第二章 数据认知与预处理

2.1 数据认知
- 2.1.1 属性类型
- 2.1.2 常用统计描述方法
- 2.1.3 样本紧密性

2.2 预处理
- 2.2.1 数据清洗
- 2.2.2 数据规范化

预习并回答以下问题

- 天气={晴、多云、阴、下雨}，请问天气是什么类型的属性？如何用独热编码来表示？
- 常用的中心趋势度量有哪些？分别适合哪些类型属性？
- 欧式距离怎么用点积表示？
- IQR怎么计算？
- 属性之间的相关性一般怎么衡量？

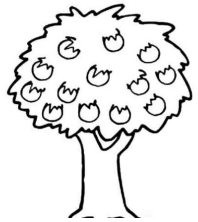
2.1.1 属性类型

假设一个西瓜数据集收集了一定数目的西瓜样本，每个样本由：色泽、重量、甜度三个属性描述。下表给出了三个西瓜样本在以上三个属性的观测值

序号	色泽	重量	甜度
1	青绿	3.3	高
2	浅白	3.5	一般
3	浅白	2.9	高

每一行记录对应一个“对象”或“样本”

每一列对应一个“属性”或“特征”。



到底叫什么？

来自不同领域的研究者喜欢用自己领域特有的名词，所以同一个概念可能有多个名字。比如计算机科学领域喜欢用“对象”来代表一条记录，而统计学用“样本”。类似的，在统计领域把用于对样本进行描述的一个方面叫“属性”，而在机器学习或计算机视觉领域一般叫“特征”。

属性类型

属性（特征）主要分为以下几种类型：

数值类型（有大小）

连续属性(continuous)：取连续值

如房屋面积、价格。

离散

等级属性(ordinal)：取离散值但有大小

如收入等级取高、中、低；评价分1-5颗星。

类别属性(categorical)：取离散值且没有大小

如性别、颜色。

注意：这里的属性类型从其取值方式来定义，需要与保存数据时python里面的数据类型（dtype）区分。比如等级属性或类别属性具体记录时可能用了int64，也可能是object。如果是保存为int64的等级属性，一般无需处理。但如果是用int64表示的类别属性，如用1表示男性，2表示女性，则需要根据具体情况来决定是否需要进一步处理。

属性类型转化

连续属性⟷离散属性之间的相互转化

1.不同的算法可能处理不同类型的属性：

如决策树、朴素贝叶斯处理离散属性； 而回归、SVM、神经网络等方法处理连续属性。

2.原始数据很可能包含多种类型属性，需要进行一定转化使其符合后续分析算法的输入要求。

属性类型转化

连续属性离散化（转成等级属性）：**类似模拟信号转成数字信号**

基本思想是区间划分(bining):把属性观测值分成多个区间(bin)，每个区间代表一个等级。

主要划分方法：距离、等频，以及基于聚类的方法等。

等距划分：每个区间的取值范围一样大

等频划分：每个区间包含的样本数一样多

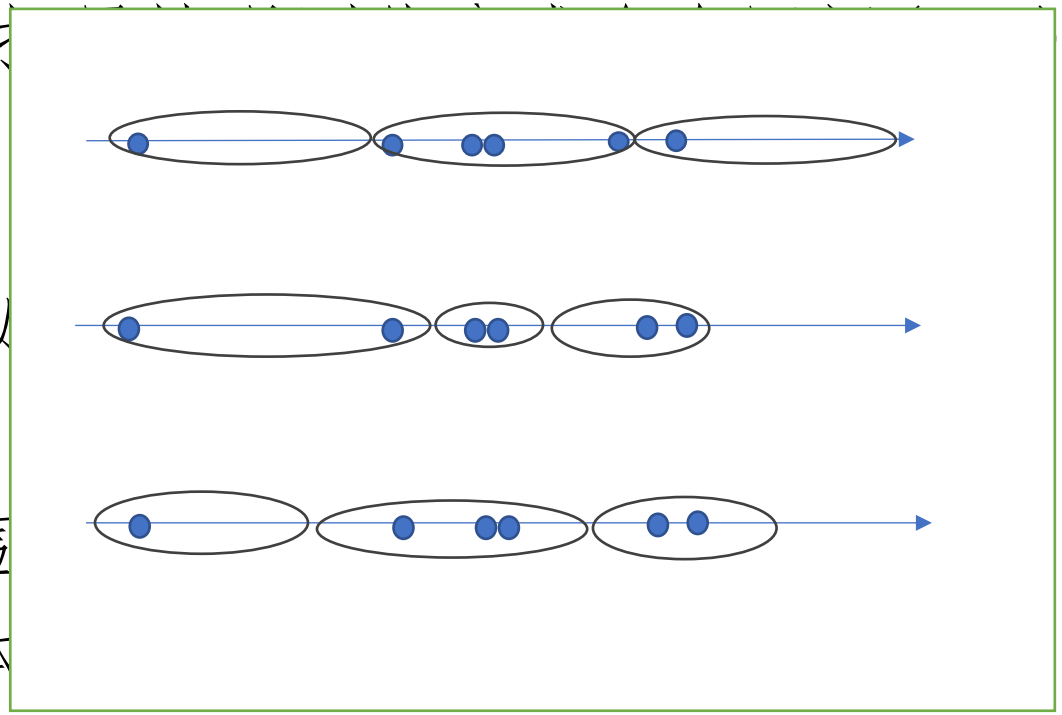
聚类：对连续属性的观测值进行聚类，聚类得到的簇对应一个区间

属性类型转化

连续属性离散化（转成等级属性）：**类似模拟信号转成数字信号**

基本思想是区间划分(bining)：

每个区间代表一个等级。



主要划分方法：距离、等频，以

等距划分：每个区间的取值范围

等频划分：每个区间包含的样本

聚类：对连续属性的观测值进行聚类，聚类得到的簇对应一个区间

聚类：根据样本之间的相似度/距离对样本进行分组。

属性类型转化

离散属性连续化：离散属性转换后可以被当作连续属性来处理。

- 等级属性可以用大小不同的值代表不同等级。

如:用1，0.5，0.1分别代表高、中、低三个等级；

- 类别属性可以用one-hot（独热）编码来表示该特征，编码的长度等于该属性可以取的不同值的个数。

例如：

用两维one-hot编码表示*性别*，[1，0]表示男，[0，1]表示女，即把“性别”⟹“是否男性”、“是否女性”两个**二值(binary)**属性。

例子：属性类型转换

原始数据

序号	色泽	重量	甜度
1	青绿	3.3	高
2	浅白	3.5	一般
3	浅白	2.9	高

色泽(类别属性)，可能的取值为浅白、青绿、乌黑三种，转成3维one-hot编码；
甜度（等级属性），有高、一般、低三个等级。

离散属性转换后的数据

序号	是否浅白	是否青绿	是否乌黑	重量	甜度
1	0	1	0	3.3	1.0
2	1	0	0	3.5	0.5
3	1	0	0	2.9	1.0

属性个数从3增加到5。

数据矩阵

序号	是否浅白	是否青绿	是否乌黑	重量	甜度
1	0	1	0	3.3	1.0
2	1	0	0	3.5	0.5
3	1	0	0	2.9	1.0

第4个属性

$$\boldsymbol{X}_{3\times5}=\begin{bmatrix}0 & 1 & 0 & 3.3 & 1.0\\ 1 & 0 & 0 & 3.5 & 0.5\\ 1 & 0 & 0 & 2.9 & 1.0\end{bmatrix}$$

第1个样本

第1个样本向量：$\boldsymbol{x}_1=(0,1,0,3.3,1.0)$

第2个样本向量：$\boldsymbol{x}_2=(1,0,0,3.5,0.5)$

第3个样本向量：$\boldsymbol{x}_3=(1,0,0,2.9,1.0)$

其中x_{ij}表示样本i中第j个属性的观测值或特征权重。如$x_{14}=3.3$。

每个样本表示成一个5维向量，每个属性对应一个维度(dimension) 。

sklearn和pandas提供的类别转换

- 独热编码-特征数目增加

sklearn.preprocessing.OneHotEncoder，可以通过categories选项指定每个属性的取值，默认自动从数据中获得。

pd.get_dummies，跳过数值类型对象，即无论是整型还是浮点型，都会保留原始列不变，自动从数据中获得。

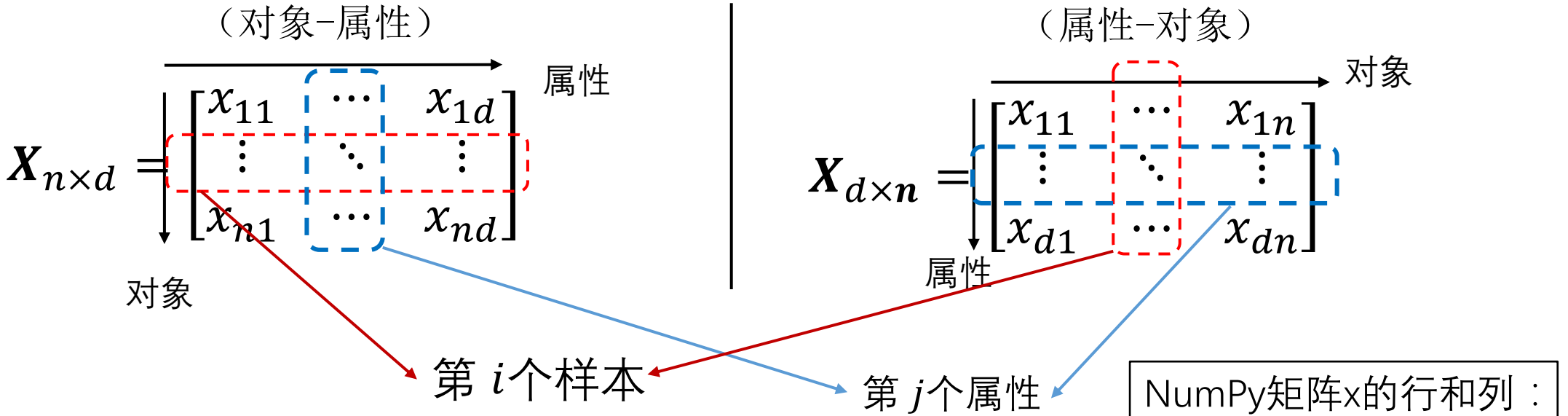
- 类别属性转数值-特征数目不变

sklearn.preprocessing.OrdinalEncoder 或pd.factorize()

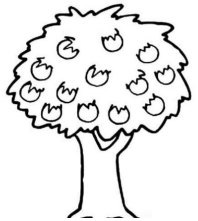
每个类别用一个整数来表示。一般情况下分类问题中标签的处理就采用该种方式。但如果是属性，并且在算法中需要参与计算相似度，则不适合用该方式。

数据矩阵（两种形式）

怎么表示由个d个属性描述的包含n个对象的数据集？



NumPy矩阵x的行和列：
第i行为：x[i, :]
第j列为：x[:, j]



为啥又有两种表示呢？
对象-属性：计算机科学家喜欢，因为直接与数据表格对应。
属性-对象：经典线性代数中，向量为列向量。线性代数可是比数据挖掘的老前辈。

输入形式和数学符号说明

输入形式：

- 经典数学公式中，一般默认向量为列向量，数据矩阵为属性-对象形式。
- 由于很多数据导入是对象-属性形式，注意确认实现算法时公式或数据是否需要进行转化。
- 调用库函数时注意是否符合所要求的数据矩阵形式。

数学符号：

- 小写字母表示标量、向量的一个分量、矩阵的元素，如n, x_i , x_{ij};
- 小写且黑体的表示向量，如$\boldsymbol{x}$，大写且黑体的表示矩阵，如$\boldsymbol{X}$。
- $\boldsymbol{x} = [x_1, x_2, \dots x_d]$表示行向量，$\boldsymbol{x} = [x_1, x_2, \dots x_d]^T$表示列向量，其中$x_j$表示$\boldsymbol{x}$的第j个分量, 上标T表示转置。

第二章 数据认知与预处理

2.1 数据认知

2.1.1 属性类型

2.1.2 常用统计描述方法

2.1.3 对象紧密性

2.2 预处理

2.2.1 数据清洗

2.2.2 数据规范化

2.1.2 常用统计描述方法

用DataFrame.describe()查看统计信息

```
In [188]: x=pd.DataFrame(irisdata.data)

In [189]: x.describe()
Out[189]:
                0           1           2           3
count  150.000000  150.000000  150.000000  150.000000
mean     5.843333    3.057333    3.758000    1.199333
std      0.828066    0.435866    1.765298    0.762238
min      4.300000    2.000000    1.000000    0.100000
25%      5.100000    2.800000    1.600000    0.300000
50%      5.800000    3.000000    4.350000    1.300000
75%      6.400000    3.300000    5.100000    1.800000
max      7.900000    4.400000    6.900000    2.500000
```

注意：如果存在缺失值，那么每列对应的count的数值少于总的记录数。

如何通过pd相关方法确认是否存在缺失值？

2.1.2 常用统计描述方法

- 对单个属性的统计描述：中心趋势、离散趋势
  - ✓中心趋势：假设对应某个属性有n个观测值，那么这个属性最具有代表性的值是多少？
  - ✓离散趋势：反映数据集中其他值远离其中心值的程度，即散布程度。
- 属性之间的相关性

  不同属性之间的独立性有多强。

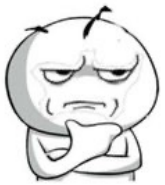
统计描述方法-中心趋势度量

假设对应某个属性有n个观测值，那么这个属性最具有代表性的值是多少？

- 均值(mean)

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

以上均值对应算术平均，是用来计算数据中心最常用的方法之一。



简单求均值有什么局限性？

- 如果每个观测值的可靠性不一样呢？
- 如果其中一个观测值是不准确的，比如特别大，对均值有什么影响呢？

• 中位数(median)

假设有n个观测值，对其排序后中间位置对应的值即中位数。

n是奇数：排在第$\frac{1}{2}(n-1)+1$位的值即中位数；

n是偶数：中位数可以是中间两个位置对应值的平均或任意一个。

例：对应属性“年龄”的5个观测值为[18, 20, 16, 22, 19]，计算其均值和中位数。

均值：$\bar{x}=\frac{1}{5}(18+20+16+22+19)=19$

中位数：排序后[16, 18, 19, 20, 22]，第3位的值是19

如果把第4个观测值从22变成32，均值和中位数有什么变化？说明什么？

- 众数(mode)

均值和中位数适用于取值分大小的属性，包括连续属性和等级属性。但不适用于类别属性，因为其属性值没有大小。

众数：一个集合中出现次数最多的值。

比如：对花朵颜色属性的一组观测值发现，出现次数最多的颜色是红色。

d[‘Title’].value_counts()

- 适用于离散属性，包括等级属性、类别属性；
- 众数可能有多个。

统计描述方法-离散趋势度量

- 方差σ^2和标准差σ：

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2 - \bar{x}^2$$

其中$\bar{x}$为观测值的均值。

标准差是衡量数据集发散程度的基本指标。一组合理的数据观测值，一般不会远离均值，超过标准差的数倍。

方差的无偏估计为: $\sigma^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$ 。 不过这个系数对挖掘算法的结果没有太多影响。

方差、标准差

年龄观测值 x=[18, 20, 16, 22, 19]

```
In [143]: age=np.array([18, 20, 16, 22, 19])

In [144]: age.mean()
Out[144]: 19.0

In [145]: age.std()
Out[145]: 2.0
```

年龄观测值 x1=[15, 23, 16, 22, 19]

```
In [149]: age1=np.array([15, 23, 16, 22, 19])

In [150]: age1.mean()
Out[150]: 19.0

In [151]: age1.std()
Out[151]: 3.1622776601683795
```

注意：

NumPy.array中的.std()方法默认采用有偏估计，即除以n，而pandas.DataFrame中.std()默认无偏估计，除以n-1。

- 极差与分位数

极差或全距(range)是指最大值和最小值的差，反映数据的最大离散程度。

$$range(x) = x_{max} - x_{min}$$

分位数

观测值递增排序后，把划分成k个大小基本相同的集合，即每个集合包含$\frac{n}{k}$个观测值。

Q2就是中位数median

比如$k = 4$时，产生3个分割点$Q_1(25\%), Q_2(50\%), Q_3(75\%)$，叫做四分位数。

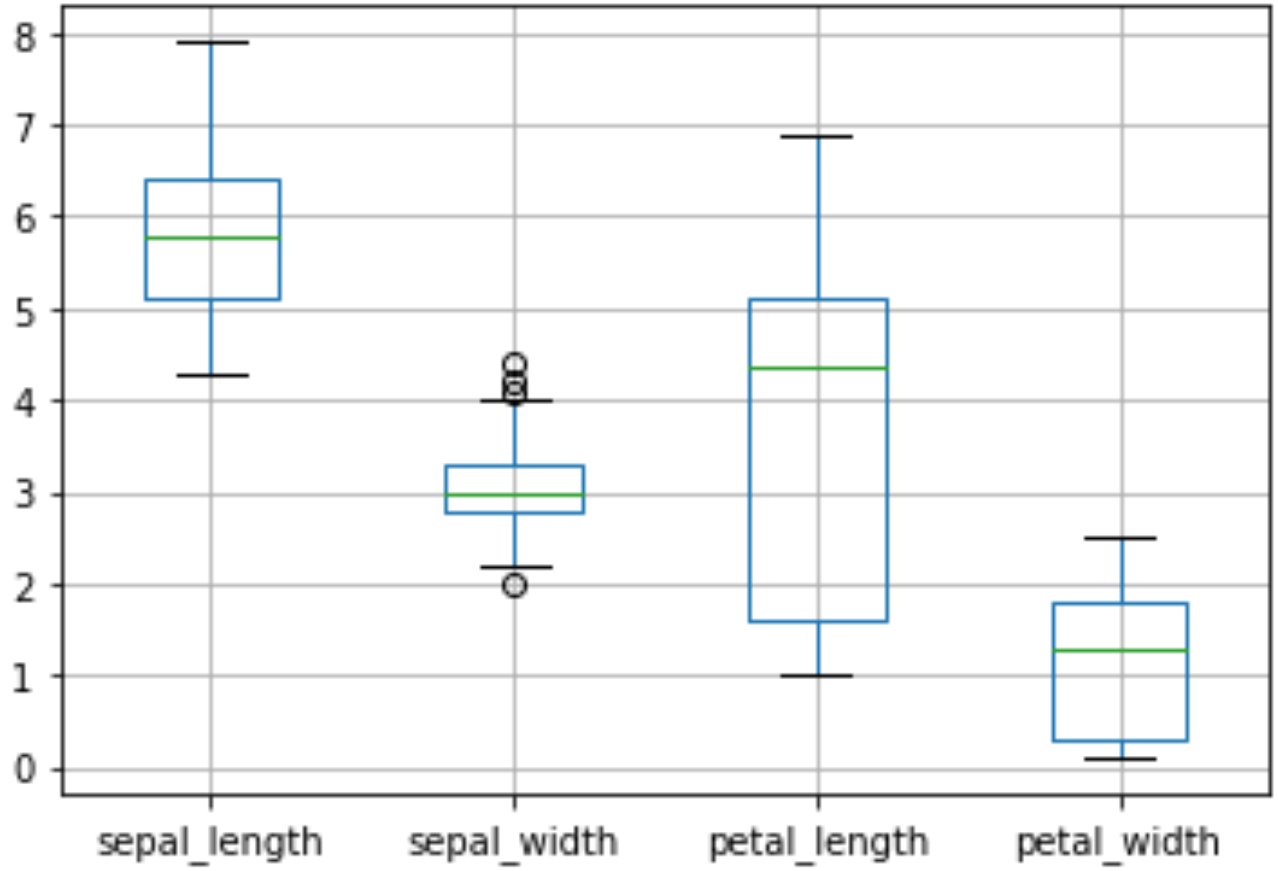
四分位极差：　　$IQR = Q_3 - Q_1$

例如有8个观测值，排序后为：[3.4，3.6，4.0，5.5，5.8，6.0，6.1，6.3]
四分后：{3.4，3.6}，{4.0，5.5}，{5.8，6.0}，{6.1，6.3}
则$Q_1 = 3.6,\ Q_2 = 5.5,\ Q_3 = 6.0$，　$IQR = 6.0 - 3.6$=2.4

注意：实际计算公式比上面复杂。
如 w = 1+(N-1)*0.25表示 Q1的位置，则 Q1 = x(y) + z*(x(y+1) - x(y))，y和z分别对应w的整数和小数部分。

箱型图 (box plot)

- 利用数据中的五个统计量：最小值(min)、第一四分位数(Q1)、中位数(Q2)、第三四分位数(Q3)与最大值(max)来描述数据的一种方法。



思考与探索：
上下的点表示什么？
这些点对boxplot中五个统计量的计算有什么影响？

属性之间的相关性

- 当前的属性集是最有效的吗？有没有冗余？

给定数据 $\boldsymbol{X}_{n\times d}=\begin{bmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{bmatrix}$

怎么衡量属性，即列向量之间的相关性？

属性之间的相关性

给定数据 $\boldsymbol{X}_{n\times d}=\begin{bmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nd} \end{bmatrix}$

协方差(co-variance)矩阵

$$\boldsymbol{C}_{d\times d}=\begin{pmatrix} c_{11} & \cdots & c_{1d} \\ \vdots & \ddots & \vdots \\ c_{d1} & \cdots & c_{dd} \end{pmatrix}$$

$$c_{ij}=\frac{1}{n}\sum_{h=1}^{n}(x_{hi}-\overline{x_i})(x_{hj}-\overline{x_j})$$

与方差类似，无偏估计为除以n-1。

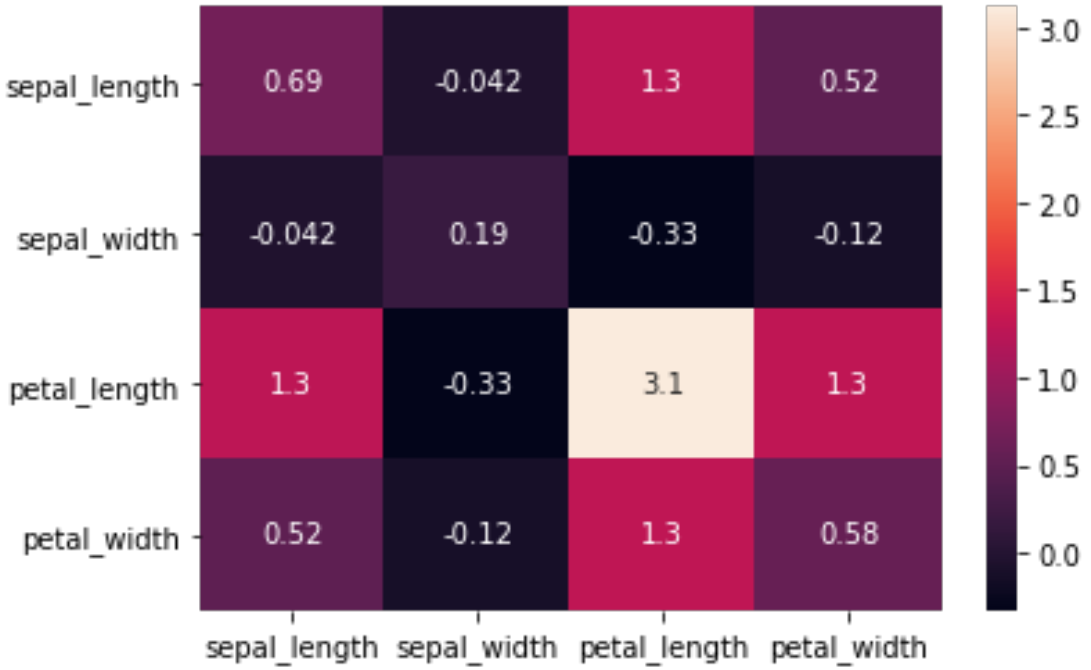
其中，$\overline{x_i}, \overline{x_j}$分别为第$i$列和第$j$列的均值。

协方差：两个变量的观测值减去各自均值后的点积（再除以n或n-1）。

$$c_{ii}=\frac{1}{n}\sum_{h=1}^{n}(x_{hi}-\overline{x_i})^2=\sigma_i^2$$

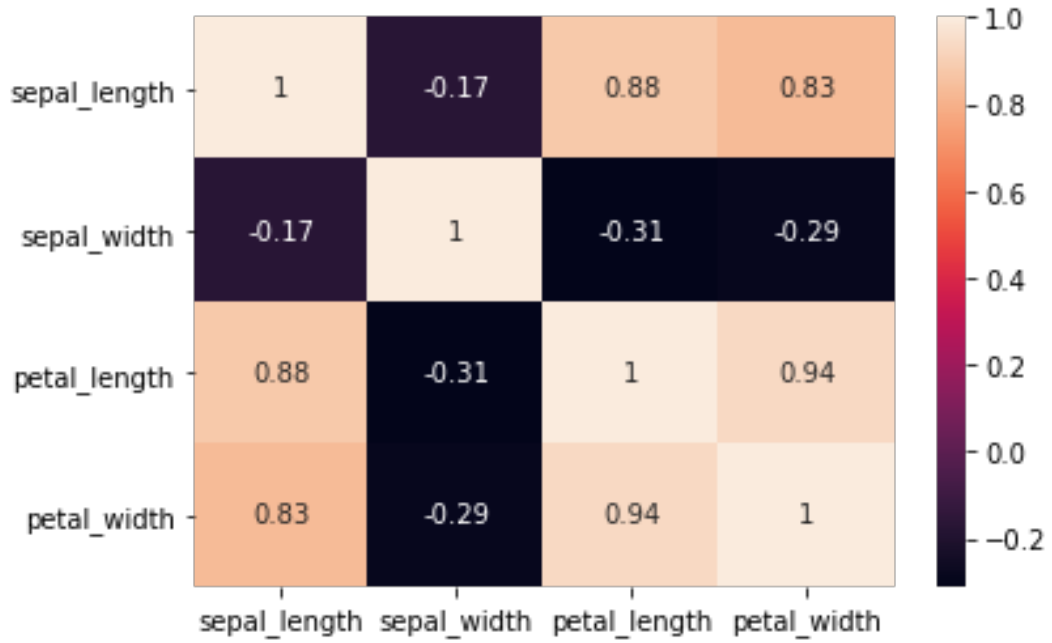
热力图

```
import seaborn as sns
plt.figure(figsize=(6, 4))
sns.heatmap(corr_matrix, annot= True)
```



协方差矩阵

corr_matrix=X.cov



相关系数（归一化的协方差）矩阵

corr_matrix=X.corr(method='spearman')

第二章 数据认知与预处理

2.1 数据认知

2.1.1 属性类型

2.1.2 常用统计描述方法

2.1.3 对象紧密性

2.2 预处理

2.2.1 数据清洗

2.2.2 数据规范化

数据分析中的基本问题

有了属性-对象矩阵表示的数据集，是不是马上就去应用挖掘算法？

下面着个基本问题可能需要先考虑：

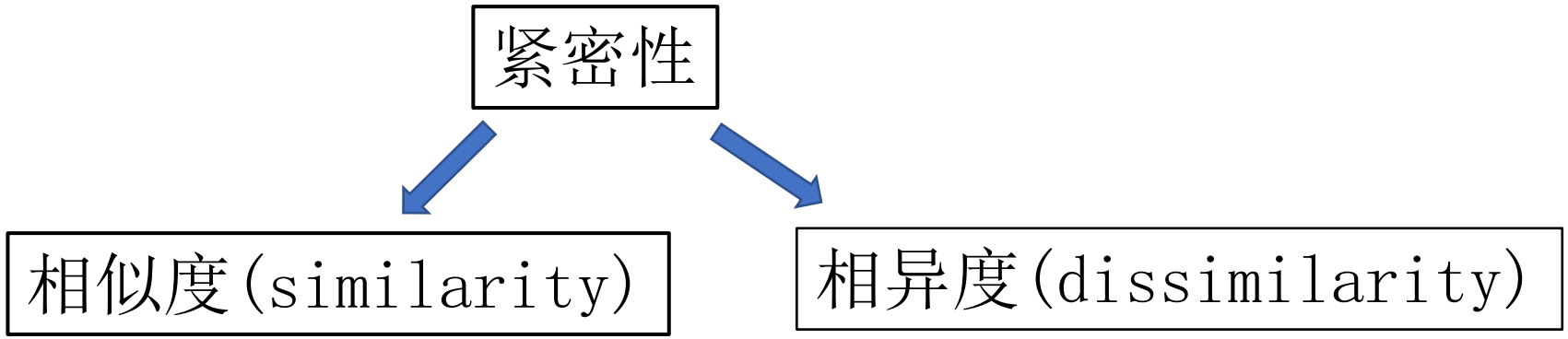
- 如何衡量对象之间的紧密性？

样本之所以被叫做“对象”，是因为数据挖掘问题是要发现由样本构成的数据集的内在结构，而样本之间的紧密性是描述样本之间关系的基本方法。

2.1.3 对象之间的紧密性

如何衡量两个样本之间的关联性是很多数据挖掘算法（聚类、最近邻分类）中的一个基本问题。有效的解决这一问题往往可以事半功倍。

如何计算两个对象（对应向量$\boldsymbol{x}_i$和$\boldsymbol{x}_j$）之间的紧密性？



向量范数(norm)

向量“范数”表示向量长度或尺寸的一种度量，常用的向量范数包括：

- l_1 范数 $\|\boldsymbol{x}\|_1 = \sum_{i=1}^{d} |x_j|$

$$dist_{Manh}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{l=1}^{d} |x_{il} - x_{jl}| = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_1$$

- l_2 范数 $\|\boldsymbol{x}\|_2 = \sqrt{\sum_{l=1}^{d} x_l{}^2} = \boldsymbol{x}^T\boldsymbol{x}$

$$dist_{Euc}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{l=1}^{d} (x_{il} - x_{jl})^2} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2$$

- l_∞ 范数 $\|\boldsymbol{x}\|_\infty = max_l(\{|x_1|, \ldots |x_d|\})$

$$dist_{che}(\boldsymbol{x}_i, \boldsymbol{x}_j) = max_l\{|x_{il} - x_{jl}|\} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_\infty$$

基于距离的相异度

- 欧式距离(Euclidean distance)

$$dist_{Euc}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt{\sum_{l=1}^{d}(x_{il} - x_{jl})^2}$$

- 曼哈顿距离(Manhattan distance)

$$dist_{Manh}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{l=1}^{d}|x_{il} - x_{jl}|$$

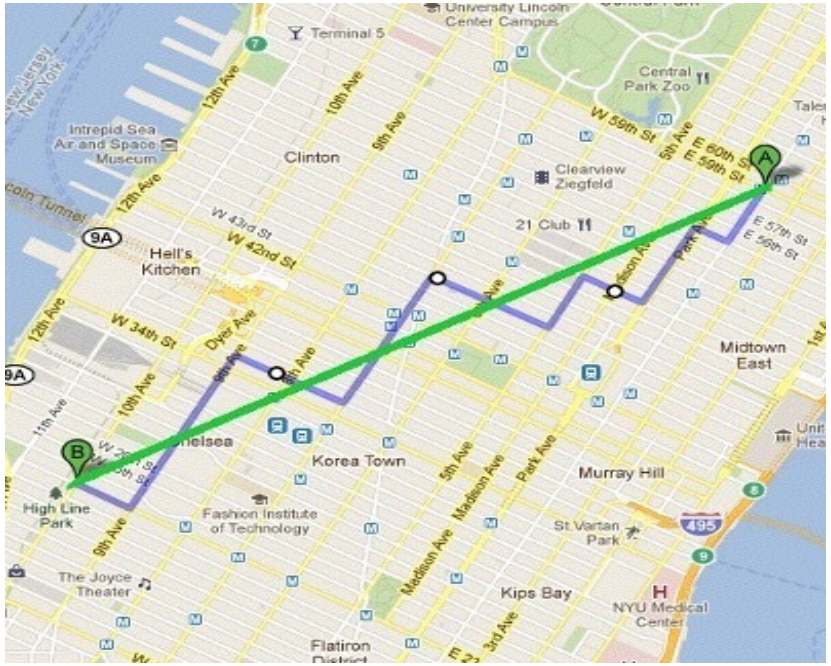
两者都满足：

非负性： $d(\boldsymbol{x}, \boldsymbol{y}) \geq 0$

同一性：$d(\boldsymbol{x}, \boldsymbol{x}) = 0$

对称性：$d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$

三角不等式： $d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z}) + d(\boldsymbol{z}, \boldsymbol{y})$



曼哈顿距离又叫城市街区距离

基于距离的相异度

- 切比雪夫距离(Chebyshev distance)

$$dist_{che}(\boldsymbol{x}_i, \boldsymbol{x}_j) = max_l|x_{il} - x_{jl}|$$

- 明可夫斯基距离(Minkowski distance)

$$dist_{Mink}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sqrt[p]{\sum_{i=1}^{d}(x_{il} - x_{jl})^p}$$

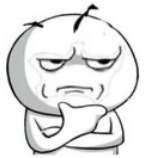
欧式距离、曼哈顿距离、切比雪夫距离是明可夫斯基距离的三种特殊形式，分别对应$p = 2$, $p = 1$，和 $p = \infty$。

向量点积(dot product)

两个向量$\boldsymbol{x}_i$和$\boldsymbol{x}_j$的点积为对应分量乘积之和：

$$\boldsymbol{x}_i{}^T\boldsymbol{x}_j = \sum_{l=1}^{d} x_{il}x_{jl} = <\boldsymbol{x}_i, \boldsymbol{x}_j>$$

关于点积：
- 输入两个向量，输出为一个标量，所以又叫标积。
- 可以理解为两个向量的一种相关性
- 满足：$\boldsymbol{x}^T\boldsymbol{y} = \boldsymbol{y}^T\boldsymbol{x}$
- 一个向量的l_2范数的平方为自己对自己的点积，即 $\|\boldsymbol{x}\|_2^2 = \boldsymbol{x}^T\boldsymbol{x}$



欧式距离的平方 $\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$怎么用点积表示？

余弦相似度（Cosine similarity）

余弦相似度用两个向量之间的夹角来衡量两者之间的相似度。夹角越小，相似度越大，其取值范围为[-1,1]，计算公式：

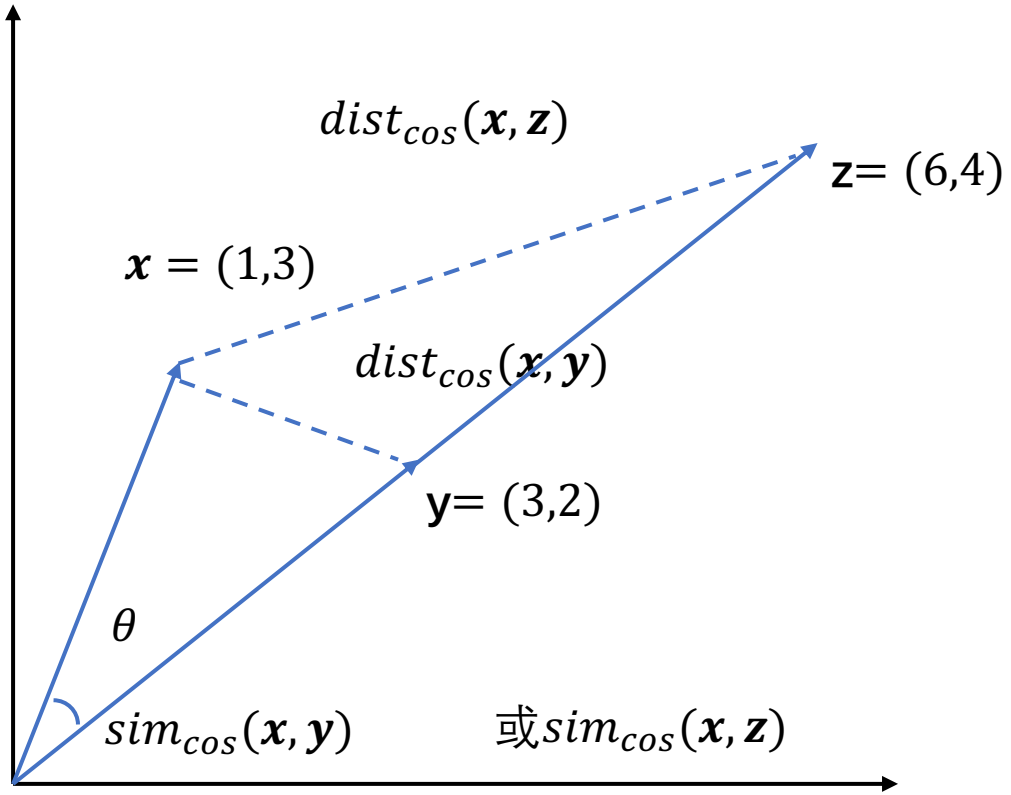
$$sim_{cos}(\boldsymbol{x}_i, \boldsymbol{x}_j) = \frac{\boldsymbol{x}_i{}^T \boldsymbol{x}_j}{\|\boldsymbol{x}_i\|\|\boldsymbol{x}_j\|} = \frac{\sum_{l=1}^{d} x_{il} x_{jl}}{\sqrt{\sum_{l=1}^{d} x_{il}{}^2}\sqrt{\sum_{l=1}^{d} x_{jl}{}^2}}$$

两种特殊情况：

- 对非负向量，即$\boldsymbol{x}_i \geq 0, \boldsymbol{x}_j \geq 0$,两者之间的余弦相似度为[0,1]。
- 对单位向量，即$\|\boldsymbol{x}_i\|$=1，$\|\boldsymbol{x}_j\| = 1$,则余弦相似度退化为两者的点积。

余弦距离：$dist_{cos}(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1 - sim_{cos}(\boldsymbol{x}_i, \boldsymbol{x}_j)$

欧式距离 vs 余弦距离



欧式距离：两点的绝对距离，与向量方向和长度都相关。
余弦相似度：两个向量之间的夹角，与向量长度无关。

汉明距离(Hamming distance)

- 适用于类别型数据。定义为向量中对应分量值不相同的元素个数。
- 普遍用于二值数据，即每个分量值为0或1。1表示对应属性出现，0表示不出现。

如$\boldsymbol{x}_i = (1,0,1,0,0,1), \boldsymbol{x}_j = (0,1,1,0,0,1)$,则

$$dist_{Ham}(\boldsymbol{x}_i, \boldsymbol{x}_j)=2$$

独热(one-hot)编码是一种特殊的二值向量，只有一个分量为1，其他为0。

Jaccard相似度

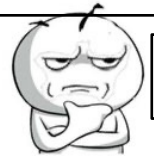
- 两个集合之间交集的相对大小，适用于用集合表示的数据。

$$\text{Jaccard(S,T)}=\frac{|S\cap T|}{|S\cup T|}$$

以上公式中，| |表示集合大小，即包含的元素个数。

例：用户购物记录$S=\{2,3,5,13\}, T=\{3,5,13,35\}$分别表示两个用户购买的商品的id，这两个记录之间的相似度为：

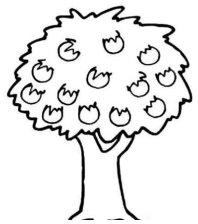
$$\text{Jaccard(S,T)}=\frac{|\{3,5,13\}|}{|\{2，\ 3，\ 5，\ 13，\ 35\}|}=\frac{3}{5}$$



以上例子余弦相似度可以用吗？怎么用？

怎么选？

1. 欧式距离是最普遍的距离度量，用于一般性数据；
2. 对稀疏高维数据，余弦距离更加有效，比如文本相似度；



把一个文档用出现的词来表示，又叫词袋(bag-of-words)模型。假设有n条评论，一共包含m个不同的词，即词典大小为m。则每条评论被表示为一个m维向量，每个元素对应某个词在该评论的权重，如出现次数。

由于单条评论的长度很短，数据被表示成高维稀疏向量，此时，余弦相似度往往比欧式距离更有效。

推导单位向量的欧式距离和余弦距离之间的关系

对象间的关系矩阵

一个数据集包中n个对象之间的关系可以用对象-对象的关系矩阵来表示：

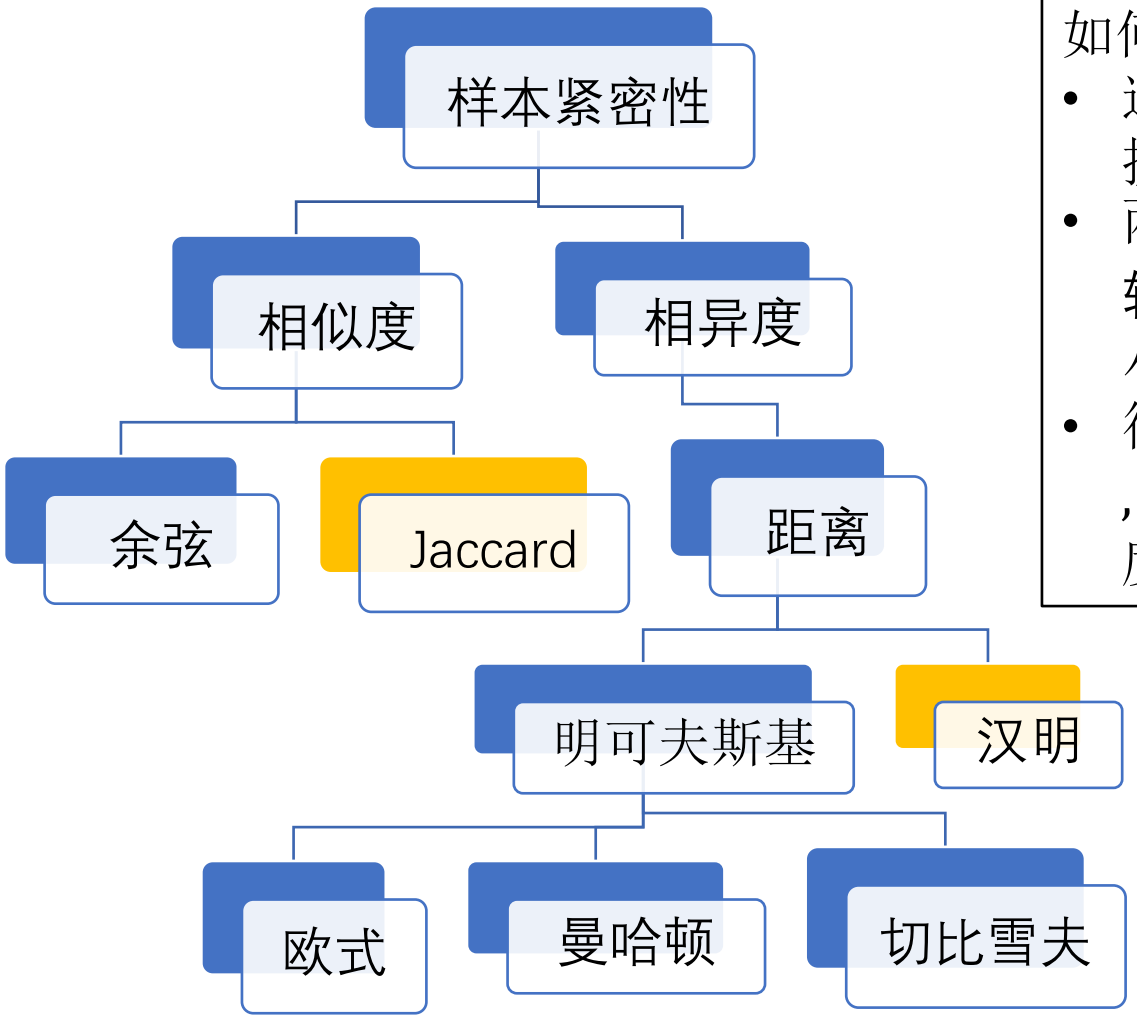
$$\boldsymbol{R_{n\times n}} = \begin{bmatrix} r_{11} & \cdots & r_{1n} \\ \vdots & \ddots & \vdots \\ r_{n1} & \cdots & r_{nn} \end{bmatrix}$$

具体关系即紧密性可以是相似度，也可以是相异度。

相似度/相异度矩阵的计算量问题：

- 一般情况下，关系是对称的即$r_{ij} = r_{ji}$。所以对n个对象的数据集，只需要计算$\frac{1}{2}(n^2-n)$ 个值。
- 计算量是$O(n^2)$,对大规模数据集不适合，需要借助更加快速的近似算法。

相似度/相似度总结



如何得到关系矩阵？

- 通过左图中的常用度量计算得到（黄色的两种可直接用于类别属性）；
- 两种关系的转化：如基于已有的相似度通过$1/s_{ij}$转换得到d_{ij}。实际编程实现时，在分母加一个很小的数如10^{-7}以防分母为0。
- 很多时候要求$s_{ij} \in [0,1]$，$s_{ii} = 1$，或$d_{ij} \in [0,1]$，$d_{ii} = 0$，因此需要对有些方法计算得到的相似度、相异度矩阵进行合理的归一化。

第二章 数据认知与预处理

2.1 数据认知

2.1.1 数据类型

2.1.2 常用统计描述方法

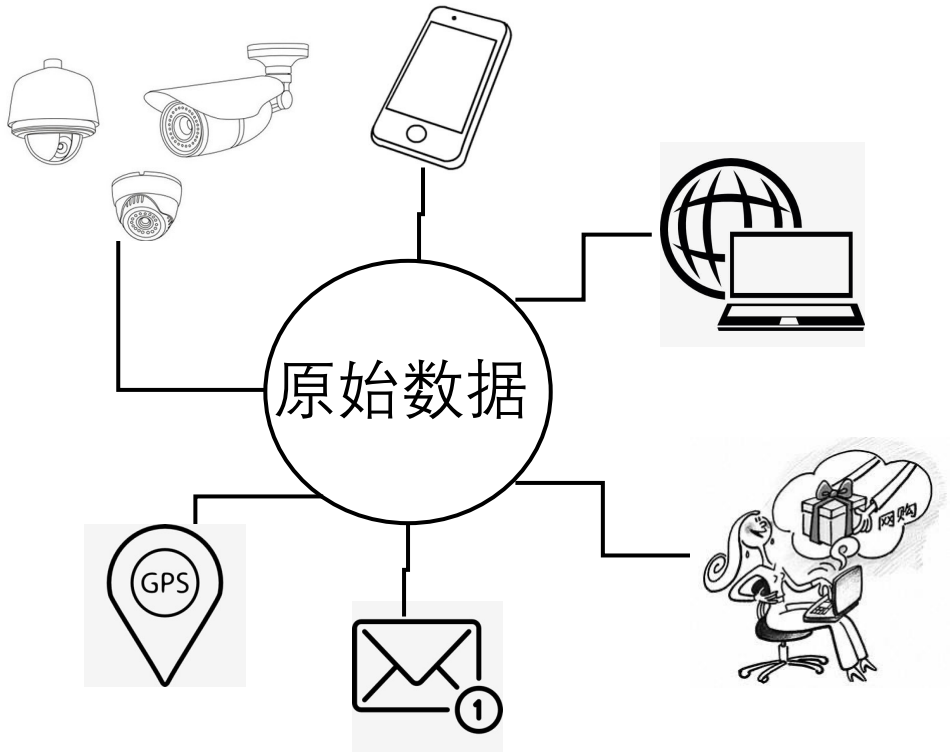
2.1.3 对象紧密性和属性相关性

2.2 预处理

2.2.1 数据清洗

2.2.2 数据规范化

数据清洗



如何处理缺失值

- 为什么会有缺失值?人为或计算机输入错误、采集数据的设备故障、转换文件时出现数据丢失等。

- 怎么处理缺失值？
  - 直接删除存在缺失值的记录：不能有效利用这些记录中的其他信息。

- 常用方法：
  1. 使用全局常量填充。
  2. 使用属性的均值或中位数来填充。
  3. 如果数据集有类别标签，使用同一类样本该属性的均值或中位数来填充。
  4. 使用最有可能的值来填充，该数通过基于推理和归纳的算法确定。

查看每个属性的缺失情况

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6462 entries, 0 to 6461
Data columns (total 27 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Cust_No                     6462 non-null   int64
 1   Target                      6462 non-null   int64
 2   Birth_Place                 6462 non-null   int64
 3   Gender                      6462 non-null   int64
 4   Age                         6462 non-null   int64
 5   Marriage_State              4465 non-null   float64
 6   Work_Years                  6462 non-null   int64
 7   Unit_Kind                   2445 non-null   object
 8   Title                       2767 non-null   float64
 9   Year_Income                 6462 non-null   float64
 10  Couple_Year_Income          2008 non-null   float64
 11  L12_Month_Pay_Amount        6462 non-null   float64
 12  Ast_Curr_Bal                6462 non-null   float64
 13  Std_Cred_Limit              6462 non-null   int64
 14  Loan_Curr_Bal               6462 non-null   float64
 15  ZX_Max_Account_Number       6462 non-null   int64
 16  ZX_Max_Link_Banks           6462 non-null   int64
 17  ZX_Max_Overdue_Account      6462 non-null   int64
 18  ZX_Link_Max_Overdue_Amount  6462 non-null   int64
 19  ZX_Total_Overdu_Months      6462 non-null   int64
 20  ZX_Max_Overdue_Duration     6462 non-null   int64
 21  ZX_Max_Credits              6462 non-null   int64
 22  ZX_Max_Credit_Banks         6462 non-null   int64
 23  ZX_Max_Overdue_Credits      6462 non-null   int64
```

查看每个特征**是否存在**缺失值
df.isnull().any()

```
Cust_No           False
Target            False
Nation             True
Birth_Place       False
Gender            False
Age               False
Marriage_State     True
```

每个属性的**缺失值个数**：
df.isnull().sum()

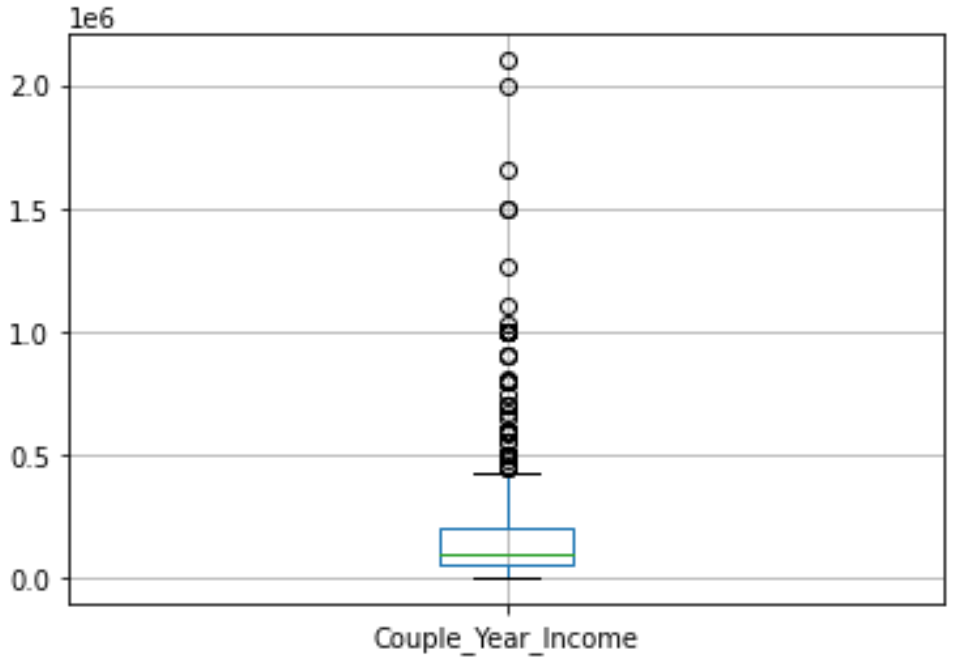
```
Cust_No              0
Target               0
Nation            2137
Birth_Place          0
Gender               0
Age                  0
Marriage_State    1997
```

打印缺失率最高的6个属性

```
df_missing_stat = pd.DataFrame(df.isnull().sum()/df.shape[0],columns=['missing_rate']).reset_index()
df_missing_stat.sort_values(by='missing_rate',ascending=False)[:6]
```

	index	missing_rate
10	Couple_Year_Income	0.689260
7	Unit_Kind	0.621634
8	Title	0.571804
5	Marriage_State	0.309037
0	Cust_No	0.000000
16	ZX_Max_Link_Banks	0.000000

统计信息与boxplot




```
df['Couple_Year_Income'].describe()
```

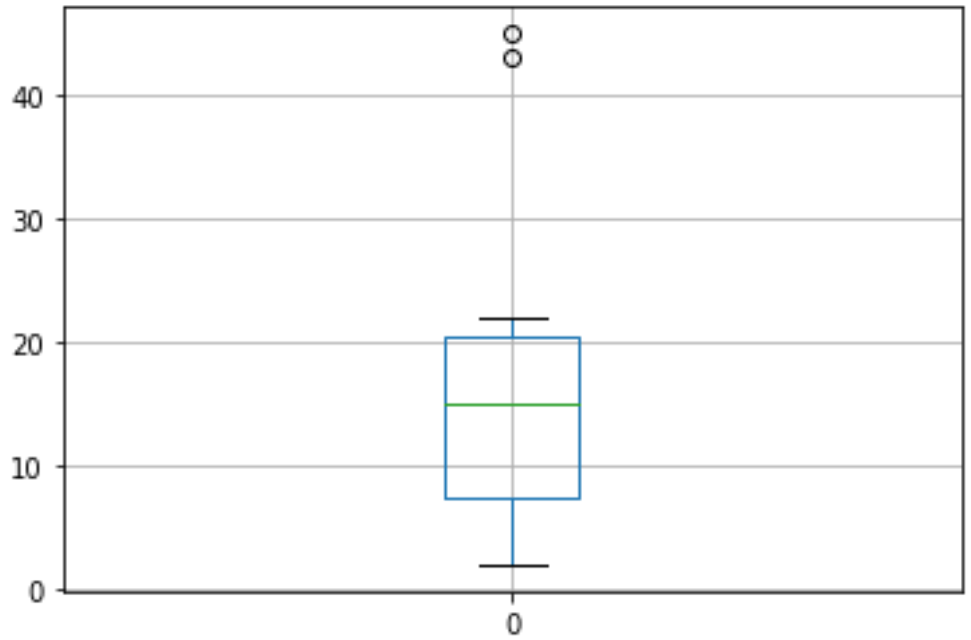
count	2.008000e+03
mean	1.495545e+05
std	1.875145e+05
min	0.000000e+00
25%	5.000000e+04
50%	1.000000e+05
75%	2.000000e+05
max	2.100000e+06

```
df=np.array([2,4,6,8,12,14,16,18,20,22,45,43])
df2=np.array([2,4,6,8,12,14,16,18,20,22])
```

```
df=pd.DataFrame(df)
df.describe()
```

]:

	0
count	12.000000
mean	17.500000
std	13.892444
min	2.000000
25%	7.500000
50%	15.000000
75%	20.500000
max	45.000000



Boxplot：
- 小于Q1-1.5*IQR和大于Q3+1.5*IQR的值被视为异常值；
- 画图时，最小值、最大值不包括异常点；
- Q1、Q2、Q3依然是原始值（describe返回的值），即计算时包含异常点；

```
df2=pd.DataFrame(df2)
df2.describe()
```

3]:

	0
count	10.000000
mean	12.200000
std	6.957011
min	2.000000
25%	6.500000
50%	13.000000
75%	17.500000
max	22.000000

注意：实际计算公式比上面复杂。
如 w = 1+(N-1)*0.25表示Q1的位置，则 Q1 = x(y) + z*(x(y+1) - x(y))，y和z分别对应w的整数和小数部分。

先删除异常值再用中位数填补缺失值

基本方法：小于Q1-1.5*IQR和大于Q3+1.5*IQR的值被视为异常值

- 以Couple_Year_Income为例子

```
item = 'Couple_Year_Income'
iqr = df[item].quantile(0.75) - df[item].quantile(0.25)
q_abnormal_L = df[item] < df[item].quantile(0.25) - 1.5 * iqr
q_abnormal_U = df[item] > df[item].quantile(0.75) + 1.5 * iqr
#取异常点的索引
print(item + '中有' + str(q_abnormal_L.sum() + q_abnormal_U.sum()) + '个异常值')
item_outlier_index = df[q_abnormal_L|q_abnormal_U].index

###删除异常值
df.drop(index = item_outlier_index,inplace=True)
print(df.shape)

#用中位数填补缺失值
df[item] = df[item].fillna(df[item].median())
```

第二章 数据认知与预处理

2.1 数据认知
2.1.1 数据类型
2.1.2 常用统计描述方法
2.1.3 对象紧密性和属性相关性
2.2 预处理
2.2.1 数据清洗
2.2.2 数据规范化

数据规范化或标准化(standardization)

房屋数据

序号	面积(m^2)	房间数目	离地铁站距离（km）	交付房价(万/平米)
1	120	3	1.5	2.5
2	90	2	1.0	2.0
3	90	3	2.0	1.8

不同属性具有不同的单位和取值范围，直接进行运算，如求距离会忽略某些取值范围小的属性或让结果仅由取值范围大的属性来决定。

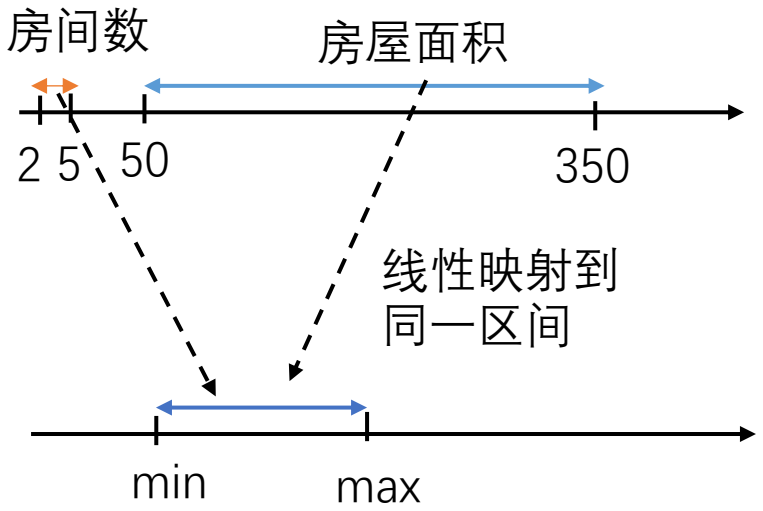
- 基于以上数据计算样本1和样本2之间的欧式距离：

$$d_{12} = \|\boldsymbol{x}_1 - \boldsymbol{x}_2\| = \sqrt{(120-90)^2+(3-2)^2+(1.5-1.0)^2+(2.5-2.0)^2} = 30.02$$

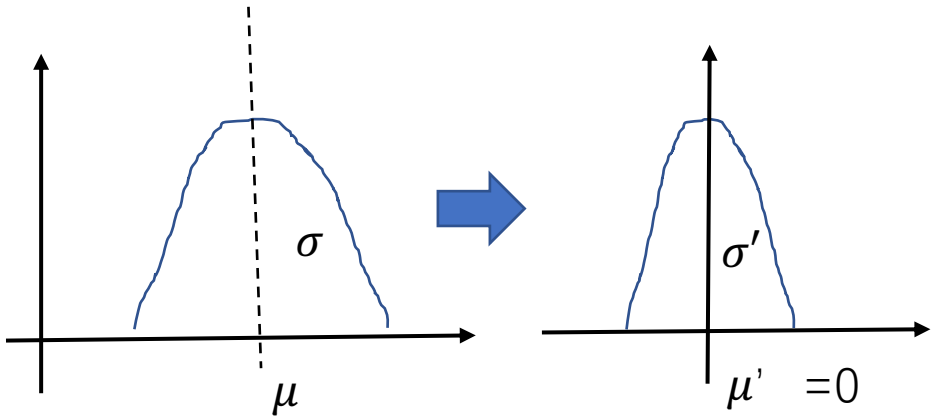
$$\approx \sqrt{(120-90)^2} = 30.0$$

基本由第一个属性决定

数据规范化或标准化



方法一、最小-最大（min-max）规范化：通过线性变换把原始数据取值范围映射到某个统一的区间。



方法二、零均值(zero-mean)规范化：通过属性的均值和标准差进行规范化。

最小-最大(min-max)规范化

- 假设某个属性原始取值范围[min, max]，通过以下公式将其映射到$[min', max']$:

$$x'_j = \frac{x_j - min}{max - min}(max' - min') + min'$$

- 假设房屋面积的原始取值范围为[50, 350]，现通过公式将表格中该列数据映射到[0,1]:

$$x'_1 = \frac{120-50}{350-50}(1-0) + 0 = 0.23$$

$$x'_2 = x'_3 = \frac{90-50}{350-50}(1-0) + 0 = 0.13$$

最小-最大(min-max)规范化

- 假设房间数目、离地铁站距离、交付时房价的取值范围分别为[2, 5]、[0.5, 3.5]、[1.5, 3.5]，通过同样方式把所有属性都规范化到[0,1]把所有属性归一化到[0,1]后的房屋数据

序号	面积	房间数目	离地铁站距离	交付房价
1	0.23	0.33	0.33	0.50
2	0.13	0.00	0.16	0.25
3	0.13	0.33	0.50	0.15

基于min-max标准化后的数据计算样本1和样本2的欧式距离

$$d_{12} = \|x_1 - x_2\| = \sqrt{(0.23-0.13)^2+(0.33-0)^2+(0.33-0.16)^2+(0.50-0.25)^2} = 0.43$$

$$> \sqrt{(0.23-0.13)^2} = 0.1$$

所有属性都被合理考虑到

零均值(zero-mean)规范化

- 假设所有样本在某个属性的取值的均值为μ， 标准差为σ， 则把属性通过以下零均值规范化变成均值为0，标准差为1的分布：

$$x'_j = \frac{x_j - \mu}{\sigma}$$

零均值在有些地方也叫Z分数(z-score)规范化。

如房屋面积的均值为100，标准差（有偏）为14.14，则

$$x'_1 = \frac{120-100}{14.14} = 1.4$$

$$x'_2 = x'_3 = \frac{90-100}{14.14} = -0.71$$

零均值(zero-mean)规范化

```
In [33]: ms=np.mean(x,axis=0)

In [34]: st=np.std(x,axis=0)

In [35]: x1=(x-ms)/st
```

注意：NumPy.array.std()默认除以n, 用ddof=1选项设置无偏。pandas默认无偏。

对所有属性进行零均值规范化后

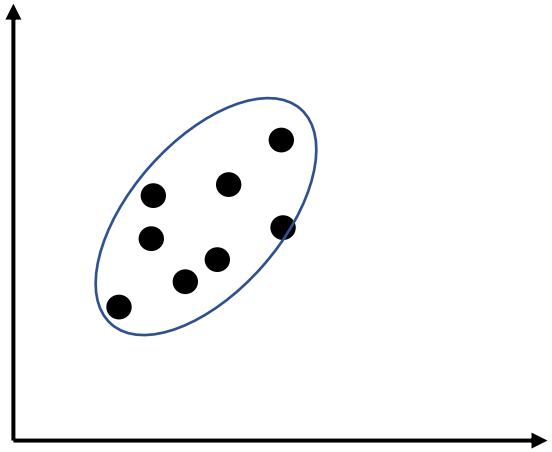
序号	面积(m^2)	房间数目	离地铁站距离（km）	交付房价(万/平米)
1	1.4	0.71	0.0	1.4
2	-0.71	-1.4	-1.2	-0.34
3	-0.71	0.71	1.2	-1

计算样本1和样本2的欧式距离

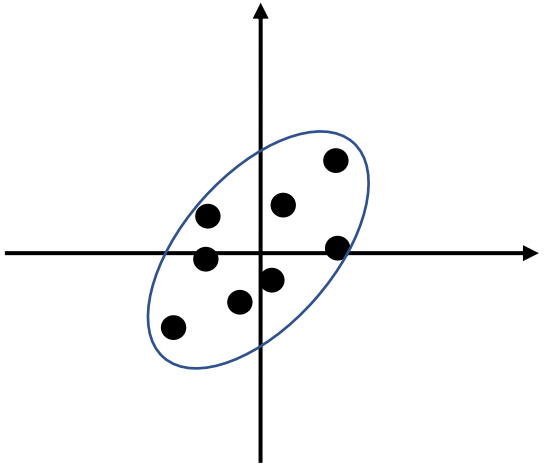
$$d_{12} = \|x_1 - x_2\| = \sqrt{(1.4+0.71)^2+(0.71+1.4)^2+(0.0+1.2)^2+(1.4+0.34)^2}=3.66$$
$$> \sqrt{(1.4+0.71)^2}=2.11$$

中心化

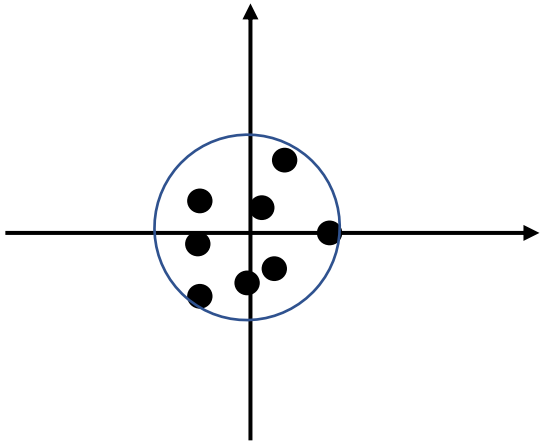
- 只是把均值变成0，方差不变，叫中心化。



原始数据



中心化后的数据
$(\boldsymbol{\mu}=0)$



规范化后的数据
$(\boldsymbol{\mu}=0,\ \boldsymbol{\sigma}^2=1)$

中心化后的方差和协方差

- 方差$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$，由于$\bar{x} = 0$，则中心化后数据的方差

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} x_i^2$$

- 协方差

$c_{ij} = \frac{1}{n}\sum_{h=1}^{n}(x_{hi} - \bar{x_i})(x_{hj} - \bar{x_j})$，由于中心化后每列均值为0，所以协方差为

$$c_{ij} = \frac{1}{n}\sum_{h=1}^{n} x_{hi}x_{hj}$$

对应的协方差矩阵为：$C = X^T X$ （这里$X_{n\times p}$为对象-属性矩阵）

拉格朗日乘子法（Lagrange Multipliers）

- 适用于等式约束下的优化问题:

$$\min_x f(x)$$
$$\text{s.t. } h_i(x) = 0 \ \ for \ \ i = 1,..q$$

- 基本思想：通过引入额外变量（拉格朗日乘子），把对具有约束条件的优化问题转换为无约束的优化问题来求解。
- 引入拉格朗日乘子$\alpha_i \geq 0$, 构建拉格朗日函数

$$\mathrm{L}(x, \alpha_i) = f(x) + \sum_{i=1}^{q} \alpha_i \ h_i(x)$$

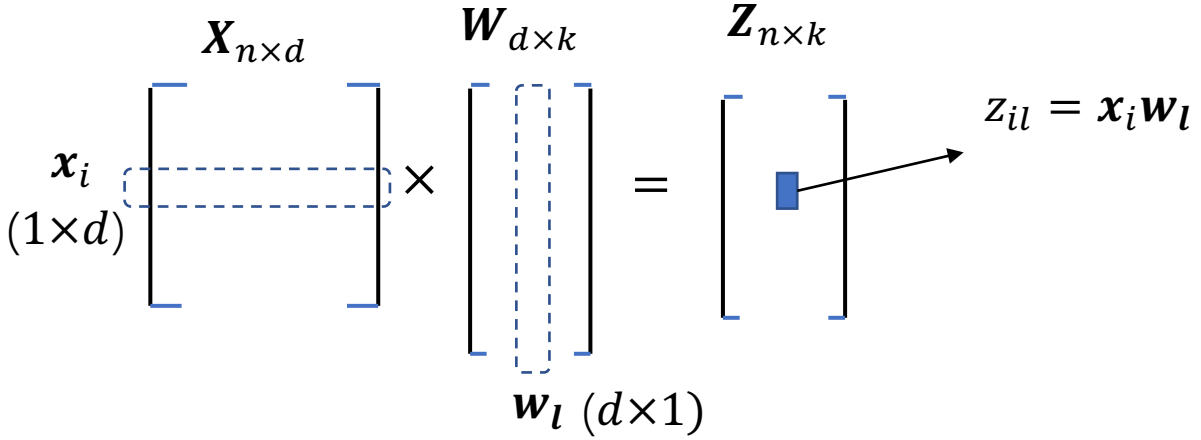
令$\mathrm{L}(x, \alpha_i)$对x和α_i的偏导为零

$$\frac{\partial L(x,\alpha_i)}{\partial x} = f'(x) + \sum_{i=1}^{q} \alpha_i \ h_i'(x) \ = 0$$

对每个 i，令 $\frac{\partial L(x,\alpha_i)}{\partial \alpha_i} = h_i(x) = 0$

投影后均值的计算

假设数据集表示为对象-属性矩阵$\boldsymbol{X}_{n\times d}$，通过矩阵$\boldsymbol{W}_{d\times k}$把样本$\boldsymbol{x}_i$线性映射到$k$维向量$\boldsymbol{z}_i$。



推导： $\overline{z_l} = \overline{\boldsymbol{x}}\boldsymbol{w}_l$

$\overline{\boldsymbol{x}}$是$\boldsymbol{X}$按列求均值得到的横向量；

$\overline{z_l} = \frac{1}{n}\sum_{i=1}^{n} z_{il}$为$\boldsymbol{Z}$第$l$列的均值。

$$\overline{z_l} = \frac{1}{n}\sum_{i=1}^{n} z_{il}$$
$$= \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\,\boldsymbol{w}_l$$
$$= \left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}_i\right)\boldsymbol{w}_l$$
$$= \overline{\boldsymbol{x}}\boldsymbol{w}_l$$