边界点。不在任何核心点邻域内的点为噪声点。

(6) 伪代码

```
算法：DBSCAN
输入：数据集D，参数MinPts和Eps
输出：簇集合
(1) 将数据集D中的所有对象标记为未处理状态
(2) for 数据集D中每个对象p do
(3)   if  p已经归入某个簇或标记为噪声 then
(4)      continue;
(5)   else
(6)      检查对象p的Eps邻域 N_Eps(p) ;
(7)      if  N_Eps(p) 包含的对象数小于MinPts then
(8)            标记对象p为边界点或噪声点;
(9)      else
(10)           标记对象p为核心点，并建立新簇C;
(11)           for  N_Eps(p) 中所有尚未被处理的对象q  do
(12)               检查其Eps邻域,若包含至少MinPts个对象,则将其中未归入任何一个簇的对象加入C;
(13)           end for
(14)     end if
(15)   end if
(16) end for
```

五　关联分析

1. 支持度、置信度、提升度

(1) 支持度 $\sup(\mathrm{A},\mathrm{B})=\dfrac{\sigma(A\bigcap B)}{N}=P(A\bigcap B)$，找频繁项集

(2) 置信度 $\mathrm{conf}(\mathrm{A},\mathrm{B})=\dfrac{\sigma(A\bigcap B)}{\sigma(A)}=P(B\mid A)$，找强关联规则

(3) 提升度 $\mathrm{lift}(\mathrm{A},\mathrm{B})=\dfrac{\mathrm{conf}(\mathrm{A},\mathrm{B})}{\sup(\mathrm{B})}=\dfrac{P(A\bigcap B)}{P(A)\times P(B)}$

(4) 记法：$\forall \mathrm{X}\in \text{transaction}, \text{buys}(\mathrm{X},\mathrm{A})\wedge\text{buys}(\mathrm{X},\mathrm{B})\Rightarrow\text{buys}(\mathrm{X},\mathrm{C})$　[sup, conf]

2. Apriori 算法

(1) 算法流程

① 设定 k=1

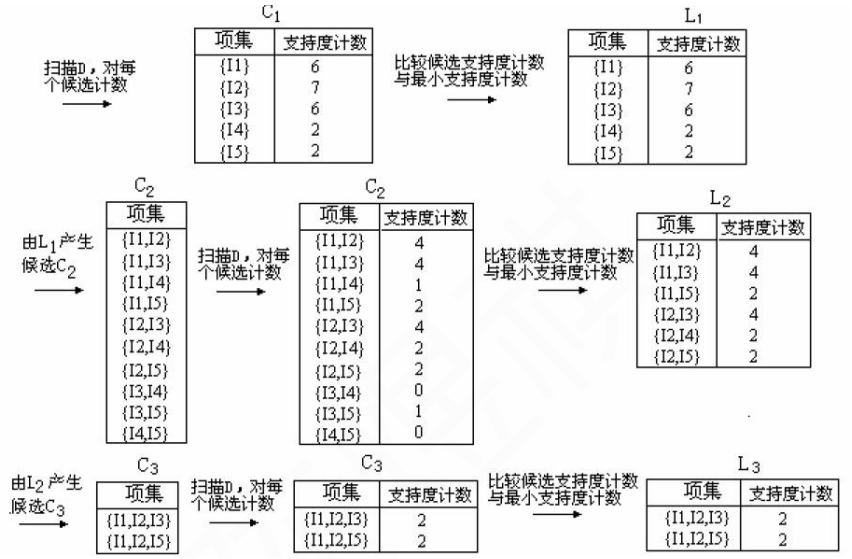
② 扫描一遍事务数据库，确定每个项的支持度，生成频繁 1-项集

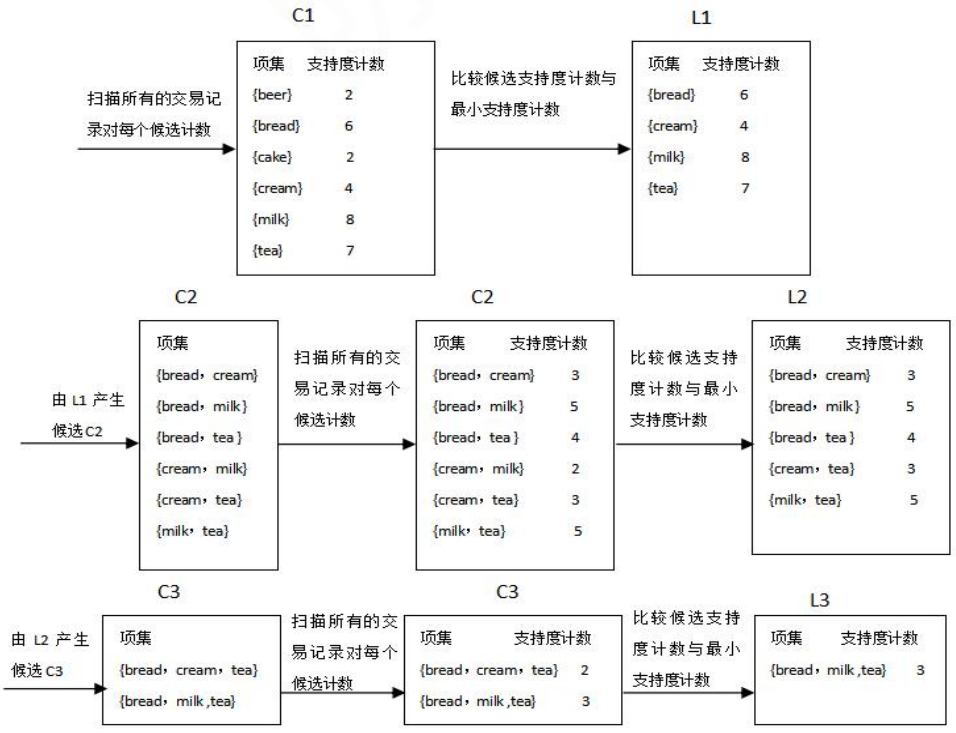
③ 重复生成(k+1)-项集

1) 候选项集的产生：由长度为 k 的频繁项集生成长度为 k+1 的项集
2) 候选项集前剪枝：对每个候选项集，若具有长度为 k 的非频繁项集，则删除该候选项集
3) 支持度计算：统计剩下的候选项集的支持度（得到 C_k）
4) 候选项集后剪枝：删除非频繁的候选项集（得到 L_k）
5) 设定 k=k+1

(2) 频繁项集内部按字典序排序，合并时使用连接方式（AB+AC 得到 ABC）

(3) 前剪枝时逐个剔除项，检查每个 k-项集是否频繁，有一个不频繁就剪枝





(4) 由频繁项集生成强关联规则：枚举每个频繁项集 Y 的所有非空真子集 X，得到规则 $X \Rightarrow Y - X$；若符合最小置信度阈值 min_conf 要求则输出。

3. FP-Growth 算法

(1) 构建 FP-tree

① 事务内各项，按支持度降序预处理，并剔除支持度计数不足阈值的项

② 逐个读入事务，映射到 FP-tree 的一条前缀路径（项:次数）

③ 维护一个链接具有相同项的节点的指针列表

(2) 发现频繁项集

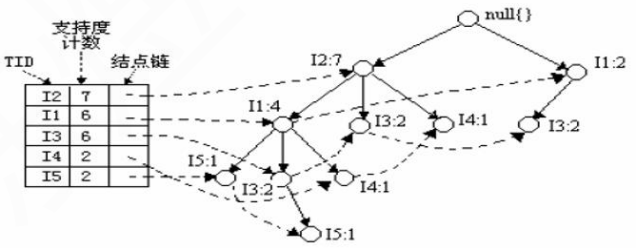
① 对每个项，抽取条件模式基：介于树根节点与所查找项之间的前缀路径，次数取所查找项的次数。（{x,y,z}:3, {u,v}:2）

② 将条件模式基看作新的事务库，对每个项生成一颗条件 FP-树

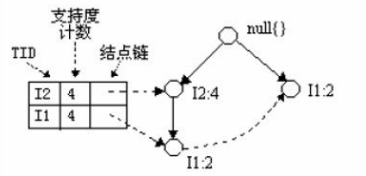
③ 从条件 FP-树中得到频繁项集

图见书 P203、204

TID	每个事务建树的路径
T100	$I_2I_1I_5$
T200	I_2I_4
T300	I_2I_3
T400	$I_2I_1I_4$
T500	I_1I_3
T600	I_2I_3
T700	I_1I_3
T800	$I_2I_1I_3I_5$
T900	$I_2I_1I_3$



项	条件模式库	Fp-tree	频繁模式
I5	$\{I_2I_1: 1\}$ $\{I_2I_1I_3: 1\}$	$<I_2: 2, I_1:2>$	$\{I_2I_5:2\}$ $\{I_1I_5:2\}$ $\{I_2I_1I_5:2\}$
I4	$\{I_2I_1:1\}$,$\{I_2:1\}$	$<I_2:2>$	$\{I_2I_4:2\}$
I3	$\{I_2I_1:2\}$ $\{I_2:2\}$ $\{I_1:2\}$	$<I_2:4, I_1:2>$ $<I_1:2>$	$\{I_2I_3:4\}$ $\{I_1I_3:4\}$ $\{I_2I_1I_3:2\}$
I1	$\{I_2:4\}$	$<I_2:4>$	$\{I_2I_1:4\}$



具有条件结点 I3 的条件 FP-树

六　离群点挖掘

1. 基于距离的方法

(1) 对象 p 的 k-最近邻距离 k_distance(p)：除 p 外第 k 近的点的距离（在边缘线上的点可能有多个）

(2) 离群因子 $OF1(x,k)=\frac{\sum_{y\in N(x,k)}\text{distance}(x,y)}{|N(x,k)|}$，即邻居距离平均值

(3) 伪代码

算法：基于距离的离群点检测算法

输入：数据集D，最近邻个数k

输出：离群点对象列表

```
(1) for all 对象x do
(2)     确定x的k-最近邻集合N(x,k);
(3)     确定x的离群因子OF1(x,k);
(4) end for
(5) 将OF1(x,k)降序排序，确定离群因子大的若干对象为离群点
(6) return
```

---

2. 基于相对密度的方法

(1) 局部邻域密度 $\text{density}(x,k)=\left(\frac{\sum_{y\in N(x,k)}\text{distance}(x,y)}{|N(x,k)|}\right)^{-1}$，即邻居距离平均值的倒数

(2) 相对密度 $\text{relative density}(x,k)=\frac{\sum_{y\in N(x,k)}\text{density}(y,k)/|N(x,k)|}{\text{density}(x,k)}$，即邻居密度均值÷自身密度

(3) 离群因子 $\text{OF2}(x,k)=\text{relative density}(x,k)$

(4) 伪代码

```
算法：基于相对密度的离群点检测算法
输入：数据集D，最近邻个数k
输出：离群点对象列表
(1) for all 对象x do
(2)     确定x的k-最近邻集合N(x,k);
(3)     使用x的最近邻，确定x的密度density(x,k);
(4) end for
(5) for all 对象x do
(6)     确定x的相对密度relative density(x,k)，并赋值给OF2(x,k)
(7) end for
(8) 将OF2(x,k)降序排序，确定离群因子大的若干对象为离群点
(9) return
```

3. 基于聚类的方法（两阶段离群点挖掘方法 TOD）

(1) 设数据集 D 被一趟聚类算法划分为 k 个簇 C1~k

(2) 离群因子 $\mathrm{OF3}(p)=\sum_{i=1}^{k}\frac{|C_i|}{|D|}\times d(p,C_i)$，即到簇中心距离的加权平均值

(3) 平均值 Ave_OF，标准差 Dev_OF

(4) 满足条件 $\mathrm{OF3}(p)\geq \mathrm{Ave\_OF}+\beta\cdot\mathrm{Dev\_OF}\quad(1\leq\beta\leq 2)$ 的判定为离群点。通常取 $\beta=1$。