数据挖掘与机器学习

潘斌

panbin@nankai.edu.cn

范孙楼227

上节回顾

- 数据预处理的方法
  - 汇总统计
  - 可视化

本节提要

- 降维（特征提取、特征选择）
- 概念学习

数据预处理中的特征提取

- 维归约（特征提取）
  - 维度较低时，许多数据挖掘算法的效果会更好
    - 删除不相关的特征，降低噪声，避免维数灾难
    - 降低了算法的时间和内存需求
    - 模型更易理解，容易让数据可视化
  - 维归约常用方法
    - 主成分分析（PCA）
    - 线性判别分析（LDA)

- 特征提取问题
  - 已知给定的M个原始特征
  - 经过数学变换得到m个特征（m < M）

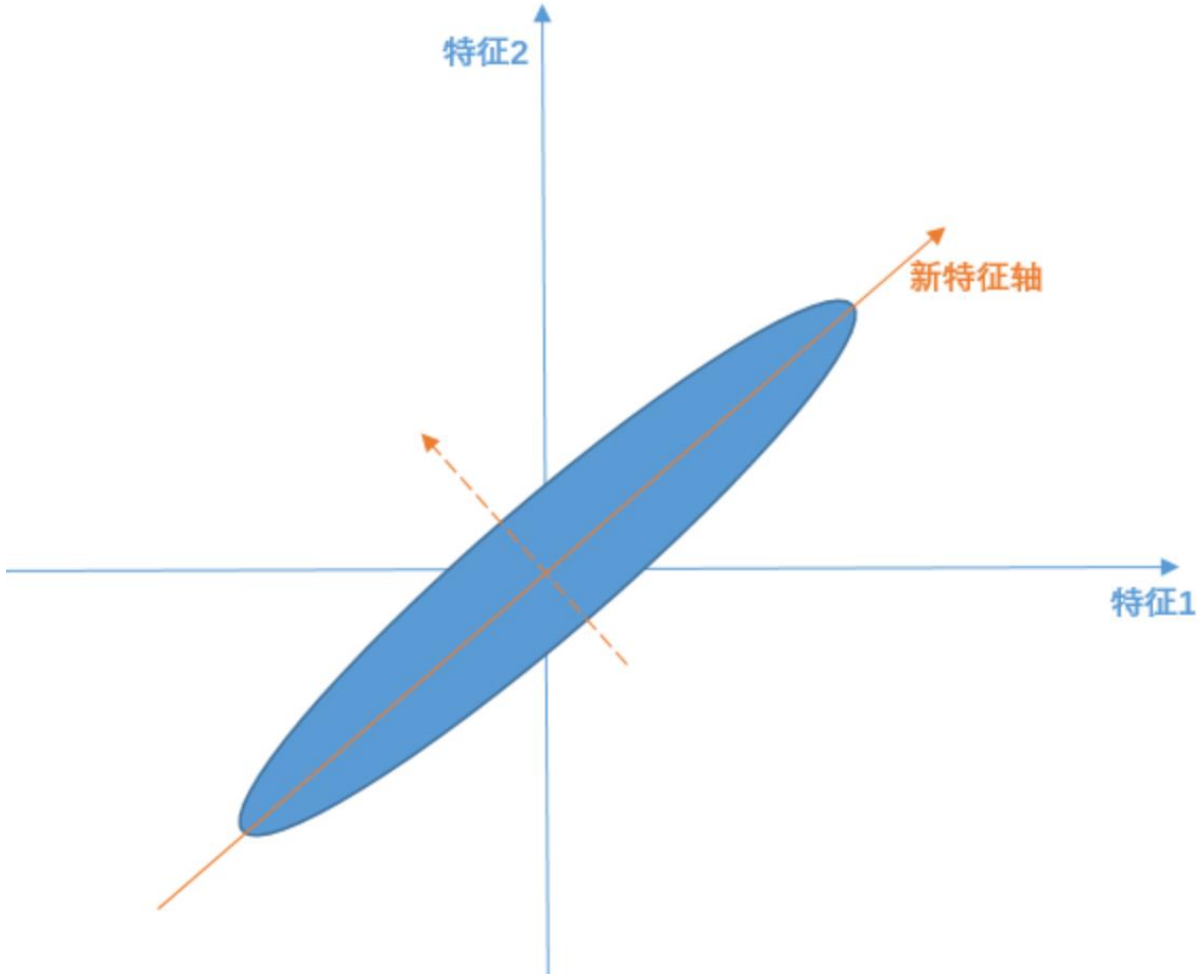
PCA

- 主成分分析
  - Principal Component Analysis (PCA)

  - 已知给定的M个原始特征
  - 经过线性组合（正交）变换，得到一组按“重要性”从大到小排列的特征
  - 取前m个特征

PCA

主轴方向方差大

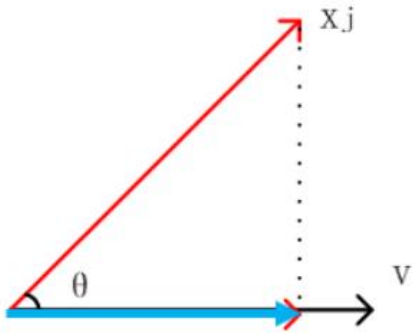


PCA

- **PCA问题**
  - 设原始特征向量x，维数为M，线性组合的新特征向量y
    - y的各分量 $y_i = \sum_{j=1}^{m} a_{ij} x_j = a_i^T x$
    - 基本约束条件 $a_i^T a_i = 1$
    - 矩阵形式 $y = A^T x$
  - 求最佳的正交变换矩阵A，使得新特征的方差达到极值

PCA

◆投影数据

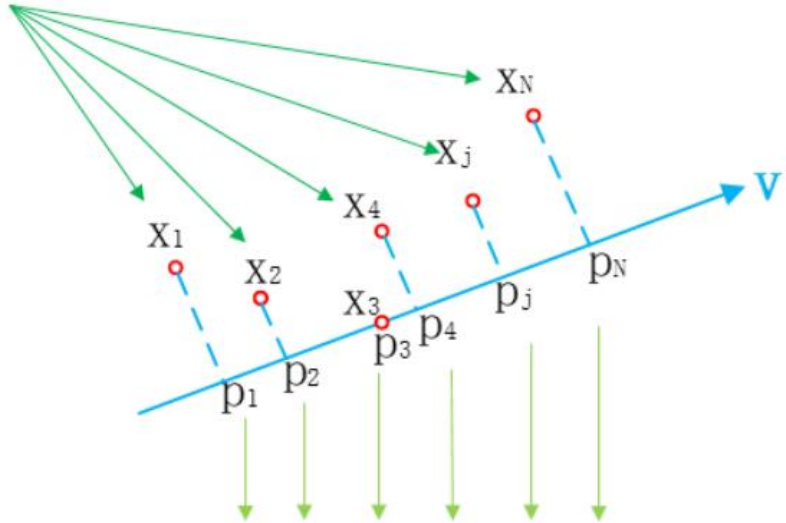


◆xj的投影长度

$$\|X_j\| \cdot \cos\theta = \|X_j\| \cdot \frac{\langle X_j, V \rangle}{\|X_j\| \cdot \|V\|} = \frac{\langle X_j, V \rangle}{\|V\|} = \langle X_j, V \rangle$$

◆样本数据，n维 $X_j = \left[X_{j1}, X_{j2}, \cdots X_{jn}\right]^T$

投影

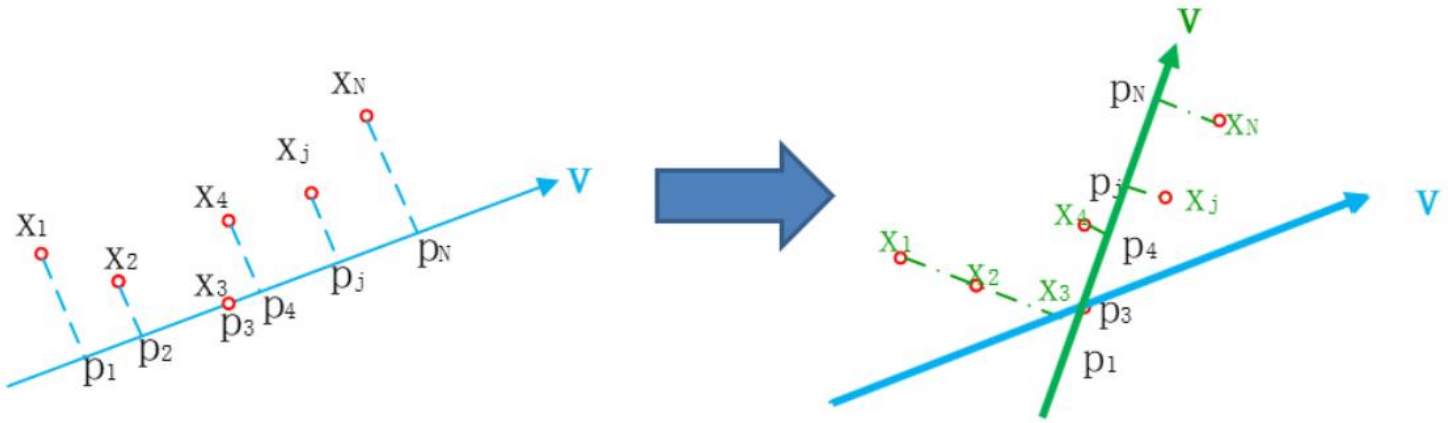


向量内积 → $p_1, p_2, \cdots, p_N$

◆投影数据，N个

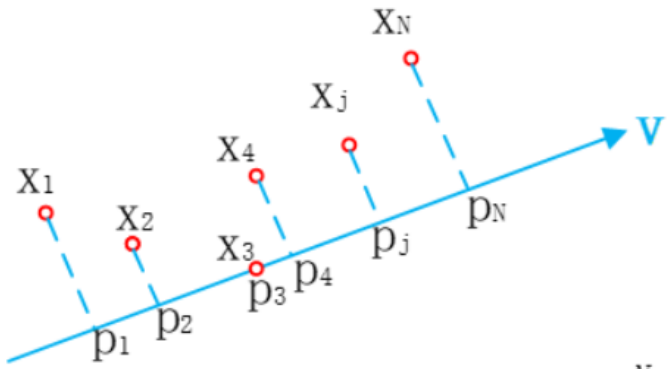
PCA

- 重要：PCA用投影数据的方差来表征原始数据的信息大小——投影数据的方差越大，表明数据分散程度越大，其中包含的信息量越大。
- 考虑到，沿不同的向量v进行投影，将得到不同的投影数据，那么投影数据方差也将发生变化，即其所表征的信息大小也将发生变化。



PCA

- 表征原始数据的信息大小——投影数据的方差



投影数据去除均值

$$\sigma^2 = \frac{1}{N}\cdot\sum_{i=1}^{N}\left(v^T X_j - 0\right)^2 = \frac{1}{N}\cdot\sum_{i=1}^{N}\left(v^T X_j \cdot v^T X_j\right)$$

$$= \frac{1}{N}\cdot\sum_{i=1}^{N}\left(v^T X_j \cdot X_j^T v\right) = v^T\left(\frac{1}{N}\cdot\sum_{i=1}^{N}\left(X_j \cdot X_j^T\right)\right)v$$

$$= v^T\left(C_x\right)v$$

PCA

- 优化问题建模
- 为了尽可能大地反映样本数据的信息，我们需要确定某个单位向量v，使得样本数据在其上的投影数据具有最大方差。建模如下：

$$\begin{cases} \max\limits_{v} \sigma^2 = \max v^T C_X v \\ s.t. \ \|v\| = 1 \end{cases}$$

PCA

- 优化问题求解
- 利用“拉格朗日方程”求解该最优化：

$$f(v,\lambda) = v^T C_x v - \lambda(\|v\| - 1) = v^T C_x v - \lambda(v^T v - 1)$$

$$\begin{cases} \dfrac{\partial}{\partial v} f(v,\lambda) = 2C_x v - 2\lambda v = 0 \\ \dfrac{\partial}{\partial \lambda} f(v,\lambda) = v^T v - 1 = 0 \end{cases}$$

$$\begin{cases} C_x v = \lambda v \\ v^T v = 1 \end{cases}$$

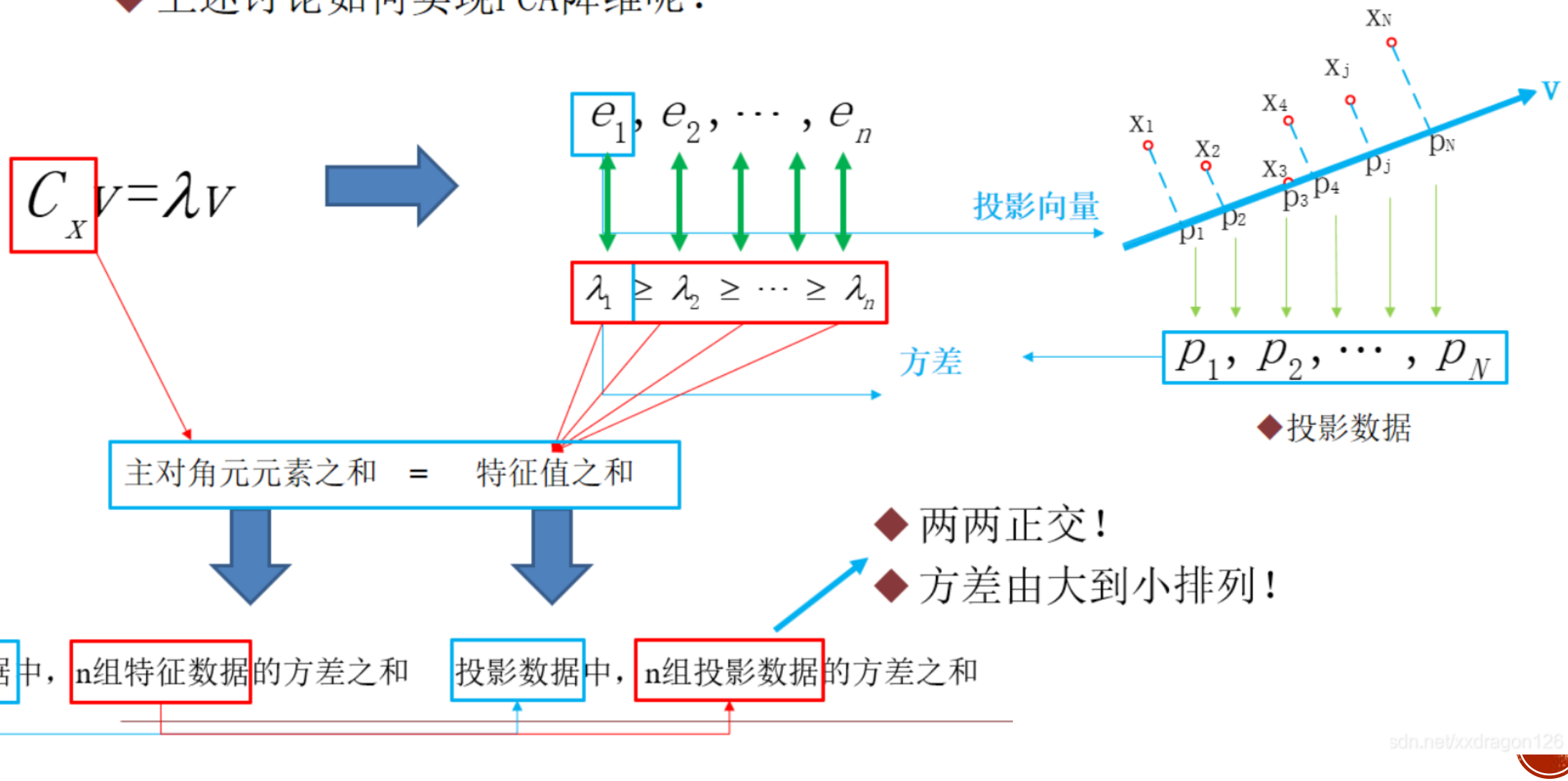
v是Cx的特征向量

$$\max \sigma^2 = \max v^T C_x v = \max v^T \lambda v = \max \lambda$$

方差是Cx的特征向量

PCA

◆ 上述讨论如何实现PCA降维呢？

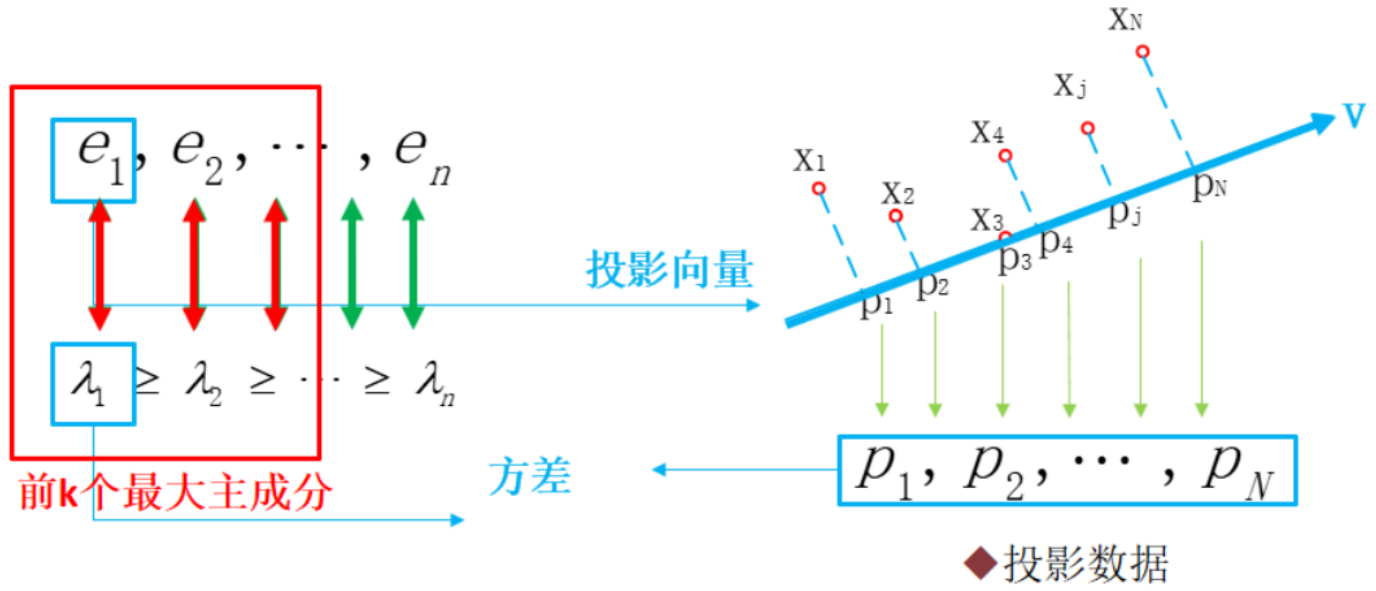


PCA(缺点)

◆ 小结PCA降维过程

$$r = \frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{n} \lambda_j}$$

方差占比（贡献率）



◆投影数据

◆ 即，这k组投影数据对原始数据的表征能力为r(称为“贡献率”)。

◆ 特别地，当k取n时，r=1，意味着这k个主成分能够完全表征原始数据。

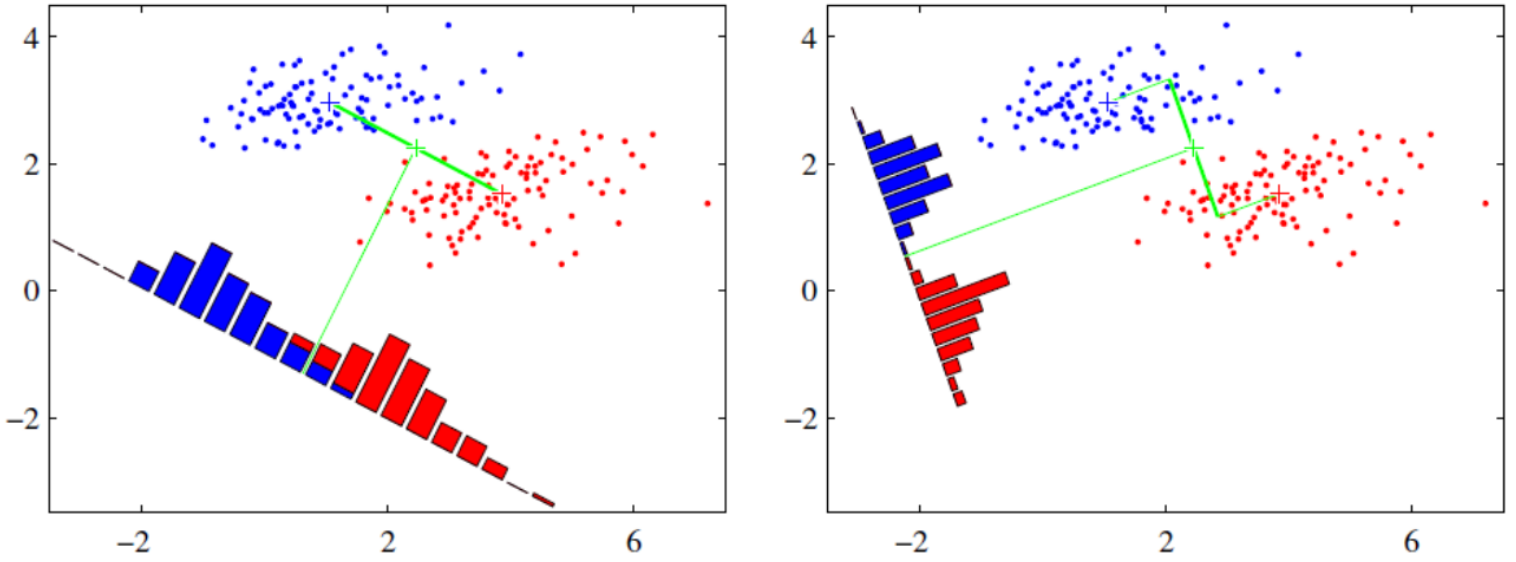
此时即完成了将原始n维数据降维成新的k(k≤n)组特征数据。

LDA

- **Fisher投影准则**
  - 已知给定的M个原始特征
  - 经过数学投影得到1个特征
  - 求最佳投影向量p^*

$$\max J_F(p) = \frac{p^T S_B p}{p^T S_W p}$$

同类尽可能近，不同类尽可能远

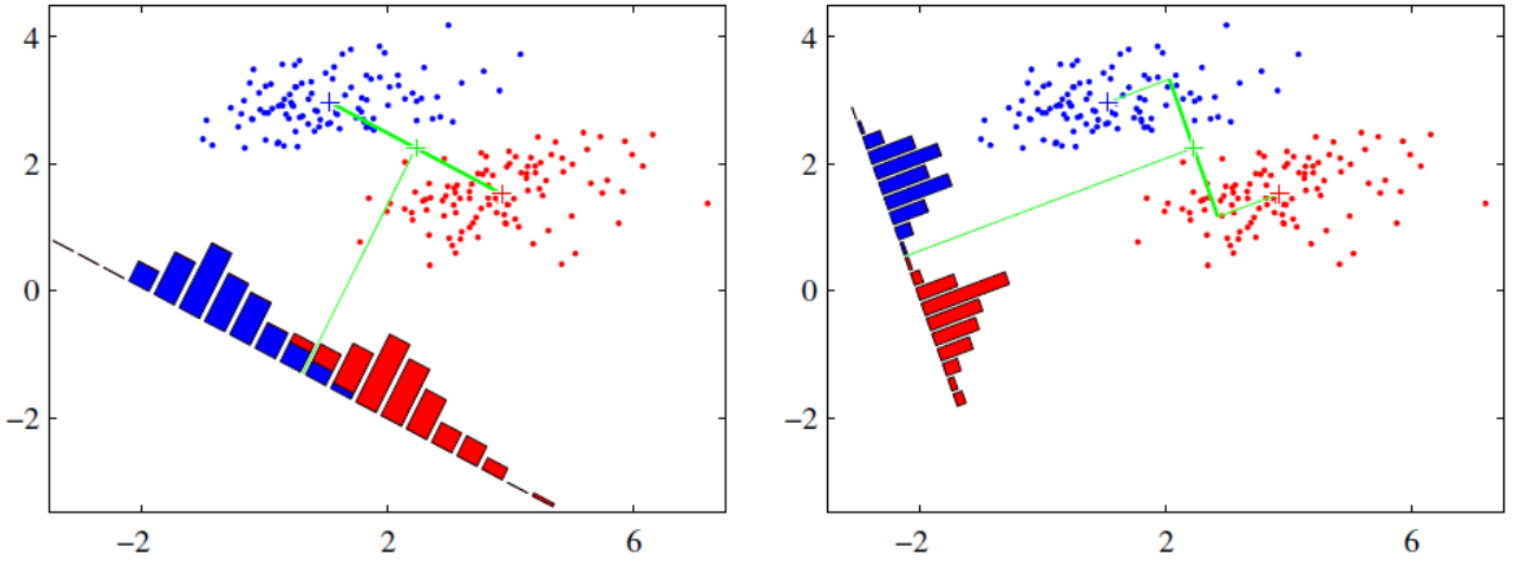


LDA

- **LDA**投影准则
  - 已知给定的M个原始特征
  - 经过数学投影得到m个特征
  - 求最佳投影矩阵P*

$$\max J_F(p) = \frac{p^T S_B p}{p^T S_W p}$$

同类尽可能近，不同类尽可能远



LDA

- 优化准则

$$\begin{aligned}\mathbf{S}_w &= \boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1 \\ &= \sum_{\boldsymbol{x}\in X_0} (\boldsymbol{x}-\boldsymbol{\mu}_0)(\boldsymbol{x}-\boldsymbol{\mu}_0)^{\mathrm{T}} + \sum_{\boldsymbol{x}\in X_1} (\boldsymbol{x}-\boldsymbol{\mu}_1)(\boldsymbol{x}-\boldsymbol{\mu}_1)^{\mathrm{T}}\end{aligned}$$

$$\mathbf{S}_b = (\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)^{\mathrm{T}}$$

令分母为1

$$\begin{aligned}\min_{\boldsymbol{w}} \quad & -\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_b\boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^{\mathrm{T}}\mathbf{S}_w\boldsymbol{w} = 1\end{aligned}$$

拉格朗日对偶乘子法

数据预处理中的特征选择

- 特征子集选择（特征选择）
  - 降低维度的又一方法
  - 冗余特征（Redundant features ）
    - 重复包含在一个或多个其他属性中的许多或所有信息
      - 一种产品的购买价格和所支付的销售税额
  - 不相关特征（Irrelevant features）
    - 包含对于相关数据挖掘任务几乎完全没用的信息
      - 学生的ID号码对于预测学生总平均成绩

几种常见的子集选择方法

- 暴力（Brute-force）方法
  - 将所有可能的特征子集作为感兴趣的数据挖掘算法的输入，然后选择产生最好结果的子集
- 过滤（Filter）方法
  - 在数据挖掘算法运行前进行特征选择
- 包装（Wrapper）方法
  - 将数据挖掘算法作为黑盒子找到最好的属性子集，通常并不枚举
- 嵌入（Embedded）方法
  - 特征选择自然的作为数据挖掘算法的一部分，算法本身决定使用哪些属性和忽略哪些属性

- 过滤算法
  - Relief(Kira &Rendell,1992)
    - 设计一个“相关统计量”来度量特征的重要性（如：单特征分类正确率）
      - 一个向量，每个分量对应一个初始特征
      - 特征子集的重要性由相应相关统计量分量之和来决定
    - 选择方法
      - 指定一个阈值，选择比其大的分量对应的特征
      - 指定子集中特征的个数k，选择最大的k个特征

- 确定相关统计量

给定训练集 $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$, 对每个示例 $\boldsymbol{x}_i$,

"猜中近邻" (near-hit): $\boldsymbol{x}_i$ 的同类样本中的最近邻 $\boldsymbol{x}_{i,\mathrm{nh}}$
"猜错近邻" (near-miss): $\boldsymbol{x}_i$ 的异类样本中的最近邻 $\boldsymbol{x}_{i,\mathrm{nm}}$

相关统计量对应于属性 j 的分量为

$$\delta^j = \sum_i -\mathrm{diff}(x_i^j, x_{i,\mathrm{nh}}^j)^2 + \mathrm{diff}(x_i^j, x_{i,\mathrm{nm}}^j)^2 \ ,$$

其中 x_a^j 表示样本 $\boldsymbol{x}_a$ 在属性 j 上的取值, $\mathrm{diff}(x_a^j, x_b^j)$ 取决于属性 j 的类型: 若属性 j 为离散型, 则 $x_a^j = x_b^j$ 时 $\mathrm{diff}(x_a^j, x_b^j) = 0$, 否则为 1; 若属性 j 为连续型, 则 $\mathrm{diff}(x_a^j, x_b^j) = |x_a^j - x_b^j|$, 注意 x_a^j, x_b^j 已规范化到 $[0, 1]$ 区间.

包装算法

- LVW(Las Vegas Wrapper, Liu & Setiono,1996)

- 本质上是对特征集进行有放回采样，找到所有采样中的最优值

输入：数据集 D;
特征集 A;
学习算法 $\mathfrak{L}$;
停止条件控制参数 T.

过程：

```
E = ∞;
d = |A|;
A* = A;
t = 0;
while t < T do
    随机产生特征子集 A';
    d' = |A'|;
    E' = CrossValidation(L(D^A'));
    if (E' < E) ∨ ((E' = E) ∧ (d' < d)) then
        t = 0;
        E = E';
        d = d';
        A* = A'
    else
        t = t + 1
    end if
end while
```

输出：特征子集 A^*

LVW 算法描述

初始化

在特征子集A'上通过交叉验证估计学习器误差

嵌入算法

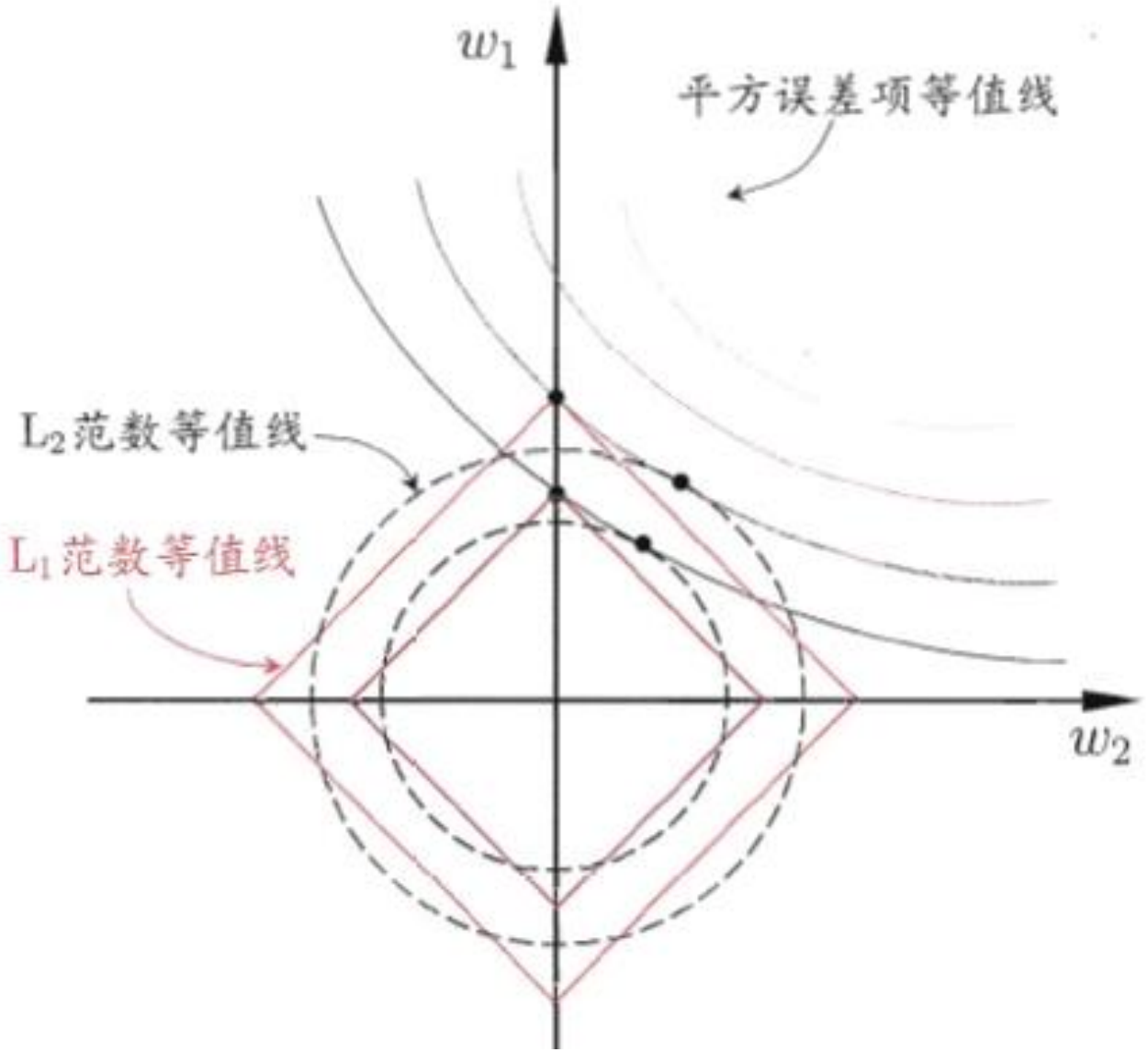
给定数据集 $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)\}$, 其中 $\boldsymbol{x} \in \mathbb{R}^d, y \in \mathbb{R}$. 考虑最简单的线性回归模型, 优化目标为

$$\min_{\boldsymbol{w}} \sum_{i=1}^{m} (y_i - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i)^2 \ .$$

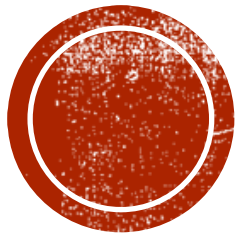
采用 L_1 范数, 则有

$$\min_{\boldsymbol{w}} \sum_{i=1}^{m} (y_i - \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}_i)^2 + \lambda \|\boldsymbol{w}\|_1 \ .$$

其中正则化参数 $\lambda > 0$. 称为 LASSO (Least Absolute Shrinkage and Selection Operator) [Tibshirani, 1996]).



- L1范数切点在轴上，故稀疏
- L2范数切点在象限内，故无稀疏
- L0范数在哪？



概念学习

Concept learning

- 概念（concept）：一个对象或事件集合，它是从更大的集合中选取的子集，或者是在这个较大集合中定义的布尔函数。
  - 如，从动物的集合中选取鸟类
    - 在动物集合中定义的函数，它对鸟类产生true 并对其他动物产生false
- 概念学习：从样例中逼近布尔值函数。是指从有关某个布尔函数的输入输出训练样例中，推断出该布尔函数。

- 一个概念学习的例子：Aldo Enjoy Sport

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- 输入某天的各种属性Sky、AirTemp、Humidity…等条件，Aldo是否进行水上运动?

- 术语定义
  - 概念定义在一个实例（instance）集合之上，这个集合表示为X。
    - 例中，X是所有可能的日子，每个日子由Sky、AirTemp、Humidity、Wind、Water和Forecast六个属性表示。
  - 待学习的概念或函数称为目标概念（target concept），记作c。
    - c可以是定义在实例X上的任意布尔函数，即c:X→{0, 1}。
    - 例中，目标概念对应于属性EnjoySport的值，当EnjoySport=Yes时c(x)=1，当EnjoySport=No时c(x)=0。

- 训练样例（training examples）：X中的一个实例x以及它的目标概念值c(x)。
  - 用序偶<x,c(x)>来描述训练样例
  - 符号D用来表示训练样例的集合。
- 正例（positive example）：对于c(x)=1的实例，也称为目标概念的成员。
- 反例（negative example）：对于c(x)=0的实例，也称为非目标概念成员。

- 给定目标概念c的训练样例集，学习器面临的问题是假设或估计c。
- 使用符号H来表示所有可能假设的集合，称之为假设空间。
  - 通常H依设计者所选择的假设表示而定。
  - H中每个假设h表示X上定义的布尔函数，即h:X→{0,1}。
  - 目标：寻找H中的假设h，使对于X中的所有x，h(x)=c(x)。

- 获取h的方法：归纳学习
- 归纳学习：从特殊的样例得到普遍的规律
  - 归纳学习算法最多只能保证输出的假设与训练样例（特殊）相拟合。
- 归纳学习假设（The Inductive Learning Hypothesis）：任一假设如果在足够大的训练样例集中很好地逼近目标函数，它也能在未见实例中很好地逼近目标函数。（数据同分布）

- 独立同分布
- 独立：每一个样本的出现或者生成，都是独立事件，即任意两个样本之间不相关
- 同分布：抽样内样本服从总体的分布（文字识别不平衡）

- 表示假设（Representing Hypotheses）
  - 很多种可能的表示方法
  - 一个简单的形式：实例的各属性约束的合取式^（Conjunction）
  - EnjoySport例中，令每个假设为6个约束（或变量）的向量，每个约束对应一个属性可取值范围，为
    - 由“?”表示任意值（如，AirTemp=?）
    - 明确指定的属性值（如，AirTemp=Warm）
    - 由“ø”表示不接受任何值（如，AirTemp=ø）
    - 如，
      - < Sunny , ?, ?, Strong , ? , Same >
      - <?, ?, ?, ?, ?, ?>　　所有的样例都是正例（最一般，任取皆正）
      - < ø, ø, ø, ø, ø, ø >　所有的样例都是反例（最特殊，任取皆零）

- 任何概念学习任务能被描述为
  - 实例的集合（总体）
  - 实例集合上的目标函数（目标）
  - 训练样例的集合（样本）
  - 候选假设的集合（假设）
- EnjoySport 概念学习任务

EnjoySport 概念学习任务

- 已知：
  - 实例集 X：可能的日子，每个日子由下面的属性描述：
    - *Sky*（可取值为 *Sunny*，*Cloudy* 和 *Rainy*）
    - *AirTemp*（可取值为 *Warm* 和 *Cold*）
    - *Humidity*（可取值为 *Normal* 和 *High*）
    - *Wind*（可取值为 *Strong* 和 *Weak*）
    - *Water*（可取值为 *Warm* 和 *Cool*）
    - *Forecast*（可取值为 *Same* 和 *Change*）
  - 假设集 H：每个假设描述为 6 个属性 *Sky*，*AirTemp*，*Humidity*，*Wind*，*Water* 和 *Forecast* 的值约束的合取。约束可以为“?”（表示接受任意值），“∅”（表示拒绝所有值），或一特定值。
  - 目标概念 c: *EnjoySport*: $X \rightarrow \{0, 1\}$
  - 训练样例集 D：目标函数的正例和反例
- 求解：
  - H 中的一假设 h，使对于 X 中任意 x，$h(x)=c(x)$。

- 当假设的表示形式选定后，那么就隐含地为学习算法确定了所有假设的空间
  - 这些假设是学习程序所能表示的
  - 也是它能够学习的
  - 例，Enjoy-Sport的假设空间
- 概念学习可以看作一个搜索的过程
  - 搜索范围：假设的表示所隐含定义的整个空间
  - 搜索目标：能够最好地拟合训练样例的假设

- 假设的一般到特殊序关系
  - 考虑下面两个假设
    - h1=<sunny, ?, ?, Strong, ?, ?>
    - h2=<Sunny, ?, ?, ?, ?, ?>
  - 任何被h1划分为正例的实例都会被h2划分为正例，因此h2比h1更一般。
- 利用这个关系，无需列举所有假设，就能在无限的假设空间中进行较彻底的搜索

- more-general-than-or-equal-to（更一般或相等）
- 定义：令h_j和h_k为在X上定义的布尔函数。定义一个more-general-than-or-equal-to关系，记做$\geq_g$。称$h_j \geq_g h_k$当且仅当

$$(\forall x \in X)[(h_k(x)=1) \rightarrow (h_j(x)=1)]$$

- 对X中任意实例x和H中任意假设h，我们说x满足h当且仅当h(x)=1。
- 给定假设h_j和h_k，h_j more-general-than-or-equal-to h_k，当且仅当任意一个满足h_k的实例同时也满足h_j。
- h_j严格的more-general-than h_k（写作$h_j >_g h_k$），当且仅当$(h_j \geq_g h_k) \wedge \neg(h_k \geq_g h_j)$。
- 逆向的关系“比……更特殊”：h_j more-specific-than h_k，当h_k more-general-than h_j。

- Instance, Hypotheses, and More-General-Than

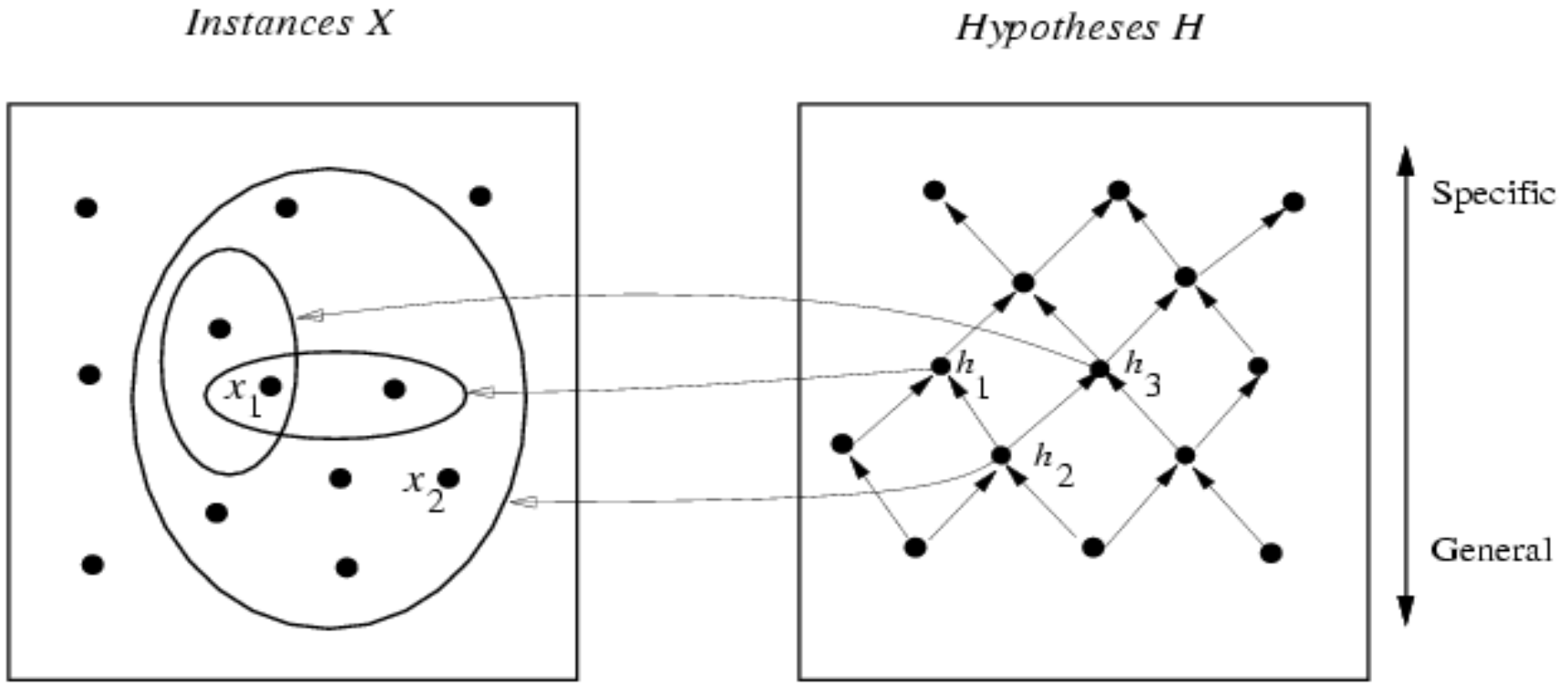
$h_1 = \langle Sunny, ?, ?, Strong, ?, ? \rangle$

$h_2 = \langle Sunny, ?, ?, ?, ?, ? \rangle$

$h_3 = \langle Sunny, ?, ?, ?, Cool, ? \rangle$

$x_1 = \langle Sunny, Warm, High, Strong, Cool, Same \rangle$

$x_2 = \langle Sunny, Warm, High, Light, Warm, Same \rangle$



x_1= <Sunny, Warm, High, Strong, Cool, Same>

x_2= <Sunny, Warm, High, Light, Warm, Same>

h_1= <Sunny, ?, ?, Strong, ?, ?>

h_2= <Sunny, ?, ?, ?, ?, ?>

h_3= <Sunny, ?, ?, ?, Cool, ?>

- 练习：给出下列假设的偏序关系。

h1: <*Sunny, Warm, ?, Strong, ?, ?*>

h2: <*Sunny, ?, ?, Strong, ?, ?*>

h3: <*Sunny, Warm, ?, ?, ?, ?*>

h4: <*?, Warm, ?, Strong, ?, ?*>

h5: <*Sunny, ?, ?, ?, ?, ?*>

h6: <*?, Warm, ?, ?, ?, ?*>

- 一致（Consistent）
- 定义：一个假设h与训练样例集合D一致，当且仅当对D中每一个样例<x,c(x)>，h(x)=c(x)。

$$Consistent(h,D) \equiv (\forall \langle x,c(x)\rangle \in D)\ h(x)=c(x)$$

  - 与“满足”不同
    - 一个样例x在h(x)=1时称为满足假设h，不论x是目标概念的正例还是反例。
    - 这一样例是否与h一致与目标概念有关，即是否h(x)=c(x)。

- 求解h的算法
  - **Find-S Algorithm**
    - 基本思想：
      - 使用more_general_than偏序的搜索算法
      - 沿着偏序链，从较特殊的假设逐渐转移到较一般的假设。
      - 在每一步，假设只在需要覆盖新的正例时被泛化。
      - 每一步得到的假设，都是在那一点上与训练样例一致的最特殊的假设。

- Find-S Algorithm
  - 将h初始化为H中最特殊假设
  - 对每个正例x
    - 对h的每个属性约束ai
    - 如果x满足ai
    - 那么不做任何事
    - 否则，将h中ai替换为x满足的紧邻的更一般约束
  - 输出假设h

Find-S算法实例

- 将h初始化为H中最特殊假设
- 对每个正例x
  - 对h的每个属性约束ai
  - 如果x满足ai
  - 那么不做任何事
  - 否则，将h中ai替换为x满足的紧邻的更一般约束
- 输出假设h

$h_0 \leftarrow \langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle$

$x_1 = \langle Sunny\ Warm\ Normal\ Strong\ Warm\ Same \rangle, +$

$h_1 \leftarrow \langle Sunny, Warm, Normal, Strong, Warm, Same \rangle$

$x_2 = \langle Sunny\ Warm\ High\ Strong\ Warm\ Same \rangle, +$

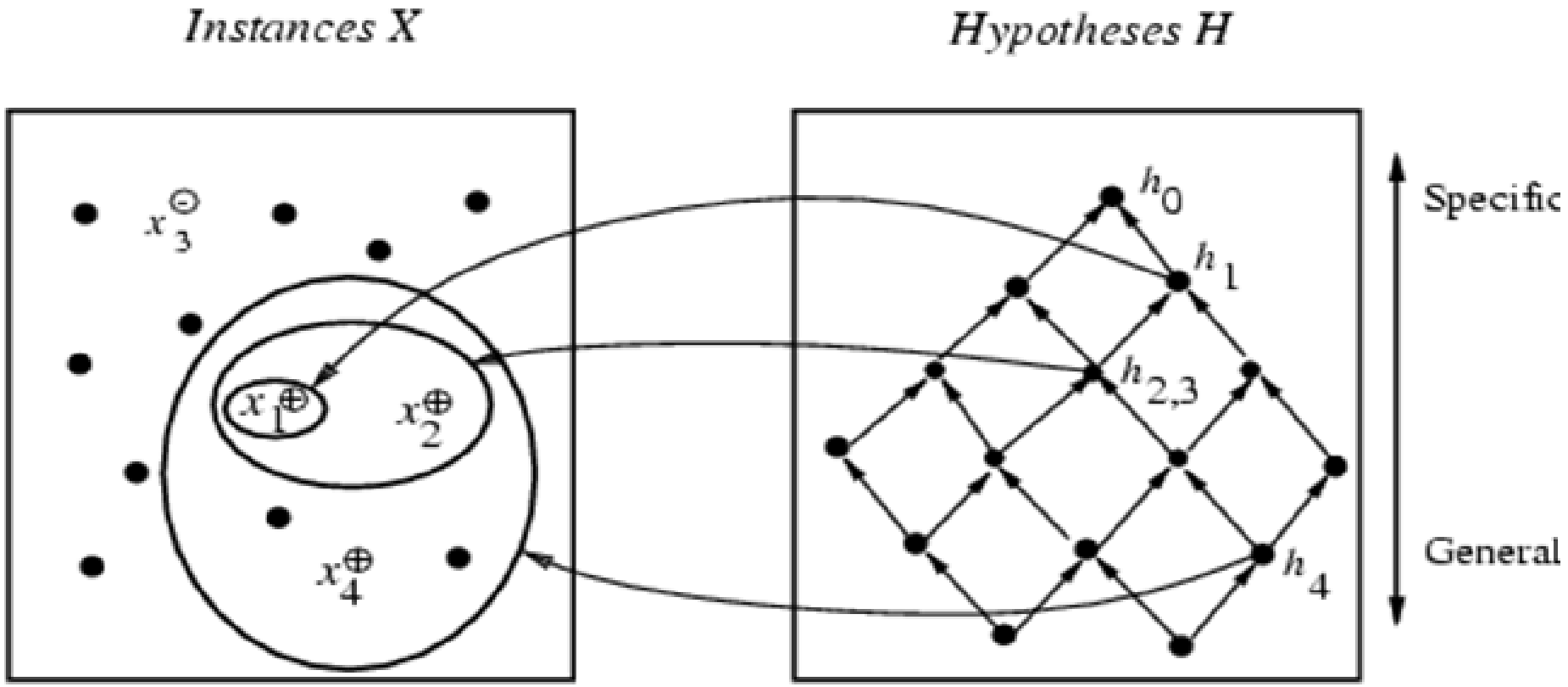
$h_2 \leftarrow \langle Sunny, Warm, ?, Strong, Warm, Same \rangle$

$x_3 = \langle Rainy\ Cold\ High\ Strong\ Warm\ Change \rangle, -$

$h_3 \leftarrow \langle Sunny, Warm, ?, Strong, Warm, Same \rangle$

$x_4 = \langle Sunny\ Warm\ High\ Strong\ Cool\ Change \rangle, +$

$h_4 \leftarrow \langle Sunny, Warm, ?, Strong, ?, ? \rangle$


Instances X
Hypotheses H
$x_3^{\ominus}$
$x_1^{\oplus}$
$x_2^{\oplus}$
$x_4^{\oplus}$
h_0
h_1
$h_{2,3}$
h_4
Specific
General

练习

Outlook	Temperature	Humidity	Wind	PlayTennis
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Mild	High	Strong	No
Sunny	Cool	Normal	Weak	Yes
Overcast	Hot	Normal	Weak	Yes

- Find-S学习算法的特点及不足
  - Find-S的重要特点：对以属性约束的合取式^（conjunction）描述的假设空间H，保证输出为H中与正例一致的最特殊的假设。
  - 存在的问题（反例；容错性）

- 变型空间（Version space，版本空间）
  - 定义：关于假设空间H和训练样例集D的变型空间，标记为$VS_{H,D}$，是H中与训练样例集D一致的所有假设构成的子集。

$$VS_{H,D} = \{h \in H \mid Consistent(h,D)\}$$

  - 与训练样例集一致的所有假设组成的集合
  - 包含的是目标概念的所有合理的变型

- 求解h的算法
  - 列表后消除算法（The List-Then-Eliminate Algorithm）
    - 变型空间VersionSpace←包含H中所有假设的列表
    - 对每个训练样例<x, c(x)> 从变型空间中移除所有h(x)≠c(x)的假设h
    - 输出VersionSpace中的假设列表
      - 只要假设空间是有限的，就可使用
      - 保证得到所有与训练数据一致的假设
      - 非常繁琐地列出H中的所有假设，大多数实际的假设空间无法做到

- 变型空间举例
  - EnjoySport

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

结果如→

每来一个样本，压缩一次假设空间

h1: <*Sunny, Warm, ?, Strong, ?, ?*>

h2: <*Sunny, ?, ?, Strong, ?, ?*>

h3: <*Sunny, Warm, ?, ?, ?, ?*>

h4: <*?, Warm, ?, Strong, ?, ?*>

h5: <*Sunny, ?, ?, ?, ?, ?*>

h6: <*?, Warm, ?, ?, ?, ?*>

- 求解h的算法：候选消除算法
  - 变型空间的两个边界定义
    - 一般边界（General Boundary）G：
      - H中与D（训练集）相一致的极大一般成员的集合。

$$G \equiv \{ g \in H \mid Consistent(g, D) \wedge (\neg \exists g' \in H)[(g' >_g g) \wedge Consistent(g', D)]\}$$

    - 特殊边界（Specific Boundary）S:
      - 在H中与D相一致的极大特殊成员的集合。

$$S \equiv \{ s \in H \mid Consistent(s, D) \wedge (\neg \exists s' \in H)[(s >_g s') \wedge Consistent(s', D)]\}$$

  - 从变型空间的边界慢慢往中间找

求解h的算法

候选消除算法

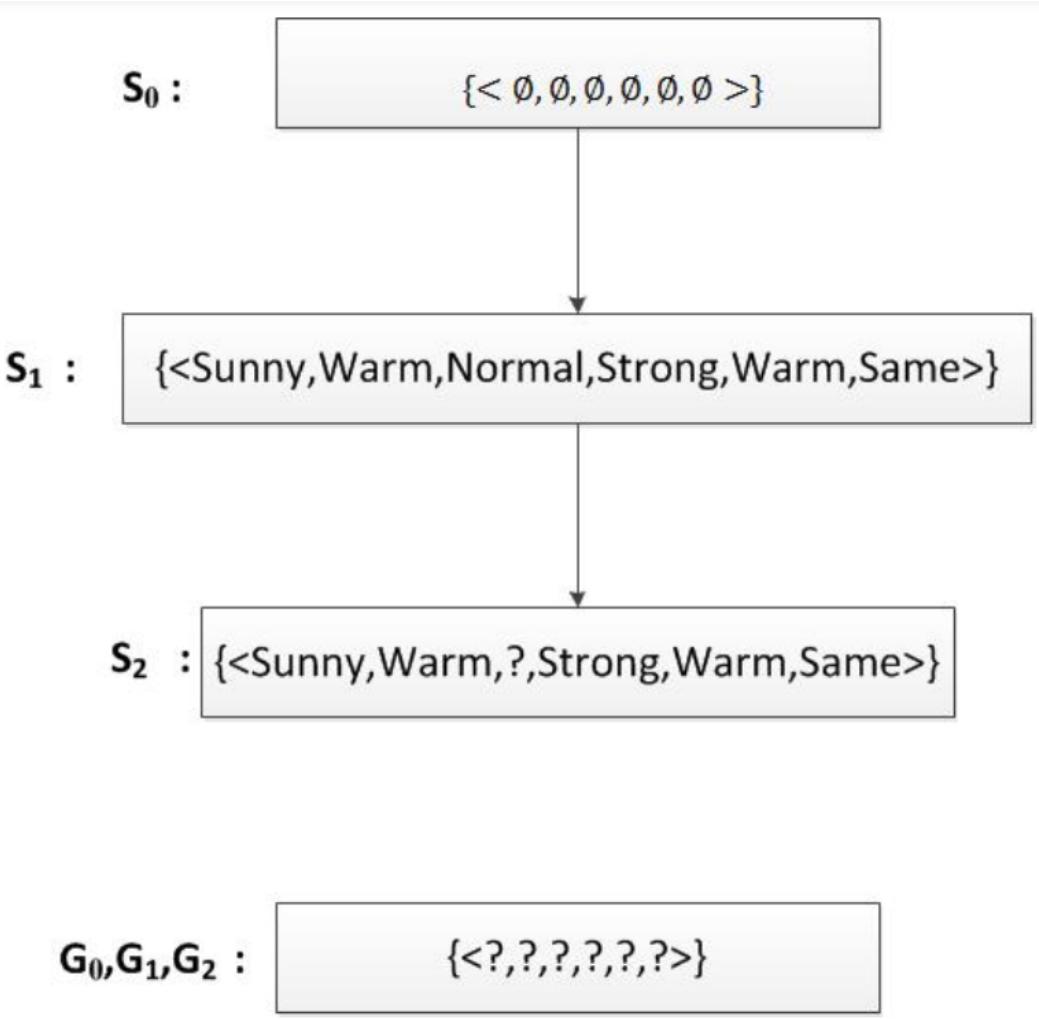
将G集合初始化为H中极大一般假设

将S集合初始化为H中极大特殊假设

对每个训练例d，进行以下操作：

- 如果d是一正例
  - 从G中移去所有与d不一致的假设
  - 对S中每个与d不一致的假设s
    - 从S中移去s
  - 把s的所有的极小一般化式h加入到S中，其中h满足
    - h与d一致，而且G的某个成员比h更一般
  - 从S中移去所有这样的假设：它比S中另一假设更一般
- 如果d是一个反例
  - 从S中移去所有d不一致的假设
  - 对G中每个与d不一致的假设g
    - 从G中移去g
  - 把g的所有的极小特殊化式h加入到G中，其中h满足
    - h与d一致，而且S的某个成员比h更特殊
  - 从G中移去所有这样的假设：它比G中另一假设更特殊

- 如果d是一正例
  - 从G中移去所有与d不一致的假设
  - 对S中每个与d不一致的假设s
    - 从S中移去s
  - 把s的所有的极小一般化式h加入到S中，其中h满足
    - h与d一致，而且G的某个成员比h更一般
  - 从S中移去所有这样的假设：它比S中另一假设更一般

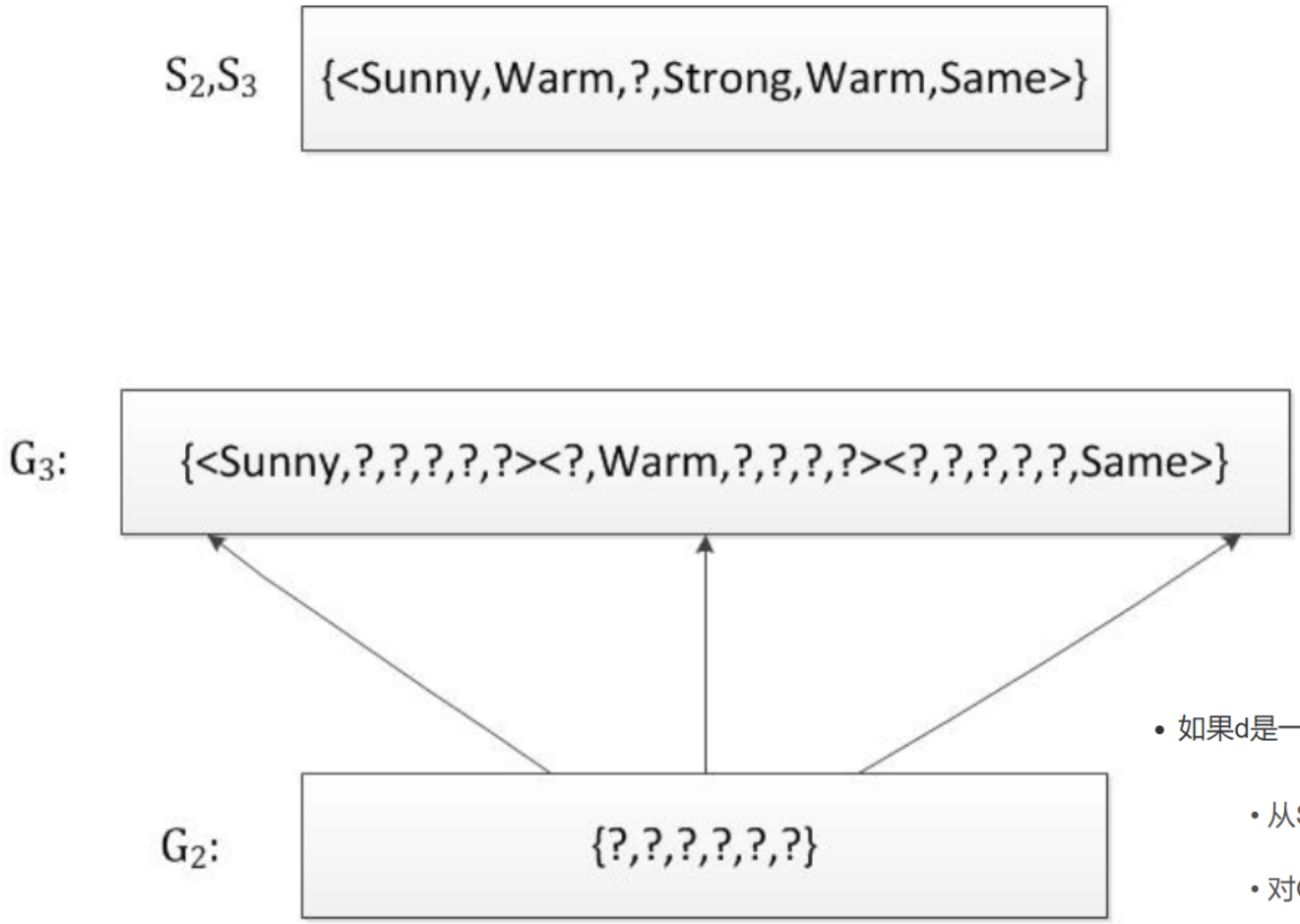


- 分别看是否满足S，G

训练样例：

1.<Sunny,Warm,Normal,Strong,Warm,Same>,EnjoySport=Yes

2.<Sunny,Warm,High,Strong,Warm,Same>,EnjoySport=Yes

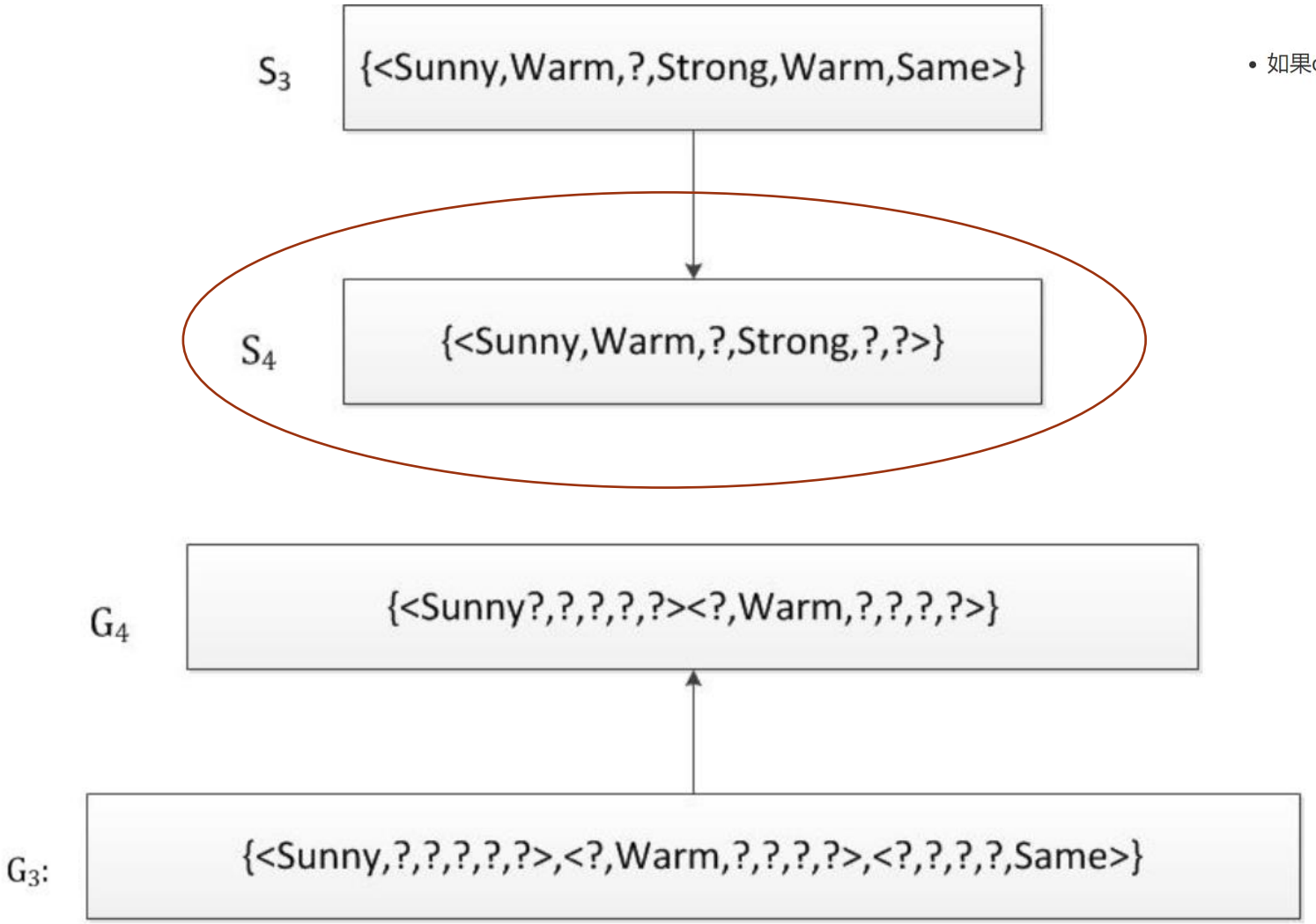


训练样例:

3.<Rainy,Cold,High,Strong,Warm,Change>,EnjoySport=No

- 分别看是否满足S，G
- G3中6选3
- 另外3个不在变型空间中
- G3中有多个可选的极大一般假设

- 如果d是一个反例
  - 从S中移去所有d不一致的假设
  - 对G中每个与d不一致的假设g
    - 从G中移去g
    - 把g的所有的极小特殊化式h加入到G中，其中h满足
      - h与d一致，而且S的某个成员比h更特殊
    - 从G中移去所有这样的假设：它比G中另一假设更特殊



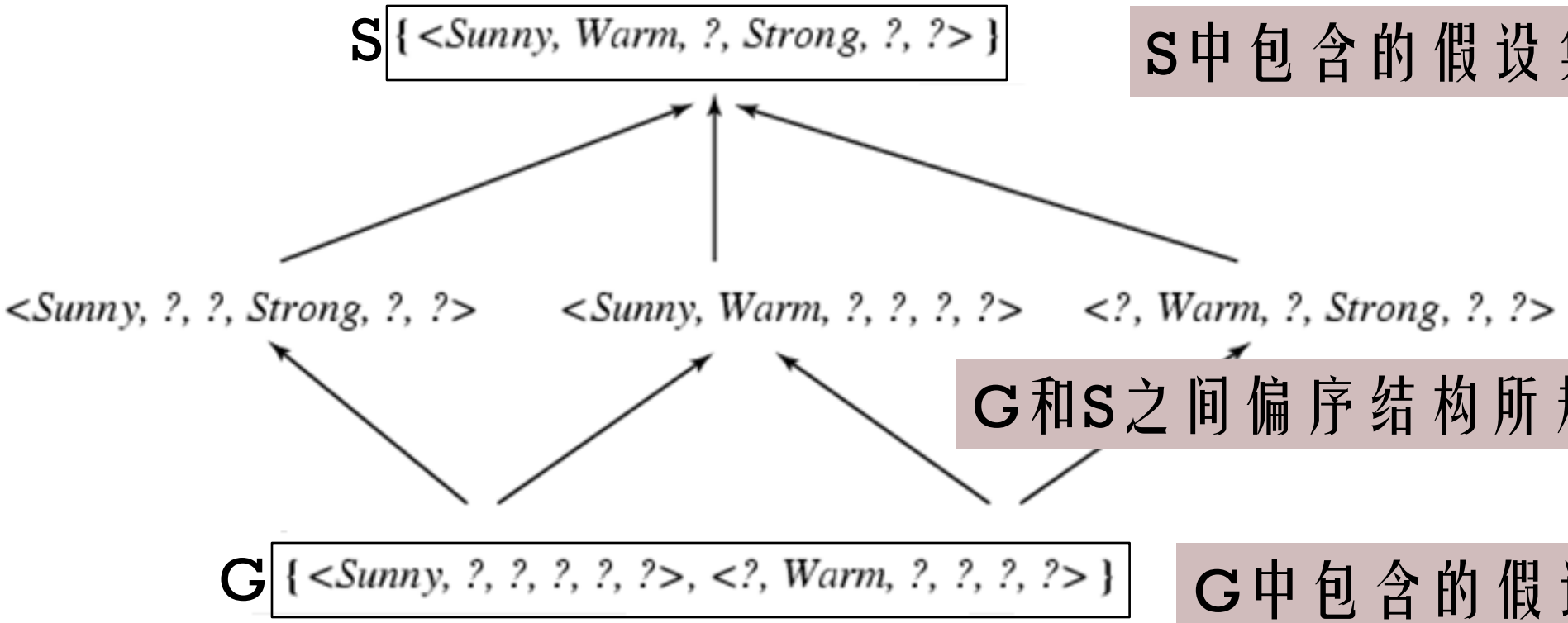
- 如果d是一正例
  - 从G中移去所有与d不一致的假设
  - 对S中每个与d不一致的假设s
    - 从S中移去s
  - 把s的所有的极小一般化式h加入到S中，其中h满足
    - h与d一致，而且G的某个成员比h更一般
  - 从S中移去所有这样的假设：它比S中另一假设更一般

- 分别看是否满足S，G
- 红框为最终结果

训练样例：

4.<Sunny,Warm,High,Storage,Cool,Change>,EnjoySport=Yes

本例中的变型空间



S中包含的假设集

G和S之间偏序结构所规定的假设

G中包含的假设集

- 变型空间表示定理
- 令X为一任意的实例集合，H为X上定义的布尔假设的集合。令c:X→{0,1}为X上定义的任一目标概念，并令D为任一训练样例集合{<x,c(x)>}。对所有的X,H,c,D以及定义好的S和G，变型空间表示如下：

$$VS_{H,D}=\{h\in H|(\exists s\in S)(\exists g\in G)(g\geq_g h\geq_g s)\}$$

  - 上述定理表明：
    - 变型空间由G、S、G和S之间的偏序结构所规定的假设h组成

- 候选消除算法讨论
  - 候选消除算法输出与训练样例一致的所有假设的集合
  - 候选消除算法在描述这一集合时不需要明确列举所有成员
  - 利用more_general_than偏序结构，可以得到一个一致假设集合的简洁表示
  - 候选消除算法的缺点：同Find-S一样，容错性能差

- 候选消除算法收敛到正确目标概念的条件
  - 训练样例中没有错误
  - H中确实包含描述目标概念的正确假设

- 继续探讨：一个有偏的假设空间
  - 在EnjoySport这个例子中，假设空间限制为只包含属性值的合取（同时发生，交集）。（有偏）
  - 这一限制，导致假设空间不能够表示最简单的析取（发生一件）形式的目标概念。

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Cool	Change	Yes
2	Cloudy	Warm	Normal	Strong	Cool	Change	Yes
3	Rainy	Warm	Normal	Strong	Cool	Change	No

“三个属性问题，Sky=Sunny或Sky=Cloudy”

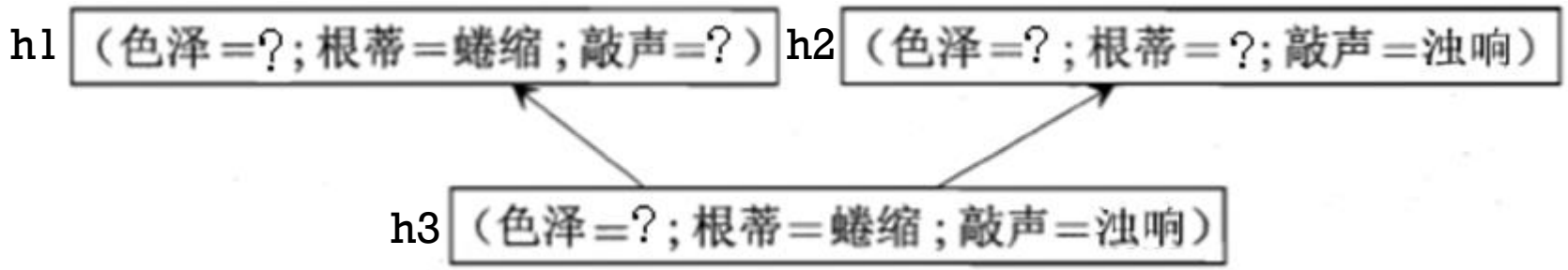
- 归纳推理的一个重要的基本属性：
  - 学习器如果不对目标概念的形式做预先的假定，那么它从根本上无法对未见的样本（实例）进行分类（如：设定为合取形式、选最特殊h的作为解）。
- 归纳偏置（**Inductive Bias**）：
  - 对归纳学习进行的某种形式的预先假定

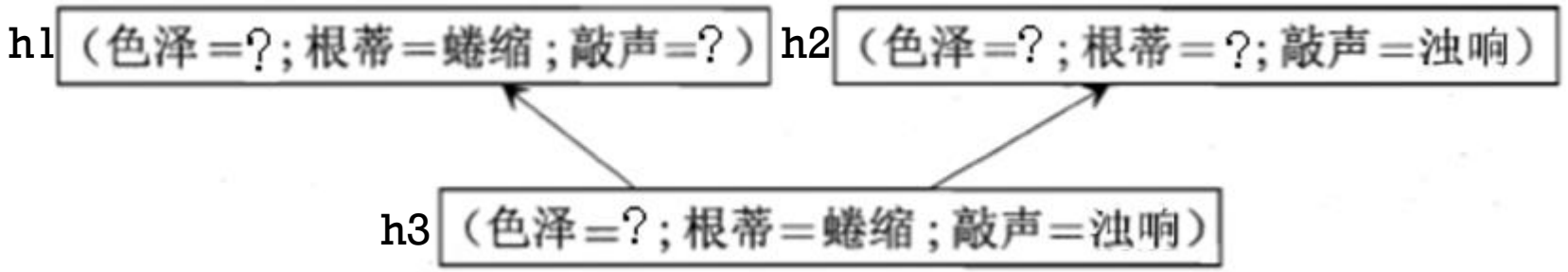
- 归纳偏好（归纳偏置的偏好）：
  - 假如通过不同学习算法得到三个与训练集一致的假设，但是他们对应的模型在遇到相同的问题时，会产生不同的预测结果。那么，应该选择哪种模型？我们无法通过训练模型得知哪个模型“更好”。这时，学习算法本身的“偏好”就会起到决定性作用。机器学习算法在学习过程中对某种类型假设的偏好，称为：“归纳偏好”。

- 西瓜数据集：学习概念“好瓜”

编号	色泽	根蒂	敲声	好瓜
1	青绿	蜷缩	浊响	是
2	乌黑	蜷缩	浊响	是
3	青绿	硬挺	清脆	否
4	乌黑	稍蜷	沉闷	否

- 变型空间：

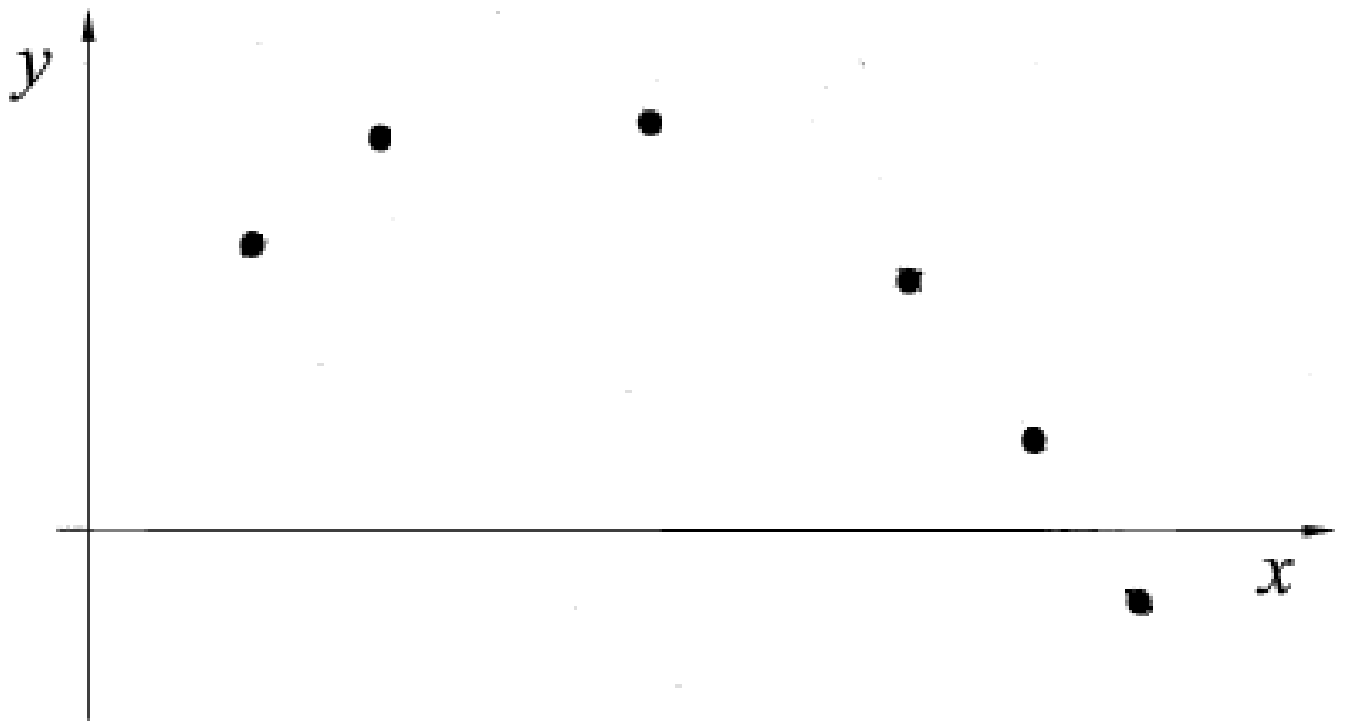


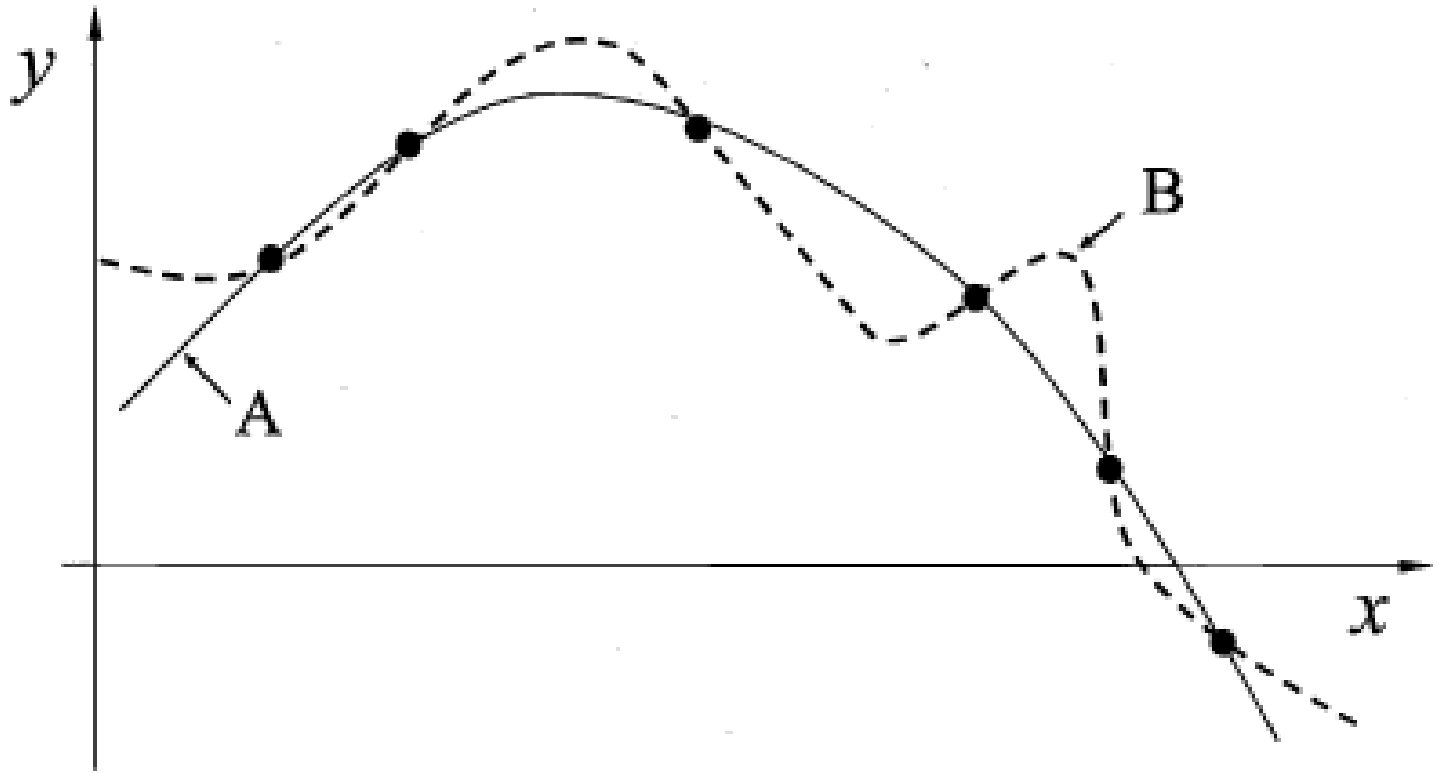


- 新瓜（好瓜）：色泽=青绿；根蒂=蜷缩；敲声=清脆
  - 算法偏好尽可能特殊的模型：否
  - 算法偏好尽可能一般的模型，且由于某种原因它更“相信”根蒂：是
  - 算法偏好多数表决：否

- 归纳偏好常表现为正则化项
- 可依据“奥卡姆剃刀”原理进行选择：若有多个假设与观察一致，则选择最简单的那个（如无必要，勿增实体）

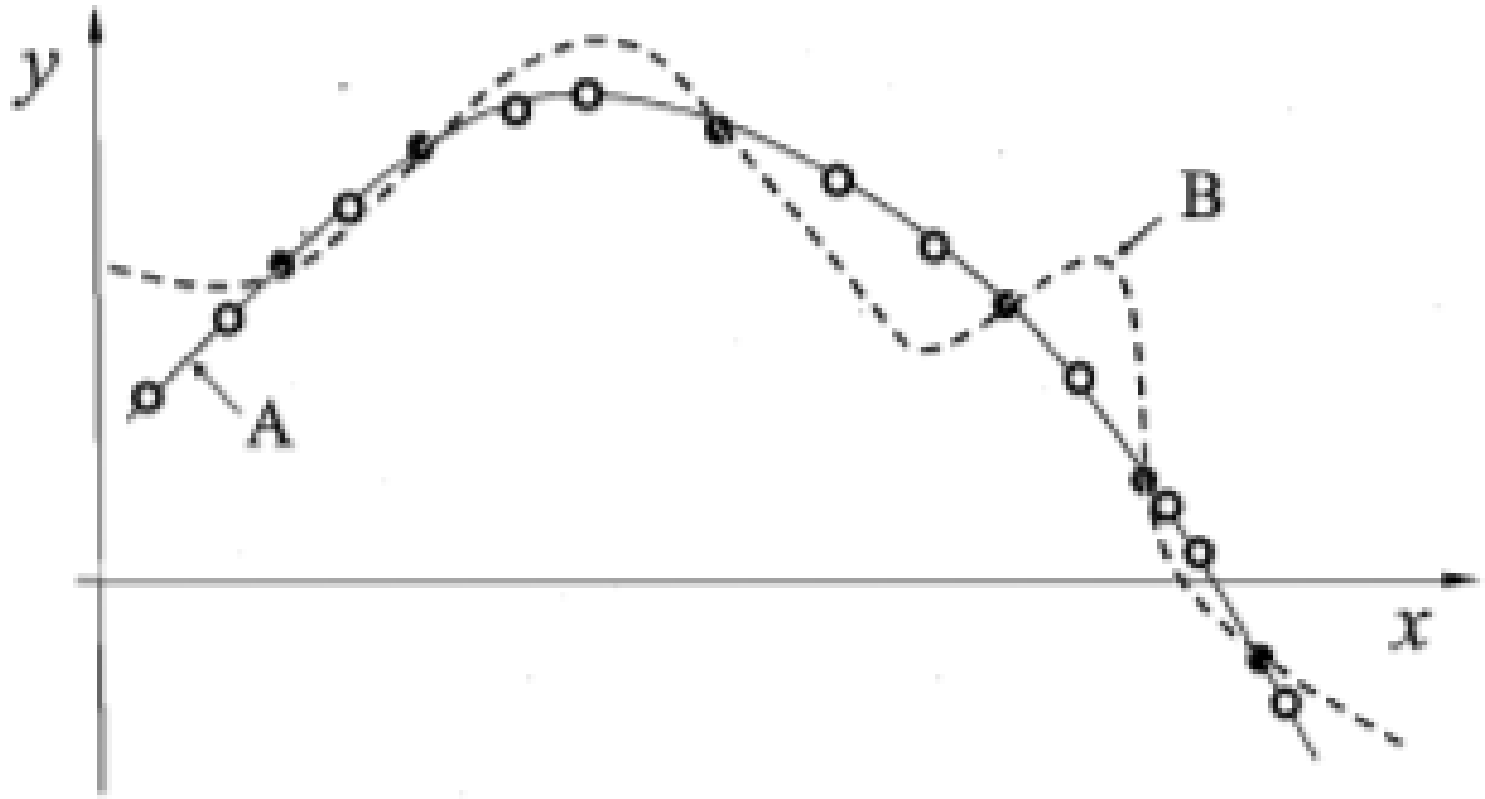
- 考察一个回归学习



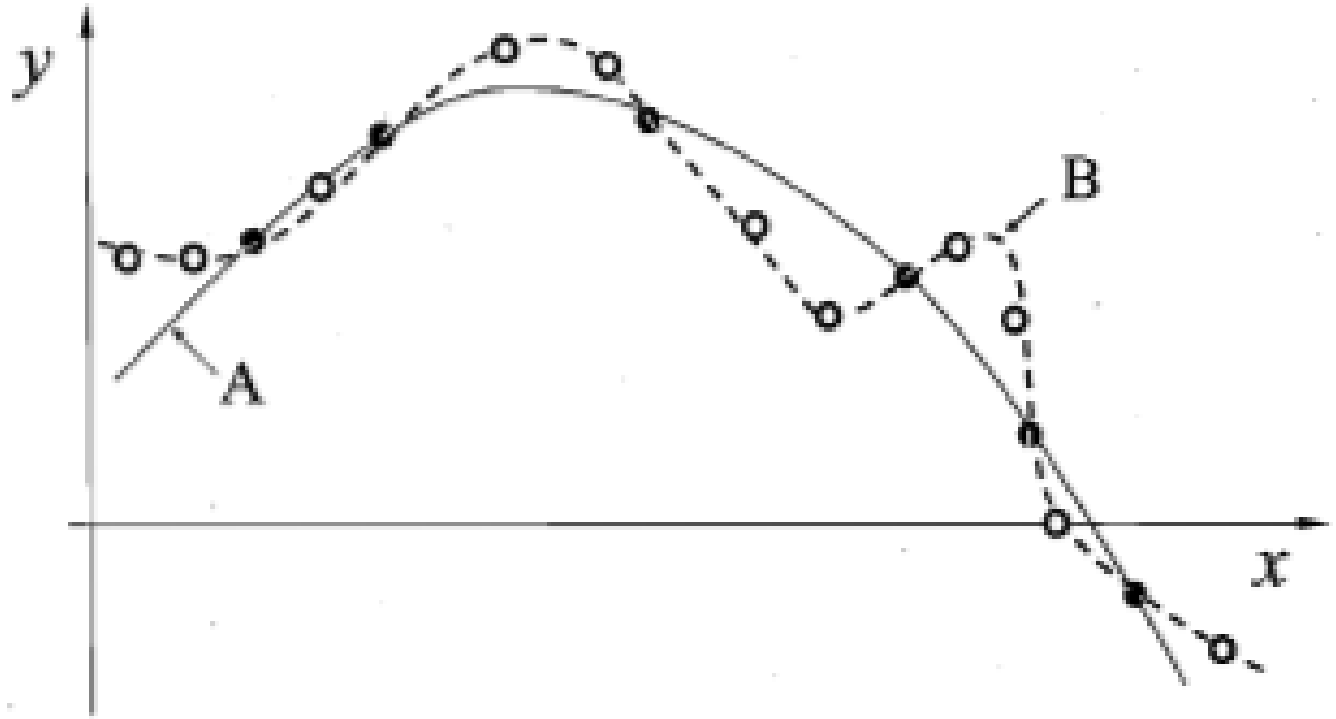


存在多条曲线与有限样本训练集一致

- 通过学习算法得到A、B两条拟合曲线



- 根据“奥卡姆剃刀”原理，A优于B



- 但B优于A的情况也是完全可能存在的

对于一个学习算法fa，若它在某些问题上比学习算法fb好，则必然存在在另一些问题上，fb比fa好。这个结论对任何算法均成立。“没有免费的午餐”定理证实，无论学习算法fa多聪明、学习算法fb多笨拙，它们的期望性能竟然相同（训练集外误差）。

为简单起见，假设样本空间 $\mathcal{X}$ 和假设空间 $\mathcal{H}$ 都是离散的. 令 $P(h|X,\mathfrak{L}_a)$ 代表算法 $\mathfrak{L}_a$ 基于训练数据 X 产生假设 h 的概率，再令 f 代表我们希望学习的真实目标函数. $\mathfrak{L}_a$ 的“训练集外误差”，即 $\mathfrak{L}_a$ 在训练集之外的所有样本上的误差为

$$E_{ote}(\mathfrak{L}_a|X,f)=\sum_{h}\sum_{\boldsymbol{x}\in\mathcal{X}-X}P(\boldsymbol{x})\,\mathbb{I}\left(h\left(\boldsymbol{x}\right)\neq f\left(\boldsymbol{x}\right)\right)P\left(h\mid X,\mathfrak{L}_a\right)\ ,$$

其中 $\mathbb{I}(\cdot)$ 是指示函数，若 $\cdot$ 为真则取值 1，否则取值 0.

若 f 均匀分布，则有一半的 f 对 $\boldsymbol{x}$ 的预测与 $h(\boldsymbol{x})$ 不一致.

对所有可能的 f 按均匀分布对误差求和，有

$$\begin{aligned}
\sum_f E_{ote}(\mathfrak{L}_a|X,f) &= \sum_f \sum_h \sum_{\boldsymbol{x}\in\mathcal{X}-X} P(\boldsymbol{x})\,\mathbb{I}(h(\boldsymbol{x})\neq f(\boldsymbol{x}))\,P(h\mid X,\mathfrak{L}_a) \\
&= \sum_{\boldsymbol{x}\in\mathcal{X}-X} P(\boldsymbol{x}) \sum_h P(h\mid X,\mathfrak{L}_a) \sum_f \mathbb{I}(h(\boldsymbol{x})\neq f(\boldsymbol{x})) \\
&= \sum_{\boldsymbol{x}\in\mathcal{X}-X} P(\boldsymbol{x}) \sum_h P(h\mid X,\mathfrak{L}_a) \frac{1}{2} 2^{|\mathcal{X}|} \\
&= \frac{1}{2} 2^{|\mathcal{X}|} \sum_{\boldsymbol{x}\in\mathcal{X}-X} P(\boldsymbol{x}) \sum_h P(h\mid X,\mathfrak{L}_a) \\
&= 2^{|\mathcal{X}|-1} \sum_{\boldsymbol{x}\in\mathcal{X}-X} P(\boldsymbol{x})\cdot 1\,.
\end{aligned}$$

总误差竟然与学习算法无关!

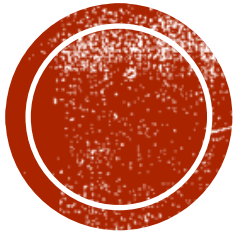
对于任意两个学习算法 $\mathfrak{L}_a$ 和 $\mathfrak{L}_b$,

$$\sum_f E_{ote}(\mathfrak{L}_a|X,f) = \sum_f E_{ote}(\mathfrak{L}_b|X,f)$$

这就是“没有免费的午餐”定理 (No Free Lunch Theorem, 简称 NFL定理) [Wolpert, 1996; Wolpert and Macready, 1995].

- 优化算法的等价性
- 任何优化算法都不比穷举法好
- 为什么还要研究最优化和机器学习算法呢?

- NFL定理有一个重要前提：所有“问题”出现的机会相同、或所有问题同等重要。但实际情形并不是这样的。很多时候，我们只关注自己正在试图解决的问题，希望为它找到一个解决方案，至于这个解决方案在别的问题上是否为好方案，我们并不关心。
- 脱离具体问题，空泛的谈论“什么学习算法更好”毫无意义
- 收敛速度



THE END !