新媒体大数据分析

New Media Big Data Analysis

第三章 数据建模

黄振亚，朱孟潇，张凯

课程主页：
http://staff.ustc.edu.cn/~huangzhy/Course/NM2025.html

助教：齐畅，朱家骏
bigdata_2025@163.com

决策树剪枝

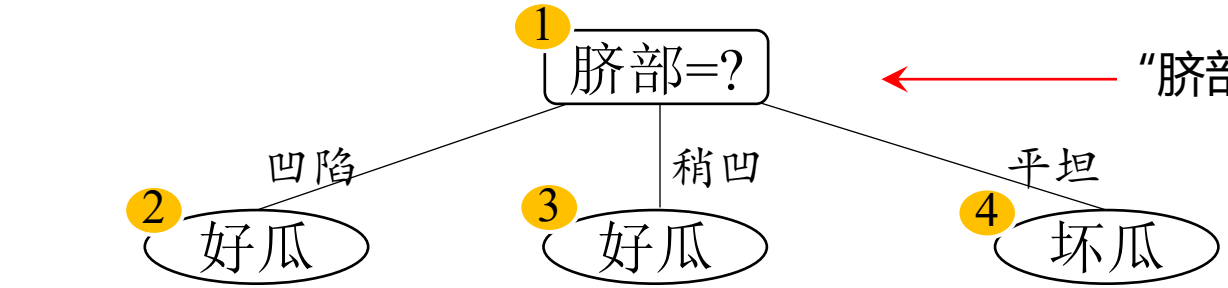
3. 决策树剪枝

- 在生成决策树之后，我们还将根据实际情况，对决策树进行剪枝
  - 剪枝的原因在于训练过程的“过拟合”问题
    - 如果训练集与测试集效果都不好，说明出现“欠拟合”
    - 如果训练集效果好，而测试集效果不好，说明出现“过拟合”
- 过拟合出现的原因：训练过程中过度迁就训练数据特性，而导致构造出过于复杂、过于细枝末节的决策树，泛化能力较差
- 解决这一问题的办法在于对已生成的决策树进行简化，即“剪枝”
- 包括两种策略：预剪枝、后剪枝

决策树剪枝

3. 决策树剪枝：预剪枝

- 在生成决策树的过程中即进行剪枝，称作“预剪枝”
  - 每个节点划分前，衡量当前节点的划分能否提高决策树的泛化能力
  - 通过提前停止生成分枝对决策树进行剪枝，可以利用信息增益等测度来对分枝生成情况（优劣）进行评估



“脐部=?”

验证集精度

划分前: 42.9%

划分后: 71.4%

预剪枝决策: 划分

“色泽=?”

验证集精度

划分前: 71.4%

划分后: 57.1%

预剪枝决策: 禁止划分

“根蒂=?”

验证集精度

划分前: 71.4%

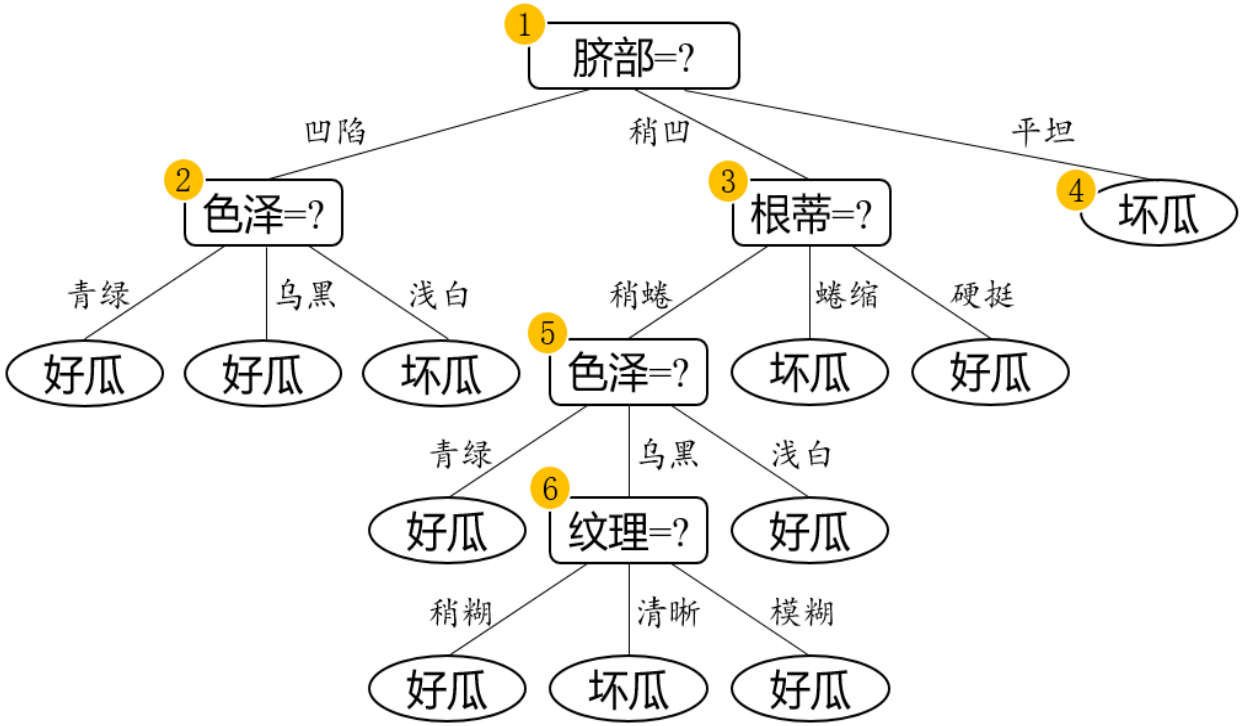
划分后: 71.4%

预剪枝决策: 禁止划分

决策树剪枝

- **3. 决策树剪枝：后剪枝**
  - 在生成决策树之后再进行剪枝，称作“后剪枝”
  - 自底向上 考察每个非叶子节点，考虑将该节点替换成叶子节点后能否提高泛化性能

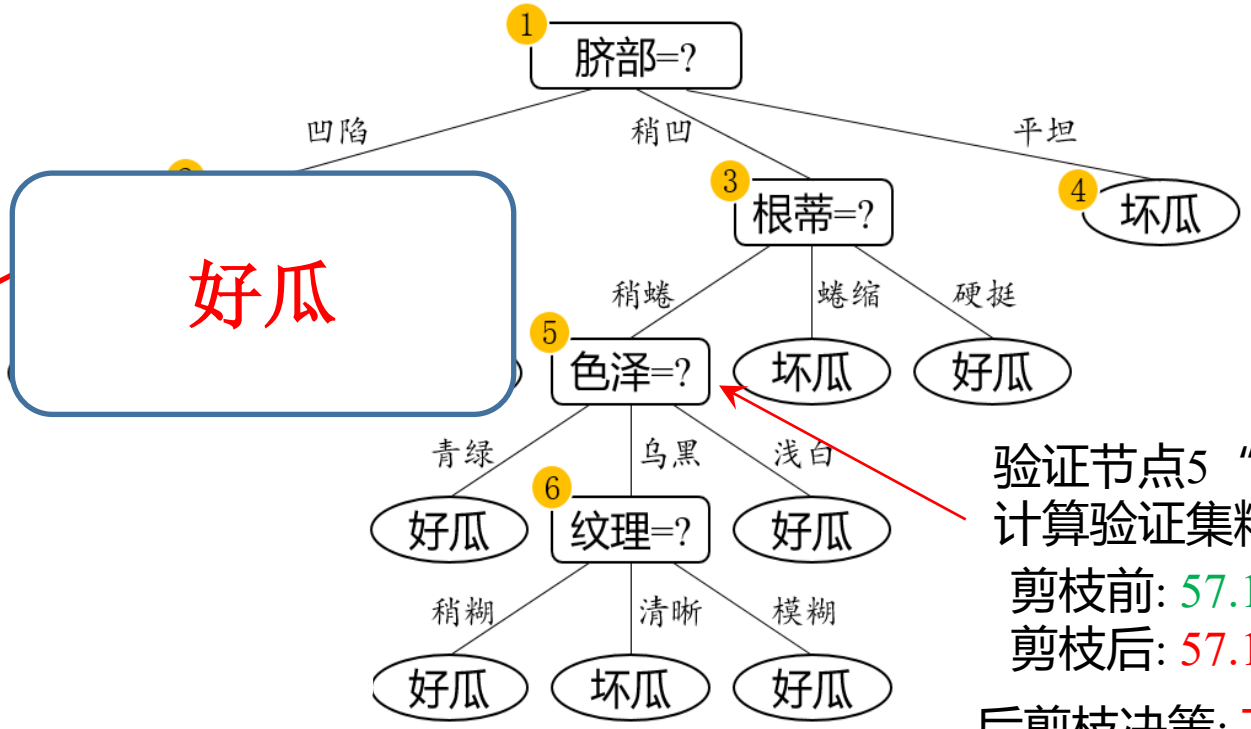
剪枝前



决策树剪枝

- **3. 决策树剪枝：后剪枝**
  - 自底向上 考察每个非叶子节点，考虑将该节点替换成叶子节点后能否提高泛化性能

剪枝后



验证节点2“色泽”计算验证集精度

剪枝前: 57.1%

剪枝后: 71.4%

后剪枝决策: 剪枝

验证节点5“色泽”计算验证集精度

剪枝前: 57.1%

剪枝后: 57.1%

后剪枝决策: 不剪枝

决策树剪枝

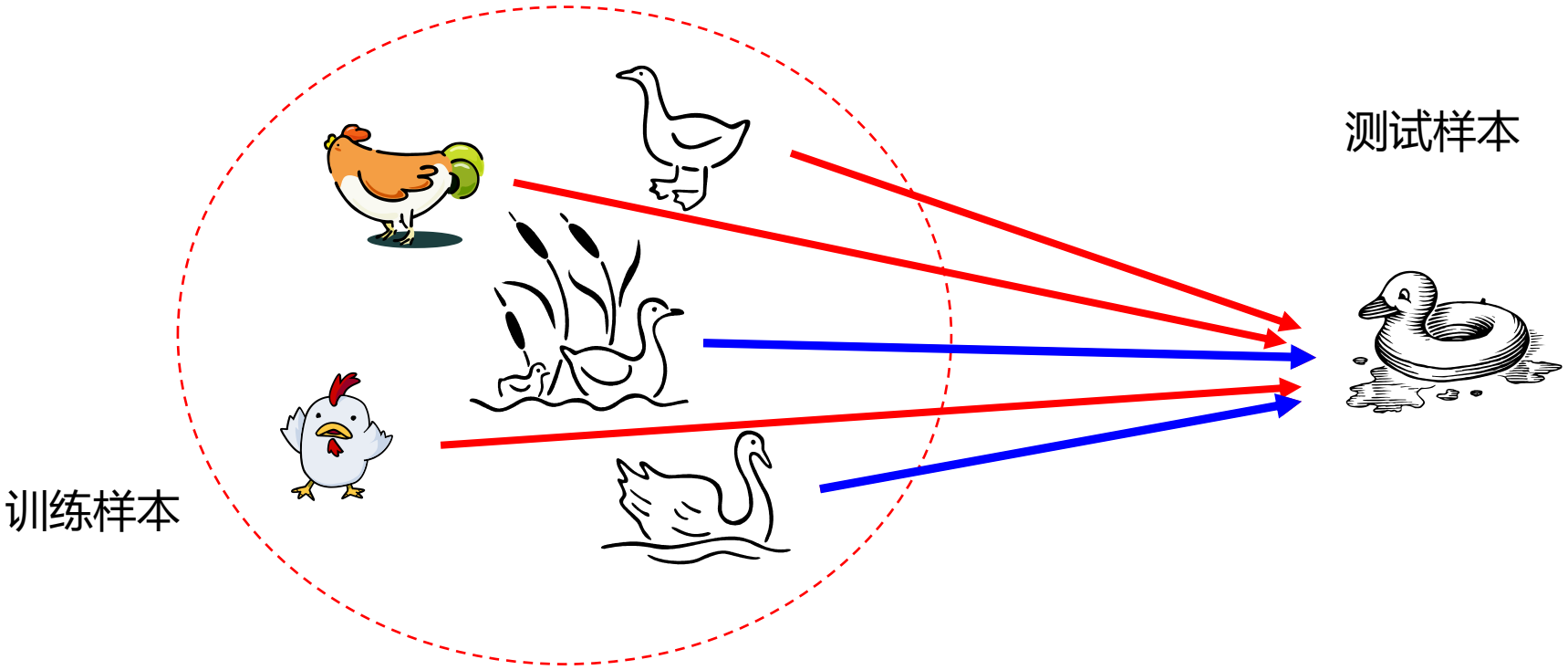
- **3. 决策树剪枝**：比较两种剪枝策略
  - 从过程上看，后剪枝方法经过了“构建”到“剪枝”这样的过程，显然它要比事前剪枝需要更多的计算时间
  - 对应的，后剪枝可以获得更可靠的决策树
  - 实际使用时：先剪枝可以与后剪枝方法相结合，从而构成一个混合的剪枝方法

分类与预测

- 有监督学习：分类与预测
- 常用方法
  - 规则方法
  - 决策树
  - 最近邻方法
  - 支持向量机（SVM）
  - 集成方法
- 类不平衡问题
- 分类的评价指标

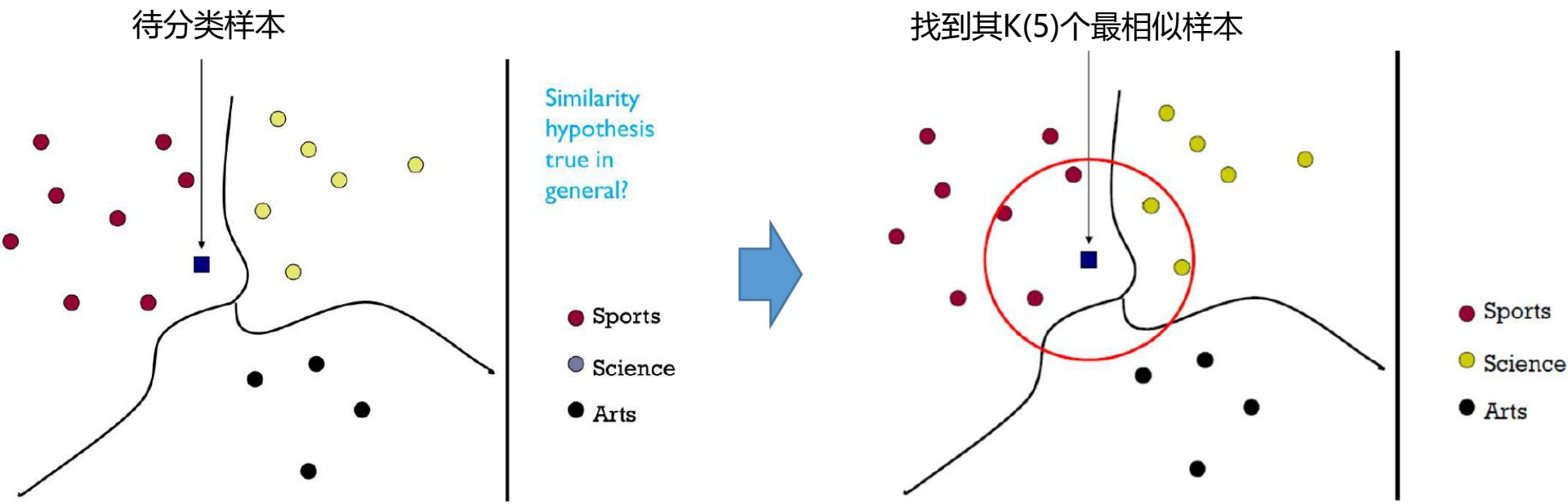
分类：K近邻方法

- 分类——K近邻方法
  - 问题：判断测试样本是什么动物？



分类：K近邻方法

- 分类——K近邻方法
  - 对数据空间内的样本，可提出相似样本假设
    - 表征上相近的样本应该属于同一个类别
  - K近邻思想：用K个最相似样本的类别来预测未知样本的类别(投票方法)
  - 核心问题：距离度量、K的取值



K近邻方法：距离度量

- K近邻方法核心问题：距离度量
  - K近邻分类的效果严重依赖于距离度量
  - 对于高维空间而言，最基本的度量方式为欧式距离

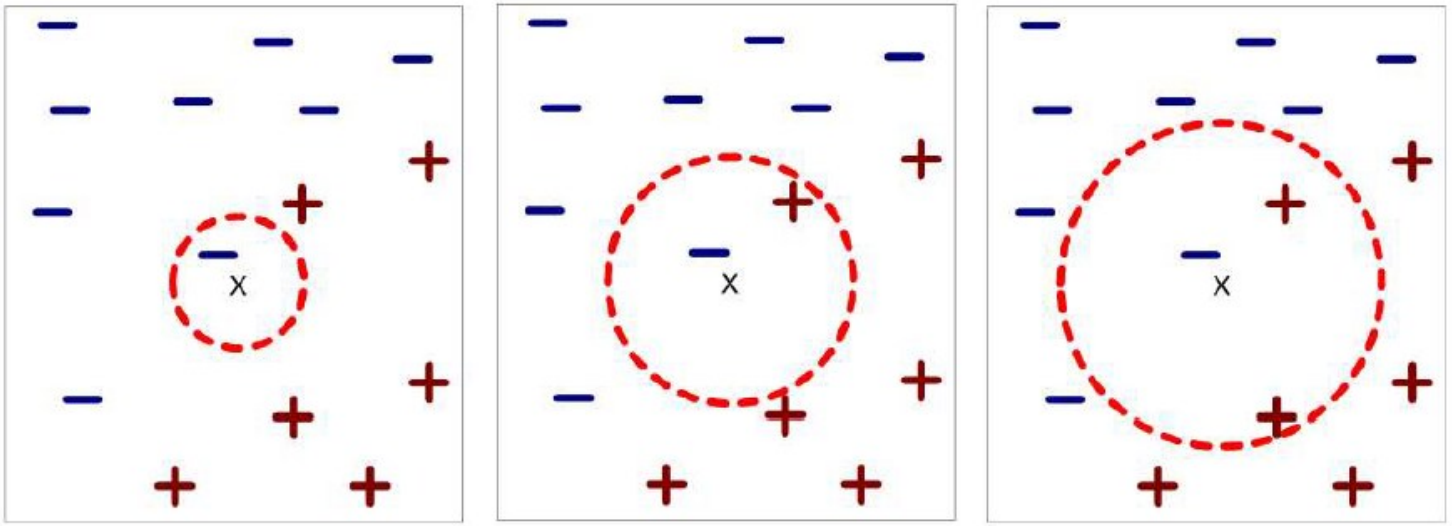
$$d(x, x') = \sqrt{\sum_i (x_i - x'_i)^2}$$

  - 离散0/1向量，则可使用汉明距离（Hamming）代替
  - 除此之外，对于文本而言（如采用TF-IDF），可使用余弦相似度
  - 其他可采用的度量如马氏距离等

回顾第二章：数据集成

K近邻方法：K的取值

- K近邻方法核心问题：K的取值
  - K近邻分类的效果同样严重依赖于 K 的取值（**即邻居的数量**）
    - K太小，容易受噪声干扰；
    - K太大，可能导致错误涵盖其他类别样本



(a) 1-nearest neighbor　(b) 2-nearest neighbor　(c) 3-nearest neighbor

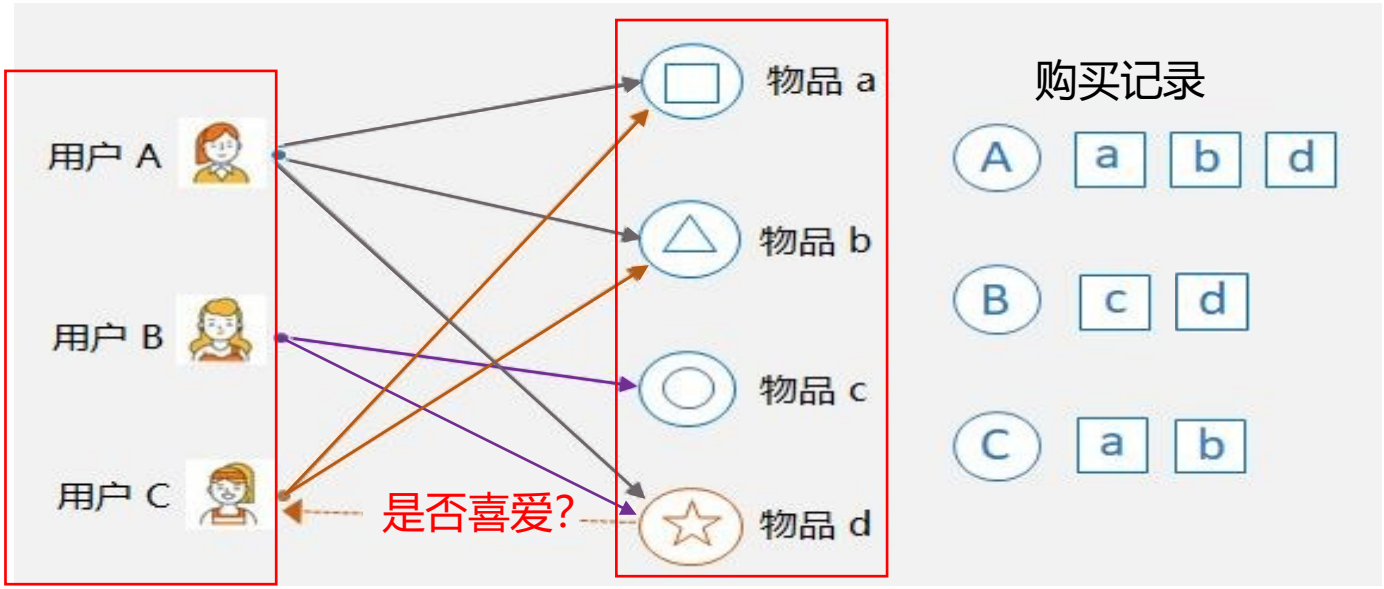
不同的K值，结果不同

分类：K近邻方法

- K近邻方法的特点
  - K近邻方法是一种典型的基于实例的学习
    - 使用具体实例进行预测，而不需要对数据进行抽象（如提取特征）
  - K近邻方法是一种消极学习，不需要模型，但分类过程开销很大
    - 相比之下，积极学习方法训练模型较为费时费力，但基于模型分类很快
  - K近邻方法基于局部信息进行判别，受噪声影响很大
  - K近邻方法需要慎重选择度量并预处理数据，否则可能被误导
    - 例如，借助身高体重进行分类，身高波动范围不大，而体重差距巨大
    - （回顾第二章：数据集成）已知：小明(160,60000)；小王(160,59000)；小李(170，60000)。小明与谁的体型更相似？

K近邻方法实例

- K近邻思想的应用实例：推荐系统的UCF、ICF模型



UCF: 基于K个相似用户对物品的评分

- 用户A、B购买过物品d，且与C相似，可用他们对d的平均喜爱程度作为C对d的喜爱程度

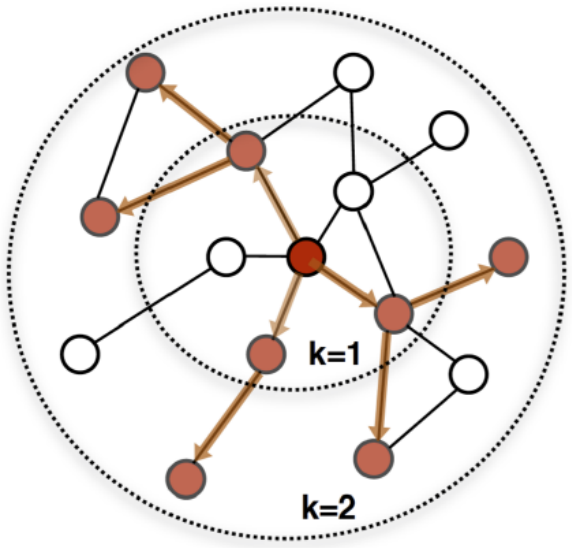
ICF: 基于用户对K个相似物品的评分

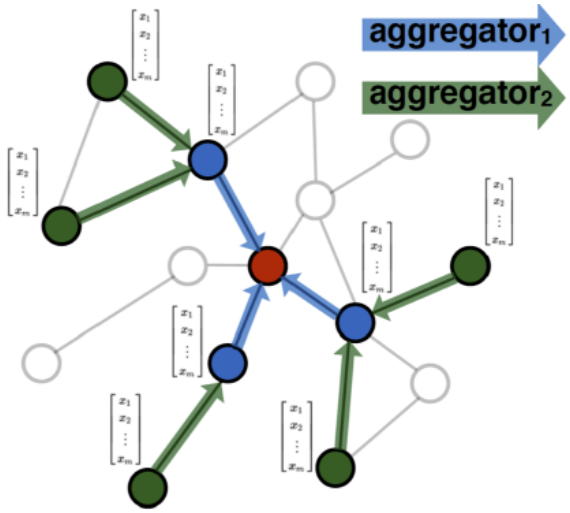
- C购买过a,b，若物品d与a,b相似，可用C对他们的平均喜爱程度作为对物品d的喜爱程度

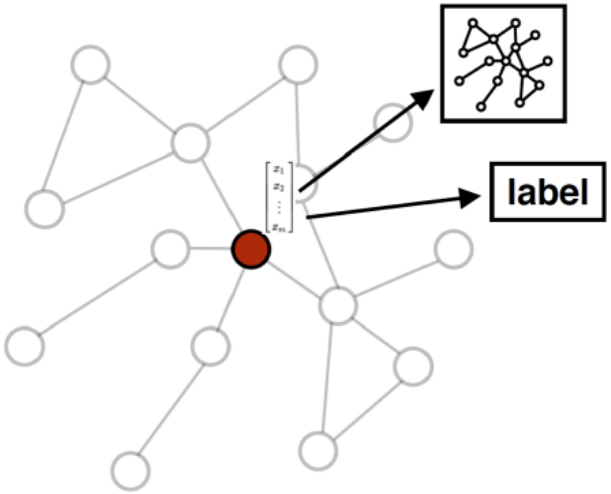
- Sarwar, Badrul, et al. Item-based collaborative filtering recommendation algorithms. WWW 2001.
- Amazon recommendations: Item-to-item collaborative filtering. IEEE Internet computing, 2003

K近邻方法实例

- K近邻思想的应用实例：图神经网络中的近邻
  - 基本思想：将K个邻居节点的信息传播到当前节点
  - 距离度量：基于注意力机制计算(GAT模型)







- Hamilton, Will, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs." NeurIPS 2017

KNN的实例

- Transformer
  - Google Brain 2017的提出的一篇工作
  - 针对RNN的弱点进行重新设计，解决了RNN效率问题和传递中的缺陷

- **Original paper:** Vaswani et al. Attention Is All You Need. In *NIPS*, 2017.

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

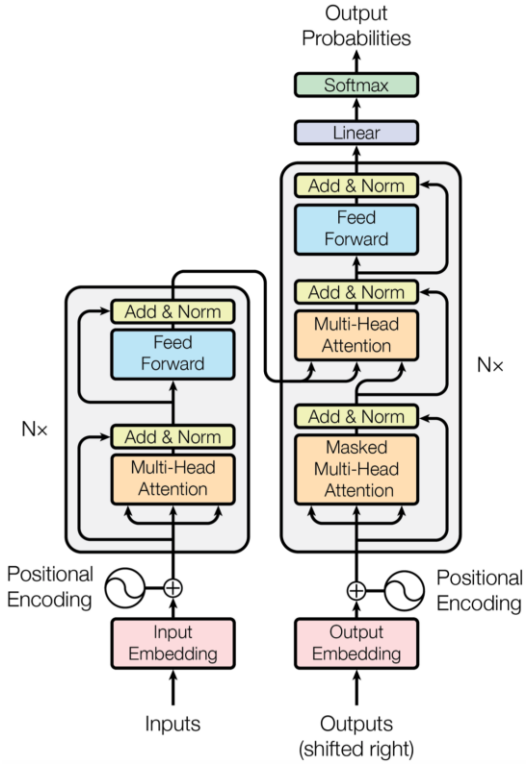
Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

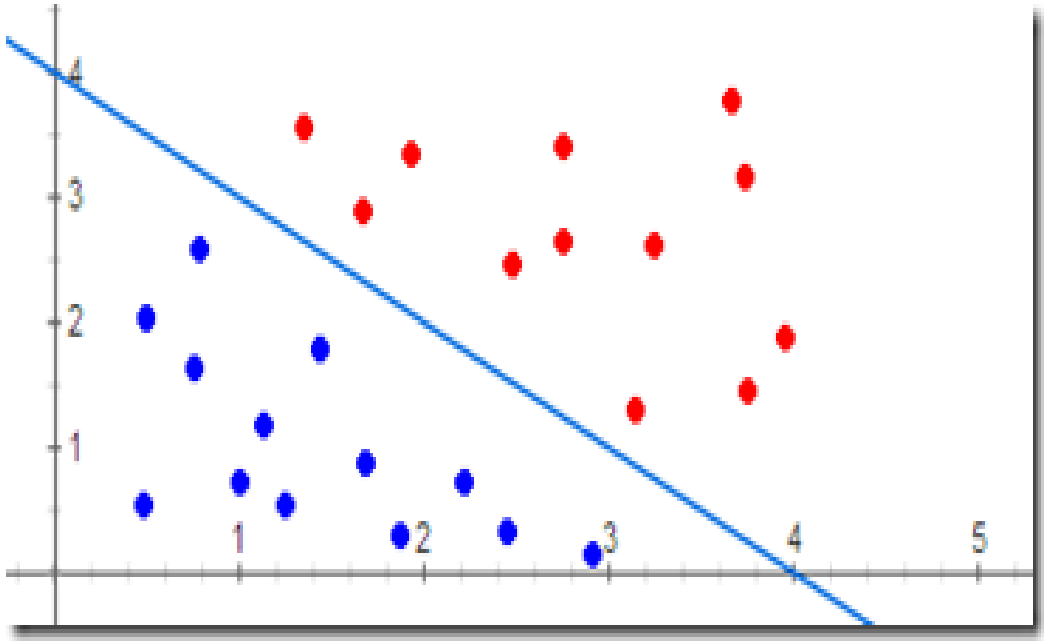


分类与预测

- 有监督学习：分类与预测
- 常用方法
  - 规则方法
  - 决策树
  - 最近邻方法
  - 支持向量机（SVM）
  - 集成方法
- 分类的评价指标
- 类不平衡问题

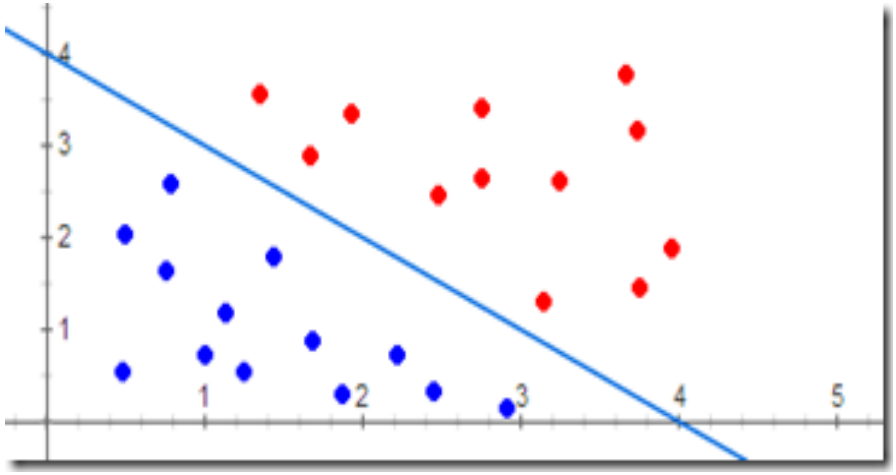
分类：感知机

- 分类——感知机（perceptron）
  - 目标：寻找一条直线S（高维时是超平面）划分不同类别的数据
  - 输入：样本的特征向量X={x}, $x \in R^d$
  - 输出：样本类别y $\in \{-1, +1\}$

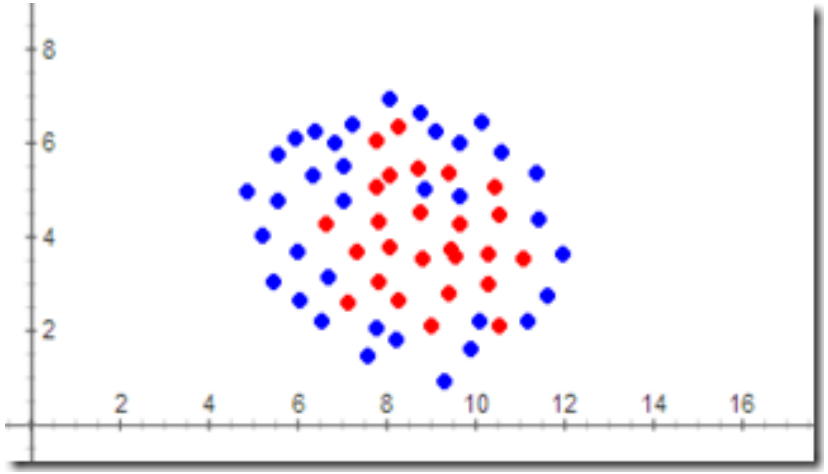


分类：感知机

- 分类——感知机（perceptron）
  - 1957年由Rosenblatt提出，是神经网络与支持向量机的基础
  - 感知机的前提：样本空间线性可分
    - 左例中，可以用一条直线将+1类和-1类完美分开，称这个样本空间是线性可分的
    - 右例的样本是线性不可分的，感知机不能处理这种情况

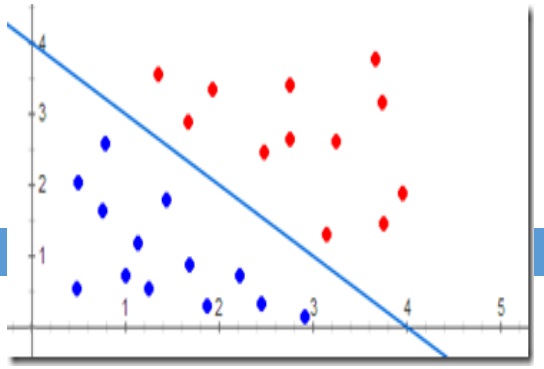


线性可分



线性不可分

分类：感知机



- 感知机的基本概念
  - 模型：符号函数 $f(x) = sign(w \cdot x + b) = \begin{cases} +1, & w \cdot x + b \geq 0 \\ -1, & w \cdot x + b < 0 \end{cases}$
  - 分界超平面 S： $w \cdot x + b = 0$
  - 学习目标：最小化 误分类点 到 超平面 的 总距离
    - 点(x_0, y_0)到超平面S: $w \cdot x + b = 0$的距离$\frac{1}{\|w\|} |w \cdot x_0 + b|$
    - 误分类点(x_i, y_i)到超平面S: $w \cdot x_i + b = 0$的距离为 $-\frac{1}{\|w\|} y_i (w \cdot x_i + b)$

推导过程省略

x_i错分时，若y_i为-1，则计算的$(w \cdot x_i + b) > 0$
若y_i为+1，则计算的$(w \cdot x_i + b) < 0$

  - 损失函数：$\underset{w,b}{\operatorname{argmin}} L(w, b) = -\sum_{x_i \in M} y_i (w \cdot x_i + b)$，
  - 学习策略：找到参数w和b，使得损失函数最小

它是连续可导的，这就使得我们比较容易求得其最小值

分类：感知机

- 感知机学习算法：梯度下降 — 课后学习

$$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i (w \cdot x_i + b)$$

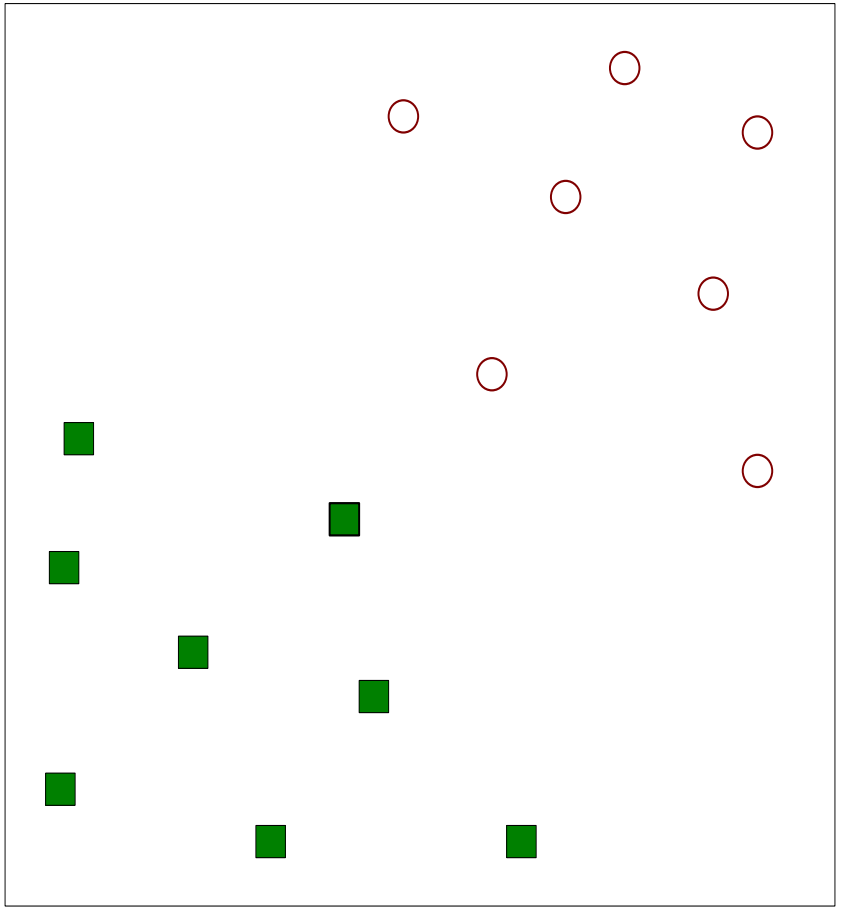
  - 随机初始化w_0和b_0
  - 梯度下降不断地极小化损失函数
    - 每次随机选取一个误分类点对w和b进行更新。
    - 设误分类点为(x_i, y_i)，那么损失函数$L(w,b)$的梯度为：

$$\nabla_w L(w,b) = -y_i \cdot x_i$$
$$\nabla_b L(w,b) = -y_i$$

    - 接下来对w, b进行更新$w \leftarrow w + \gamma y_i \cdot x_i, b \leftarrow b + \gamma y_i$，其中$\gamma (0 \le \gamma \le 1)$为步长，也、称为学习速率（learning rate）。
    - 通过迭代，直到损失函数为0 (无误分点)

分类：支持向量机

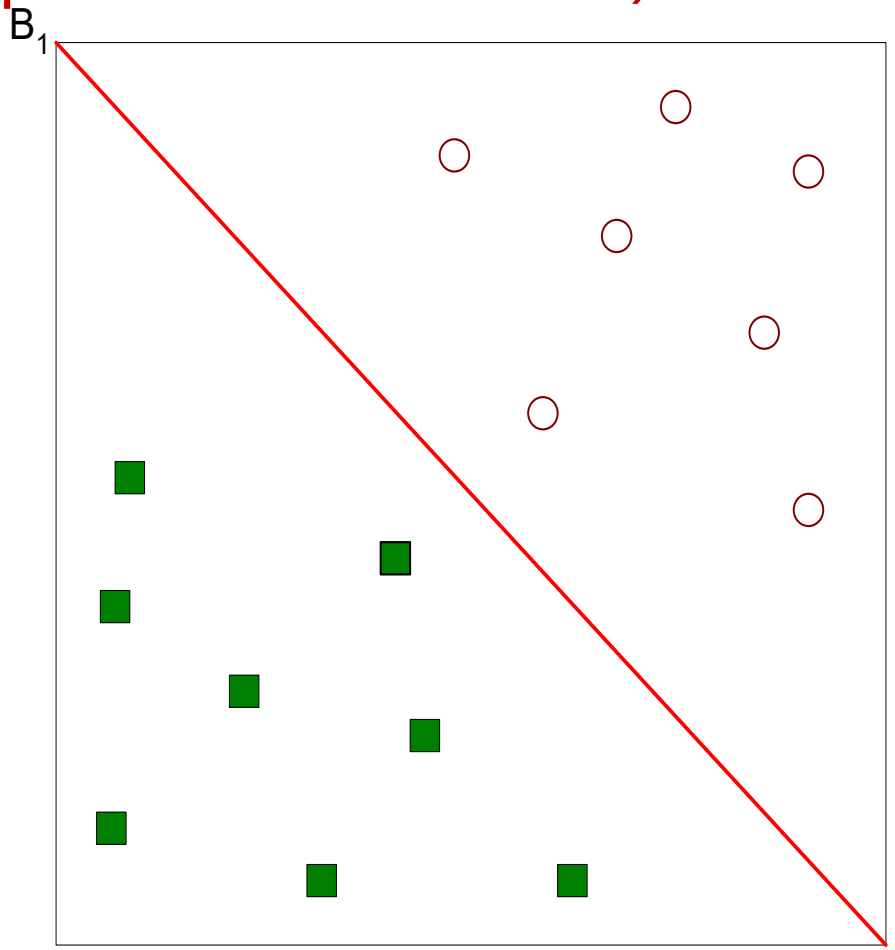
- 分类——支持向量机（Support Vector Machine）



分类：支持向量机

- 分类——支持向量机（Support Vector Machine）

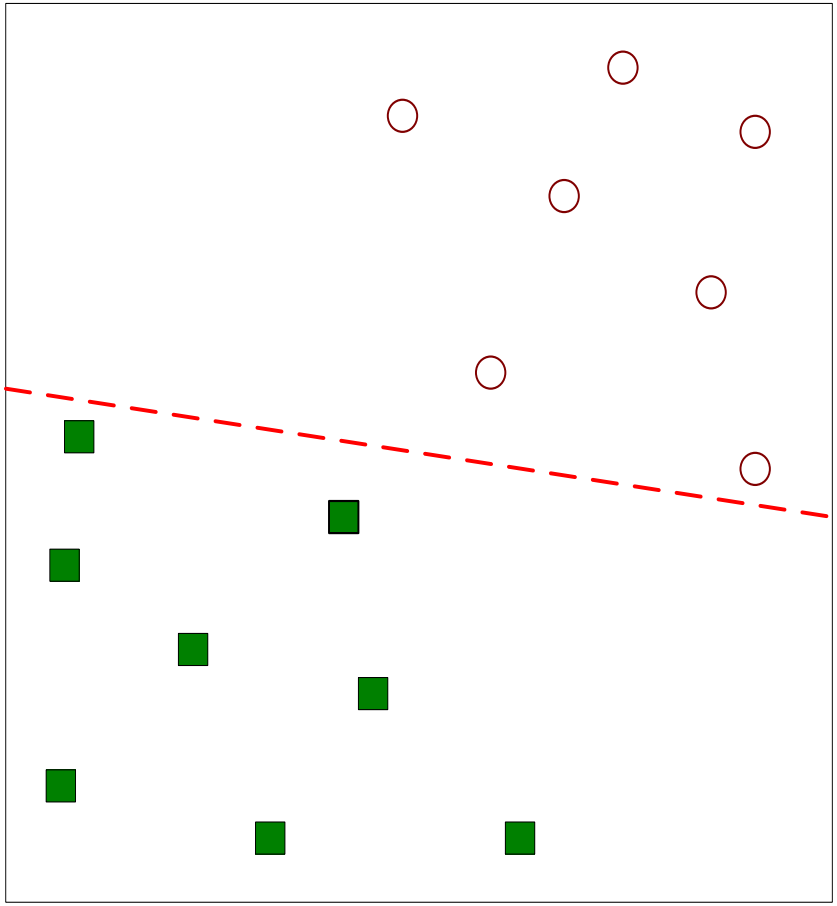
一个可行解



分类：支持向量机

- 分类——支持向量机（Support Vector Machine）

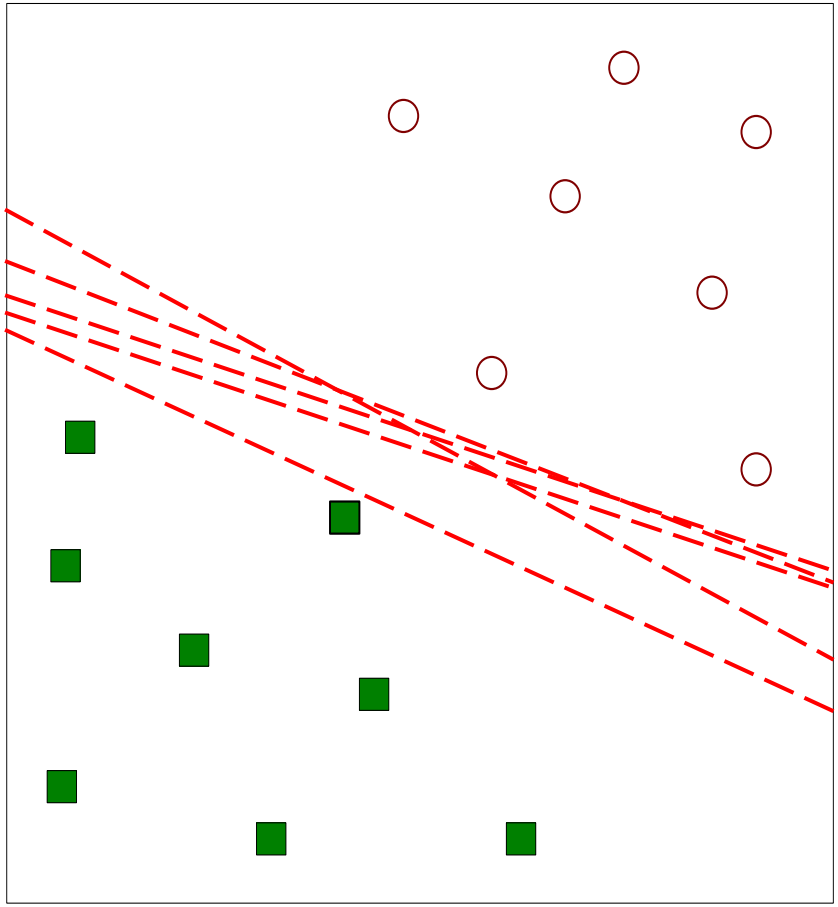
另一个可行解



分类：支持向量机

- 分类——支持向量机（Support Vector Machine）

其他可行解



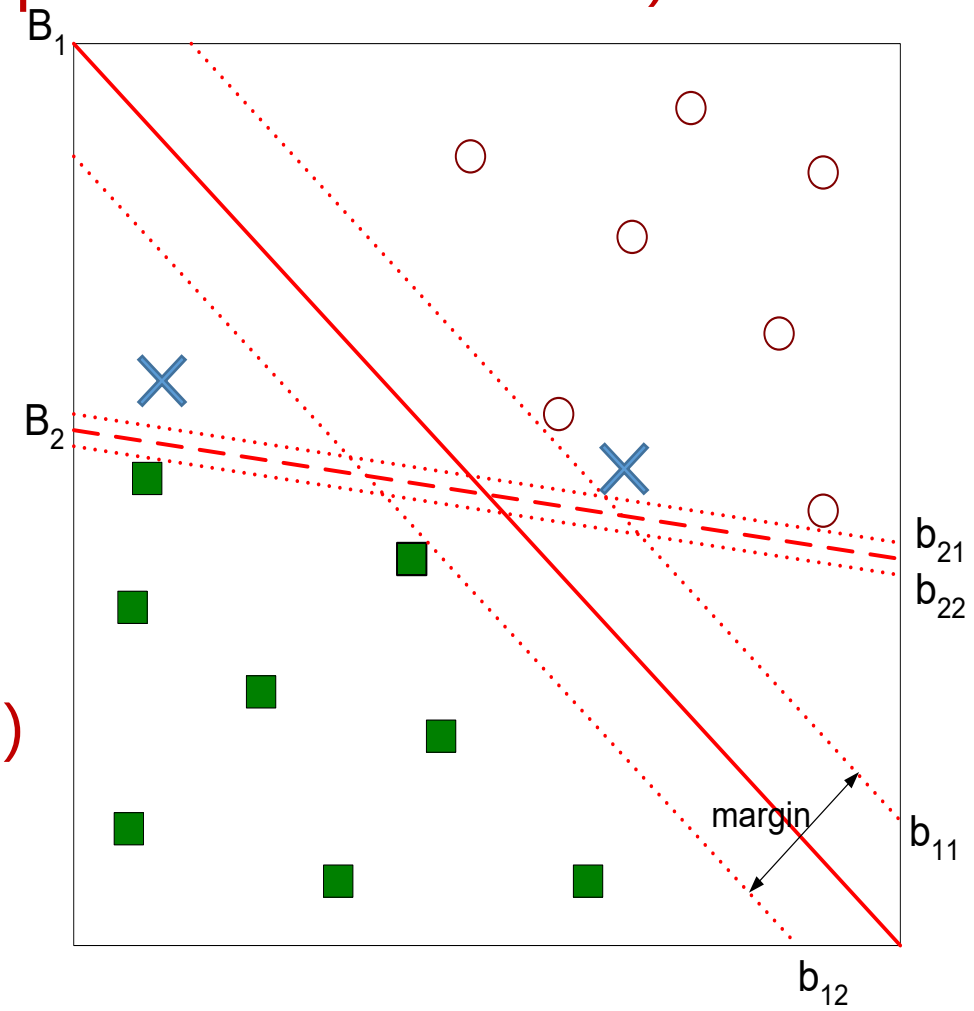
分类：支持向量机

- 分类——支持向量机（Support Vector Machine）

B1与B2，哪个更好？

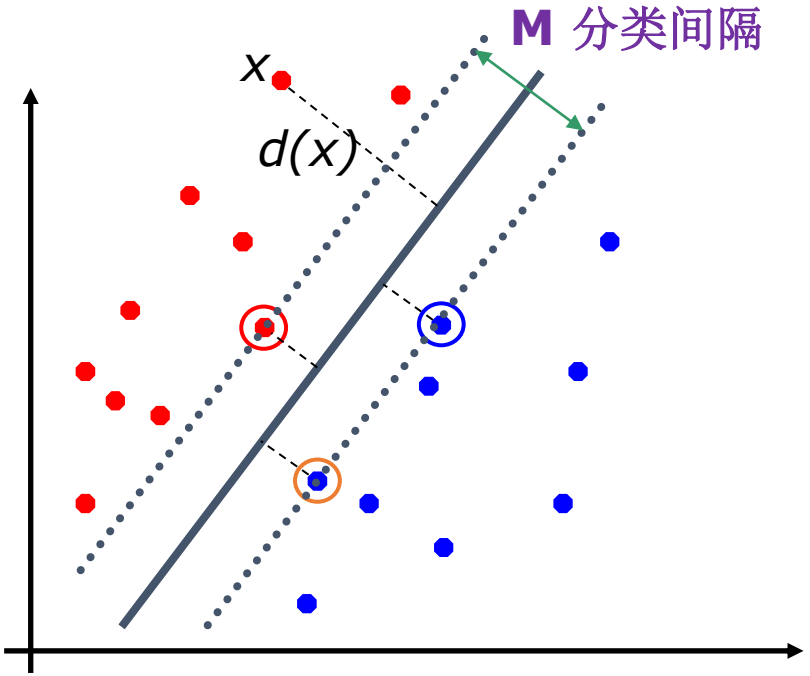
B1 保证分类正确（区分）

分类间隔大 （更容易区分）



分类：支持向量机

- 分类——支持向量机
  - 因此，应该选择“正中”的最大间隔超平面（分类间隔最大）
    - 容忍性好，泛化能力强
    - 在线性可分的条件下，符合这样条件的超平面“存在且唯一”
  - 问题：如何找到最优的超平面？
  - 最大化分类间隔

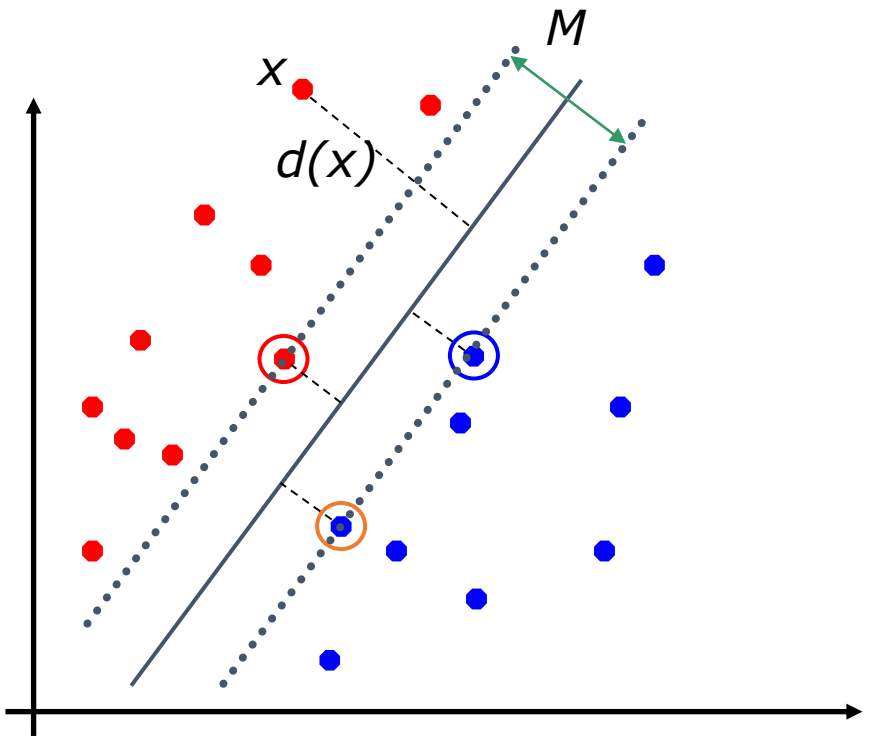


分类：支持向量机

- 分类——支持向量机
  - 理解：最大化分类间隔
    - 区分能力更强
    - 概率的角度: 最难的点置信度最大
    - 即使我们在选边界的时候犯了小错误，使得边界有偏移，仍然有很大概率保证可以正确分类绝大多数样本
    - 很容易实现交叉验证，因为边界只与极少数的样本点有关
    - 有一定的理论支撑
    - 实验结果验证了其有效性

保证分类正确性—当前

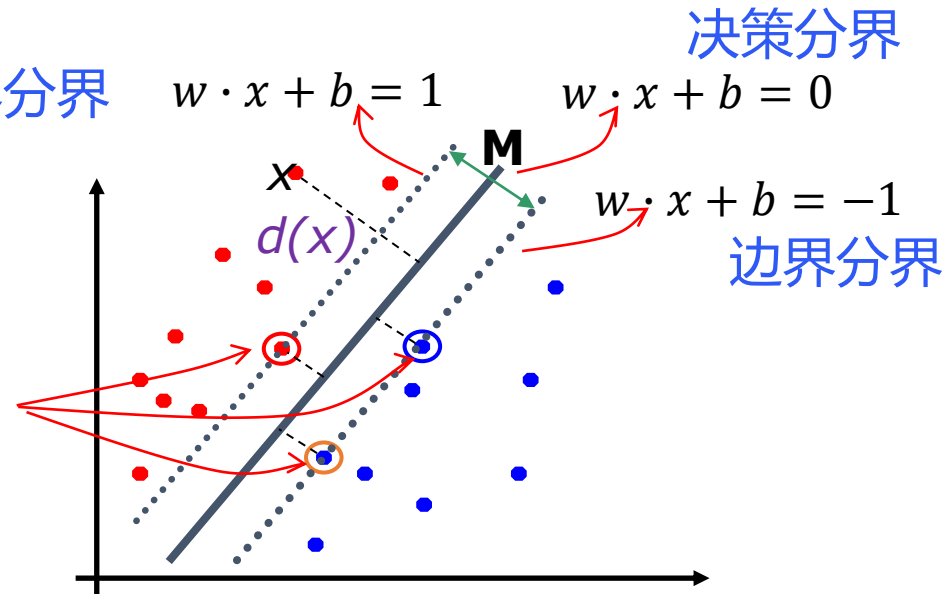
保证分类质量—未来



分类：支持向量机

- 支持向量机的基本概念
  - 模型：符号函数$y = sign(w \cdot x + b) = \begin{cases} +1, w \cdot x + b \geq 0 \\ -1, w \cdot x + b < 0 \end{cases}$
  - 决策分界面(Decision Boundary)：$w \cdot x + b = 0$
  - 边界分界面(Margin Boundary)：$w \cdot x + b = \pm 1$
  - 支持向量(Support Vectors)：满足$w \cdot x + b = \pm 1$的样本

两个边界之间的距离**d(x)**



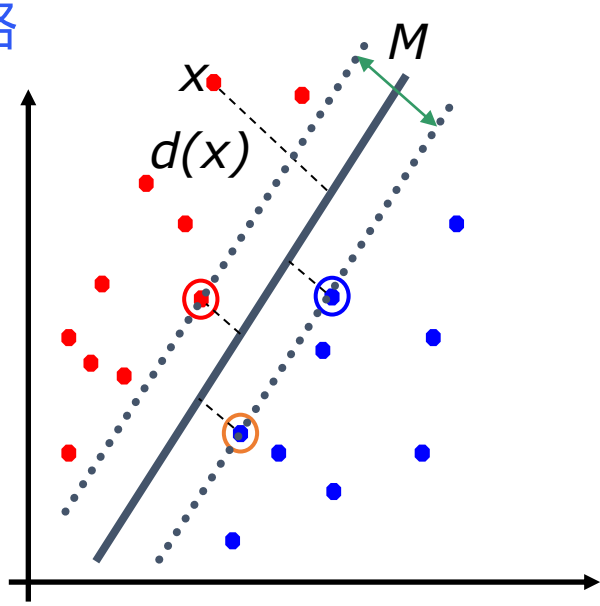
分类：支持向量机

- 支持向量机
  - **两个边界之间的距离d(x)：支持向量到决策分界面的距离**
    - 点x_i到平面$w \cdot x + b = 0$的距离为$\frac{1}{\|w\|} y_i(w \cdot x_i + b)$
    - 支持向量到决策平面的距离为$\frac{1}{\|w\|}$

最大化

- 两个边界之间的距离：$\frac{2}{\|w\|}$　推导过程省略
  - 目标函数：$\text{argmax}\, L(w,b) = \frac{2}{\|w\|}$
  - 学习策略：找到参数w和b，使得目标最大

$$\underset{w,b}{\text{argmax}}\, L(w,b) = \frac{2}{\|w\|}$$
$$s.t. \quad y_i\left(\mathrm{w}^T \cdot x_i + b\right) \geq 1, \qquad \mathrm{i} = 1,2,\ldots,\mathrm{m}$$



分类：支持向量机

- 支持向量机学习算法
  - 优化任务转化

$$\underset{w,b}{argmax}\ L(w,b)=\frac{2}{\|w\|}$$
$$s.t.\quad y_i\left(\mathrm{w}^T\cdot x_i+b\right)\geq 1,\qquad \mathrm{i}=1,2,\ldots,\mathrm{m}$$

优化目标：平方和系数都是为了求导方便

等价

$$\underset{w,b}{\mathrm{argmin}}\ L(w,b)=\frac{1}{2}\|w\|^2$$
$$s.t.\quad y_i\left(\mathrm{w}^T\cdot x_i+b\right)\geq 1,\qquad \mathrm{i}=1,2,\ldots,\mathrm{m}$$

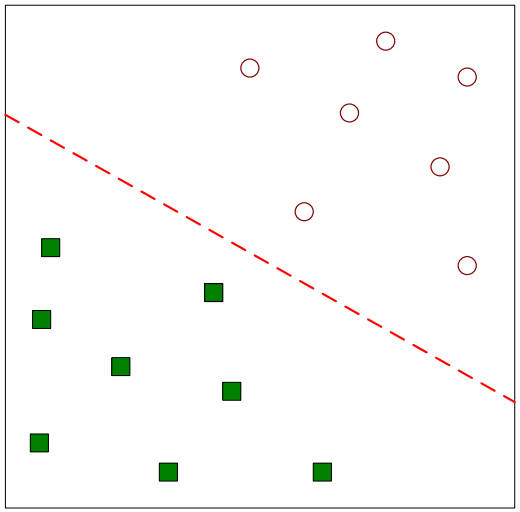
  - (课后学习)注意到，该形式符合凸二次规划问题特征，可以借助拉格朗日对偶性，通过求解对偶问题加以求解
  - (课后学习)具体而言，求解方式可采用序列最小优化算法（SMO）

```
from sklearn.svm import LinearSVC
```

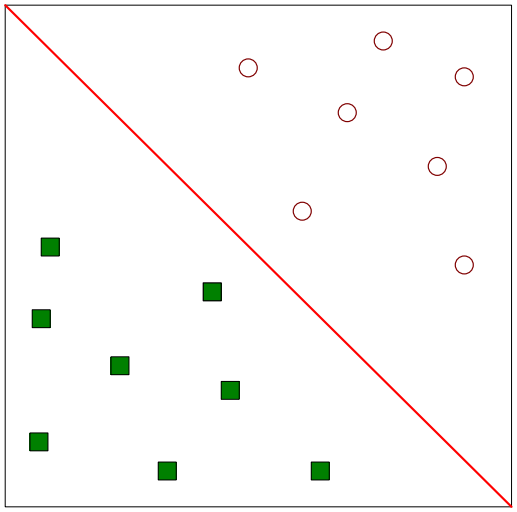
分类：支持向量机

- 感知机与支持向量机对比
  - 相同点：均采用模型$f(x) = sign(w \cdot x + b)$
  - 不同点：采用不同的优化目标

感知机



SVM



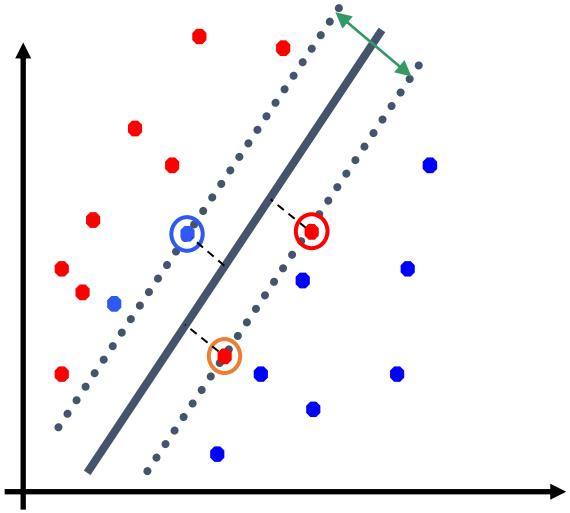
优化目标：

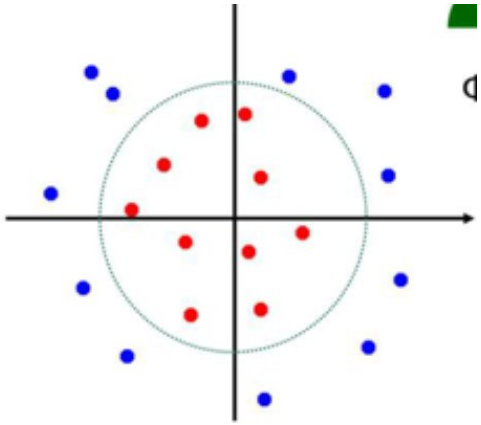
$$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i(w \cdot x_i + b)$$

$$\min_{w,b} L(w,b) = \frac{1}{2}\|w\|^2$$
$$s.t.\, y_i\left(w^T x_i + b\right) \geq 1$$

SVM总结

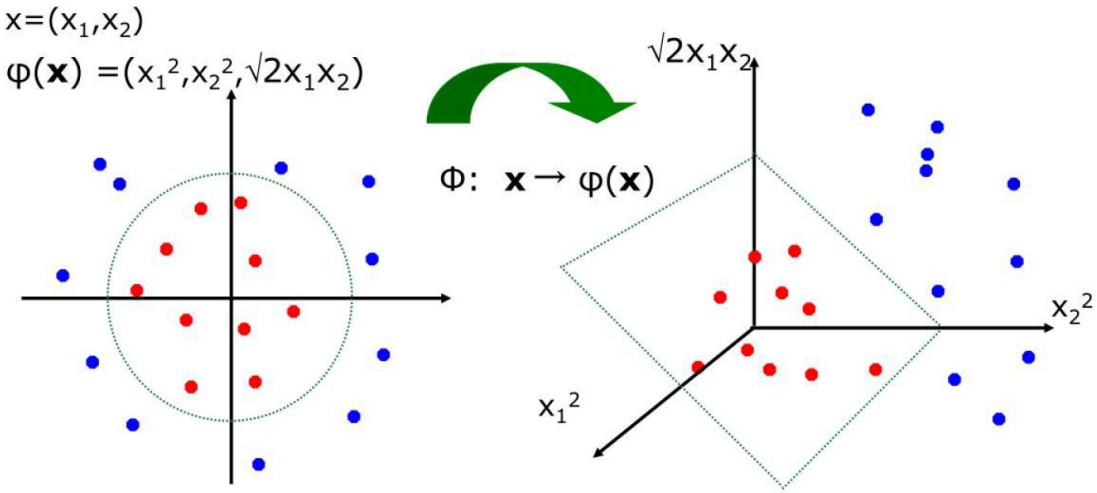
- 支持向量机
  - 优点：强分类器
    - 区分分类结果
    - 最大化间隔：更容易区分分类结果
    - 有数学理论保证
    - 只有支持向量在影响，优化简单
  - 缺点： Hard Margin SVM
    - 只能处理线性可分问题
    - 线性不可分
      - 软间隔：soft margin SVM（课后学习）
    - 线性完全不可分





线性不可分问题

- 线性不可分问题
  - 如果数据集线性不可分，不存在这样一个超平面，怎么办？
    - 解决方法：将样本映射到一个更高维的特征空间，使得在这个特征空间线性可分
  - **核函数：**
    - 核函数的目的，在于将高维空间下的SVM求解时需要的内积运算转化为低维空间下的核函数计算，从而避免可能遇到的“维度灾难”问题



核函数

- 线性不可分问题与核函数——（课后学习）
  - 常见的核函数如下表所示

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^{\top} \boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^{\top} \boldsymbol{x}_j)^d$	$d \geq 1$为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\delta^2}\right)$	$\delta > 0$为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\delta}\right)$	$\delta > 0$
Sigmoid核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\beta \boldsymbol{x}_i^{\top} \boldsymbol{x}_j + \theta)$	tanh为双曲正切函数, $\beta > 0, \theta < 0$

  - 其中，线性核函数与高斯核函数（径向基）是最为常用的
  - 常见挑选核函数方法一般为以下两种：
    - 穷举法：一个个试过来，选择效果最好的一种
    - 混合法：将多个不同的核函数混合起来使用

分类与预测

- 有监督学习：分类与预测
- 常用方法
  - 规则方法
  - 决策树
  - 最近邻方法
  - 感知机，支持向量机（SVM）
  - 集成方法
- 分类的评价指标
- 类不平衡问题