



## 第三章 机器学习初步 复习

# 目录

- 机器学习概述
  - 思路
  - 实际操作流程
  - 过拟合和欠拟合
- 机器学习类型
  - 有监督学习（重点：分类）
  - 无监督学习（重点：聚类）
- 数据准备
  - 训练集、测试集、验证集
  - 数据集的划分
  - 不平衡数据
- 有监督学习相关的性能评估
  - 回归
  - 分类 ※

## 3. 聚类

### ◆ 案例：kNN分类器

1. 思路
2. 步骤
3. 优缺点
4. Pycharm示例
5. 改进

### ◆ 案例：k-means聚类分析

1. 思路
2. 步骤
3. 优缺点
4. Pycharm示例
5. 改进

# 机器学习概述

- 思路

实际问题→获取数据→数据预处理→特征工程→模型训练与调优→模型评估→最终模型

- 实际操作（有监督学习）

收集数据并给定标签

训练模型

测试，评估

- 实际操作（无监督学习）

收集数据

训练模型

测试，评估

```
def train(train_images, train_labels):  
    # build a model for images -> labels...  
    return model  
  
def predict(model, test_images):  
    # predict test_labels using the model...  
    return test_labels
```

# 机器学习概述

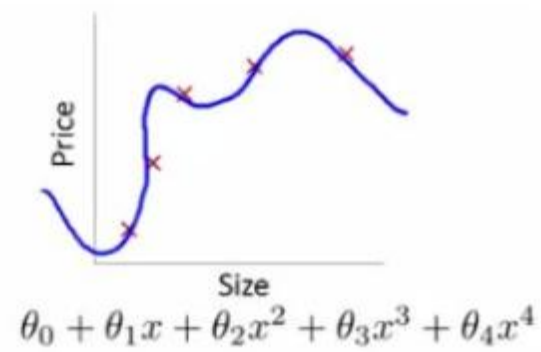
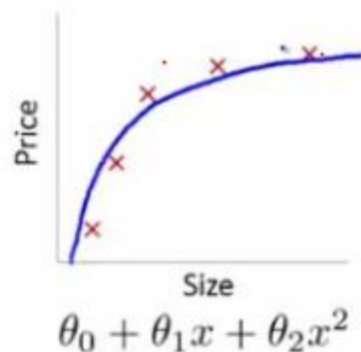
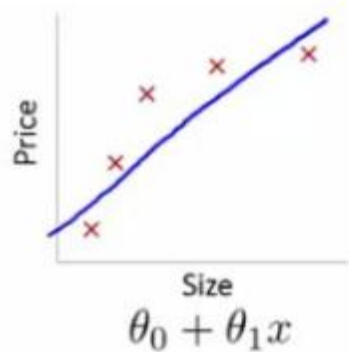
- 过拟合和欠拟合问题
  - 过拟合：模型在训练集上表现很好，但没有理解数据背后的规律，在测试集上表现很差，泛化能力差。
  - 过拟合出现原因：
    - 样本选取有误，导致数据不足以代表预定的分类规则
    - 噪音干扰过大
    - 模型复杂度过
  - 过拟合改善方法：
    - 增加数据样本数，减少特征数
    - 减少噪声影响
    - 使用正则化约束或DropOut
    - 调整参数和超参数
    - .....

# 机器学习概述

- 过拟合和欠拟合问题
  - 欠拟合：模型不能在训练集上获得足够低的误差，无法学习到数据背后的规律。
  - 欠拟合出现原因
    - 模型复杂度过低
    - 特征量过少
  - 欠拟合改善方法：
    - 模型复杂化
    - 增加数据特征数
    - 调整参数和超参数
    - .....

# 机器学习概述

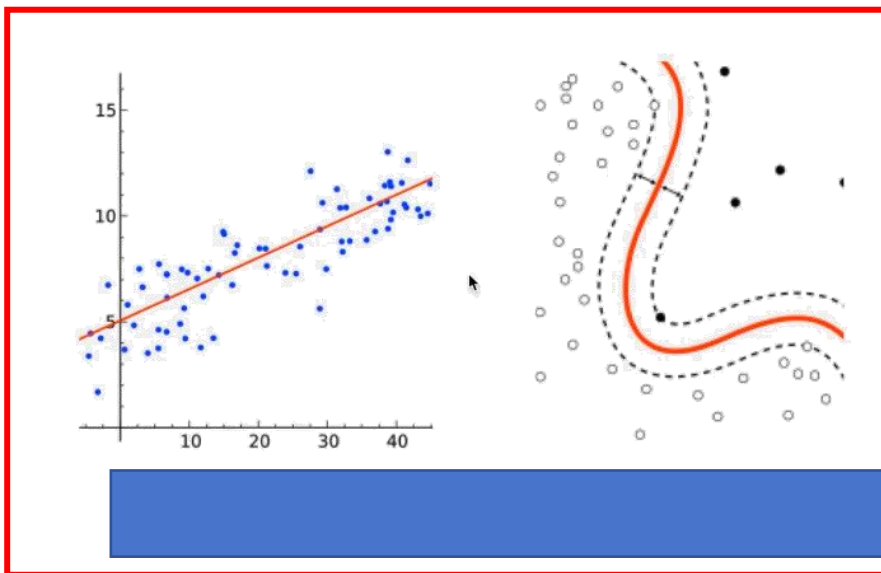
- 过拟合和欠拟合问题



判断过拟合和欠拟合？

# 机器学习类型

## 监督学习



- 经典分类器
  - k近邻 (kNN)
  - 决策树 (DT)
  - 朴素贝叶斯 (NB)
  - 支持向量机 (SVM)
  - 多层感知机 (MLP)
  - .....

### ◆ 分类

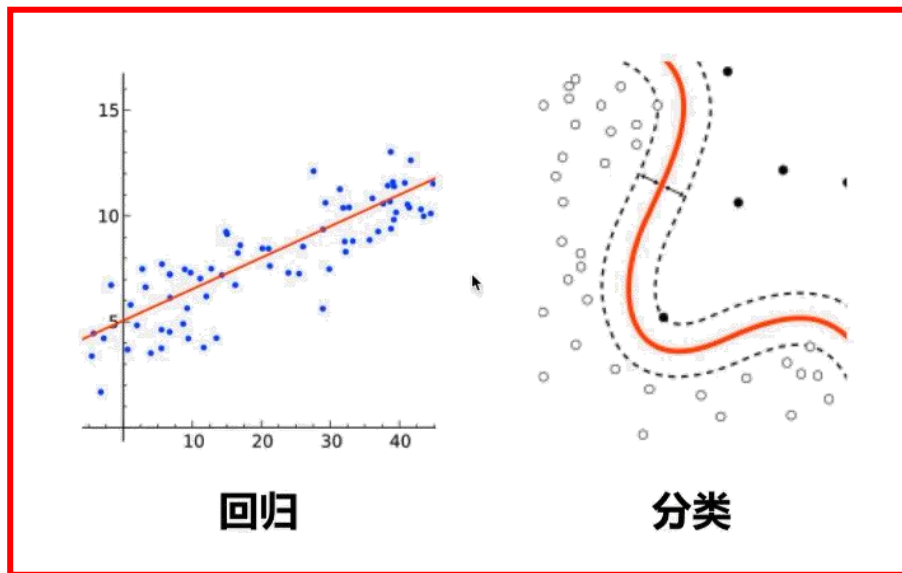
输入：一组带label的训练数据，label表明样本所属类别。

训练：使用训练集学习模型参数，学习拟合数据的分类器。

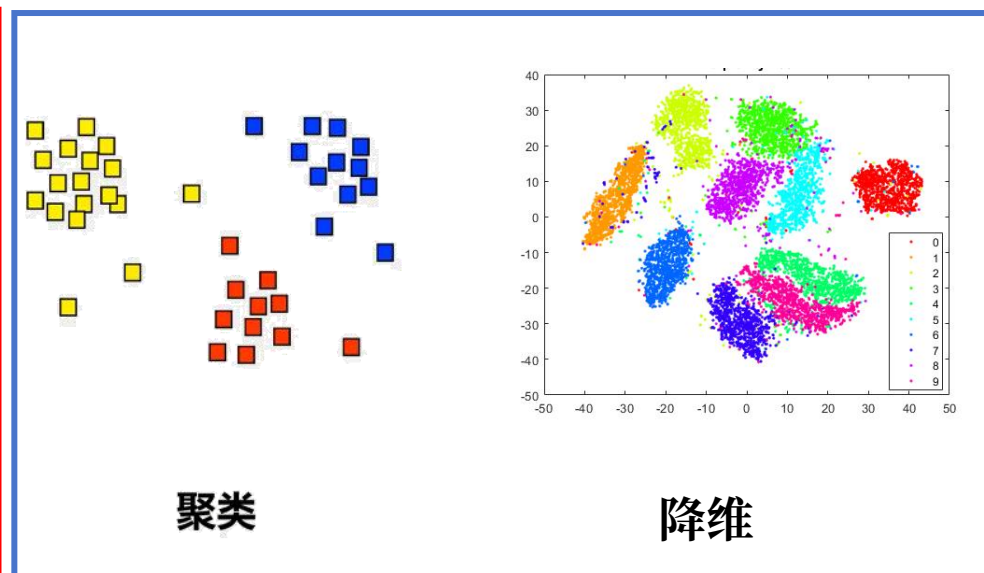
输出：预测的样本label。

# 机器学习类型

## 监督学习



## 无监督学习



### ◆ 聚类

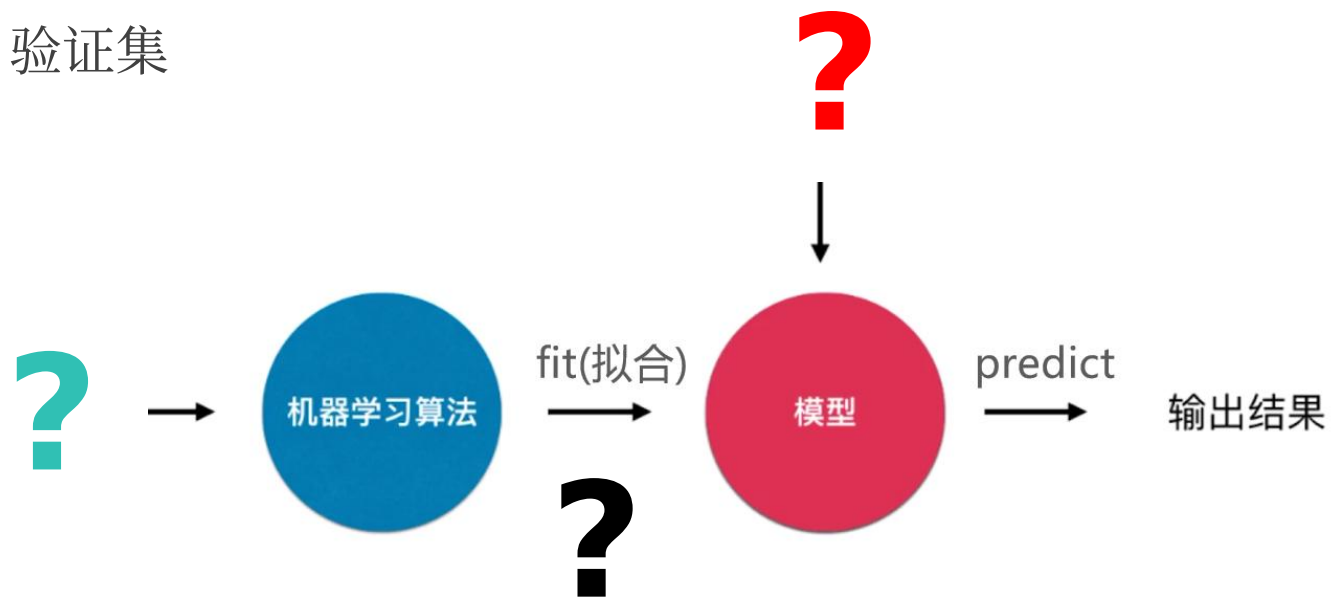
输入：一组不带label的训练数据

训练：使用训练集学习模型参数的分类器。

输出：对数据的类标识。

# 数据准备

- 训练集、测试集、验证集

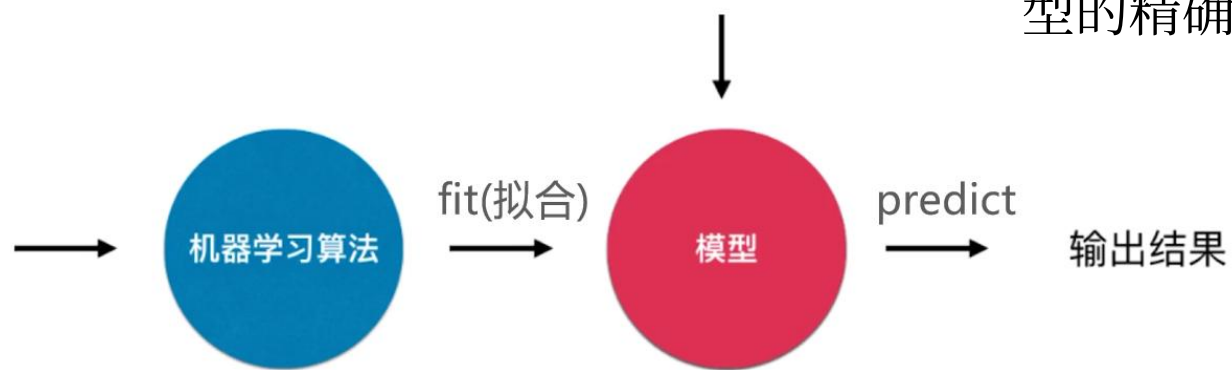


# 数据准备

- 训练集、测试集、验证集

## 测试集 (Test Set) :

为了测试已经训练好的模型的精确度。



## 训练集 (Training Set) :

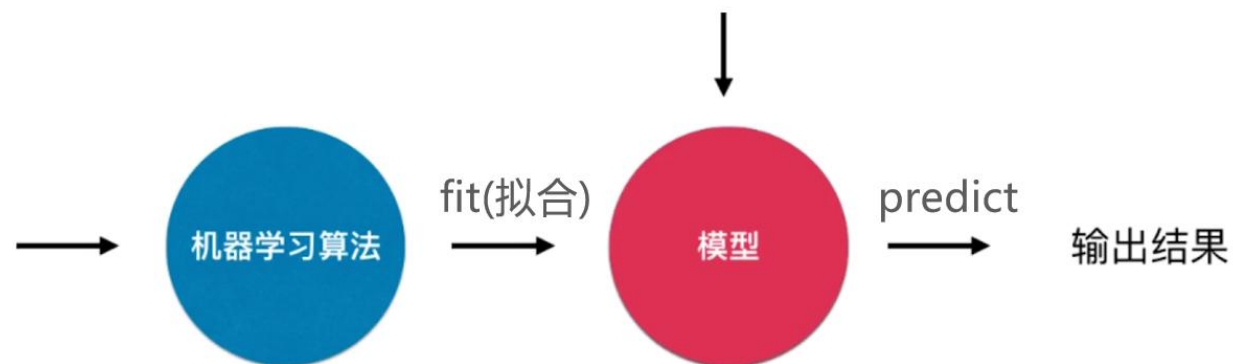
帮助我们训练模型，简单的说就是通过训练集的数据让我们确定拟合曲线的参数。

## 【可选】验证集 (Validation Set) :

也叫做开发集 (Dev Set)，用来做模型选择 (model selection)，辅助模型超参数的构建、优化及确定

# 数据准备

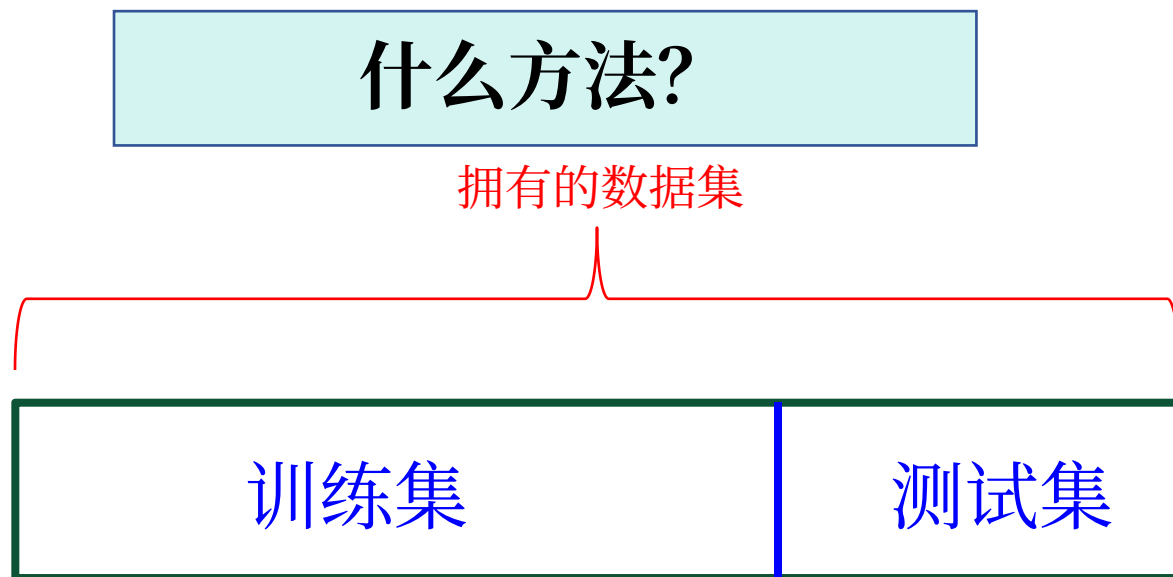
- 训练集、测试集、验证集



测试集与训练集“互斥”

# 数据准备

- 数据集的划分
  - 留出法 (hold-out)

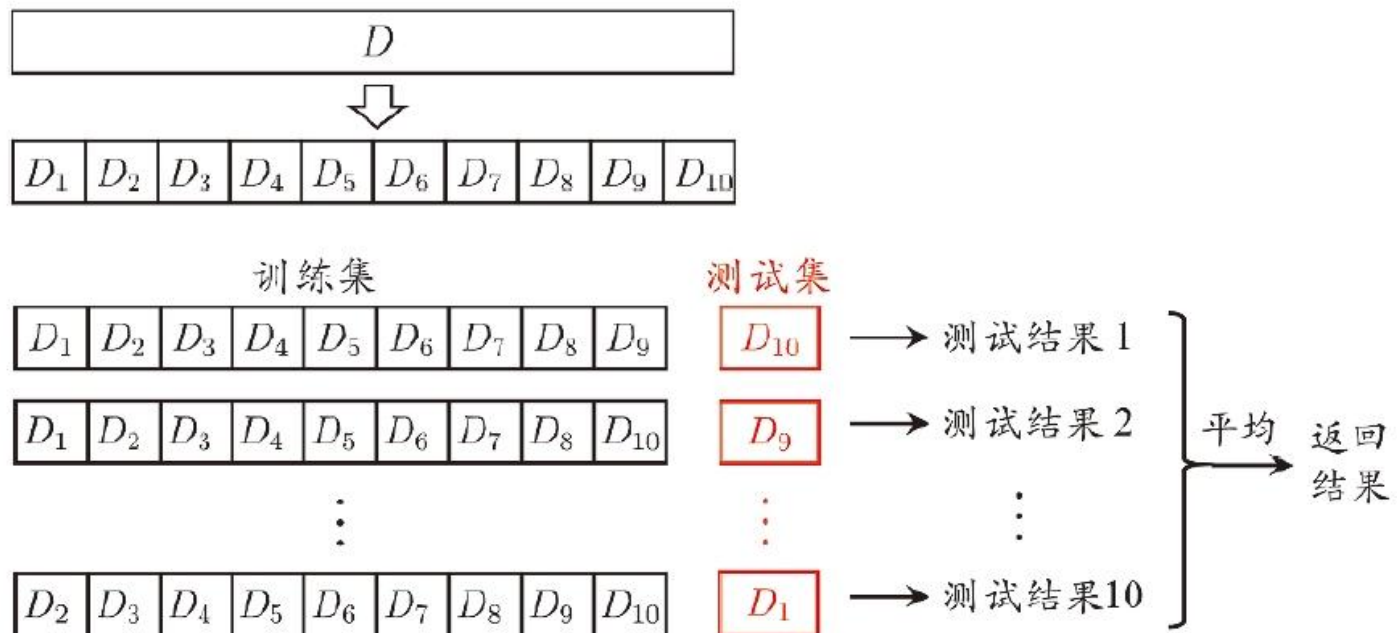


# 数据准备

## • 数据集的划分

- 留出法 (hold-out)
- K折交叉验证法 (cross validation)

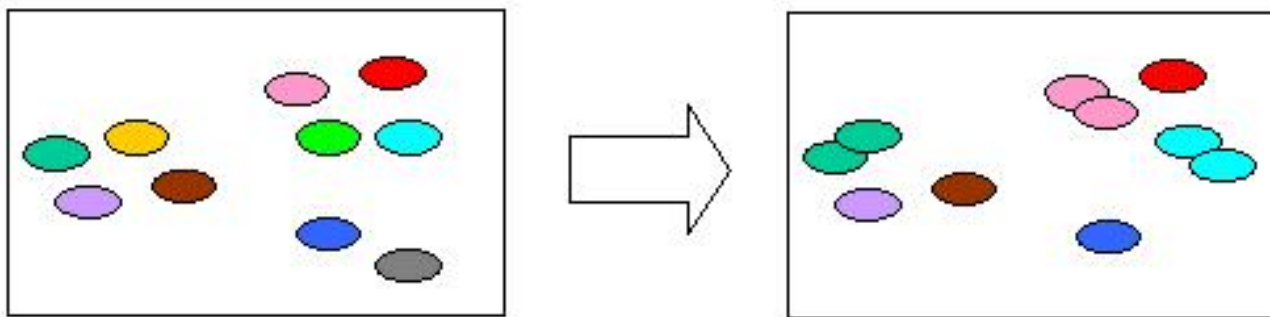
什么方法？图中 $k=?$



# 数据准备

- 数据集的划分

- 留出法 (hold-out)
- K折交叉验证法 (cross validation)
- 自助法



- 基于“自助采样” (bootstrap sampling), 亦称“有放回采样”、“可重复采样”
- 训练集与原样本集同规模, 但数据分布有所改变

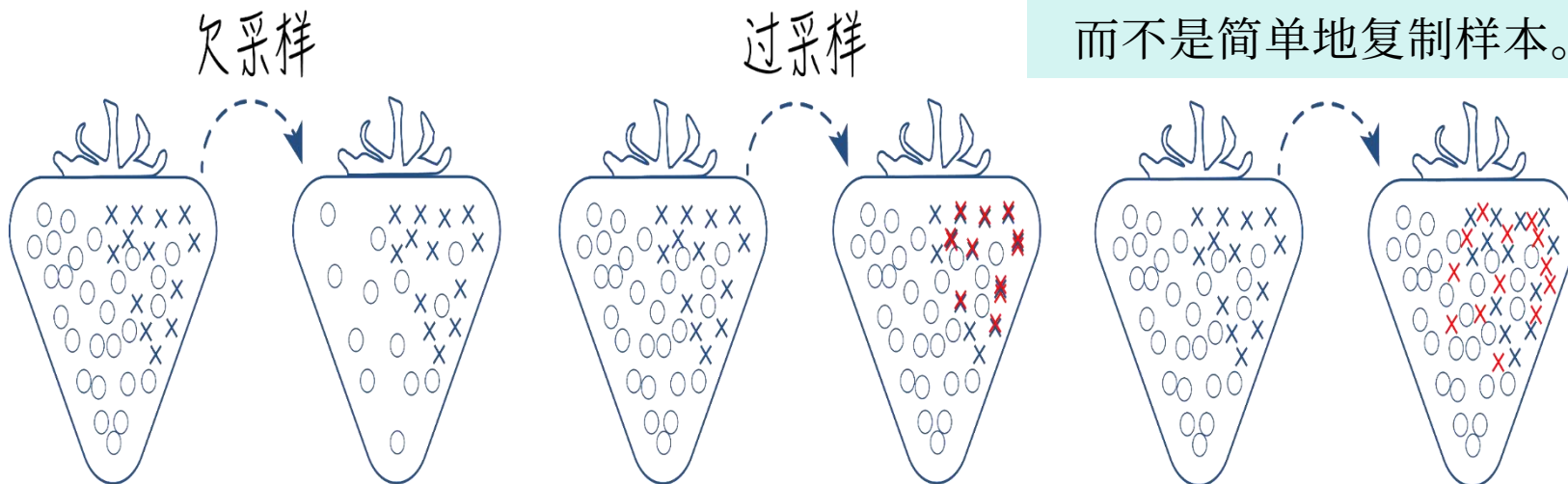
# 数据准备

## • 不平衡数据 (Imbalanced Data)

各个类别的样本量极不均衡的数据。以二分类问题为例，假设正类的样本数量远大于负类的样本数量

常见处理方法：

1) 采样法



SMOTE(算法的思想是合成新的少数类样本，而不是简单地复制样本。)

# 数据准备

- 不平衡数据 (Imbalanced Data)

各个类别的样本量极不均衡的数据。以二分类问题为例，假设正类的样本数量远大于负类的样本数量

常见处理方法：

1) 采样法

2) 代价敏感学习：为不同类别的样本提供不同的权重，  
让模型有倾向性地进行学习

# 回归模型的性能评估

- 均方误差:

$$E(f; D) = \frac{1}{m} \sum_{i=1}^m (f(\mathbf{x}_i) - y_i)^2$$

- 相关系数

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}$$

- $R^2$

$$SSR = \sum_{i=1}^{i=n} (\hat{y}_i - \bar{y})^2$$

$$SST = \sum_{i=1}^{i=n} (y_i - \bar{y})^2$$

$$SSE = \sum_{i=1}^{i=n} (y_i - \hat{y}_i)^2$$

$$R^2 = \frac{SSR}{SST} = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

模型可解释的方差占总方差的比例

# 分类模型的性能评估

(二分类任务)

- 混淆矩阵

真实情况	预测结果	
	正例	反例
正例	$TP$ (真正例)	$FN$ (假反例)
反例	$FP$ (假正例)	$TN$ (真反例)

- 由混淆矩阵可得到：
  - 准确率 (accuracy, ACC)
  - 灵敏度/敏感度 (sensitivity)
  - 精确度 (precision)
  - 特异度 (specificity)

# 分类模型的性能评估

(二分类任务)

- 由混淆矩阵可得到:

准确率 (accuracy, ACC) : 描述分类器的分类准确率

$$\text{Sen} = \text{TP} / \text{P} = \text{TP} / (\text{TP} + \text{FN})$$

灵敏度/敏感度 (sensitivity) : 所有正例中, 被分对的比例, 也称作查全率 (true positive rate) 或召回率 (recall)

$$\text{ACC} = (\text{TP} + \text{TN}) / (\text{TP} + \text{FP} + \text{FN} + \text{TN})$$

# 分类模型的性能评估

(二分类任务)

- 由混淆矩阵可得到:

精确度 (precision) : 被分为正例的示例中, 实际为正例的比例

$$\text{Pre} = \text{TP} / (\text{TP} + \text{FP})$$

特异度 (specificity) : 所有负例中, 被分对的比例, 也称作 true negative rate

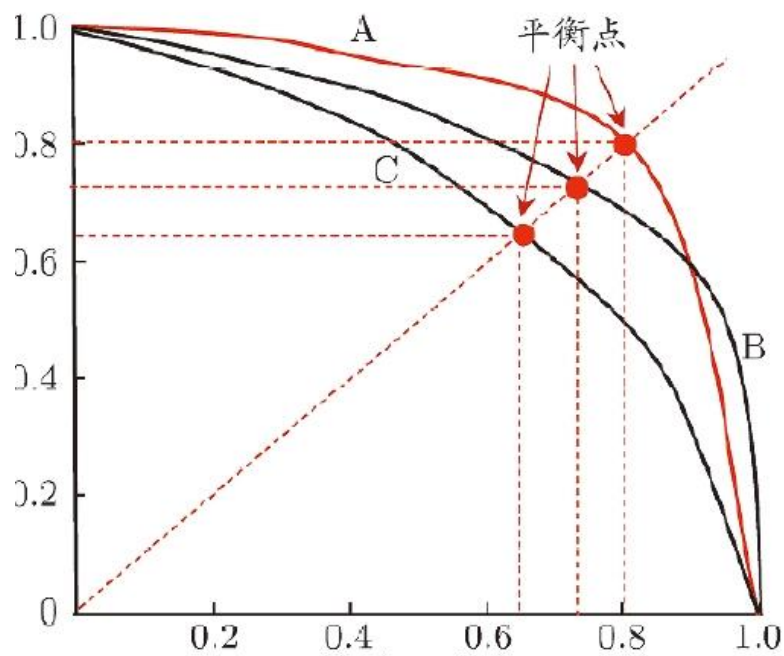
$$\text{Spe} = \text{TN} / (\text{FP} + \text{TN})$$

# 分类模型的性能评估

(二分类任务)

- PR图和BEP

精确率precision (纵坐标) vs 召回率recall (横坐标)



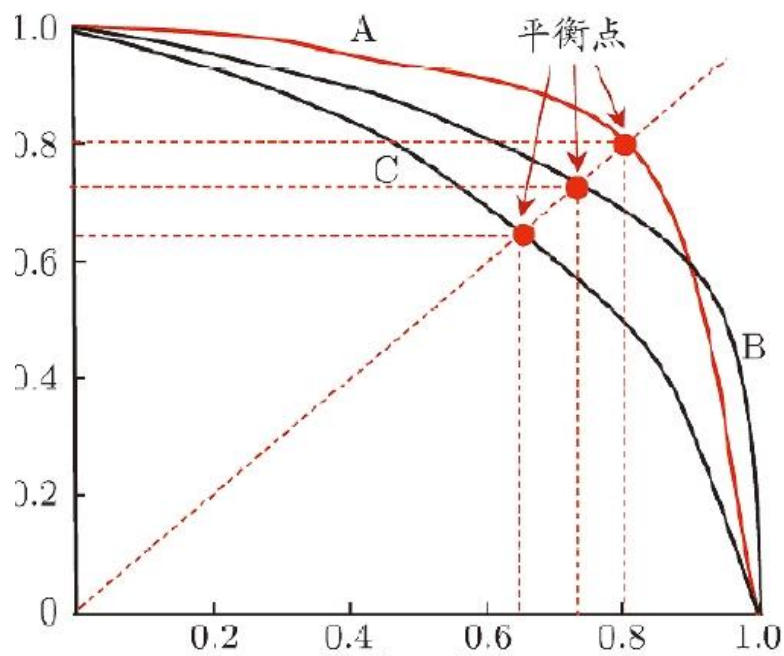
- 如果一个学习器的PR曲线被另一个学习器的PR曲线完全包住，则可断言后者的性能优于前者，例如：
- A和B优于C

# 分类模型的性能评估

(二分类任务)

- PR图和BEP

精确率precision (纵坐标) vs 召回率recall (横坐标)



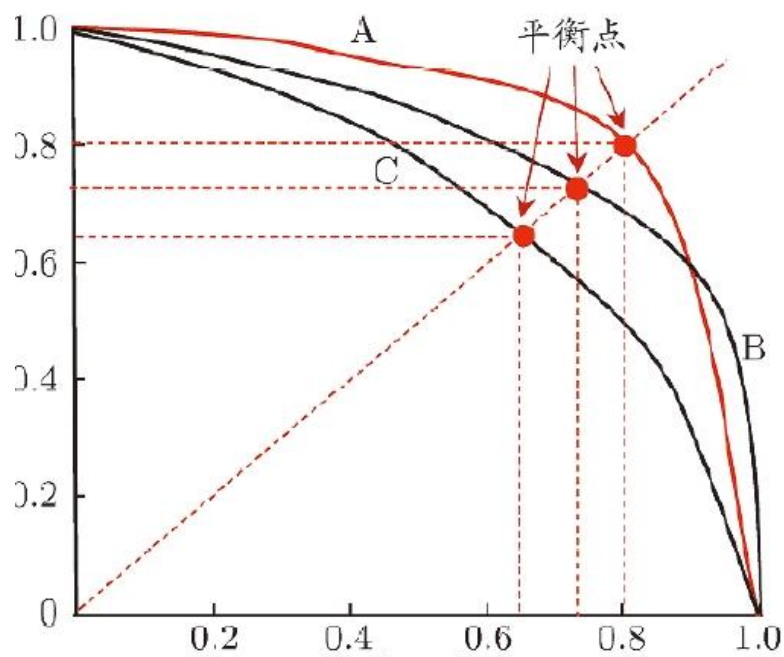
- 但是A和B的性能无法直接判断?
- 计算BEP、F1值, 或根据曲线下方的面积大小来进行比较
- BEP (平衡点) 是 $P=R$ 时的取值, 如果这个值较大, 则说明学习器的性能较好。

# 分类模型的性能评估

(二分类任务)

- PR图和BEP

精确率precision (纵坐标) vs 召回率recall (横坐标)



比较A、B、C的性能?

BEP:

- 学习器 A 优于 学习器 B
- 学习器 A 优于 学习器 C
- 学习器 B 优于 学习器 C

# 分类模型的性能评估

(二分类任务)

- F1值

精确率precision vs 召回率recall

若对 
$$F1 = \frac{2 \times P \times R}{P + R} = \frac{2 \times TP}{\text{样例总数} + TP - TN}$$

$$F_{\beta} = \frac{(1 + \beta^2) \times P \times R}{(\beta^2 \times P) + R}$$

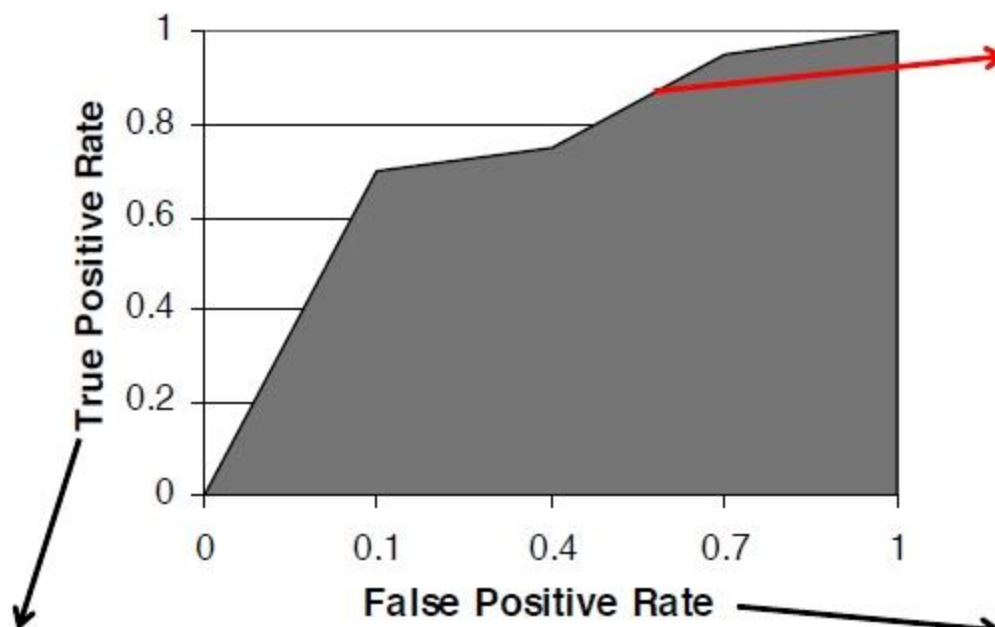
$\beta > 1$  时查全率有更大影响;  $\beta < 1$  时查准率有更大影响

# 分类模型的性能评估

(二分类任务)

- 曲线下方的面积

AUC: Area Under the ROC Curve, 面积越大, 性能越好



# 聚类模型的性能评估

- ◆ 均一性:  $p$

类似于精确率

$$p = \frac{1}{k} \sum_{i=1}^k \frac{N(C_i == K_i))}{N(K_i)}$$

- ◆ 完整性:  $r$

类似于召回率

$$r = \frac{1}{k} \sum_{i=1}^k \frac{N(C_i == K_i))}{N(C_i)}$$

- ◆ V-measure:

均一性和完整性的加权平均

$$V = \frac{(1 + \beta^2) * pr}{\beta^2 * p + r}$$

- ◆ 轮廓系数

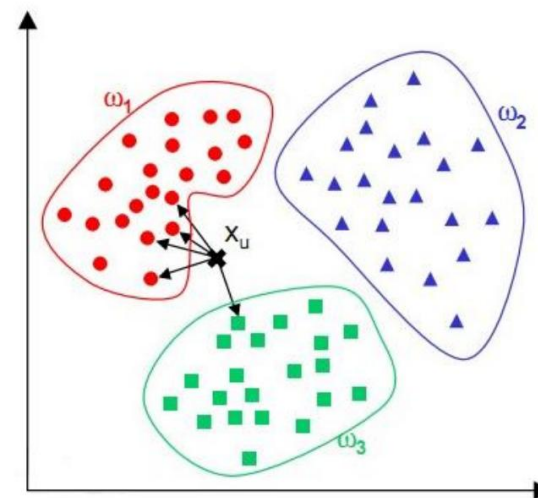
$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

簇内不相似度 $a(i)$ : 样本 $i$ 到同簇其它样本的平均距离

簇间不相似度 $b(i)$ : 样本 $i$ 到其它簇的所有样本的平均距离

# kNN方法解决分类问题

- 思路：基于实例
  - 计算待分类样本与训练集中的所有样本的距离；
  - 取距离最小的前k个点，根据“少数服从多数”的原则，将该样本划分为出现次数最多的那个类别；
  - 注意参数k的影响



# kNN方法解决分类问题

- 步骤
  - 对于未知类别属性数据集中的点:
  - 计算已知类别数据集中的点与当前点的距离
  - 按照距离依次排序
  - 选取与当前点距离最小的K个点
  - 确定前K个点所在类别的出现概率
  - 返回前K个点出现频率最高的类别作为当前点预测分类

## 关键是?

L1 (Manhattan) distance

$$d_1(I_1, I_2) = \sum_p |I_1^p - I_2^p|$$

L2 (Euclidean) distance

$$d_2(I_1, I_2) = \sqrt{\sum_p (I_1^p - I_2^p)^2}$$

# kNN方法解决分类问题

- 优点
  - 简单有效
  - 计算复杂度和训练集中的文档数目成正比，  
也就是说，如果训练集中文档总数为 $n$ ，那么KNN 的分类时间复杂度为 $O(n)$ 。
- 缺点
  - 样本不平衡时，如一个类的样本容量很大，而其他类样本容量很小时，有可能导致当输入一个新样本时，该样本的 $K$  个邻居中大容量类的样本占多数，导致过拟合
    - 不同的样本给予不同权重项

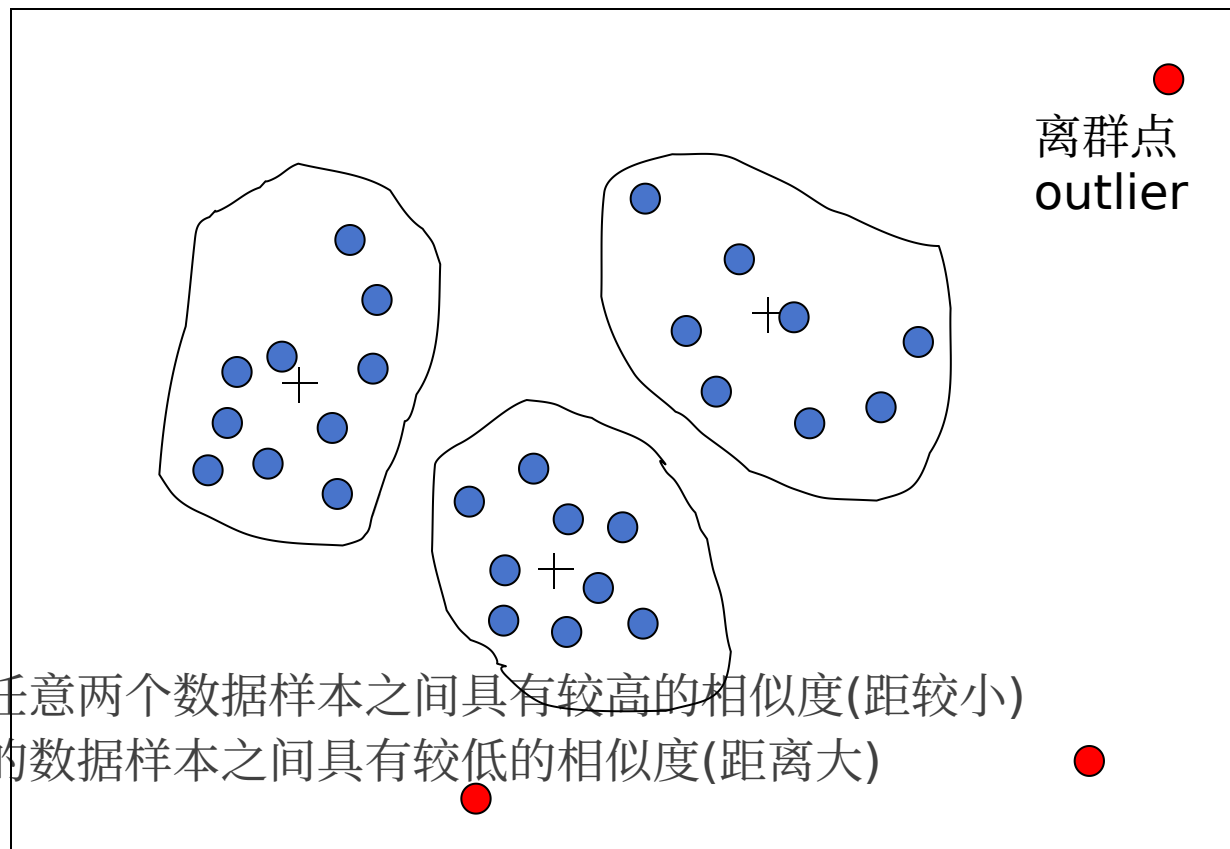


# kNN方法解决分类问题

- 改进
  - 距离加权最近邻算法
  - 伸展坐标轴或排除最小相关的特征
  - 采用网格搜索(Grid Search)等方法优化参数

# K-means方法进行聚类分析

- 思路



# K-means方法进行聚类分析

- 步骤
  - 选择 $K$ 个点作为初始质心
  - 把 $n$ 个数据样本指派到最近的质心，形成 $K$ 个簇
  - 簇内的样本相似度较高，簇间的样本相似度较低
  - 对于上一步聚类的结果，进行平均计算，得出该簇的新的聚类中心
  - 质心不发生变化，重复上述两步，直到找出有 $k$ 个簇的一个划分使得所选择的划分准则最优

关键是？

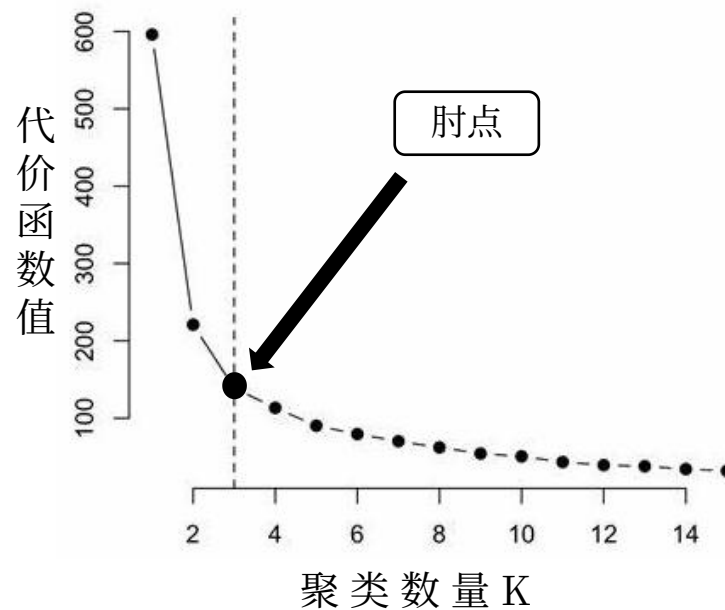
一般选择 $K=?$

# K-means方法进行聚类分析

- 步骤

K值的选择:

随着K上升, 代价函数的值迅速下降,  
在 $K=3$ 的时候达到一个肘点。  
在此之后, 代价函数的值下降得非常慢,  
所以一般选择 $K=3$ 。  
这个方法叫“肘部法则”。



肘部法则

# K-means方法进行聚类分析

- 优点
  - 复杂度较低
  - 通常以局部最优结束
  - 对于球状或团状数据分布效果非常好
- 缺点
  - 需要事先定义簇的平均值，并给出k
  - 不能处理噪声数据和孤立点
  - 不能发现非凸面形状的簇
  - 有可能会停留在一个局部最小值处

# K-means方法进行聚类分析

- 改进
  - kmedoids算法
  - 代价函数的优化
  - .....