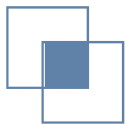


# 人工智能

Artificial Intelligence

中南大学 计算机学院





# 课程内容



1

人工智能概论

学科定义、起源与发展历程、  
人工智能与人类智能、应用场景与挑战

2

知识表示方法

状态空间表示、问题归约表示  
谓词逻辑表示、语义网络表示

3

确定性推理技术

图搜索技术、消解原理  
规则演绎、产生式系统

4

计算智能

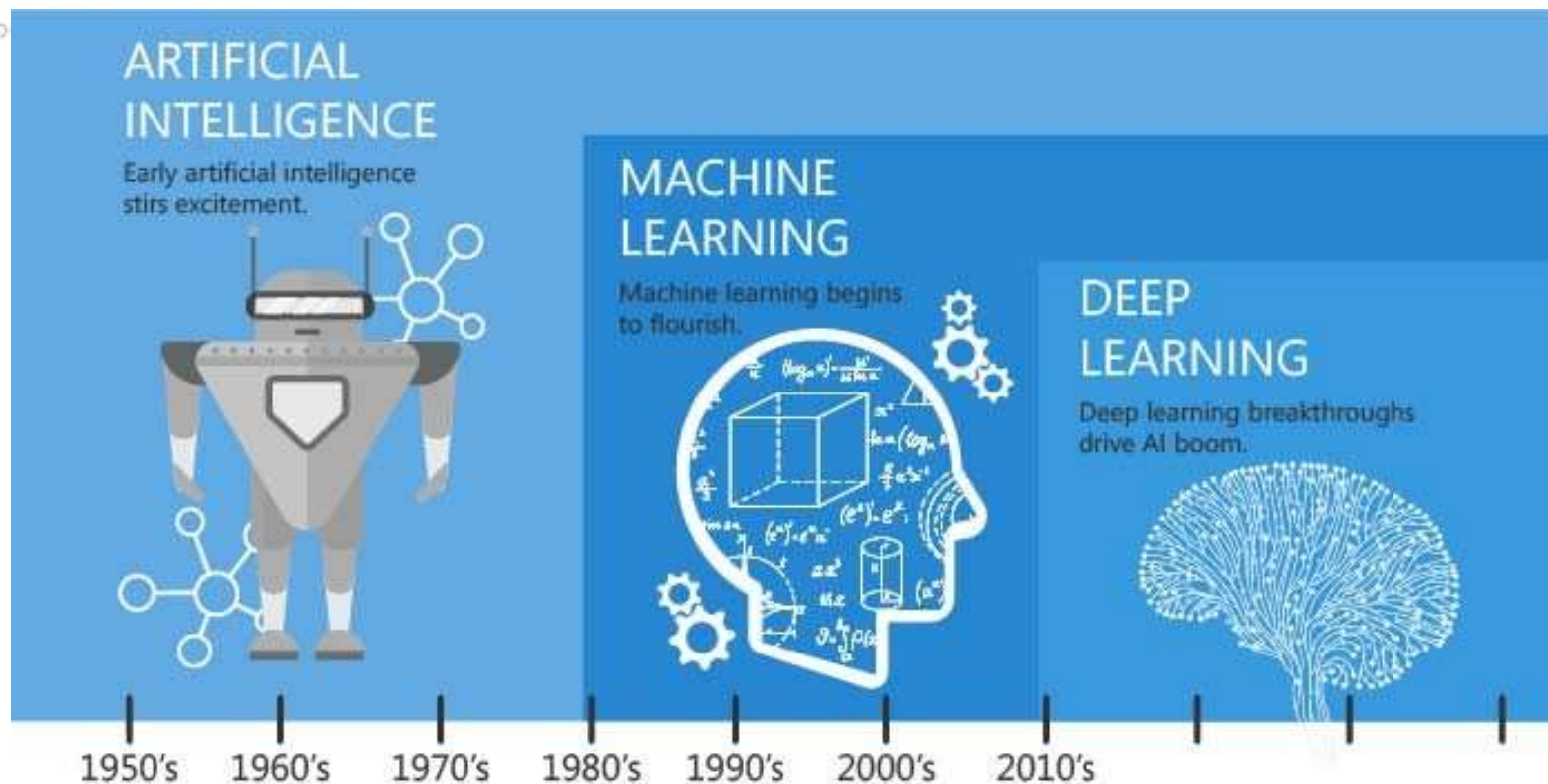
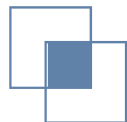
神经计算、模糊计算、遗传算法、  
粒群优化、蚁群优化

5

机器学习

决策树学习、神经网络





Since an early flush of optimism in the 1950's, smaller subsets of artificial intelligence - first machine learning, then deep learning, a subset of machine learning - have created ever larger disruptions.

- 机器学习是实现人工智能的一种方法,深度学习是机器学习一个分支。

<http://blog.aimagnifi.com/index.php/2017/10/13/what-is-the-difference-between-machine-learning-and-deep-learning/>

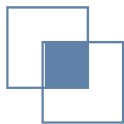




# 机器学习入门

中国科学院大学





# 目录



1

## 机器学习概论

学科定义、发展历程、  
机器学习方法、应用场景与挑战

2

## 机器学习准备

鸢尾花分类任务简介  
数据预处理、特征工程

3

## 机器学习方法

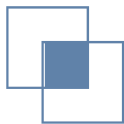
学习方法：分类、回归、聚类、其他  
机器学习模型评估

4

## 课程实践

实践：鸢尾花分类





# 目录



1

## 机器学习概论

学科定义、发展历程、  
机器学习方法、应用场景与挑战

2

## 机器学习准备

鸢尾花分类任务简介  
数据预处理、特征工程

3

## 机器学习方法

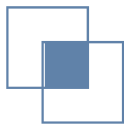
学习方法：分类、回归、聚类、其他  
机器学习模型评估

4

## 课程实践

实践：鸢尾花分类





# 机器学习概论



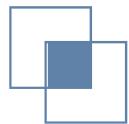
学科定义

发展历程

机器学习方法

应用场景与挑战





# 什么是机器学习



## What is machine learning?

*Machine Learning: Field of **study** that gives computers the ability to learn **without** being explicitly programmed.*

— Arthur Samuel, 1959

*Learning denotes **changes** in the system that enable a system to do the same task **more efficiently** the next time.*

— Herbert Simon, 1983

*Learning is making useful changes in the workings of our minds.*

— Marvin Minsky, 1986



亚瑟·塞缪尔  
(Arthur Samuel)



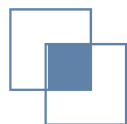
赫伯特·西蒙  
(Herbert Simon)



马文·闵斯基  
(Marvin Minsky)







# 什么是机器学习

What is machine learning?

- Machine Learning  $\langle T, P, E \rangle$ :
  - Computer automatically improves
    - at task **T**(任务)
    - according to performance metric **P**(性能)
    - through experience **E**(经验)

—— Tom Mitchell, 1997

## 下棋

**T**: 下棋

**P**: 比赛中击败对手的百分比

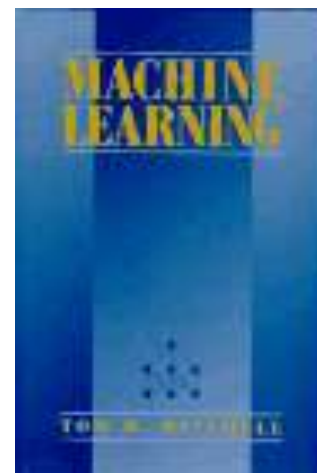
**E**: 与自己对弈的训练

## 手写体识别

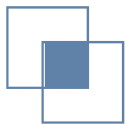
**T**: 识别手写文字

**P**: 识别的正确率

**E**: 已经做好的具有代表性分类  
的手写体数据库



汤姆·米切尔  
(Tom M. Mitchell)



# 什么是机器学习



机器学习是从人工智能中产生的一个重要学科分支，是实现智能化的关键。

机器学习 (Machine Learning) 是一门多领域**交叉学科**，涉及概率论、统计学、逼近论、凸分析、算法复杂度理论等多门学科。专门研究计算机怎样模拟或实现人类的学习行为，以获取新知识或技能，重新组织已有的知识结构使之不断改善自身的性能。

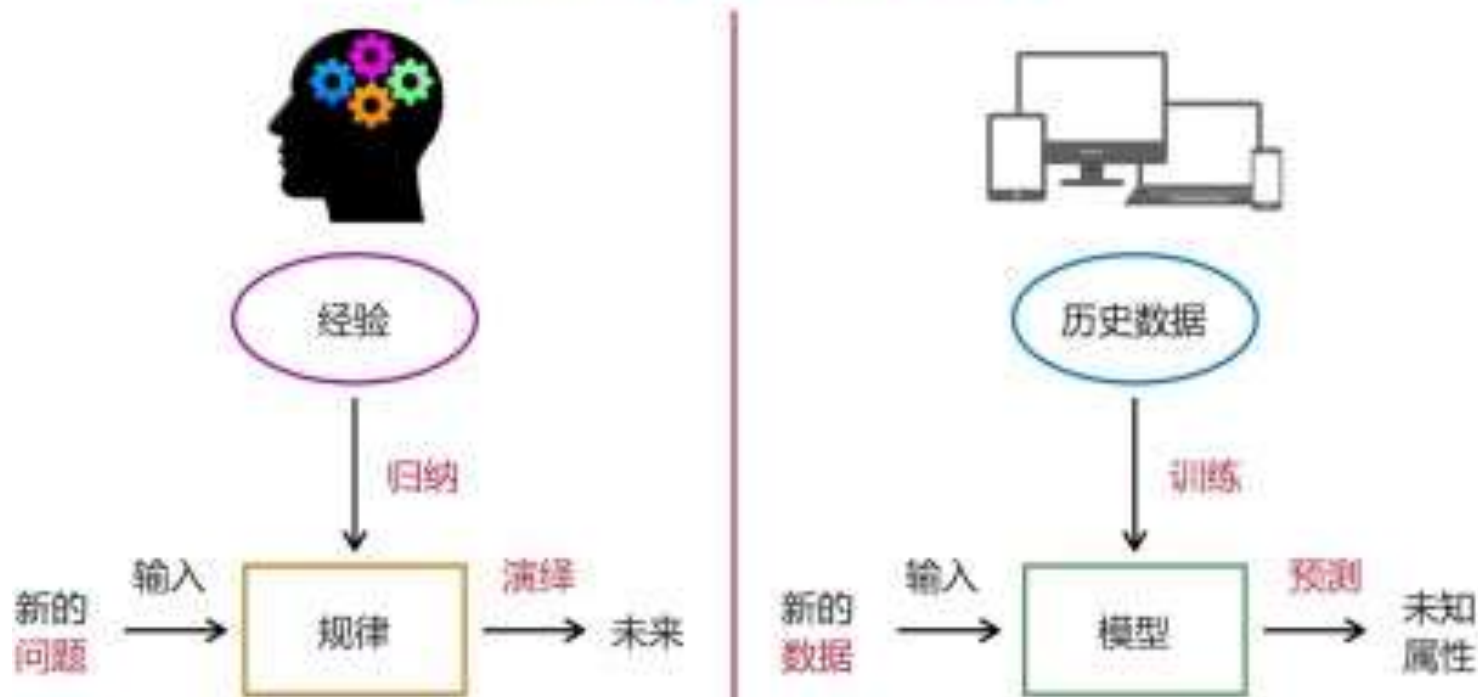
百度百科

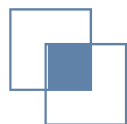
Machine learning is the study of **algorithms** and mathematical **models** that computer systems use to progressively improve their performance on a specific task. Machine learning algorithms build a mathematical model of **sample data**, known as “training data”, in order to make predictions or decisions without being explicitly programmed to perform the task.

Wikipedia



## 人类思考 vs 机器学习





# 机器学习的一般过程



$f(\text{狗}) = \text{狗}$

$f(\text{狗}) = \text{狗}$

$f(\text{狗}) = \text{狗}$

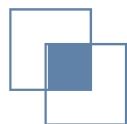
$f(\text{狗}) = \text{狗}$

$f(\text{狗}) = ?$

$f(\text{狗}) = ?$

泛化问题



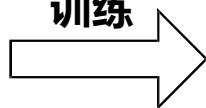


# 机器学习的一般过程

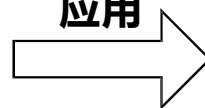


训练数据

训练



应用



未知测试数据



模型

$$\mathcal{F} = \{f | Y = f(x), x \in \mathbb{R}^n\}$$



策略

损失函数选择

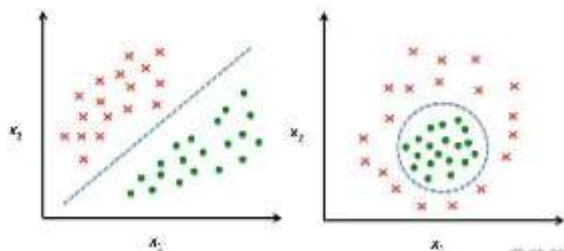
模型选择

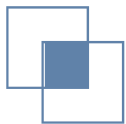


算法

在假设空间 $\mathcal{F}$ , 确定参数 $\theta$

最优化





# 机器学习概论



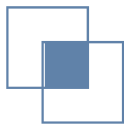
学科定义

发展历程

机器学习方法

应用场景与挑战





# 发展历程



## 推理期（20世纪50-70年代初）

- 认为只要给机器赋予逻辑推理能力，机器就能具有智能
- A. Newell 和 H. Simon 的“逻辑理论家”、“通用问题求解”程序，获得了1975年图灵奖



## 知识期（20世纪70年代中期）

- 认为要使机器具有智能，就必须设法使机器拥有知识
- E.A. Feigenbaum 作为“知识工程”之父在 1994 年获得了图灵奖



## 学科形成（20世纪80年代）

20 世纪 80 年代是机器学习成为一个独立学科领域并开始快速发展、各种机器学习技术百花齐放

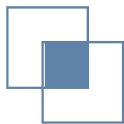
- 1980 年美国卡内基梅隆大学举行第一届机器学习研讨会
- 1990 年《机器学习:风范与方法》出版



## 繁荣期（20世纪80年代-至今）

- 20 世纪 90 年代后，统计学习方法占主导，代表为SVM
- 2006 至今，大数据分析的需求，神经网络又被重视，成为深度学习理论的基础





# 机器学习概论



学科定义

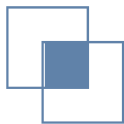
发展历程

机器学习方法

应用场景与挑战







# 机器学习方法



**有监督学习 (supervised learning)**：从给定的**有标注的训练数据集**中学习出一个函数（模型参数），当新的数据到来时可以根据这个函数预测结果。常见任务包括**分类**与**回归**。

**分类**：输出是类别标签

**Classification**: Y is discrete

Y: 年轻人(1), 老年人(-1)

X:  $x_1$  黑头发的比例, 值域 (0, 1);

$x_2$  行走速度, 值域 (0, 100) 米/每分钟.

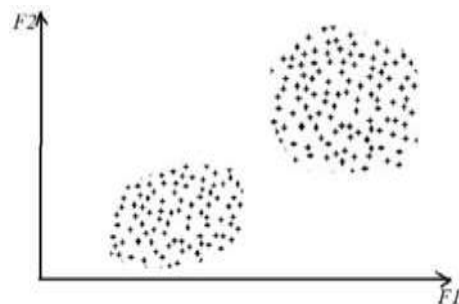
Training Data:

Y=1: (1, 99)、(0.9, 80)、(0.80, 100) ...

Y=-1: (0.2, 30)、(0.5, 50)、(0.4, 30) ...

Test:

X=(0.85, 98), Y=?



**回归**：输出是实数

**Regression**: Y is continue

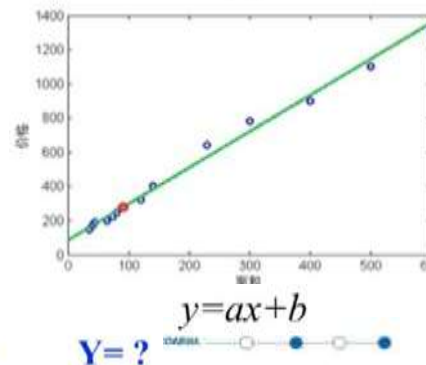
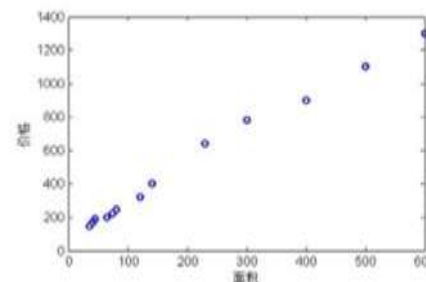
Y: 房屋价钱 (万元), 值域  $Y \geq 0$ .

X:  $x_1$ =房屋面积  $m^2$ .

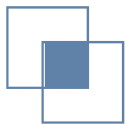
Training Data:

|     |      |
|-----|------|
| 35  | 150  |
| 40  | 170  |
| 45  | 190  |
| 65  | 200  |
| 74  | 224  |
| 80  | 245  |
| 120 | 320  |
| 140 | 400  |
| 230 | 640  |
| 300 | 780  |
| 400 | 900  |
| 500 | 1100 |
| 600 | 1300 |

Test: X=90



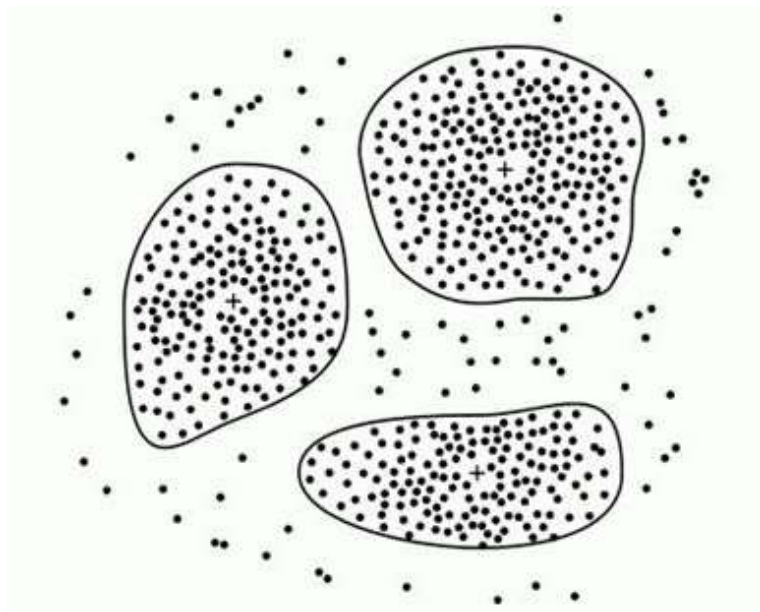
Y=?



# 机器学习方法



**无监督学习 (unsupervised learning)**：没有标注的训练数据集，需要根据样本间的统计规律对样本集进行分析，常见任务如**聚类**等。



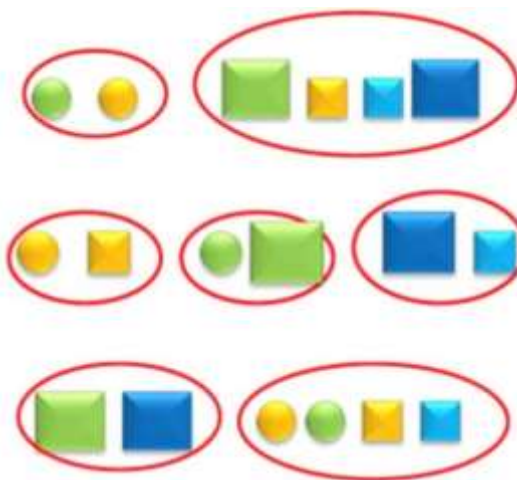
## Clustering:

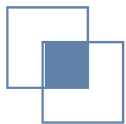
**X:** (颜色, 形状, 大小)

**Data:**



For all the data,  $Y=?$





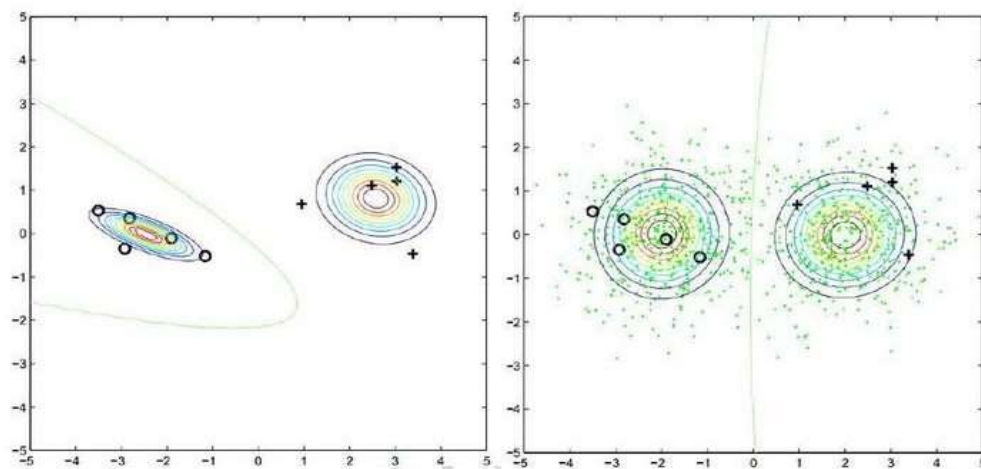
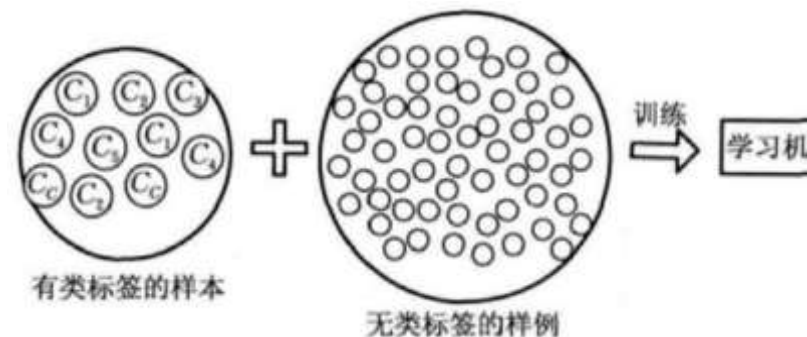
# 机器学习方法

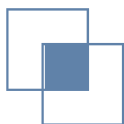


**半监督学习 (Semi-supervised learning)**：结合 **(少量的) 标注训练数据** 和 **(大量的) 未标注数据** 来进行数据的分类学习。

**两个基本假设：**

- **聚类假设**：处在相同聚类中的样本示例有较大的可能拥有相同的标记。
- **流形假设**：处于一个很小的局部区域内的样本示例具有相似的性质，因此，其标记也应该相似。





# 机器学习方法



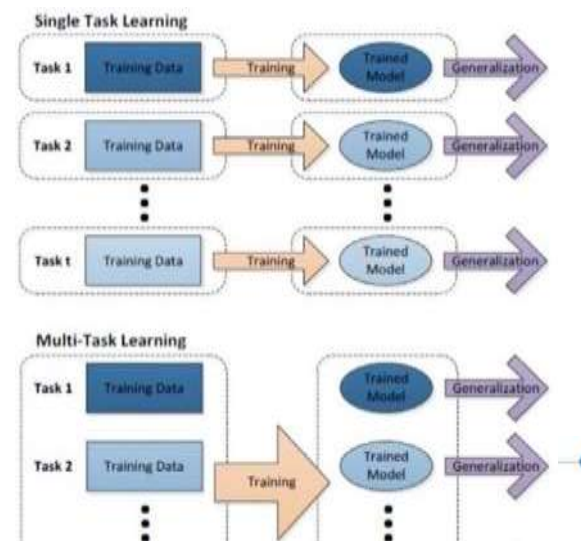
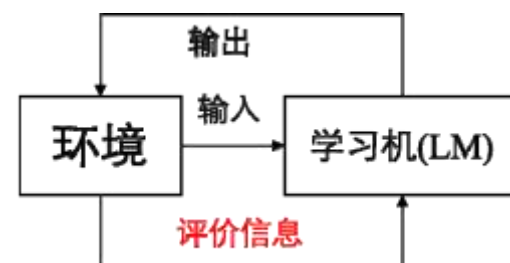
**增强学习 (Reinforcement Learning) :** 外部环境对输出只给出评价信息而非正确答案, 学习机通过强化受奖励的动作来改善自身的性能。

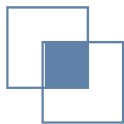
如: 让计算机学着去玩Flappy Bird

我们不需要设置具体的策略, 比如先飞到上面, 再飞到下面, 我们只是需要给算法定一个“小目标”! 比如当计算机玩的好的时候, 我们就给它一定的奖励, 它玩的不好的时候, 就给它一定的惩罚, 在这个算法框架下, 它就可以越来越好, 超过人类玩家的水平。

**多任务学习 (Multi-task Learning) :** 把多个相关 (related) 的任务放在一起同时学习。

单任务学习时, 各个任务之间的模型空间 (Trained Model) 是相互独立的, 但现实世界中很多问题不能分解为一个一个独立的子问题, 且这样忽略了问题之间所包含的丰富的关联信息。多任务学习就是为了解决这个问题而诞生的。多个任务之间共享一些因素, 它们可以在学习过程中, 共享它们所学到的信息, 相关联的多任务学习比单任务学习具备更好的泛化 (generalization) 效果。





# 机器学习概论



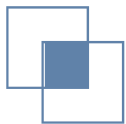
学科定义

发展历程

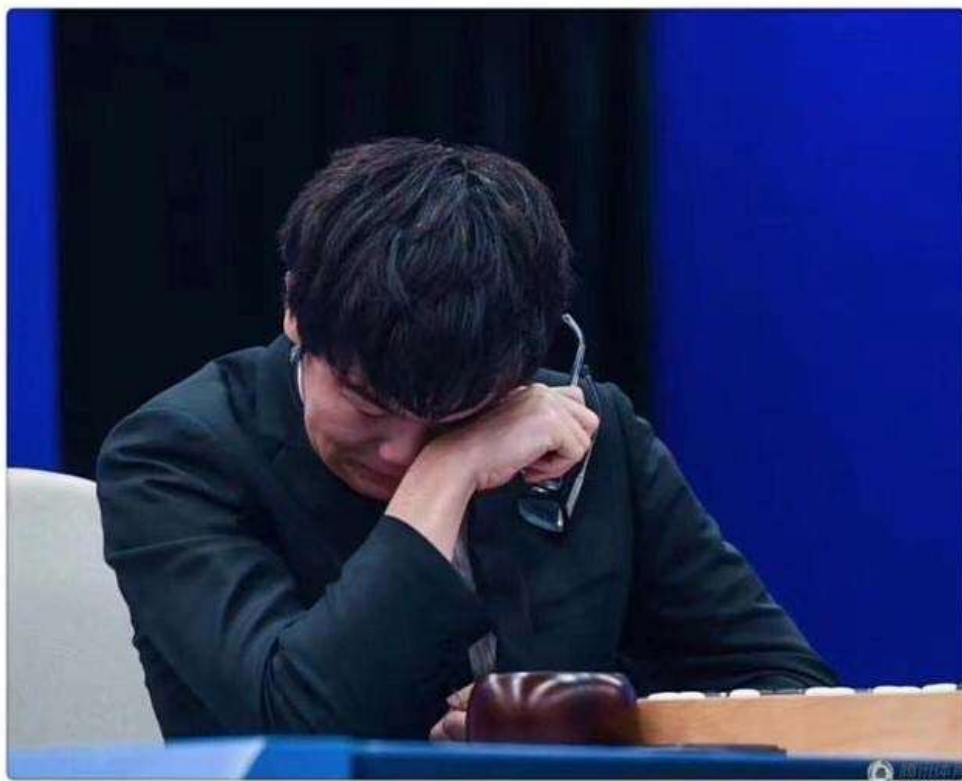
机器学习方法

应用场景与挑战





## 3:0 ! AlphaGo 完胜柯洁



**柯洁：**中国围棋职业九段棋手，世界排名第一

**AlphaGo：**Google DeepMind 开发的机器学习围棋程序

AlphaGo使用**蒙特卡罗树**搜索与两个**深度神经网络**相结合的方法，其中一个是以估值网络来评估大量的选点，而以走棋网络来选择落子。





# 无人驾驶车队亮相2018春晚



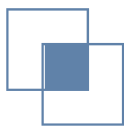
百度发布“**Apollo（阿波罗）**”软件平台，向汽车行业及自动驾驶领域提供一套完整的平台。

无人驾驶主要包括三个环节：

感知、**决策**、和控制

核心技术：异步多传感器同步+深度**数据融合**





# 机器学习已无处不在

**搜索引擎：**网页、图片、视频、新闻、学术、地图

**信息推荐：**新闻、商品、游戏、书籍

**图片识别：**人像、用品、动物、交通工具

**用户分析：**社交网络、影评、商品评论

**机器翻译、摘要生成.....**

**生物信息学.....**



相似图片



相关网页

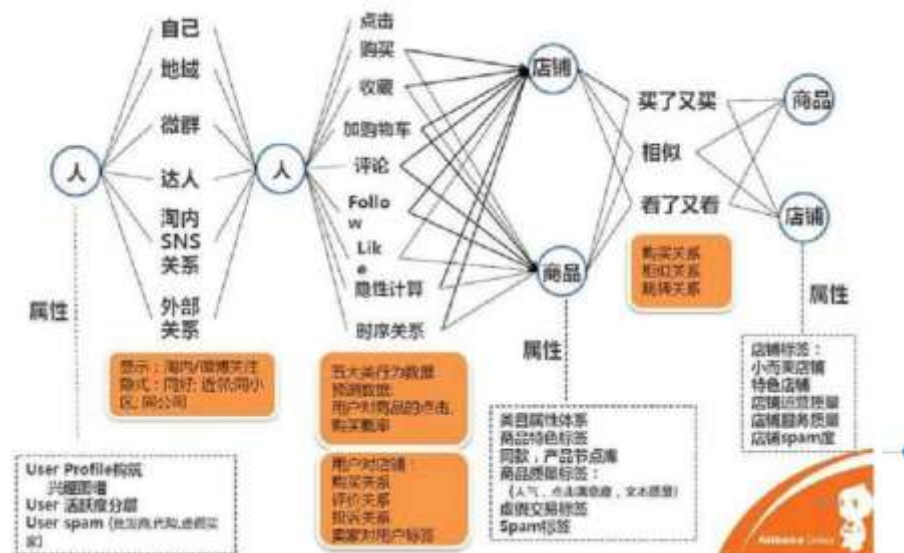


Google的成功，使得Internet搜索引擎成为一个新兴的产业

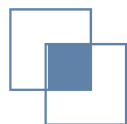
不仅有众多专营搜索引擎的公司出现（例如专门针对中文搜索的就有慧聪、百度等），而且Microsoft等巨头也开始投入巨资进行研发

Google掘到的第一桶金，来源于其创始人Larry Page和Sergey Brin提出的PageRank算法

机器学习技术正在支撑着各类搜索引擎（尤其是贝叶斯学习技术）







# 机器学习无所不能?

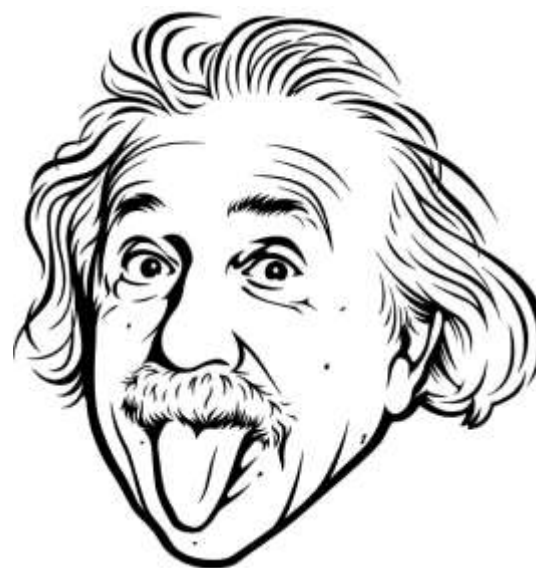


◆**问题思考**: 机器学习是否无所不能?



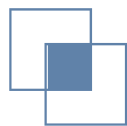
规则、计算、模式

**V.S.**



思想、创意、情感



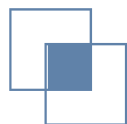


# 机器学习面临的难题与挑战



- ◆ **数据稀疏性**：训练一个模型，需要大量（标注）数据，但是数据往往比较稀疏。
- ◆ **高数量和高质量标注数据需求**：获取标定数据需要耗费大量人力和财力。而且，人会出错，有主观性。
- ◆ **冷启动问题**：对于一个新产品，在初期，要面临数据不足的冷启动问题。
- ◆ **泛化能力问题**：训练数据不能全面、均衡的代表真实数据。

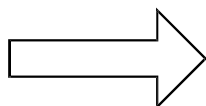




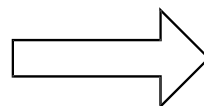
# 机器学习面临的难题与挑战



模型



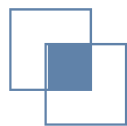
策略



算法

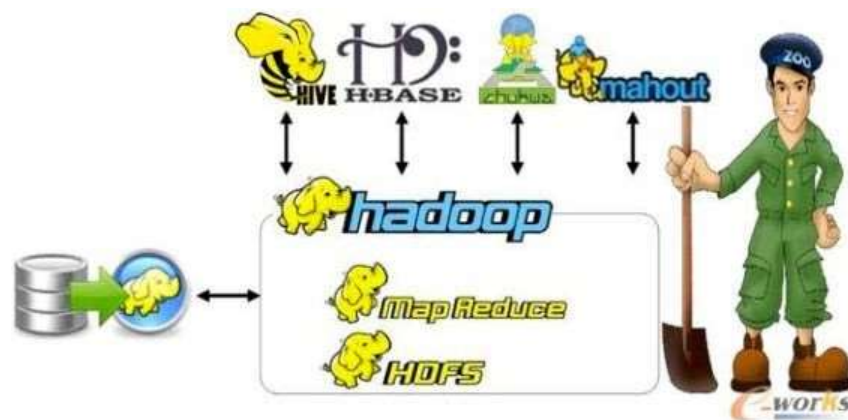
- ◆ **模型抽象困难：** 总结归纳实际问题中的数学表示非常困难。
- ◆ **模型评估困难：** 在很多实际问题中，很难形式化的、定量的评估一个模型结果的好坏。
- ◆ **寻找最优解困难：** 要解决的实际问题非常复杂，将其形式化后的目标函数也非常复杂，往往在目前还不存在一个有效的算法能找到目标函数的最优值。

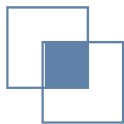




# 机器学习面临的难题与挑战

- ◆ **Scalability** 是互联网的核心问题之一。搜索引擎索引的重要网页超过 100 亿: 如果 1 台机器每秒处理 1000 网页, 需要至少 100 天。
- ◆ **速度** 是互联网核心的用户体验。线下模型训练可以花费很长时间: 比如, Google 某个模型更新一次需要几千台机器, 大约训练半年时间。但是, 线上使用模型的时候要求一定要 “快, 实时 (real-time)”
- ◆ **online learning**: 互联网每时每刻都在产生大量新数据, 要求模型随之不停更新, 所以 online learning 是机器学习的一个重要研究方向。





# 目录



1

## 机器学习概论

学科定义、发展历程、  
机器学习方法、应用场景与挑战

2

## 机器学习准备

鸢尾花分类任务简介  
数据预处理、特征工程

3

## 机器学习方法

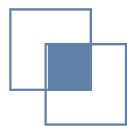
学习方法：分类、回归、聚类、其他  
机器学习模型评估

4

## 课程实践

实践：鸢尾花分类





# 机器学习准备



**问题：** 如何对如下三种鸢尾花进行分类



山鸢尾

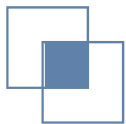


变色鸢尾



维吉尼亚鸢尾





# 机器学习准备



数据预处理

数据清洗、数据集成、数据采样



特征工程

特征编码、特征选择、特征降维、规范化



# 数据清洗

□ 对各种脏数据进行对应方式的处理，得到标准、干净、连续的数据，提供给数据统计、数据挖掘等使用。

## ● 数据的完整性

例如人的属性中缺少性别、籍贯、年龄等；  
解决方法：信息补全（使用身份证件号码推算性别、籍贯、出生日期、年龄等）；剔除；

## ● 数据的合法性

例如获取数据与常识不符，年龄大于150岁；  
解决方法：**设置字段内容**（日期字段格式为“2010-10-10”）；**类型的合法规则**（性别 in [男、女、未知]）

## ● 数据的一致性

例如不同来源的不同指标，实际内涵是一样的，或是同一指标内涵不一致；  
解决方法：建立数据体系，包含但不限于指标体系、维度、单位、频度等

## ● 数据的唯一性

例如不同来源的数据出现重复的情况等；  
解决方法：按主键去重（用sql或者excel“去除重复记录”） / 按规则去重（如不同渠道来的客户数据，可以通过相同的关键信息进行匹配，合并去重）

## ● 数据的权威性

例如出现多个来源的数据，且数值不一样；  
解决方法：为不同渠道设置权威级别，如：在家里，首先得相信媳妇说的。。。



# 数据采样

## □ 数据不平衡 (imbalance)

- ✓ **指数据集的类别分布不均。**比如说一个二分类问题，100个训练样本，比较理想的情况是正类、负类样本的数量相差不多；而如果正类样本有99个、负类样本仅1个，就意味着存在类不平衡。
- ✓ 此时预测时就算全部为正，准确率也可以达到99%，这并**不能反映模型的好坏**。

面临**不平衡数据集**的时候，传统的机器学习模型的评价方法不能精确地衡量模型的性能

## □ 解决方法

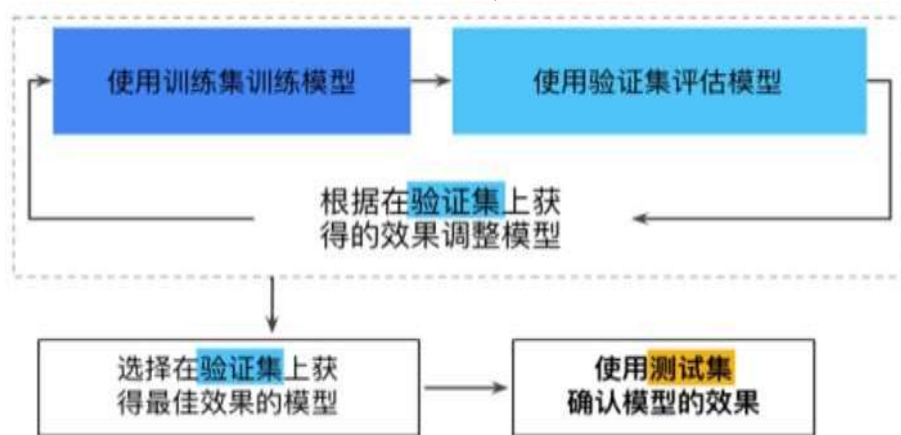
- ✓ **过采样** (Over-Sampling)  
通过随机复制少数类来增加其中的实例数量，从而可增加样本中少数类的代表性。
- ✓ **欠采样** (Under-Sampling)  
通过随机地消除占多数的类的样本来平衡类分布；直到多数类和少数类的实例实现平衡。

# 数据集拆分



## □ 机器学习中将数据划分为3份

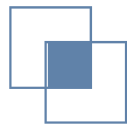
- ① **训练数据集 (train dataset)** : 用来构建机器学习模型
- ② **验证数据集 (validation dataset)** : 辅助构建模型, 用于在构建过程中评估模型, 提供无偏估计, 进而调整模型参数
- ③ **测试数据集 (test dataset)** : 用来评估



## □ 常用拆分方法

- ✓ **留出法 (Hold-Out)** : 直接将数据集划分为互斥的集合, 如通常选择 70% 数据作为训练集, 30% 作为测试集。需要注意的是保持划分后集合数据分布的一致性, 避免划分过程中引入额外的偏差而对最终结果产生影响。
- ✓ **K-折交叉验证法** : 将数据集划分为  $k$  个大小相似的互斥子集, 并且尽量保证每个子集数据分布的一致性。这样, 就可以获取  $k$  组训练 - 测试集, 从而进行  $k$  次训练和测试,  $k$  通常取值为 10。





# 机器学习准备



数据预处理

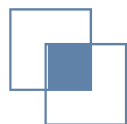
数据清洗、数据采样、数据集拆分



特征工程

特征选择、特征降维、特征编码、规范化





# 特征选择

★ 这些属性特征都有用吗?

■ 例子 1: 两类问题: 男性、女性

数据特征: 身高、体重、音频、头发长短、  
出生日期、家庭住址、籍贯、专业

■ 例子 2: 高维数据的稀疏情况

数据 1: 1, 0, 0, 0, 0, 0, 1, 3, 0, 0, 0, 0, 0, 0  
数据 2: 1, 0, 0, 0, 0, 0, 3, 3, 0, 0, 0, 0, 0, 0  
数据 3: 1, 0, 1, 0, 0, 0, 5, 3, 0, 2, 0, 0, 0, 0

特征选择



## ◆ 过滤法(Filter)

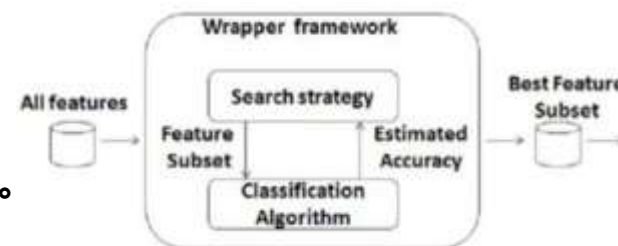
按照发散性或相关性对各特征进行评分, 设定阈值完成特征选择。

✓ 互信息: 指两个随机变量之间的关联程度, 即给定一个随机变量后, 另一个随机变量的确定性; 因而互信息取值最小为0, 意味着给定一个随机变量对确定一另一个随机变量没有关系, 越大表示另一个变量的确定性越高。

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)}$$

## ◆ 包裹法(Wrapper):

选定特定算法, 然后通过不断的启发式方法来搜索特征。



## ◆ 嵌入法(Embedded):

利用正则化的思想, 将部分特征属性的权重调整到0, 则这个特性相当于就是被舍弃了。常见的正则化有L1的Lasso, L2的Ridge, 还有一种综合L1和L2的这两个方法的Elastic Net方法。

$$\min_{w,b} \mathcal{L} = \frac{1}{n} \sum_{i=1}^n \ell(y_i, f(x_i)) + \lambda \Omega(w)$$
$$\frac{1}{2} (f(x_i) - y_i)^2$$

$\|w\|_1$   
LASSO  
Tibshirani, 1996

$\frac{1}{2} \|w\|_2^2$   
Ridge Regression  
Hoerl & Kennard, 1970

$\rho \|w\|_1 + (1 - \rho) \frac{1}{2} \|w\|_2^2$   
Elastic Net  
Zou & Hastie, 2005

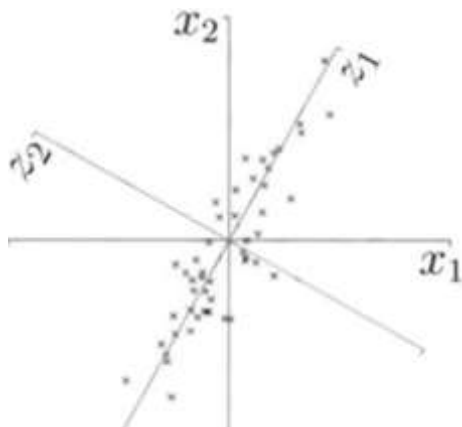
# 特征降维



特征选择完成后，可能由于特征矩阵过大，导致计算量大、训练时间长，因此**降低特征矩阵维度**也是必不可少的。

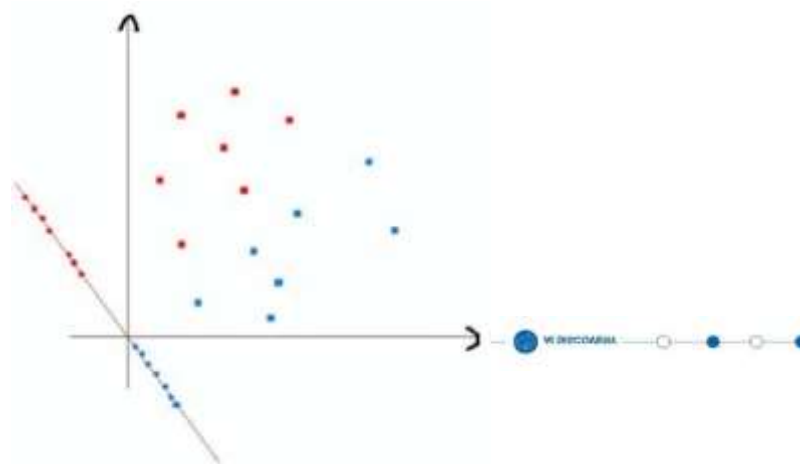
## ◆ 主成分分析(PCA)

将原始特征空间映射到彼此正交的特征向量空间，在非满秩的情况下使用SVD分解来构建特征向量。



## ◆ 线性判别分析(LDA)

给出一个标注了类别的数据集，投影到了一条直线之后，能够使得点尽量按类别区分开。



# 特征编码

数据集中经常会出现字符串信息，例如男女、高中低等，这类信息不能直接用于算法计算，需要将这些**数据转化为数值形式**进行编码，便于后期进行建模。

|   | Direction | District | Elevator | Floor | Garden   | Id           | Layout | Price  | Region | Renovation | Size  | Year |
|---|-----------|----------|----------|-------|----------|--------------|--------|--------|--------|------------|-------|------|
| 0 | 东西        | 灯市口      | NaN      | 6     | 锡拉胡同21号院 | 101102647043 | 3室1厅   | 780.0  | 东城     | 精装         | 75.0  | 1988 |
| 1 | 南北        | 东单       | 无电梯      | 6     | 东华门大街    | 101102650978 | 2室1厅   | 705.0  | 东城     | 精装         | 60.0  | 1988 |
| 2 | 南西        | 崇文门      | 有电梯      | 16    | 新世界中心    | 101102672743 | 3室1厅   | 1400.0 | 东城     | 其他         | 210.0 | 1996 |
| 3 | 南         | 崇文门      | NaN      | 7     | 兴隆都市馨园   | 101102577410 | 1室1厅   | 420.0  | 东城     | 精装         | 39.0  | 2004 |

## ◆ one-hot编码：

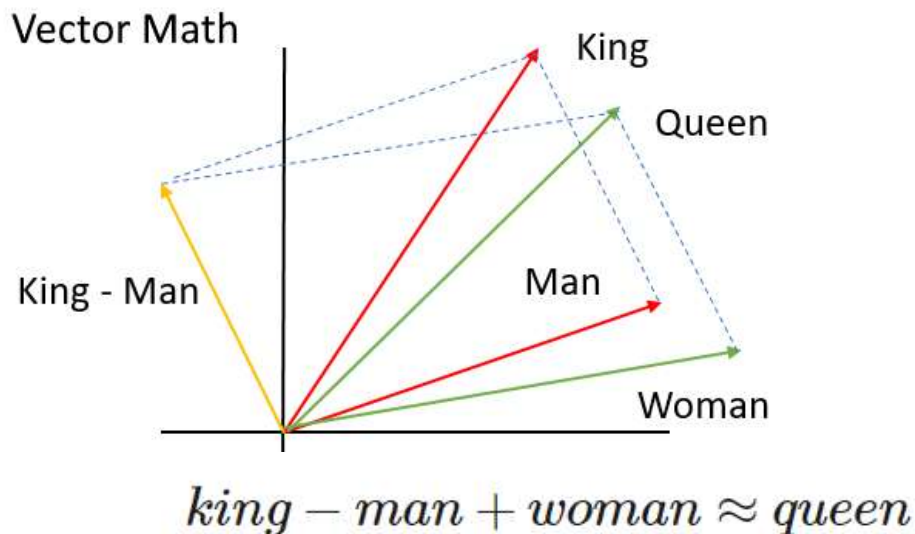
- ✓ 图中的Elevator和Renovation都是定类型 数据。除去缺失值，Elevator分类有电梯和 无电梯两种，因此可用01和10表示。
- ✓ Renovation分为有精装，简装，毛坯和其 它四种，可用0001/0010/0100/1000表示。

| "a" | "abbreviations" |     | "zoology" | "zoom" |
|-----|-----------------|-----|-----------|--------|
| 1   | 0               |     | 0         | 0      |
| 0   | 1               |     | 0         | 1      |
| 0   | 0               |     | 0         | 0      |
| .   | .               | ... | .         | .      |
| .   | .               |     | .         | .      |
| .   | .               |     | .         | .      |
| 0   | 0               |     | 0         | 0      |
| 0   | 0               |     | 1         | 0      |
| 0   | 0               |     | 0         | 1      |

# 特征编码

数据集中经常会出现字符串信息，例如男女、高中低等，这类信息不能直接用于算法计算，需要将这些**数据转化为数值形式**进行编码，便于后期进行建模。

|   | Direction | District | Elevator | Floor | Garden   | Id           | Layout | Price  | Region | Renovation | Size  | Year |
|---|-----------|----------|----------|-------|----------|--------------|--------|--------|--------|------------|-------|------|
| 0 | 东西        | 灯市口      | NaN      | 6     | 锡拉胡同21号院 | 101102647043 | 3室1厅   | 780.0  | 东城     | 精装         | 75.0  | 1988 |
| 1 | 南北        | 东单       | 无电梯      | 6     | 东华门大街    | 101102650978 | 2室1厅   | 705.0  | 东城     | 精装         | 60.0  | 1988 |
| 2 | 南西        | 崇文门      | 有电梯      | 16    | 新世界中心    | 101102672743 | 3室1厅   | 1400.0 | 东城     | 其他         | 210.0 | 1996 |
| 3 | 南         | 崇文门      | NaN      | 7     | 兴隆都市馨园   | 101102577410 | 1室1厅   | 420.0  | 东城     | 精装         | 39.0  | 2004 |



◆ **语义编码**：one-hot编码无法体现数据间的**语义关系**，对于一些有关联的文本信息来说，无法真正体现出数据关联。

- ✓ 对于这一类信息通常采用词嵌入（word embedding）的方式是比较好的选择。
- ✓ 目前在这一领域比较好的方法是基于google的word2vec方法。



# 规范化

不同属性具有不同量级时会导致：①数量级的差异将导致量级较大的属性占据主导地位；②数量级的差异将导致迭代收敛速度减慢；③依赖于样本距离的算法对于数据的数量级非常敏感。

## ◆ 标准化

通过减去均值然后除以方差（或标准差），将数据按比例缩放，使之落入一个小的特定区间。

$$x = (x - \mu) / \sigma$$

适用于：如果数据的分布本身就服从正态分布，就可以用这个方法。

## ◆ 区间缩放

将属性缩放到一个指定的最大和最小值（通常是1-0）之间。

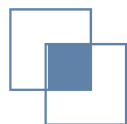
$$x = (x - \min) / (\max - \min)$$

## ◆ 归一化

将某一属性特征的模长转化成1。

$$x' = \frac{x}{\sqrt{\sum_j x[j]^2}}$$





# 机器学习准备



**问题：** 如何根据鸢尾花的花萼和花瓣大小将其分为三种不同的品种？



山鸢尾



变色鸢尾



维吉尼亚鸢尾

**特  
征**

| 花萼长度 | 花萼宽度 | 花瓣长度 | 花瓣宽度 |
|------|------|------|------|
| 5.1  | 3.3  | 1.7  | 0.5  |
| 5.0  | 2.3  | 3.3  | 1.0  |
| 6.4  | 2.8  | 5.6  | 2.2  |

model



**结  
果**

| 品种 (标签)    |
|------------|
| 0 (山鸢尾)    |
| 1 (变色鸢尾)   |
| 2 (维吉尼亚鸢尾) |





# 目录



1

## 机器学习概论

学科定义、发展历程、  
机器学习方法、应用场景与挑战

2

## 机器学习准备

鸢尾花分类任务简介  
数据预处理、特征工程

3

## 机器学习方法

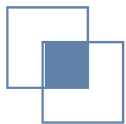
学习方法：分类、回归、聚类、其他  
机器学习模型评估

4

## 课程实践

实践：鸢尾花分类





# 机器学习方法



机器学习方法

分类问题

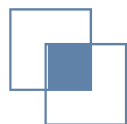
回归问题

聚类问题

其他问题

机器学习模型评估





# 机器学习方法分类



## 分类问题

决策树

贝叶斯

支持向量机

逻辑回归

集成学习

## 回归问题

线性回归

岭回归

Lasso回归

## 聚类问题

K-means

高斯混合聚类

密度聚类

层次聚类

谱聚类

## 其他问题

隐马尔可夫模型

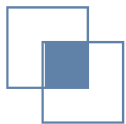
LDA主题模型

条件随机场

神经网络

深度学习





# 机器学习方法



机器学习方法

分类问题

回归问题

聚类问题

其他问题

机器学习模型评估

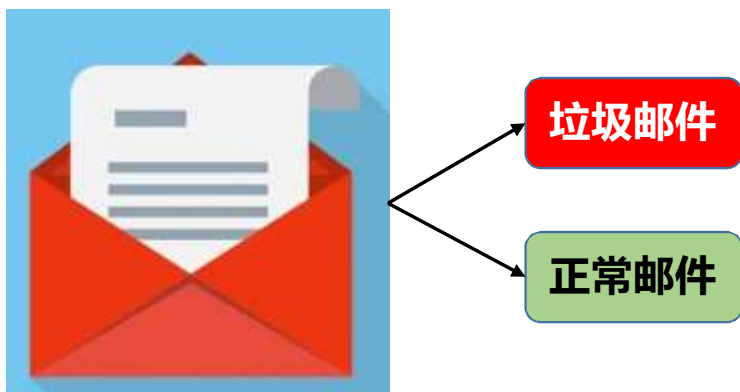


# 分类问题

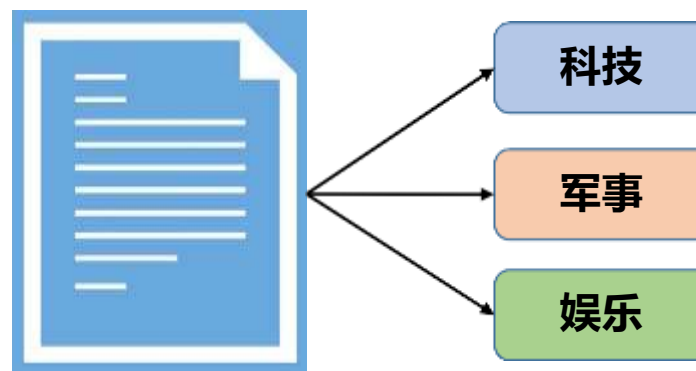
**分类问题**是**监督学习**的一个核心问题，它从数据中学习一个分类决策函数或分类模型(分类器 (classifier) )，对新的输入进行输出预测，输出变量取有限个离散值。

## □ 分类在我们日常生活中很常见

✓ 二分类问题



✓ 多分类问题



## □ 核心算法

✓ 决策树、贝叶斯、SVM、逻辑回归

# 决策树

**决策树 (decision tree)** 是一个树结构，每个**非叶节点**表示一个特征属性，每个**分支边**代表这个特征属性在某个值域上的输出，每个**叶节点**存放一个类别。

**决策过程：**从根节点开始，测试待分类项中相应的特征属性，并按照其**值**选择输出分支，直到到达叶子节点，将叶子节点存放的类别作为决策结果。

**示例：**假如我买了一个西瓜，它的特点是**纹理清晰、根蒂硬挺**，如何根据右侧决策树判断是好瓜还是坏瓜？

● 给定训练数据，**如何构建决策树**呢？

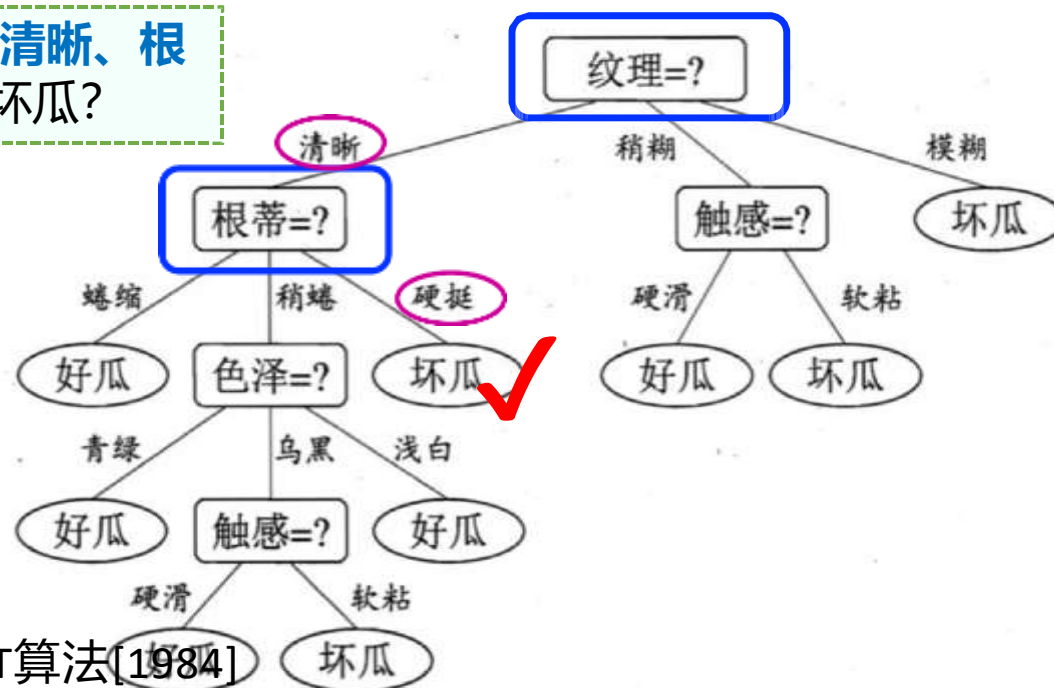
1. **特征选择：**选取对训练数据具有分类能力的特征。

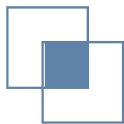
2. **决策树生成：**在决策树各个点上按照一定方法选择特征，递归构建决策树。

3. **决策树剪枝：**在已生成的树上减掉一些子树或者叶节点，从而简化分类树模型。

● 核心算法

ID3算法[1979, 1986]，C4.5算法[1993]及CART算法[1984]





# 决策树

输入: 训练集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ ; 属性集  $A = \{a_1, a_2, \dots, a_d\}$ .

过程: 函数  $TreeGenerate(D, A)$

1. 生成结点node;
2. if D中样本全属于同一类别C then
3. 将node标记为C类叶结点; **递归返回**;
4. end if
5. if A=空集OR D中样本在A上取值相同then
6. 将node标记为D中样本数(当前结点)最多的类(成为叶结点); **递归返回**
7. end if
8. 从A中选择**最优划分属性**  $a_*$ :
  - $a_* = \operatorname{argmax} Gain(D, a)$  [最大化信息增益, 偏好可取值数目较多的属性]
  - $a_* = \operatorname{argmax} Gain\_ratio(D, a)$  [最大化信息增益率, 偏好可取值数目较少的属性]
  - $a_* = \operatorname{argmin} Gini\_index(D, a)$  [最小化基尼指数]
9. for  $a_*$  的每个值  $a'_*$  do
10. 为node生成一个分支; 令  $D_{v'}$  表示D中在  $a_*$  上取值为  $a'_*$  的样本子集
11. if  $D_{v'}$  为空:
12. 将分支结点标记为D中样本数(父结点)最多的类(成为叶结点); **递归返回**;
13. else
14. 以  $TreeGenerate(D_{v'}, A \setminus \{a_*\})$  为分支结点
15. end if
16. end for

输出: 以node为根结点的一棵决策树

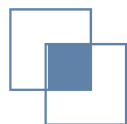
(1)当前节点包含的样本权属于同一类别, 无需划分

(2)当前属性及为空, 或是所有样本在所有属性上取值相同, 无法划分

(3) 当前结点包含的样本集合为空, 不能划分

## 决策树学习基本算法





# 决策树 特征选择



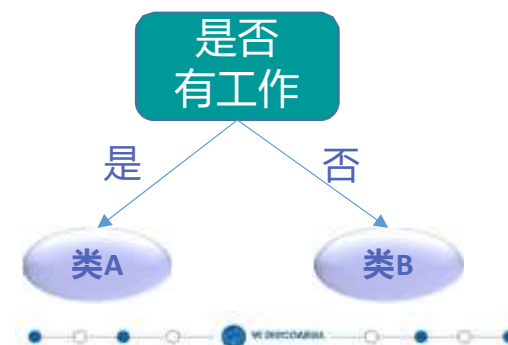
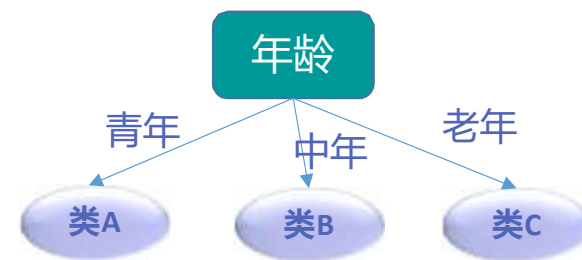
决策树构建过程中的特征选择是非常重要的一步。特征选择是决定用哪个特征来划分特征空间，**特征选择**是要选出对训练数据集具有分类能力的特征，这样可以提高决策树的学习效率。

**信息熵：**表示随机变量的不确定性，熵越大不确定性越大。

**信息增益：**信息增益 = 信息熵(前) - 信息熵(后)

**信息增益比：**信息增益比 = 惩罚参数 \* 信息增益。特征个数较多时，惩罚参数较小；特征个数较少时，惩罚参数较大。

**基尼指数：**表示集合的不确定性，基尼系数越大，表示不平等程度越高。



| 算法   | 支持模型  | 树结构 | 特征选择     |
|------|-------|-----|----------|
| ID3  | 分类    | 多叉树 | 信息增益     |
| C4.5 | 分类    | 多叉树 | 信息增益比    |
| CART | 分类、回归 | 二叉树 | 基尼系数，均方差 |



# 决策树生成



决策树生成：在决策树各个点上按照一定方法选择特征，递归构建决策树。

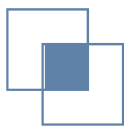
categorical  
categorical  
continuous  
class

| Tid | Refund | Marital Status | Taxable Income | Cheat |
|-----|--------|----------------|----------------|-------|
| 1   | Yes    | Single         | 125K           | No    |
| 2   | No     | Married        | 100K           | No    |
| 3   | No     | Single         | 70K            | No    |
| 4   | Yes    | Married        | 120K           | No    |
| 5   | No     | Divorced       | 95K            | Yes   |
| 6   | No     | Married        | 60K            | No    |
| 7   | Yes    | Divorced       | 220K           | No    |
| 8   | No     | Single         | 85K            | Yes   |
| 9   | No     | Married        | 75K            | No    |
| 10  | No     | Single         | 90K            | Yes   |

Training Data

## ID3决策树生成算法

1. 决定分类属性；
2. 对目前的数据表，建立一个节点N
3. 如果数据库中的数据都属于同一个类，N就是树叶，在树叶上标出所属的类
4. 如果数据表中没有其他属性可以考虑，则N也是树叶，按照少数服从多数的原则在树叶上标出所属类别
5. 否则，根据平均信息期望值GAIN值选出一个最佳属性作为节点N的测试属性
6. 节点属性选定后，对于该属性中的每个值
  - 从N生成一个分支，并将数据表中与该分支有关的数据收集形成分支节点的数据表，在表中删除节点属性那一栏如果分支数据表非空，则运用以上算法从该节点建立子树



# 决策树剪枝



在生成树的过程中，如果没有**剪枝 (pruning)** 操作，就会生成一个对训练集完全拟合的决策树，但这是对测试集非常不友好的，泛化能力不行。因此，需要减掉一些枝叶，使得模型泛化能力更强。

## 预剪枝

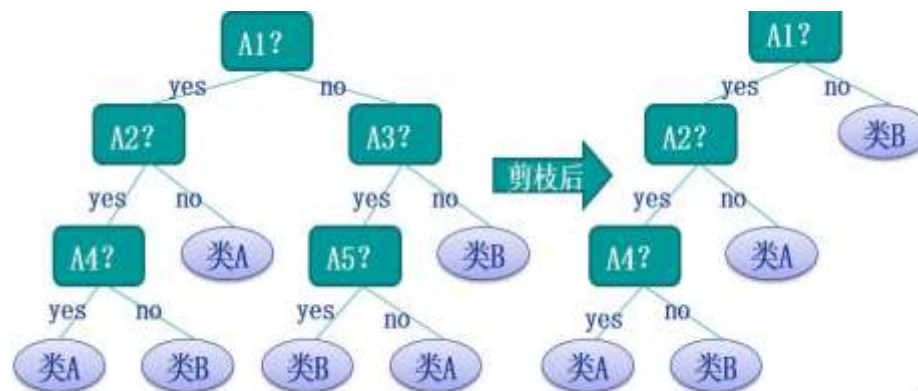
通过提前停止树的构建而对树剪枝，一旦停止，节点就是叶子，该叶子持有子集中最频繁的类。

- ✓ 定义一个高度，当决策树达到该高度时就停止生长
- ✓ 达到某个节点的实例具有相同的特征向量
- ✓ 定义一个阈值（实例个数、系统性能增益等）

## 后剪枝方法

首先构造完整的决策树，然后对那些置信度不够的结点子树用叶子结点来代替，该叶子的类标号用该结点子树中最频繁的类标记。相比于预剪枝，这种方法**更常用**，因为在预剪枝方法中精确地估计何时停止树增长很困难。

理想的决策树有三种：叶子节点数最少、叶子节点深度最小、叶子节点数最少且叶子节点深度最小。





输入: 训练集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ ;  
属性集  $A = \{a_1, a_2, \dots, a_d\}$ .

过程:

函数 **TreeGenerate**( $D, A$ )

```
1: 生成节点 node;  
2: if  $D$ 中样本全属于同一类别 $C$  then  
3:   将 node 标记为 $C$ 类叶结点; return  
4: end if  
5: if  $A = \emptyset$  OR  $D$ 中样本在 $A$ 上取值相同 then  
6:   将 node 标记为叶结点, 其类别标记为 $D$ 中样本数最多的类; return  
7: end if  
8: 从 $A$ 中选择最优化分属性 $a_*$ ;  
9: for  $a_*$ 的每一个值 $a_*^v$  do  
10:   为 node 生成一个分支; 令 $D_v$ 表示  
11:   if  $D_v$ 为空 then  
12:     将分支结点标记为叶结点  
13:   else  
14:     以 TreeGenerate( $D_v, A$ )  
15:   end if  
16: end for
```

输出: 以 node 为根结点的一颗决策树.

输入: 训练集  $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$ ;

验证集  $T = \{(x_1^T, y_1^T), (x_2^T, y_2^T), \dots, (x_m^T, y_m^T)\}$ ;

属性集  $A = \{a_1, a_2, \dots, a_d\}$ .

过程:

函数 **TreeGenerate**( $D, T, A$ )

```
1: 生成结点 node;  
2: if  $D$ 中样本全属于同一类别 $C$  then  
3:   将 node 标记为 $C$ 类叶结点; return  
4: end if  
5: if  $A = \emptyset$  OR  $D$ 中样本在 $A$ 上取值相同 then  
6:   将 node 标记为叶结点, 其类别标记为 $D$ 中样本数最多的类; return  
7: end if  
8: 从 $A$ 中选择最优化分属性 $a_*$ ;  
9: 计算 node 结点的倾向类别 $C_{\text{node}}$ ——为 $D$ 中样本数最多的类;  
10: 计算 $T$ 中被分类为 $C_{\text{node}}$ 的数量 $|T_{\text{node}}|$ ; 初始化计数 $|T_{a_*}| = 0$ ;  
11: for  $a_*$ 的每一个值 $a_*^v$  do  
12:   令 $D_v$ 表示 $D$ 中在 $a_*$ 上取值为 $a_*^v$ 的样本子集; 令 $T_v$ 表示 $T$ 中在 $a_*$ 上取值为 $a_*^v$ 的样本子集;  
13:   if  $D_v$ 为空 then  
14:     计算 $a_*^v$ 上的倾向类别 $C_v = C_{\text{node}}$ ;  
15:   else  
16:     计算 $a_*^v$ 上的倾向类别 $C_v$ ——为 $D_v$ 中样本数最多的类;  
17:   end if  
18:   计算 $T_v$ 中被分类为 $C_v$ 的数量 $|T_{C_v}|$ ;  $|T_{a_*}| += |T_{C_v}|$ ;  
19: end for  
20: if  $|T_{\text{node}}| \geq |T_{a_*}|$  then  
21:   将分支结点标记为叶结点, 其类别标记为 $D$ 中样本最多的类; return  
22: end if  
23: for  $a_*$ 的每一个值 $a_*^v$  do  
24:   为 node 生成一个分支; 令 $D_v$ 表示 $D$ 中在 $a_*$ 上取值为 $a_*^v$ 的样本子集;  
25:   if  $D_v$ 为空 then  
26:     将分支结点标记为叶结点, 其类别标记为 $D$ 中样本最多的类;  
27:   else  
28:     以 TreeGenerate( $D_v, T_v, A \setminus \{a_*\}$ ) 为分支节点  
29:   end if  
30: end for
```

输出: 以 node 为根结点的一颗决策树.



输入: 训练集  $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$   
属性集  $A = \{a_1, a_2, \dots, a_m\}$

过程:

```
函数 TreeGeneralization(D, A)
1: 生成节点 node
2: if D中样本数 < 阈值 then
3:   将 node 作为根节点返回
4: end if
5: if A = ∅ then
6:   将 node 作为根节点返回
7: end if
8: 从A中选择属性a
9: for a*的每个取值a_v do
10:   为 node 生成子节点 node_v
11:   if D_v 不为空 then
12:     node_v = TreeGeneralization(D_v, A)
13:   else
14:     node_v = 叶节点
15:   end if
16: end for
```

输出: 以 node 为根结点的决策树

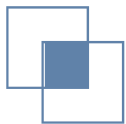
输入: 决策树  $T_0$ ;

验证集  $Test = \{(x_1^T, y_1^T), (x_2^T, y_2^T), \dots, (x_n^T, y_n^T)\}$ ;

过程:

```
1: 令  $T = T_0, k = 0$ ;
2: start 函数 TreePruning( $T, k$ )
3:   初始化  $\alpha = +\infty$ ;
4:   if  $T$  是根结点 then
5:     return  $T$ 
6:   end if
7:   for  $T$  中的每一个内部结点  $t$  do
8:     计算  $g(t) = \frac{C(t) - C(T_t)}{|T_t| - 1}$ ;
9:      $\alpha = \min(\alpha, g(t))$ ;
10:  end for
11:  for  $T$  中的每一个内部结点  $t$  (从上而下遍历) do
12:    if  $g(t) == \alpha$  then
13:      对结点  $t$  进行剪枝, 并对结点  $t$  以多数表决法决定其类, 得到树  $T$ ; break
14:    end if
15:  end for
16:   $k = k + 1, \alpha_k = \alpha, T_k = T$ ;
17:   $\{T_{list}; \alpha_{list}\} = \text{TreePruning}(T, k)$ ;
18:   $T_{list} = (T_k, T_{list}), \alpha_{list} = (\alpha_k, \alpha_{list})$ ;
19:  return  $\{T_{list}; \alpha_{list}\}$ 
20: end 函数
21:  $T_{list} = (T_0, T_{list}), \alpha_{list} = (0, \alpha_{list})$ ;
22: 利用验证集  $Test$ , 采用交叉验证法在子树序列  $T_{list} = (T_0, T_1, \dots, T_n)$  中选取最优子树  $T_\alpha$ .
```

输出:  $T_\alpha$ .



# 随机森林



随机森林(Random Forest, RF), 以决策树为基学习器构建Bagging集成的基础上, 进一步在决策树的训练过程中引入随机属性选择。







# 贝叶斯分类



贝叶斯分类是基于贝叶斯定理和属性特征条件独立性的分类方法。

贝叶斯流派的核心：Probability theory is nothing but common sense reduced to calculation.

概率论只不过是把常识用数学公式表达了出来。——拉普拉斯

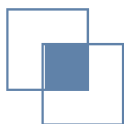
案例：假设春季感冒流行，你同桌打了一个喷嚏，那你同桌感冒了的概率是多少？

**1.计算先验概率：**你同桌没有任何症状的情况下可能得感冒的概率是多少？

**2.为每个属性计算条件概率：**如果你同桌感冒了，那他打喷嚏的概率是多少，如果他没感冒，出现打喷嚏症状的概率有多少？

**3.计算后验概率：**根据1和2求解最终问题，这才是拥有贝叶斯思想的你该做的分析。



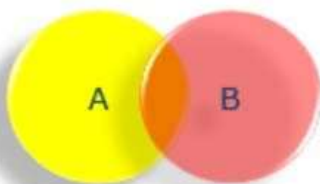


# 贝叶斯理论

## 贝叶斯公式:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

$$P(A \cap B) = P(A|B)P(B) = P(B|A)P(A)$$



Likelihood of evidence B if A is true

Prior probability

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Posterior probability of A given the evidence B

Prior probability that evidence B is true

## 推广到多个类别:

$$P(\omega_i | x) = \frac{P(x | \omega_i)P(\omega_i)}{P(x)}$$

$$P(\text{类别} | \text{特征}) = \frac{P(\text{特征} | \text{类别})P(\text{类别})}{P(\text{特征})}$$

## 朴素贝叶斯:

$$\omega_{MAP} = \arg \max_{\omega_i \in \omega} P(\omega_i | a_1, a_2, \dots, a_n)$$

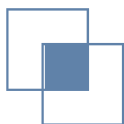
$$\omega_{MAP} = \arg \max_{\omega_i \in \omega} \frac{P(a_1, a_2, \dots, a_n | \omega_i)P(\omega_i)}{P(a_1, a_2, \dots, a_n)}$$

$$\omega_{MAP} = \arg \max_{\omega_i \in \omega} P(a_1, a_2, \dots, a_n | \omega_i)P(\omega_i)$$

$$\omega_{MAP} = \arg \max_{\omega_i \in \omega} P(\omega_i) \prod_j P(a_j | \omega_i)$$

条件独立假设





# 贝叶斯分类

举个栗子：一对男女朋友，男生向女生求婚，男生的四个特点分别是不帅，性格不好，身高矮，不上进，请你判断一下女生是嫁还是不嫁？

## ① 估计先验概率 $P(c)$

$$p(\text{嫁}) = 6/12 \text{ (总样本数)} = 1/2$$

## ② 为每个属性计算条件概率 $P(x_i|c)$

$$p(\text{不帅}|\text{嫁}) = 3/6 = 1/2$$

$$p(\text{性格不好}|\text{嫁}) = 1/6$$

$$p(\text{不帅}) = 4/12 = 1/3$$

## ③ 计算后验概率

$$\begin{aligned} p(\text{嫁}|\text{不帅, 性格不好, 身高矮, 不上进}) &= \frac{p(\text{不帅, 性格不好, 身高矮, 不上进}|\text{嫁}) * p(\text{嫁})}{p(\text{不帅, 性格不好, 身高矮, 不上进})} \\ &= \frac{p(\text{不帅}|\text{嫁}) * p(\text{性格不好}|\text{嫁}) * p(\text{身高矮}|\text{嫁}) * p(\text{不上进}|\text{嫁}) * p(\text{嫁})}{p(\text{不帅}) * p(\text{性格不好}) * p(\text{身高矮}) * p(\text{不上进})} \end{aligned}$$

$$\text{不嫁}(1/6 * 1/2 * 1 * 1/2) > \text{嫁}(1/2 * 1/6 * 1/6 * 1/6 * 1/2)$$

| 帅? | 性格好? | 身高? | 上进? | 嫁与否 |
|----|------|-----|-----|-----|
| 帅  | 不好   | 矮   | 不上进 | 不嫁  |
| 不帅 | 好    | 矮   | 上进  | 不嫁  |
| 帅  | 好    | 矮   | 上进  | 嫁   |
| 不帅 | 好    | 高   | 上进  | 嫁   |
| 帅  | 不好   | 矮   | 上进  | 不嫁  |
| 不帅 | 不好   | 矮   | 不上进 | 不嫁  |
| 帅  | 好    | 高   | 不上进 | 嫁   |
| 不帅 | 好    | 高   | 上进  | 嫁   |
| 帅  | 好    | 高   | 上进  | 嫁   |
| 不帅 | 不好   | 高   | 上进  | 嫁   |
| 帅  | 好    | 矮   | 不上进 | 不嫁  |
| 帅  | 好    | 矮   | 不上进 | 不嫁  |

分析结果：不嫁！



# 贝叶斯分类



$$\frac{p(\text{不帅}|\text{不嫁}) * p(\text{性格不好}|\text{不嫁}) * p(\text{身高矮}|\text{不嫁}) * p(\text{不上进}|\text{不嫁}) * p(\text{不嫁})}{p(\text{不帅}) * p(\text{性格不好}) * p(\text{身高矮}) * p(\text{不上进})}$$

## 拉普拉斯修正

先验概率拉普拉斯修正  $P(c) = \frac{Dc}{D} \rightarrow P(c) = \frac{Dc+1}{D+N}$

条件概率拉普拉斯修正  $P(x_i | c) = \frac{D_{c,xi}}{Dc} \rightarrow P(x_i | c) = \frac{D_{c,xi}+1}{Dc+Ni}$

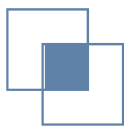
## 优点:

- (1) 算法逻辑简单,易于实现
- (2) 分类过程中时空开销小

## 缺点:

理论上, 朴素贝叶斯模型与其他分类方法相比具有最小的误差率。但是实际上并非总是如此, 这是因为朴素贝叶斯模型假设属性之间相互独立, 这个假设在实际应用中往往是不成立的, 在属性个数比较多或者属性之间相关性较大时, 分类效果不好。

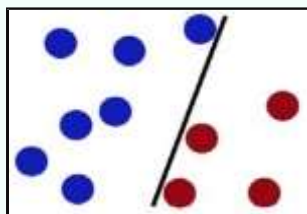




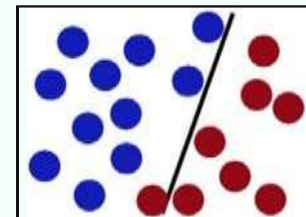
# SVM (支持向量机)

在很久以前的情人节，大侠要去救他的爱人，但魔鬼和他玩了一个游戏。

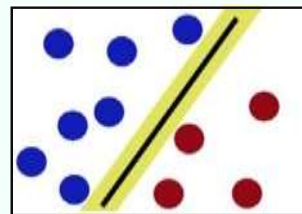
1. 魔鬼在桌子上似乎有规律放了两种颜色的球，说：“你用一根棍分开它们？要求：尽量在放更多球之后，仍然适用。”于是大侠这样放，干的似乎还不错？



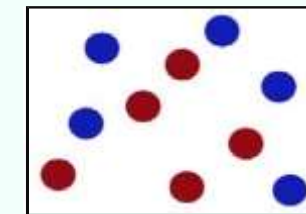
2. 然后魔鬼，又在桌上放了更多的球，似乎有一个球站错了阵营。



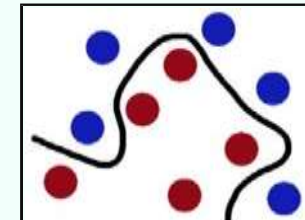
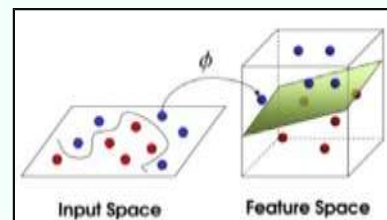
3. SVM就是试图把棍放在最佳位置，好让在棍的两边有尽可能大的间隙。



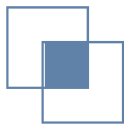
4. 魔鬼看到大侠已经学会了一个trick，于是魔鬼给了大侠一个新的挑战。



5. 现在，大侠没有棍可以很好帮他分开两种球了，现在怎么办呢？当然像所有武侠片中一样大侠桌子一拍，球飞到空中。然后，凭借大侠的轻功，抓起一张纸，插到了两种球的中间。



再之后，人们把这些球叫做「data」，把棍子叫做「classifier」，最大间隙trick叫做「optimization」，拍桌子叫做「kernelling」，那张纸叫做「hyperplane」。



# SVM (支持向量机)



回忆解析几何，点到直线的距离

$$(x, y) \text{ 到 } Ax + By + C = 0 \text{ 的距离 } \frac{|Ax + By + C|}{\sqrt{A^2 + B^2}}$$

拓展到n维空间  $\theta^T x_b = 0 \rightarrow w^T x + b = 0$

$$\frac{|w^T x + b|}{\|w\|} \quad \|w\| = \sqrt{w_1^2 + w_2^2 + \dots + w_n^2}$$

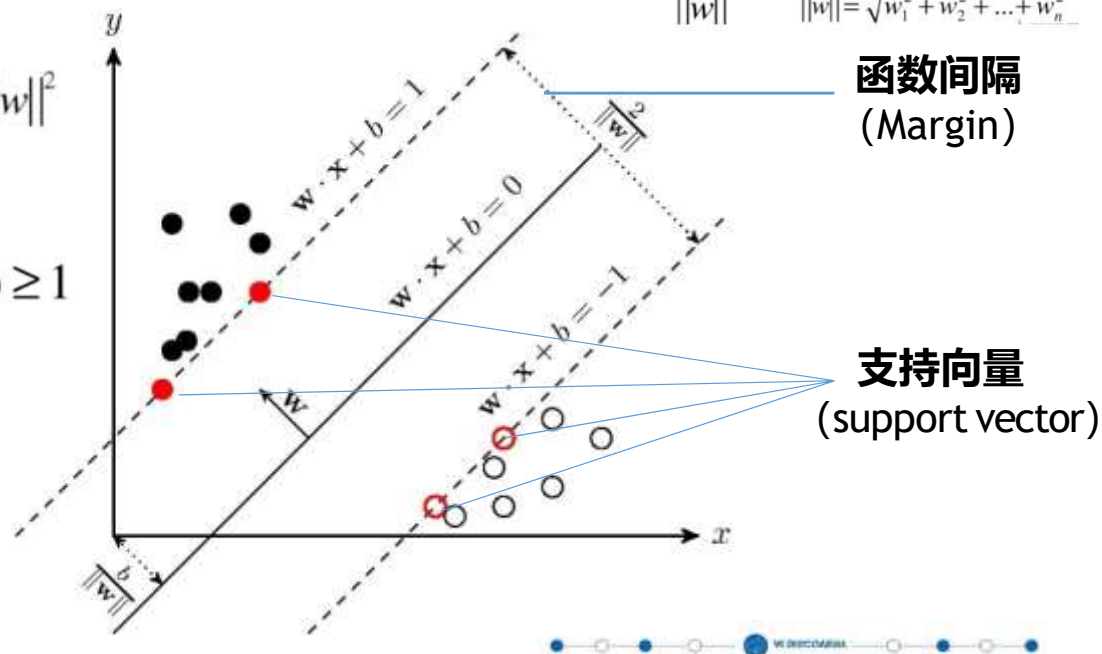
**支持向量机 (Support Vector Machine)** 是一种有监督学习方法，它尝试寻找一个最优决策边界，使距离两个类别最近的样本最远。

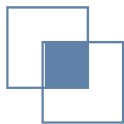
**硬间隔支持向量机：**

$$\begin{aligned} & \max \frac{1}{\|w\|} \\ \text{s.t. } & \begin{cases} \frac{w^T x^{(i)} + b}{\|w\|} \geq \frac{1}{\|w\|} & \forall y^{(i)} = 1 \\ \frac{w^T x^{(i)} + b}{\|w\|} \leq -\frac{1}{\|w\|} & \forall y^{(i)} = -1 \end{cases} \end{aligned} \quad \text{等价} \quad \begin{aligned} & \min \|w\| \quad \min \frac{1}{2} \|w\|^2 \\ \text{s.t. } & y^{(i)}(w^T x^{(i)} + b) \geq 1 \end{aligned}$$

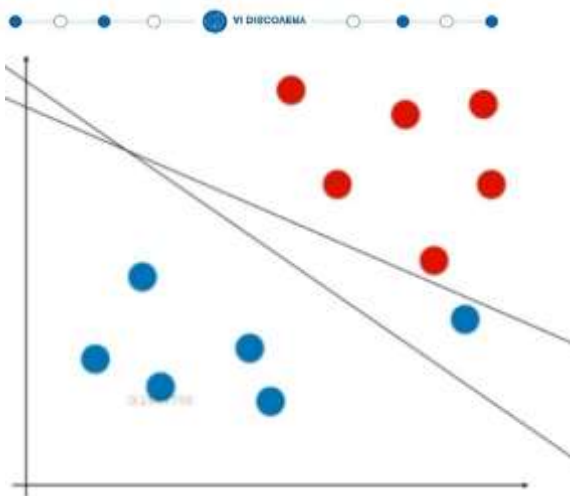
硬间隔（线性可分）支持向量机最优化问题：

$$\begin{aligned} & \min \frac{1}{2} \|w\|^2 \\ \text{s.t. } & y^{(i)}(w^T x^{(i)} + b) \geq 1 \end{aligned}$$



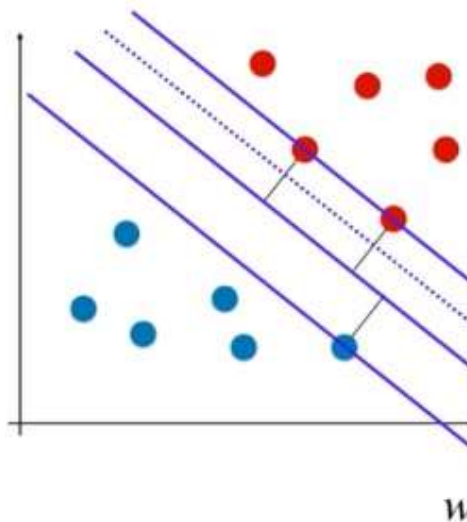


# SVM (支持向量机)



硬间隔支持向量机:

$$\min \frac{1}{2} \|w\|^2$$
$$s.t. \quad y^{(i)}(w^T x^{(i)} + b) \geq 1$$



软间隔支持向量机:

$$\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \zeta_i$$

$$s.t. \quad y^{(i)}(w^T x^{(i)} + b) \geq 1 - \zeta_i$$
$$\zeta_i \geq 0$$

L1正则

$$\min \frac{1}{2} \|w\|^2 + C \sum_{i=1}^m \zeta_i^2$$

$$s.t. \quad y^{(i)}(w^T x^{(i)} + b) \geq 1 - \zeta_i$$
$$\zeta_i \geq 0$$

L2正则

$$\min \frac{1}{2} \|w\|^2 + \sum_{i=1}^m \zeta_i$$

$$s.t. \quad y^{(i)}(w^T x^{(i)} + b) \geq 1 - \zeta_i$$

$$\zeta_i \geq 0$$

$$w^T x + b = 1$$
$$w^T x + b = 0$$
$$w^T x + b = -1$$
$$w^T x + b = 1 - \zeta$$



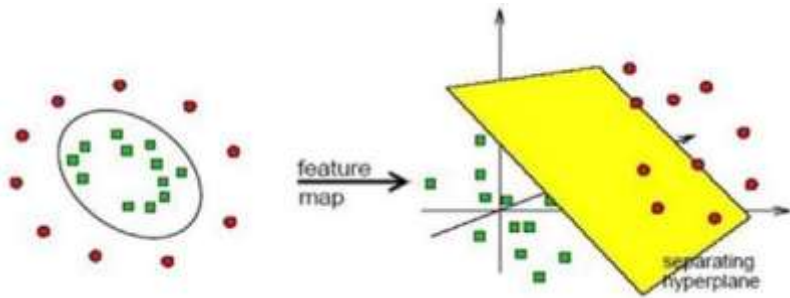


# SVM (支持向量机)



**支持向量机** (Support Vector Machine ,SVM) 是一种有监督学习方法, 它尝试寻找一个最优决策边界, 使距离两个类别最近的样本最远, 从而对分类问题提供良好的泛化能力。

## 非线性支持向量机与核函数:



- 线性核函数:  $K(x, z) = x \cdot z$
- 多项式核函数:  $K(x, z) = (x \cdot z + 1)^p$
- 高斯核函数:  $K(x, z) = \exp(-\frac{\|x - z\|^2}{2\sigma^2})$
- 混合核:  $\lambda K_1(x, z) + (1 - \lambda) K_2(x, z), 0 \leq \lambda < 1$

## SVM的优点:

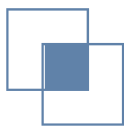
- ✓ 相对于其他训练分类算法**不需要过多样本**, 并且由于SVM引入了核函数, 所以SVM可以处理高维样本
- ✓ **结构风险最小**。这种风险是指分类器对问题真实模型的逼近与问题真实解之间的累积误差
- ✓ **非线性**, 是指SVM擅长应付样本数据线性不可分的情况, 主要通过松弛变量 (也叫惩罚变量) 和核函数技术来实现, 这一部分也正是SVM的精髓所在。

### 常用软件工具包:

LibSVM: <http://www.csie.ntu.edu.tw/~cjlin/libsvm/>

SVM-Light: <http://svmlight.joachims.org/>

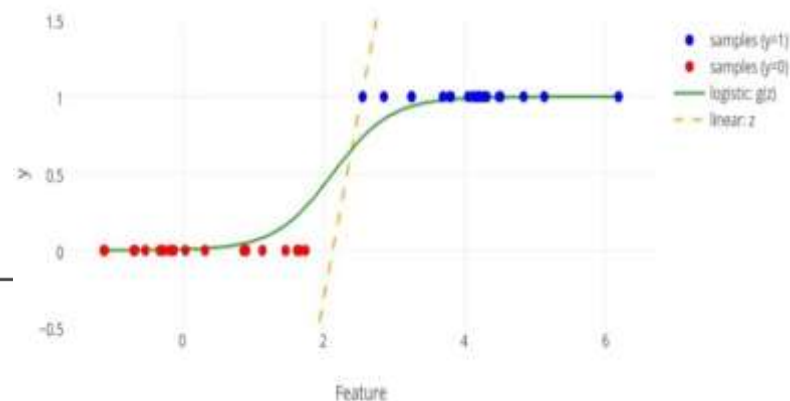
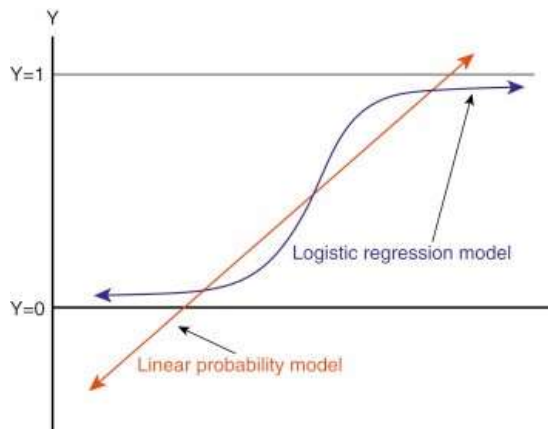
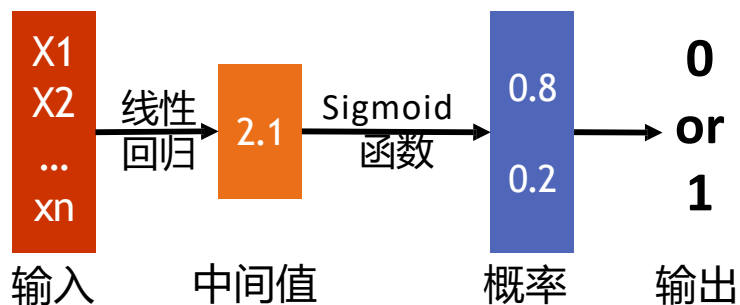
Liblinear: <http://www.csie.ntu.edu.tw/~cjlin/liblinear/>



# 逻辑回归



**logistic回归**是一个分类算法，它可以处理二元分类以及多元分类。首先逻辑回归构造广义的线性回归函数，然后使用sigmoid函数将回归值映射到离散类别。



## 二项逻辑回归模型:

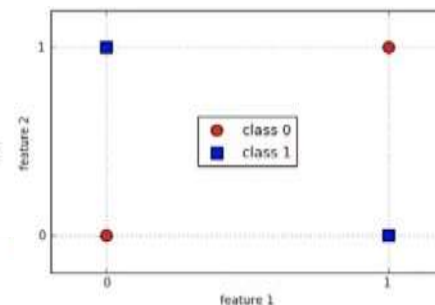
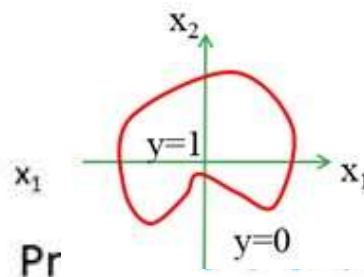
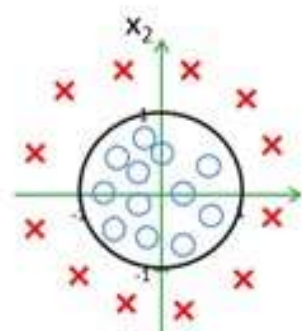
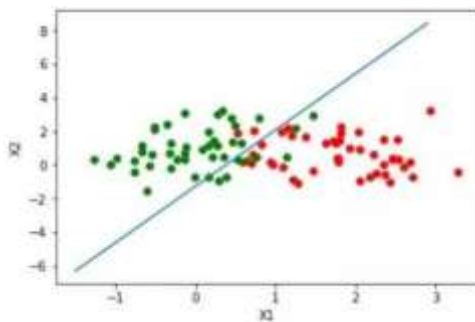
$$P(Y=1|x) = \frac{\exp(w \bullet x)}{1 + \exp(w \bullet x)}$$

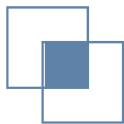
$$P(Y=0|x) = \frac{1}{1 + \exp(w \bullet x)}$$

其中:

$$w = (w^{(1)}, w^{(2)}, \dots, w^{(n)}, b)^T$$

$$x = (x^{(1)}, x^{(2)}, \dots, x^{(n)}, 1)$$





# 逻辑回归



## 二项逻辑回归模型（二分类）：

$$P(Y = 1 | x) = \frac{\exp(w \bullet x)}{1 + \exp(w \bullet x)}$$

$$P(Y = 0 | x) = \frac{1}{1 + \exp(w \bullet x)}$$

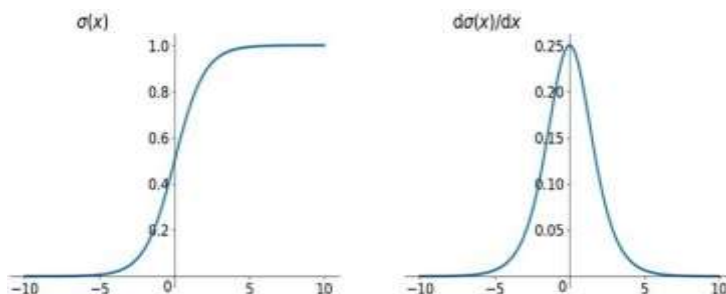
其中：

$$w = (w^{(1)}, w^{(2)}, \dots, w^{(n)}, b)^T$$

$$x = (x^{(1)}, x^{(2)}, \dots, x^{(n)}, 1)$$

一个事件的几率（odds）是指该事件发生的概率与该事件不发生的概率的比值。该事件的对数几率（log odds）或logit函数

$$\log \frac{P(Y = 1 | x)}{1 - P(Y = 1 | x)} = w \bullet x$$



$$P(Y = 1 | x) = \frac{\exp(w \bullet x)}{1 + \exp(w \bullet x)}$$

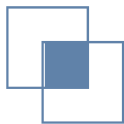
## 多项逻辑回归模型（多分类）：

$$P(Y = k | x) = \frac{\exp(w_k \bullet x)}{1 + \sum_{k=1}^{K-1} \exp(w_k \bullet x)}, k = 1, 2, \dots, K-1$$

$$P(Y = K | x) = \frac{1}{1 + \sum_{k=1}^{K-1} \exp(w_k \bullet x)}$$







# 最大熵模型

熵是随机变量不确定性的度量，不确定性越大，熵值就越大。

德国物理学家鲁道夫·克劳修斯首次提出熵的概念，用来表示任何一种能量在空间中分布的均匀程度，能量分布得越均匀，熵就越大。

举个栗子：你每次把耳机整理好，放入口袋中，下次再拿出来已经乱了。让耳机线乱掉的看不见的“力”就是熵力，耳机线喜欢变成更混乱。

最大熵原理指出，对一个随机事件的概率分布进行预测时，我们的预测应当满足全部已知的条件，而对未知的情况**不要做任何主观假设**。

最大熵原理认为，学习概率模型时，在所有可能得概率模型（分布）中，熵最大的模型是最好的模型

举个栗子：假设随机变量 $X$ 有5个取值  $\{A, B, C, D, E\}$ ，要估计各个值的概率 $P(A), P(B), P(C), P(D), P(E)$ 。

$$P(A) + P(B) + P(C) + P(D) + P(E) = 1 \quad (\text{约束})$$

根据最大熵的前提条件，我们可以假定：

$$P(A) = P(B) = P(C) = P(D) = P(E) \quad (\text{等概率})$$

所以，可以计算出来 $P(A) = P(B) = P(C) = P(D) = P(E) = 1/5$

其中， $P(A) + P(B) = 3/10$

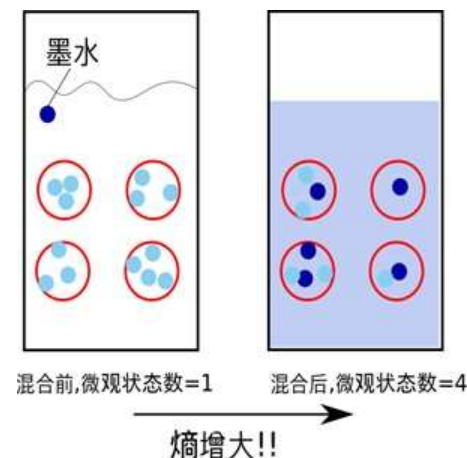
$$P(A) + P(B) + P(C) + P(D) + P(E) = 1 \quad (\text{约束})$$

$$P(A) + P(B) = 3/10 \quad (\text{约束})$$

从约束中根据等概率，

我们可以推测， $P(A) = P(B) = 3/20$ ,

$$P(C) = P(D) = P(E) = 7/30$$





# 逻辑回归与最大熵模型

## 最大熵模型:

$$\begin{aligned} \max_{P \in \mathcal{C}} H(P) &= -\sum_{s,y} \tilde{e}_s e(y|x) \log(e(y|x)) \\ \text{s.t. } E_P(f_i) &= E_P(f_i), (i=1,2,\dots,n) \\ \sum_y e(y|x) &= 1 \end{aligned}$$

(条件熵)

(约束)

其中:  $f_i(x,y) = \begin{cases} 1, & x \text{ 与 } y \text{ 满足某一事实} \\ 0, & \text{否则} \end{cases}$

## 最大熵一般形式:

$$P_w(y|x) = \frac{1}{Z_w(x)} \exp\left(\sum_{i=1}^n w_i f_i(x,y)\right)$$

$$\text{其中: } Z_w = \sum_y \exp\left(\sum_{i=1}^n w_i f_i(x,y)\right)$$

$$\text{逻辑回归: } P(Y=1|x) = \frac{\exp(w \cdot x)}{1 + \exp(w \cdot x)}$$

逻辑回归跟最大熵模型没有本质差别。(对数线性模型)

逻辑回归是最大熵相应类别为二类时的特殊情况

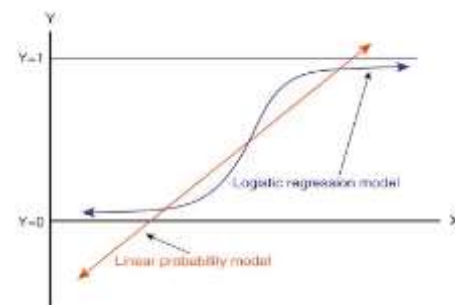
指数簇分布的最大熵等价于其指数形式的最大似然。

二项式分布的最大熵解等价于二项式指数形式(sigmoid)的最大似然;

多项式分布的最大熵等价于多项式分布指数形式(softmax)的最大似然。

## 为什么要用sigmoid函数?

$$g(z) = \frac{1}{1 + e^{-z}}$$

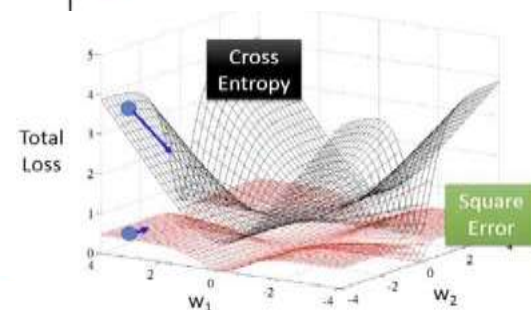


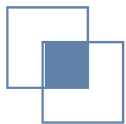
## 为什么要用对数似然损失函数?

$$P(X=n) = \begin{cases} 1-p & n=0 \\ p & n=1 \end{cases}$$

$$L(\theta) = \prod_{i=1}^m P(y=1|x_i)^{y_i} P(y=0|x_i)^{1-y_i}$$

$$\ln L(\theta) = \sum_{i=1}^m [y_i \ln P(y=1|x_i) + (1-y_i) \ln P(y=0|x_i)]$$

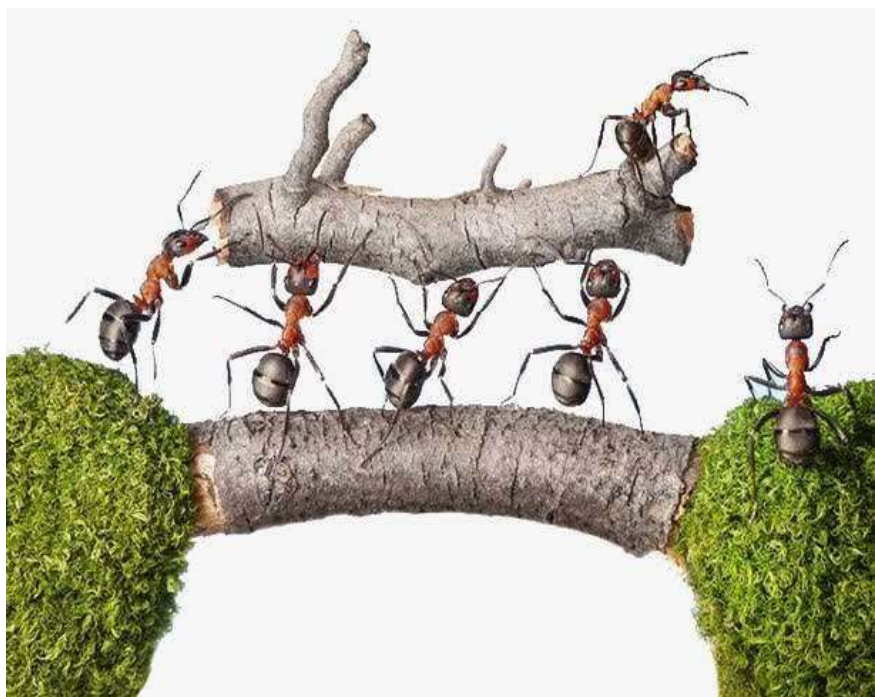




# 集成学习



**集成学习**通过将多个弱分类器集成在一起，使它们共同完成学习任务，构建一个强分类器。潜在哲学思想是“**三个臭皮匠赛过诸葛亮**”。



## ◆ 理论基础

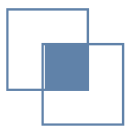
**强可学习**：在PAC学习框架中，一个概念，如果存在存在一个多项式的学习算法能够学习它，并且正确率很高，那么久称这个概念是强可学习的。

**弱可学习**：如果存在一个多项式的学习算法能够学习它，学习的正确率金币随机猜测略好，那么就称这个概念是弱可学习的。

Schapire证明强可学习与弱可学习是**等价的**，也就是说，在PAC学习框架下，一个概念强可学习的充分必要条件是这个概念若可学习的。

## ◆ 两类集成方法 Bagging(bootstrap aggregating) Boosting (提升方法)

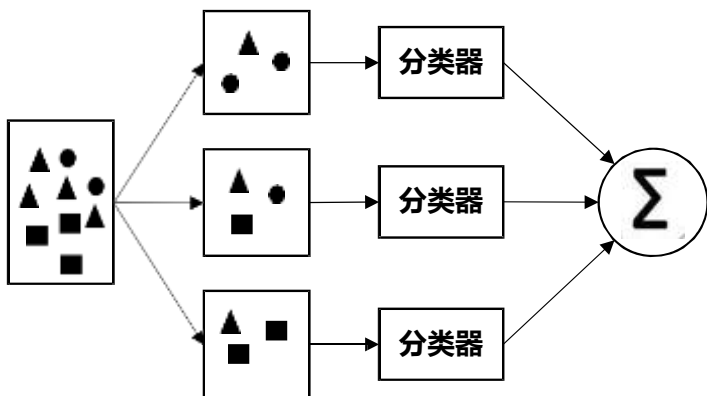




# 集成学习



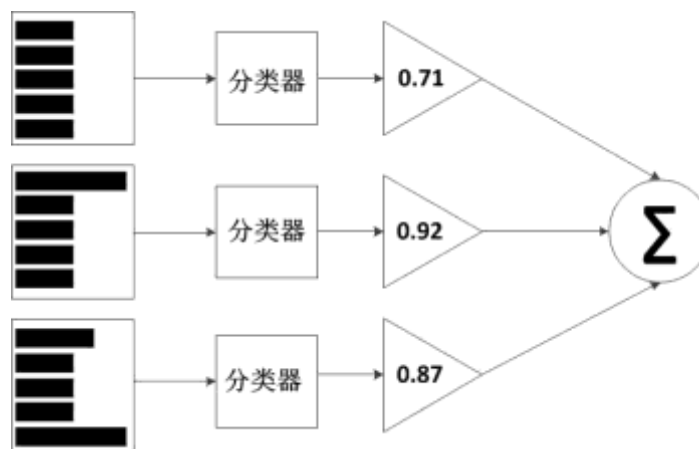
## Bagging(bootstrap aggregating)



Bagging: 基于数据**随机重抽样**的分类器构建方法

- 利用bootstrap方法从整体数据集中采取有放回抽样得到N个数据集 (如何采样?)
- 在每个数据集上学习出一个模型 (选择什么样的弱分类器?)
- 利用N个模型的输出投票得到最后的预测结果

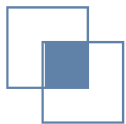
## Boosting



Boosting (Adaptive Boosting的简称), 基于错误提升 分类器性能, 通过**集中关注被已有分类器分类错误的样本**, 构建新分类器

- 初始的分布应为等概分布。
- 每次循环后提高错误样本的分布概率, 分错的样本在训练集中所占权重增大, 使得下一次循环的基分类器能够集中力量对这些错误样本进行判断。
- 计算分类器的权重, 识别率越高的基分类器权重越高, 识别率越低的基分类器权重越低。





# 集成学习



严格意义上来说，这不算是一种机器学习算法，而更像是一种优化手段或者策略，它通常是结合多个简单的弱机器学习算法，去做更可靠的决策。类似于**开会做决策**。

## □ Bagging与Boosting

都采用采样-学习-组合的方式，不同在于：

- ✓ Bagging中每个训练集互不相关，也就是每个基分类器互不相关，而Boosting中训练集要在上一轮的结果上进行调整，也使得其不能并行计算
- ✓ Bagging中预测函数是均匀平等的，但在Boosting中预测函数是加权的

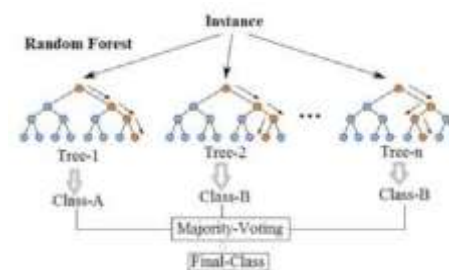
## 优点：

当先最先进的预测几乎都使用了算法集成。它比使用单个模型预测出来的结果要精确的多，在各大竞赛中得到了普遍应用。

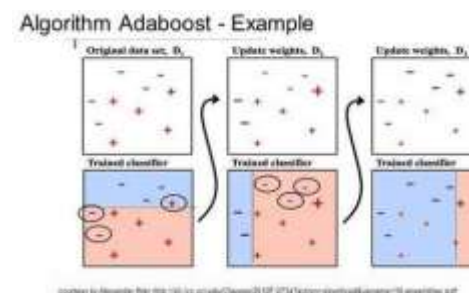
## 缺点：

需要大量的维护工作

## 代表算法：

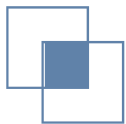


随机森林



Adboost





# 机器学习方法



机器学习方法

分类问题

回归问题

聚类问题

其他问题

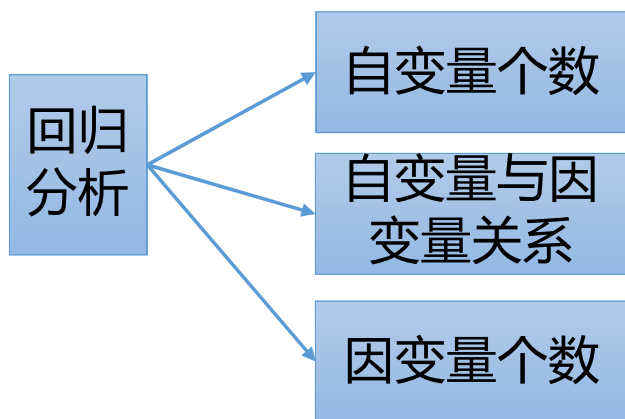
机器学习模型评估



# 回归问题

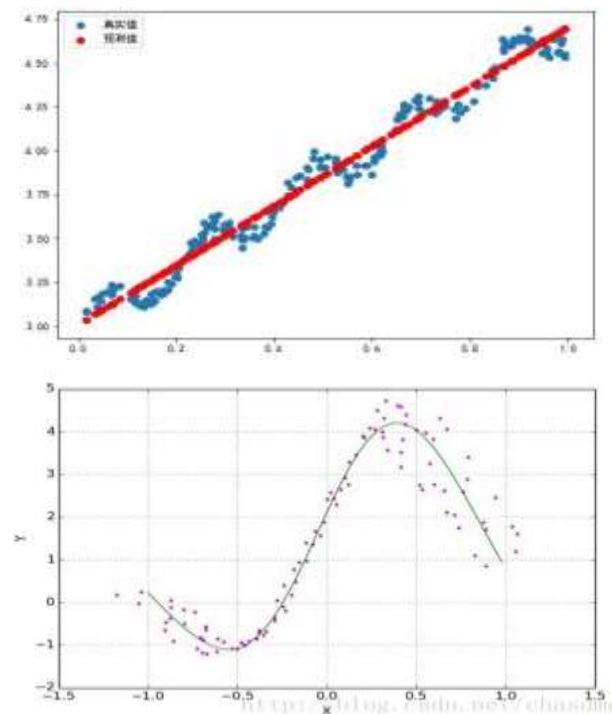


**回归分析**用于预测输入变量（**自变量**）和输出变量（**因变量**）之间的关系，特别是当输入变量的值发生变化时，输出变量值随之发生变化。

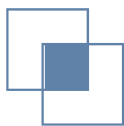


一元回归分析  
多元回归分析  
线性回归分析  
非线性回归分析  
简单回归分析  
多重回归分析

**为什么叫回归？**：达尔文表兄弟Francis Galton发明的。







# 线性回归

**线性回归算法**假设特征和结果满足线性关系。这就意味着可以将输入项分别乘以一些常量，再将结果加起来得到输出。

## 线性回归算法流程

- ① 选择拟合函数形式

$$h_{\theta}(x) = \sum_{i=0}^n \theta_i x_i = \theta^T x$$

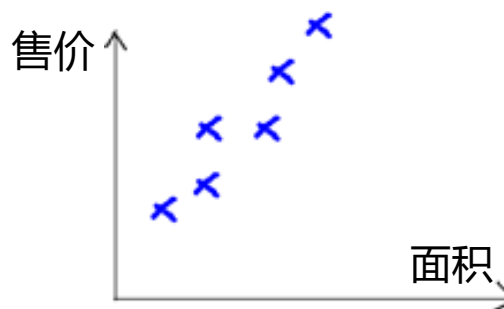
- ② 确定损失函数形式

$$\min_{\theta} J(\theta) = \frac{1}{2} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

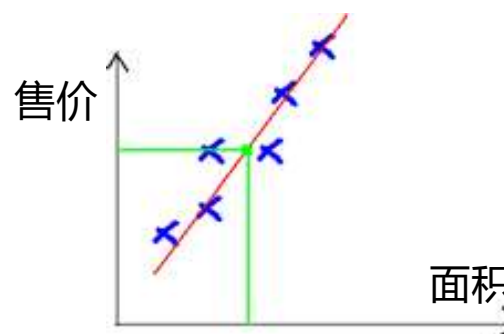
- ③ 训练算法，找到回归系数  
如最小二乘、梯度下降等

- ④ 使用算法进行数据预测

$$y = 10 * x + 3$$



以面积为x轴，售价为y轴建立房屋销售数据的特征空间表示图。



用一条曲线去尽量准的拟合这些数据，然后如果有新的输入过来，我们可以在将曲线上这个点对应的值返回。






# 线性回归扩展



**线性回归扩展算法**用简单的基函数 $\phi(x)$ 替换输入变量 $x$ 。这样我们就把线性拟合形式扩展到了固定非线性函数的线性组合。

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1x_1 + \dots + w_Dx_D$$

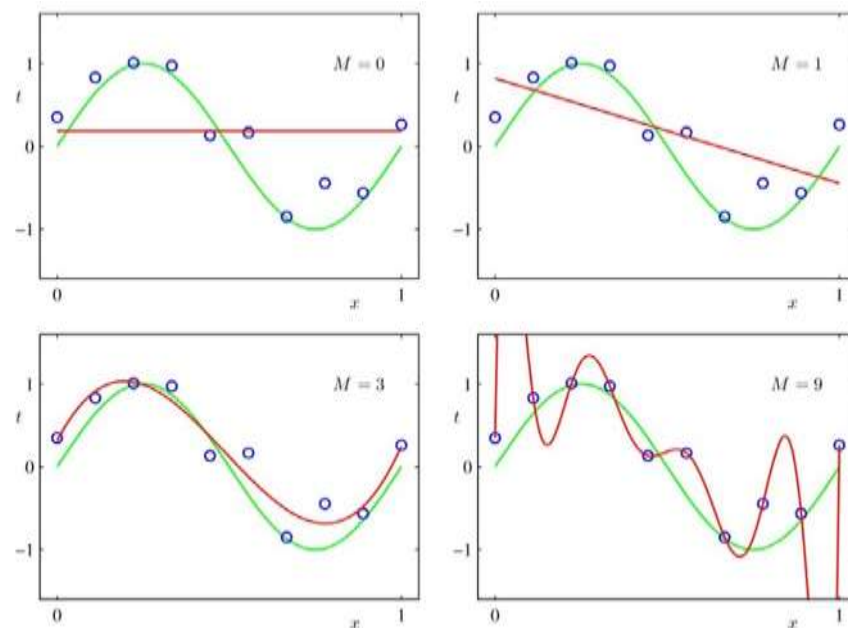

$$y(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(\mathbf{x})$$

**多项式拟合：**取  $\phi_j(x) = x^j$

$$y(x, \mathbf{w}) = w_0 + w_1x + w_2x^2 + \dots + w_Mx^M = \sum_{j=0}^M w_j x^j$$

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2$$

**过拟合：**





# 岭回归

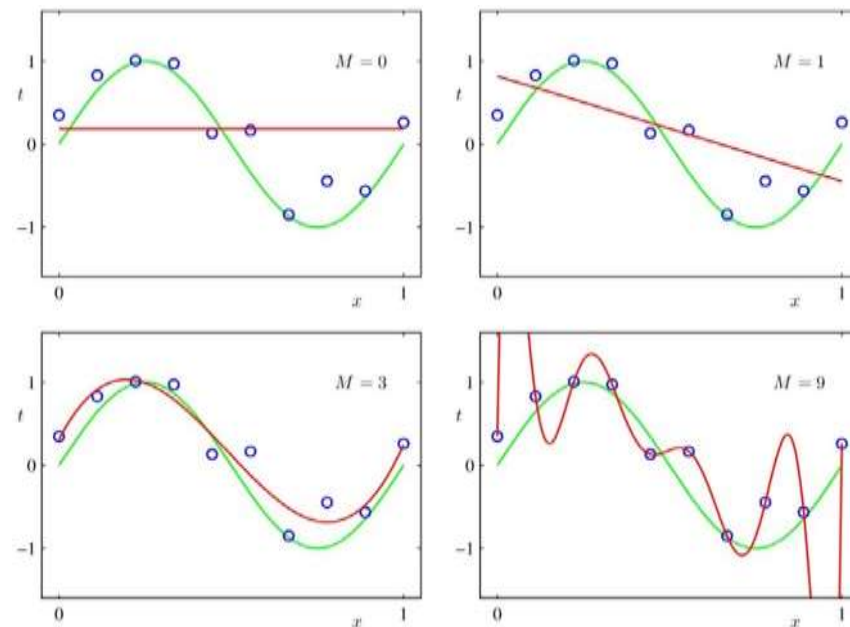


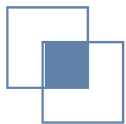
岭回归应用结构风险最小化的模型选择策略，在经验风险最小化的基础上加入正则化因子。当正则化因子选择为模型参数的二范数的时候，整个回归的方法就叫做岭回归。

$$\underset{\beta_0 \in R, \beta \in R^p}{\text{minimize}} \frac{1}{n} \sum_{i=1}^n (y_i - \beta_0 - x_i^T \beta)^2 \text{ subject to } \|\beta\|^2 \leq t$$

$$\underset{\beta_0 \in R, \beta \in R^p}{\text{minimize}} \frac{1}{n} \sum_{i=1}^n (y_i - \beta_0 - x_i^T \beta)^2 + \lambda \|\beta\|^2$$

$\lambda$ 越大，说明偏差就越大，原始数据对回归求取参数的作用就越小，当 $\lambda$ 取到一个合适的值，就能在一定意义上解决过拟合的问题：原先过拟合的特别大或者特别小的参数会被约束到正常甚至很小的值，但不会为零。





# Lasso回归



**Lasso回归**是一种压缩估计。它通过构造一个惩罚函数得到一个较为精炼的模型，使得它压缩一些系数，同时设定一些系数为零。因此保留了子集收缩的优点，是一种处理具有**复共线性数据**的**有偏估计**。

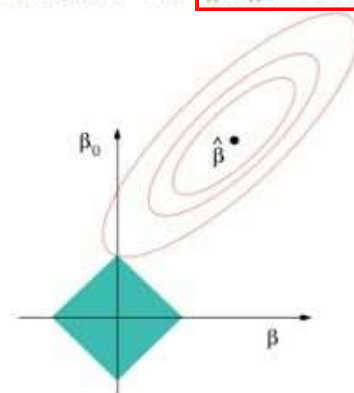
Lasso回归翻译成中文叫套索，就是拿这个东西把动物脖子套住，不要它随便跑。lasso 回归差不多这个意思，就是让回归系数不要太大，以免造成过度拟合（overfitting）。



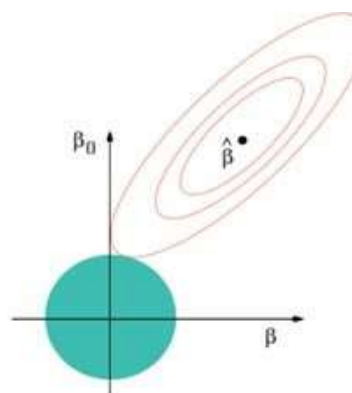
$$\underset{\beta_0 \in R, \beta \in R^p}{\text{minimize}} \frac{1}{n} \sum_{i=1}^n (y_i - \beta_0 - x_i^T \beta)^2 \quad \text{subject to} \quad \|\beta\|_1 \leq t$$

## lasso回归可以适应的情况：

样本量比较小，但是指标非常多。适用于高维统计，传统的方法无法应对这样的数据。并且lasso可以进行特征选择。



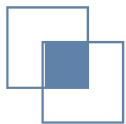
lasso 回归



岭回归

图中的红色线圈表示，损失函数的等值线，可以看到在Lasso第一范数约束下， $\beta$ 可以被约束成0。





# 机器学习方法



机器学习方法

分类问题

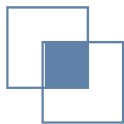
回归问题

聚类问题

其他问题

机器学习模型评估

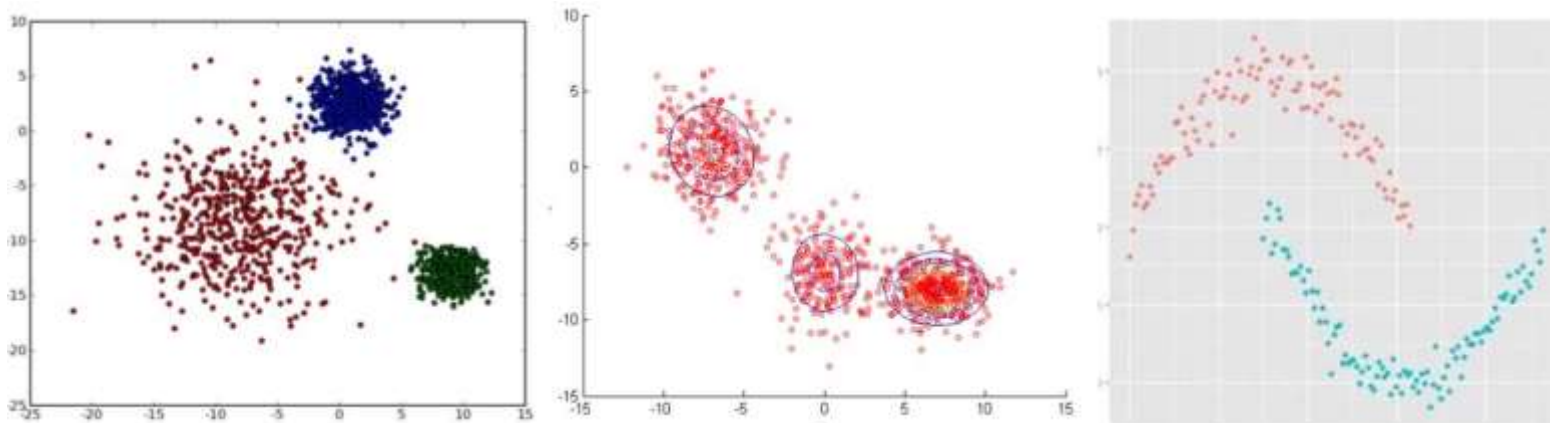




# 聚类问题



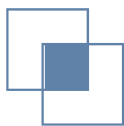
**聚类问题**是无监督学习的问题，算法的思想就是“物以类聚，人以群分”。聚类算法感知样本间的相似度，进行类别归纳，对新的输入进行输出预测，输出变量取有限个离散值。



- ✓ 可以作为一个单独过程，用于寻找数据内在的分布结构
- ✓ 可以作为分类、稀疏表示等其他学习任务的前驱过程







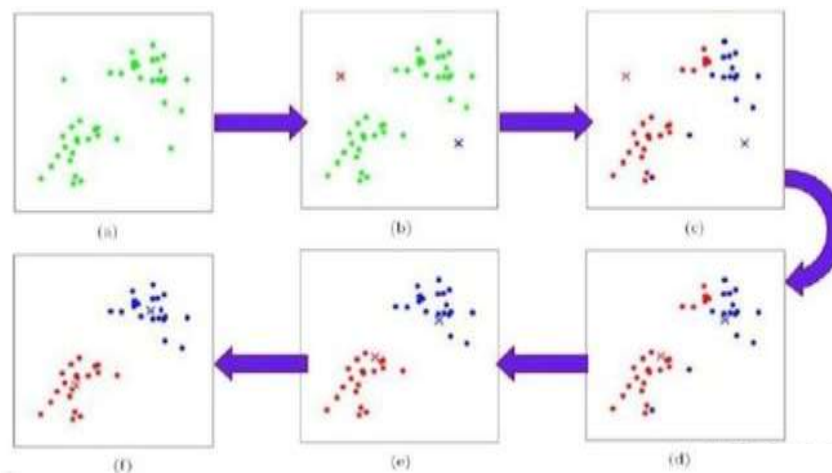
# K-means



**K-means**（又称k-均值或k-平均）聚类算法。**算法思想**就是首先随机确定k个中心点作为聚类中心，然后把每个数据点分配给最邻近的中心点，分配完成后形成k个聚类，计算各个聚类的平均中心点，将其作为该聚类新的类中心点，然后重复迭代上述步骤直到分配过程不再产生变化。

## K-means算法流程

- ① 随机选择k个随机的点（称为聚类中心）；
- ② 对与数据集中的每个数据点，按照距离k个中心点的距离，将其与距离最近的中心点关联起来，与同一中心点关联的所有点聚成一类；
- ③ 计算每一组的均值，将该组所关联的中心点移动到平均值的位置；
- ④ 重复执行2-3步，直至中心点不再变化；

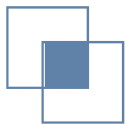


### K-Means的主要**优点**:

- ✓ 原理比较简单，实现也是很容易，收敛速度快
- ✓ 聚类效果较优
- ✓ 算法的可解释度比较强
- ✓ 主要需要调参的参数仅仅是簇数k

### K-Means的主要**缺点**:

- ✓ k值的选取不好把握
- ✓ 不平衡数据集的聚类效果不佳
- ✓ 采用迭代方法，得到的结果只是局部最优
- ✓ 对噪音和异常点比较敏感。

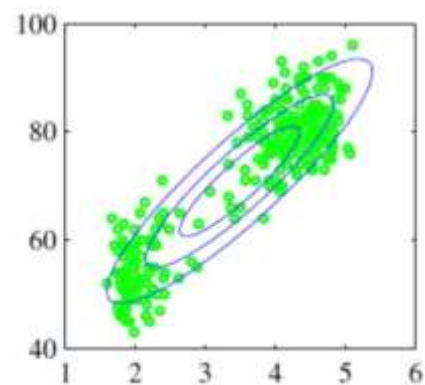
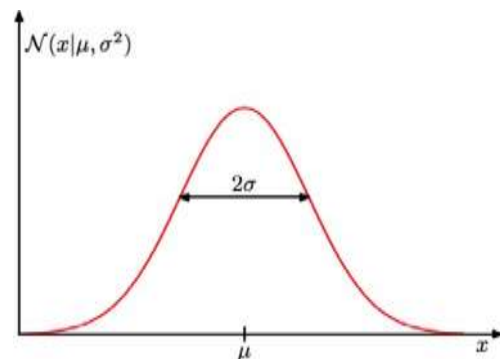


# 高斯混合模型

**高斯混合模型** (Gaussian Mixed Model) 指的是多个高斯分布函数的线性组合，是一种广泛使用的聚类算法，该方法使用了高斯分布作为参数模型。

**单高斯模型：**高斯分布 (Gaussian distribution) 有时也被称为正态分布 (normal distribution)，是一种在自然界大量的存在的、最为常见的分布形式

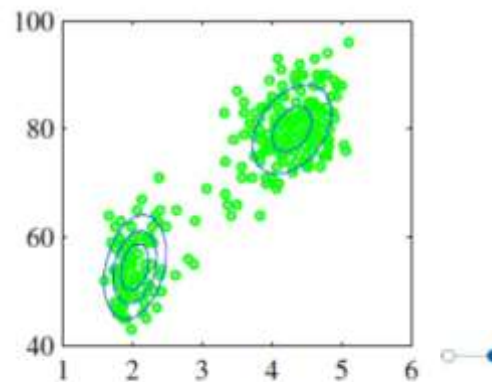
$$f(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\sigma^2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



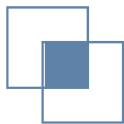
(1) 1 个高斯分布

**高斯混合模型：**混合模型是一个可以用来表示在总体分布中含有K个子分布的概率模型，换句话说，混合模型表示了观测数据在总体中的概率分布，它是一个由K个子分布组成的混合分布。

$$p(x) = \sum_{i=1}^K \phi_i \frac{1}{\sqrt{2\sigma_i^2\pi}} e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}}$$



(2) 2 个高斯分布



# 高斯混合模型求解



**EM算法**是一种迭代算法，1977年由 Dempster 等人总结提出，用于含有隐变量（Hidden variable）的概率模型参数的最大似然估计。

## • 基于高斯混合分布的聚类

### 高斯混合模型的 EM 算法

在每步迭代中，先根据当前参数来计算每个样本属于每个高斯成分的后验概率  $\gamma_{ji}$  (E步)

$$\gamma_{ji} = p_{\mathcal{M}}(z_j = i | \mathbf{x}_j) = \frac{P(z_j = i) \cdot p_{\mathcal{M}}(\mathbf{x}_j | z_j = i)}{p_{\mathcal{M}}(\mathbf{x}_j)} \\ = \frac{\alpha_i \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)}{\sum_{l=1}^k \alpha_l \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}$$

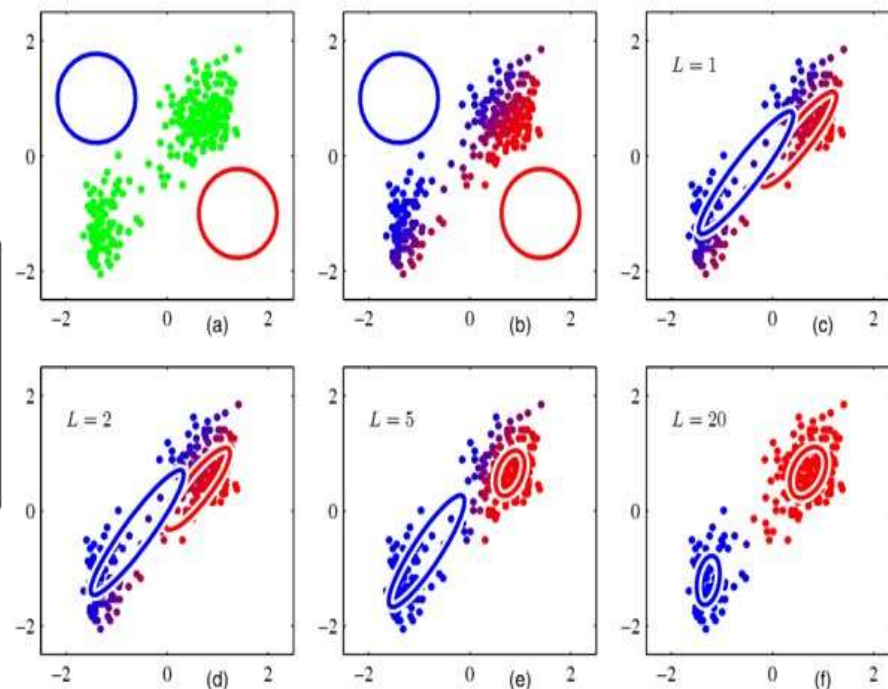


更新模型参数  $\{(\alpha_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) | 1 \leq i \leq k\}$  (M步)

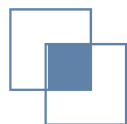
$$\text{计算新均值向量: } \boldsymbol{\mu}'_i = \frac{\sum_{j=1}^m \gamma_{ji} \mathbf{x}_j}{\sum_{j=1}^m \gamma_{ji}}; \\ \text{计算新协方差矩阵: } \boldsymbol{\Sigma}'_i = \frac{\sum_{j=1}^m \gamma_{ji} (\mathbf{x}_j - \boldsymbol{\mu}'_i)(\mathbf{x}_j - \boldsymbol{\mu}'_i)^T}{\sum_{j=1}^m \gamma_{ji}}; \\ \text{计算新混合系数: } \alpha'_i = \frac{\sum_{j=1}^m \gamma_{ji}}{m};$$

每个样本  $\mathbf{x}_j$  的簇标记  $\lambda_j = \arg \max_{i \in \{1, 2, \dots, k\}} \alpha_i \cdot p(\mathbf{x}_j | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

因此，从原型聚类的角度来看，高斯混合聚类是采用概率模型(高斯分布)对原型进行刻画，簇划分则由原型对应后验概率确定。



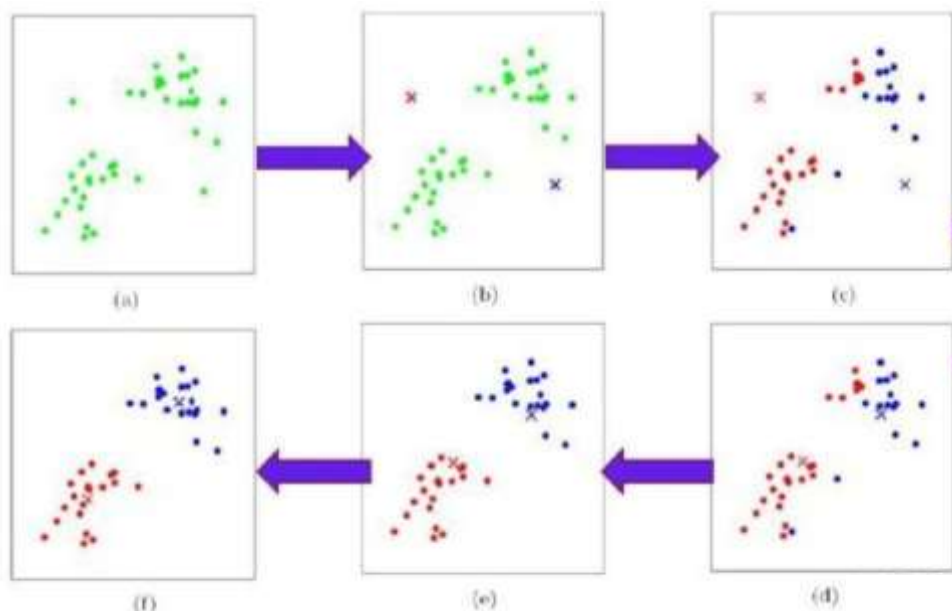




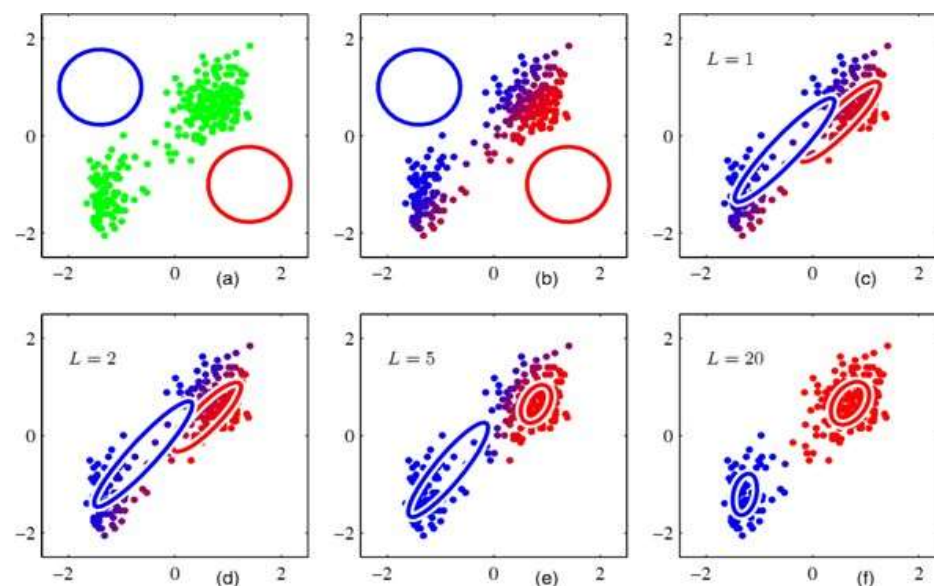
# 高斯混合模型与K-means



K-means

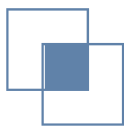


混合高斯



➤混合高斯和K-means很相似，相似点在于两者的分类受初始值影响；两者可能限于局部最优解；两者类别的个数都要靠猜测。混合高斯计算复杂度高于K-means。

➤K-means属于硬聚类，要么属于这类，要么属于那类，而GMM属于混合式软聚类，一个样本70%属于A，30%属于B。



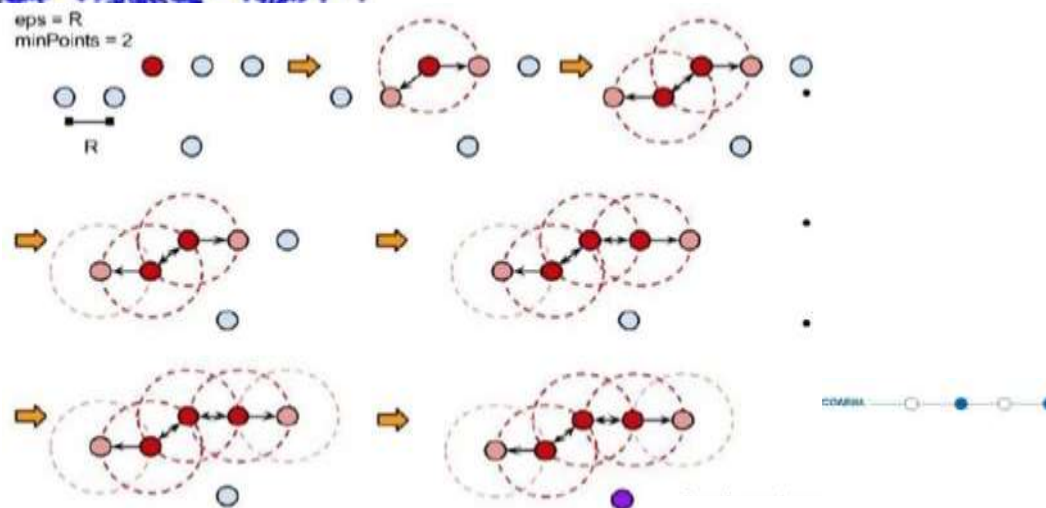
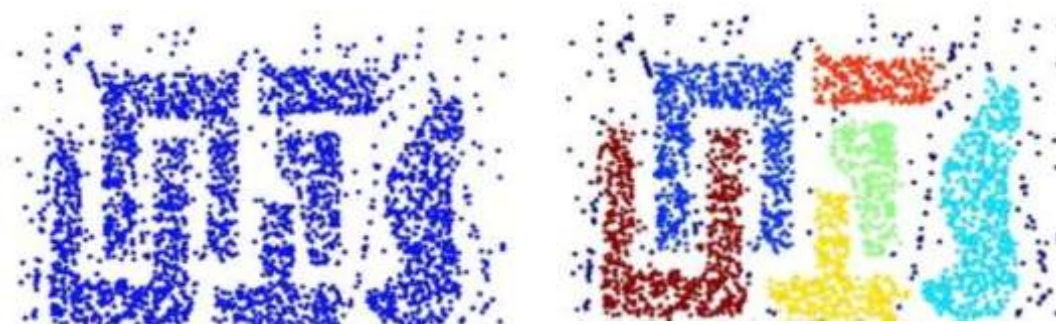
# 密度聚类

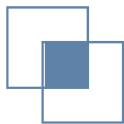
VI DISCO/IMA

**密度聚类算法**假设聚类结构能通过样本分布的紧密程度确定，算法从样本密度的角度来考察样本之间的可连接性，并基于可连接样本不断扩展聚类簇以获得最终的聚类结果。

## DBSCAN算法流程

- ① DBSCAN通过检查数据集中每个点的Eps邻域来搜索簇，如果点p的Eps邻域包含的点多于MinPts个，则创建一个以p为核心对象的簇；
- ② 然后，DBSCAN迭代地聚集从这些核心对象直接密度可达的对象，这个过程可能涉及一些密度可达簇的合并；
- ③ 当没有新的点添加到任何簇时，该过程结束





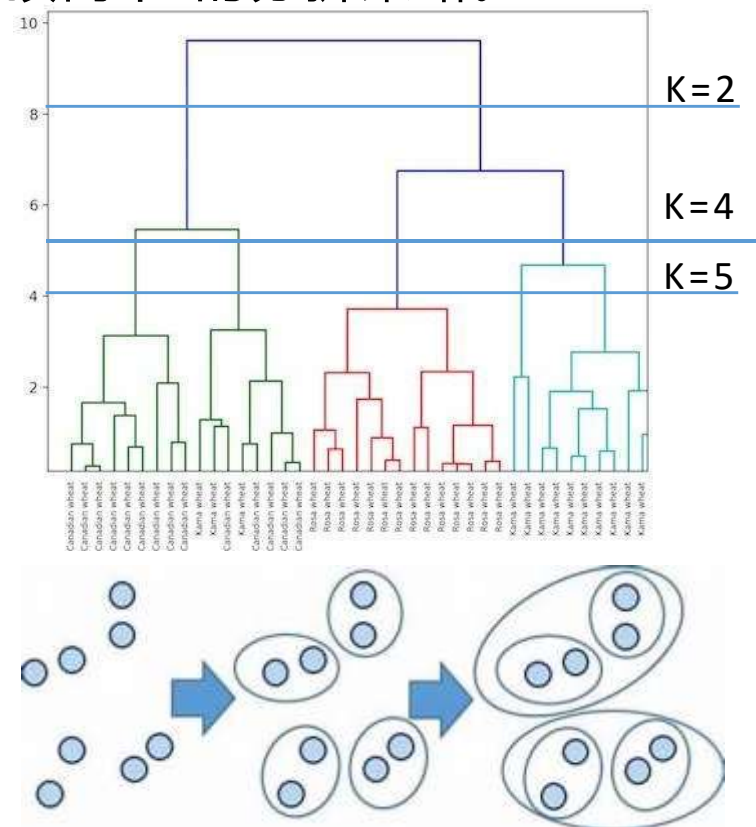
# 层次聚类



**层次聚类算法**试图在不同层次对数据集进行划分，从而形成树形的聚类结构。数据集的划分可采用“自底向上”的聚合策略，也可采用“自顶向下”的分拆策略。

## AGNES算法流程

- ① AGNES (Agglomerative NESting) 算法最初将每个对象做为一个簇，然后这些簇根据某些准则被一步步的合并，使用单链接方法；
- ② 两个簇间的相似度由这两个不同簇中距离最近的数据点对的相似度来确定。此外当两个簇最近距离超过用户给定的阈值时聚类过程就会终止；
- ③ 聚类的合并过程反复进行直到所有的对象最终满足簇数据。



# 谱聚类

**谱聚类(Spectral Clustering, SC)**是一种基于图论的聚类方法，将带权无向图划分为两个或两个以上的最优子图，使子图内部尽量相似，而子图间距离尽量距离较远，以达到常见的聚类的目的。

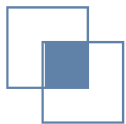
输入：样本 $x_i$ ，需要聚类的个数 $k$

- 构建相似度矩阵 $S$ ，样本间 $x_i$ 已经通过高斯相似度构建出了相似度矩阵 $S$ ，也就是邻接矩阵 $W$ 。
- 计算出度矩阵 $D$
- 计算出拉普拉斯矩阵 $L=D-W$
- 计算出 $L$ 前 $k$ 个最小的特征向量 $v_1, \dots, v_k$
- 将前 $k$ 个特征向量组合成一个矩阵 $V$ ，其中第 $i$ 列对应 $v_i$ 列向量
- 该矩阵 $V$ 的每一行对应代表 $x_i$ 的低维度的表示 $y_i$ 。
- 对所有 $y_i$ 进行 $k$ -means聚类，聚成 $k$ 类

## 谱聚类的优势：

- ✓ 只需要待聚类点之间的相似度矩阵就可以做聚类了。
- ✓ 对于不规则的数据（或者说是离群点）不是那么敏感，个人感觉主要体现在最小化图切割的公式中
- ✓  $k$ -means聚类算法比较适合于凸数据集（数据集内的任意两点之间的连线都在该数据集以内，简单理解就是圆形），而谱聚类则比较通用

谱聚类能够识别任意形状的样本空间且收敛于全局最优解，其基本思想是利用样本数据的相似矩阵(拉普拉斯矩阵)进行特征分解后得到的特征向量进行聚类。



# 机器学习方法



机器学习方法

分类问题

回归问题

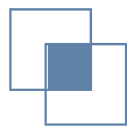
聚类问题

其他问题

机器学习模型评估







# 隐马尔可夫模型

**隐马尔可夫模型**是一个关于时序的概率模型，描述由隐马尔可夫链随机生成观测序列的过程，属于生成模型。隐马尔可夫模型在语音识别、自然语言处理、生物信息等领域有着广泛的应用。

## 隐马尔可夫模型两个假设

- ① **齐次马尔可夫性假设**，即假设隐藏的马尔可夫链在任意时刻 $t$ 的状态只依赖于其前一时刻的状态，与其他时刻的状态及观测无关，也与时刻 $t$ 无关。

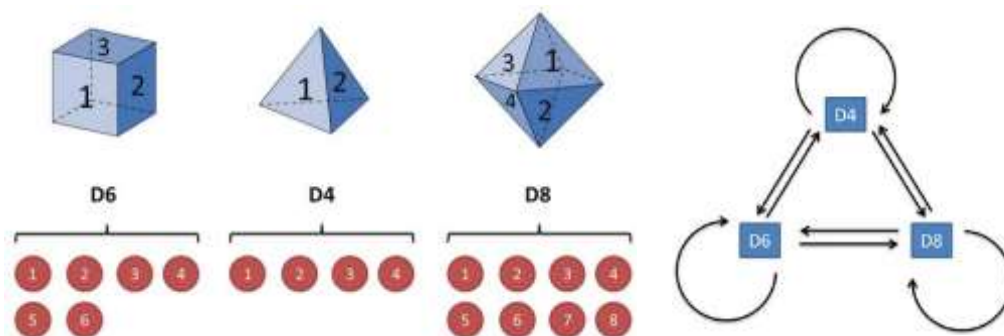
$$P(i_t | i_{t-1}, o_{t-1}, \dots, i_1, o_1) = P(i_t | i_{t-1})$$

$1, 2, \dots, T$

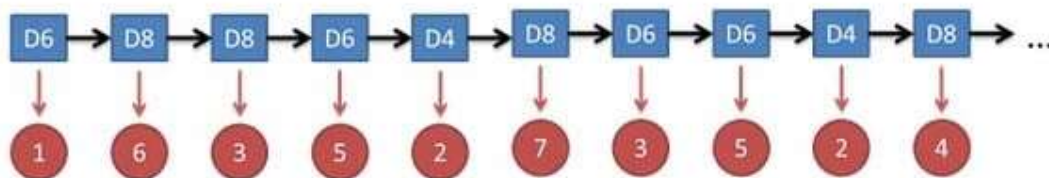
- ② **观测独立性假设**，即假设任意时刻的观测只依赖于该时刻的马尔可夫链的状态，与其他观测及状态无关。

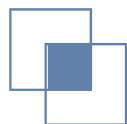
$$P(o_t | i_t, o_t, \dots, i_1, o_1) = P(o_t | i_t)$$

$1, 2, \dots, T$



隐马尔可夫模型示意图





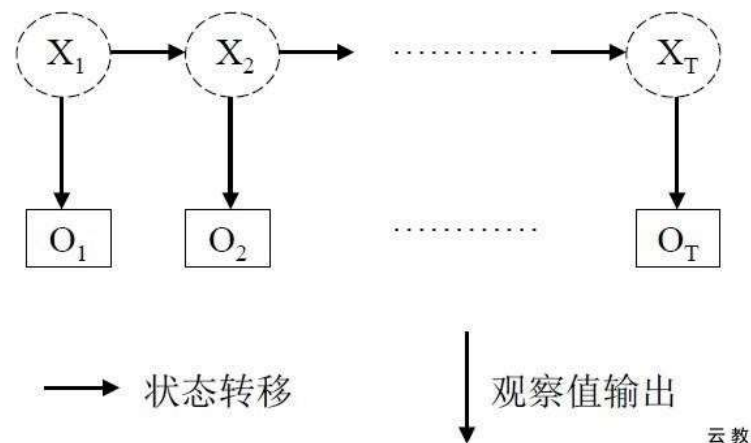
# 隐马尔可夫模型



**隐马尔可夫模型**  $\lambda = (A, B, \Pi)$ ，状态转移概率矩阵 $A$ ，初始状态概率向量 $\Pi$ ，确定了隐藏的马尔可夫链，生成不可观测的状态序列。观测概率矩阵 $B$ 确定了如何从状态生成观测，与状态序列综合确定了如何产生观测序列。

## 隐马尔可夫模型三个基本问题

- ① **概率计算问题**，给定模型 $\lambda = (A, B, \Pi)$ 和观测序列 $O = (o_1, o_2, \dots, o_T)$ ，计算在模型 $\lambda$ 下观测序列 $O$ 出现的概率 $P = (O|\lambda)$ 。
- ② **学习问题**，已知观测序列 $O = (o_1, o_2, \dots, o_T)$ 估计模型 $\lambda = (A, B, \Pi)$ 参数，使得在该模型下观测序列概率 $P = (O|\lambda)$ 最大。
- ③ **预测问题**，已知模型 $\lambda = (A, B, \Pi)$ 和观测 $O = (o_1, o_2, \dots, o_T)$ ，求对给定观测序列条件概率 $P = (I|O)$ 最大的状态序列。即给定观测序列，求最有可能的对应的状态序列。

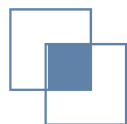


云教

**应用：**  
词性标注、中文分词、天气预测等



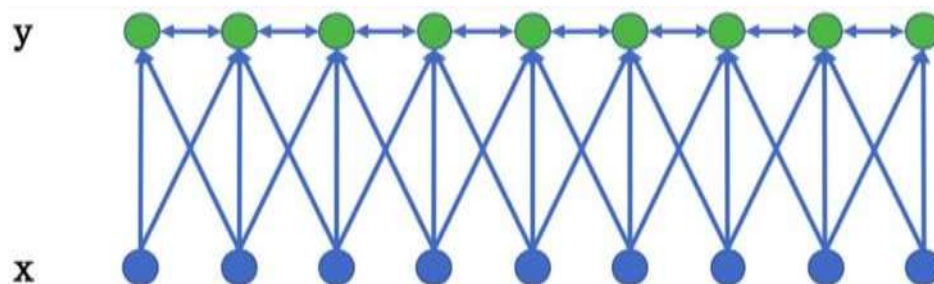




# CRF条件随机场



CRF(Conditional Random Field)条件随机场是一个序列标注模型，其优点在于 为一个位置进行标注的过程中可以利用丰富的内部及上下文特征信息。



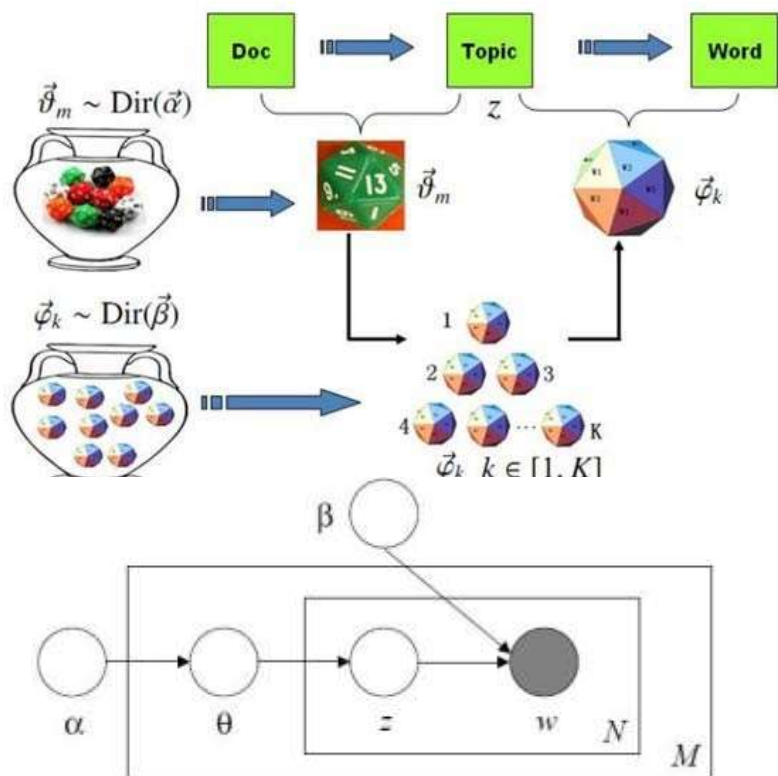
CRF由John Lafferty最早在NLP技术领域任务中进行文本标注，有多种应用场景，如：

- ✓ 分词（标注字的词位信息，由字构词）
- ✓ 词性标注（标注分词的词性，例如：名词，动词，助词）
- ✓ 命名实体识别（识别人名，地名，机构名，商品名等具有一定内在规律的实体名词）



# LDA主题模型

**LDA主题模型**是一种文档主题生成模型，是一种非监督机器学习技术。通过模拟文档生成过程，可以用来识别大规模文档集（document collection）或语料库（corpus）中潜藏的主题信息。



Topics

|         |      |
|---------|------|
| gene    | 0.04 |
| dna     | 0.02 |
| genetic | 0.01 |
| ...     |      |

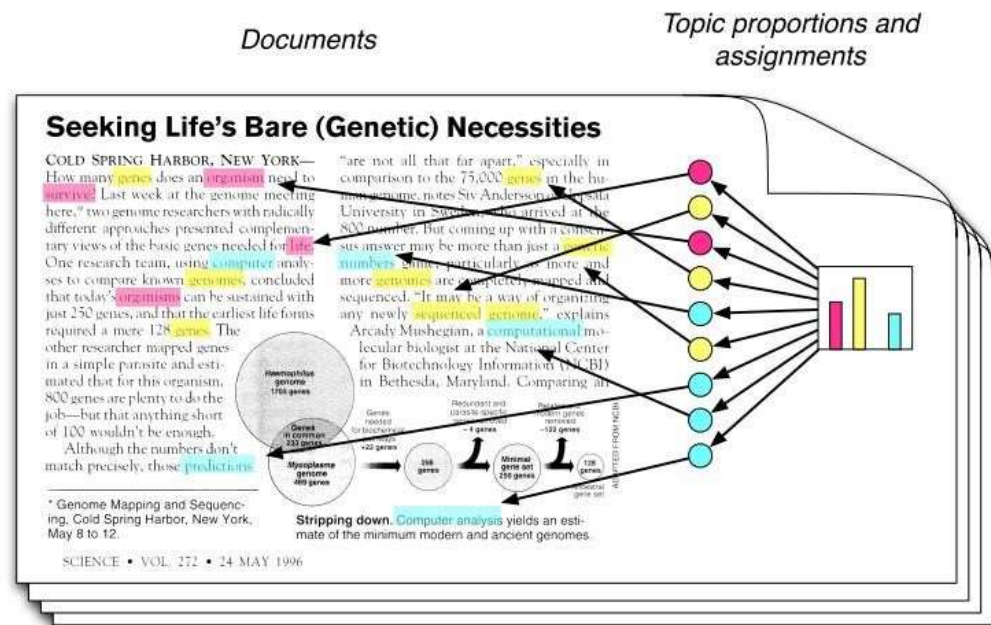
|          |      |
|----------|------|
| life     | 0.02 |
| evolve   | 0.01 |
| organism | 0.01 |
| ...      |      |

|        |      |
|--------|------|
| brain  | 0.04 |
| neuron | 0.02 |
| nerve  | 0.01 |
| ...    |      |

|          |      |
|----------|------|
| data     | 0.02 |
| number   | 0.02 |
| computer | 0.01 |
| ...      |      |





# 生成模型 v.s. 判别模型



**监督学习方法**可分为两大类，即生成方法与判别方法，它们所学到的模型称为生成模型与判别模型。

例子：例如你需要识别一种语言到底是汉语还是英语等。那么你可以有两种方法达到这个目的：

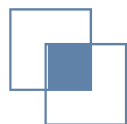
- 1、学习两种语言，你花了大量精力把汉语、英语都学会了，然后你就知道是哪种语言了。
- 2、不去学习每一种语言，你只学习汉语与英语之间的差别，然后再分类。

## 生成方法的特点：

- 从统计的角度表示数据的分布情况，能够反映同类数据本身的相似度；
- 生成方法还原出联合概率分布，而判别方法不能；
- 生成方法的学习收敛速度更快、即当样本容量增加的时候，学到的模型可以更快地收敛于真实模型；
- 当存在隐变量时，仍然可以用生成方法学习，此时判别方法不能用

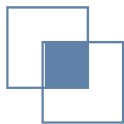
## 判别方法的特点：

- 判别方法寻找不同类别之间的最优分类面，反映的是异类数据之间的差异；
- 判别方法利用了训练数据的类别标识信息，直接学习的是条件概率 $P(Y|X)$ 或者决策函数 $f(X)$ ，直接面对预测，往往学习的准确率更高；
- 由于直接学习条件概率 $P(Y|X)$ 或者决策函数 $f(X)$ ，可以对数据进行各种程度上的抽象、定义特征并使用特征，因此可以简化学习问题；
- 缺点是不能反映训练数据本身的特性。



# 神经网络





# 机器学习方法



机器学习方法

分类问题

回归问题

聚类问题

其他问题

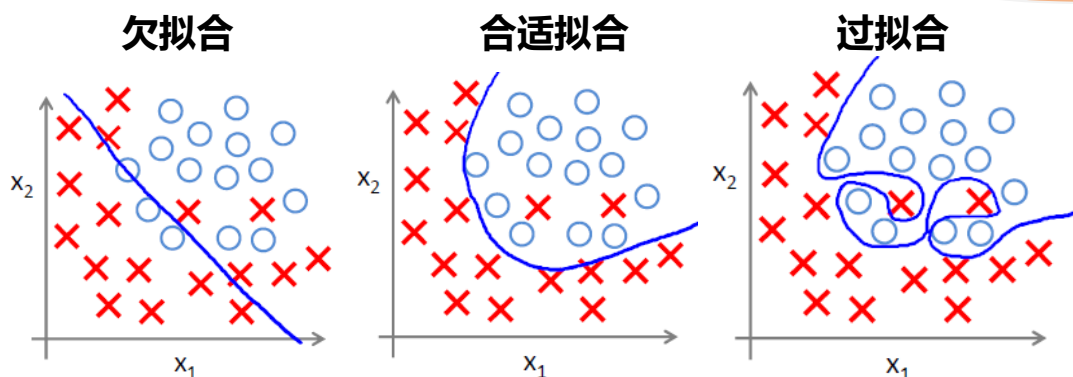
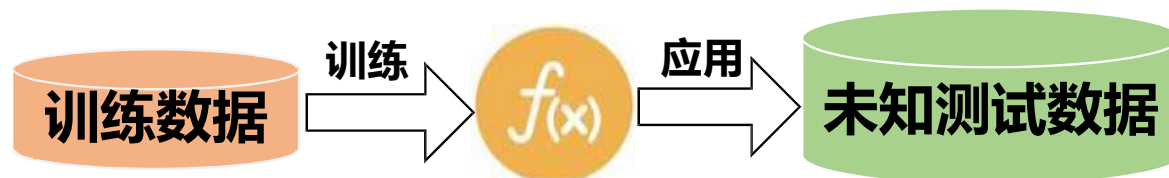
机器学习模型评估



# 模型选择

**泛化误差**：在“未来”样本上的误差

**经验误差**：在训练集上的误差



✓ AIC准则 (Akaike Information Criterion)

$$AIC = 2k - 2\ln(L)$$

✓ BIC准则 (Bayesian Information Criterion)

$$BIC = k\ln(n) - 2\ln(L)$$





# 性能评价指标-分类



**准确率(Accuracy)**是指在分类中，分类正确的记录个数占总记录个数的比。

$$accuracy = \frac{n_{correct}}{n_{total}}$$

**召回率(Recall)**也叫查全率，是指在分类中样本中的正例有多少被预测正确了。

通常，准确率高时，召回率偏低；召回率高时，准确率偏低。

## 1. 地震的预测

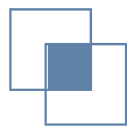
对于地震的预测，我们希望的是召回率非常高，也就是说每次地震我们都希望预测出来。

## 2. 嫌疑人定罪

基于不错怪一个好人的原则，对于嫌疑人的定罪我们希望是非常准确的。及时有时候放过了一些罪犯（召回率低），但也是值得的。







# 性能评价指标-分类

**准确率(Accuracy):** 分类正确的样本个数占所有样本个数的比例

$$accuracy = \frac{TP + TN}{TP + FN + FP + TN}$$

**平均准确率(Average per-class accuracy):** 每个类别下的准确率的算术平均

$$average\_accuracy = \frac{\frac{TP}{TP+FN} + \frac{TN}{TN+FP}}{2}$$

**精确率(Precision):** 分类正确的正样本个数占分类器所有的正样本个数的比例

$$Precision = \frac{TP}{TP + FP}$$

**召回率(Recall):** 分类正确的正样本个数占正样本个数的比例

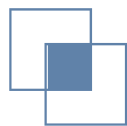
$$Recall = \frac{TP}{TP + FN}$$

分类结果混淆矩阵

| 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) |

**F1-Score:** 精确率与召回率的调和平均值, 它的值更接近于Precision与Recall中较小的值

$$F1 = \frac{2 * precision * recall}{precision + recall}$$



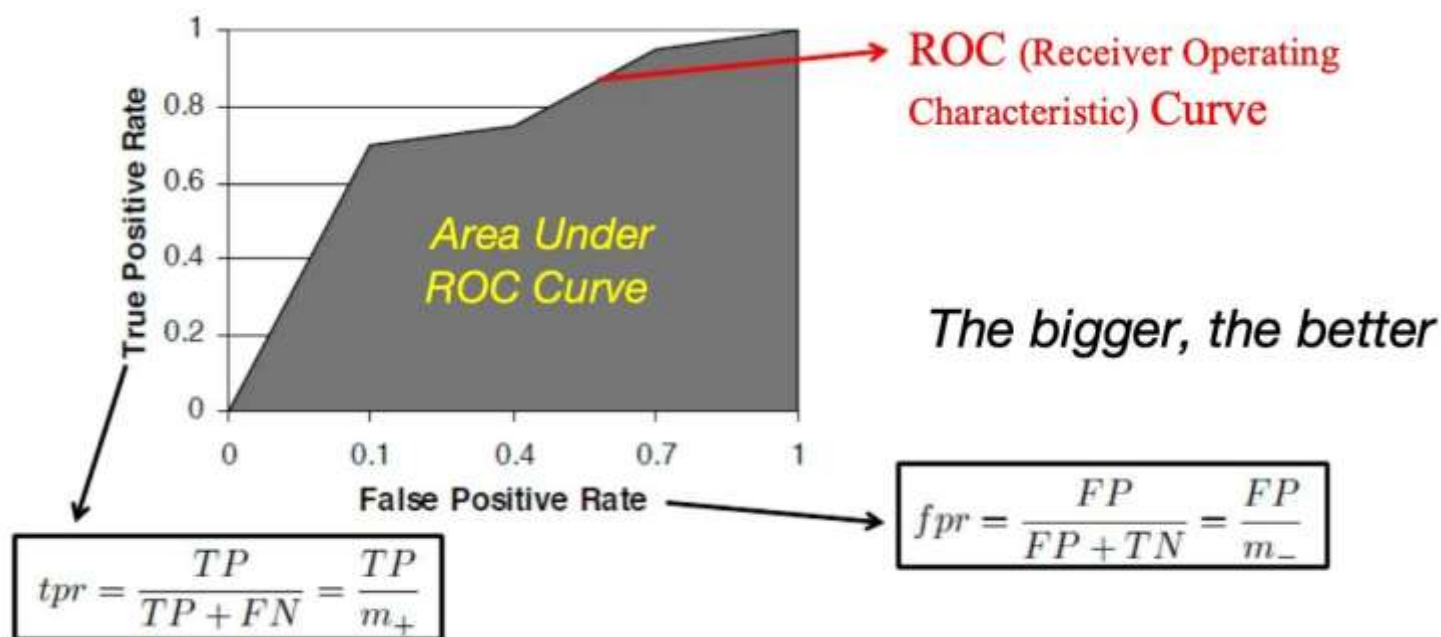
# 性能评价指标-分类



AUC(Area under the Curve(Receiver Operating Characteristic, ROC)) **ROC**: 纵轴

: 真正例率TPR; 横轴: 假正例率FPR

**AUC**是ROC曲线下的面积。一般来说, 如果ROC是光滑的, 那么基本可以判断没有太大的overfitting, 这个时候调模型可以只看AUC, 面积越大一般认为模型越好。





# 性能评价指标-分类



**PR曲线:** 根据学习器的预测结果按正例可能性大小对样例进行排序, 并逐个把样本作为正例进行预测。

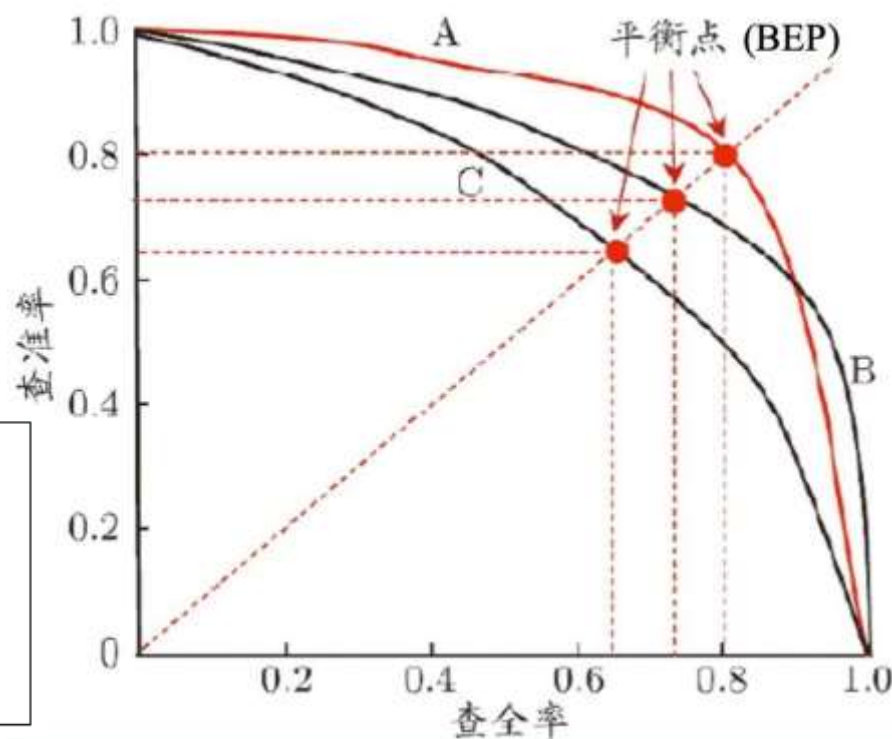
- ✓ 如果一个学习器的**PR曲线**包住了另一个, 则可以认为A的性能优于C
- ✓ 如果有交叉, 如A、B, 综合考虑PR性能, 引入**平衡点(BEP)**, 基于BEP比较, A优于B

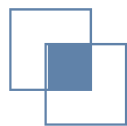
PR图:

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

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





# 性能评价指标-分类



## 宏平均&微平均

多分类问题中，若能得到**多个混淆矩阵**，例如多次训练/测试的结果，多分类的两两混淆矩阵：

**宏(macro-)**查准率、查全率、F1

$$\text{macro-}P = \frac{1}{n} \sum_{i=1}^n P_i ,$$

$$\text{macro-}R = \frac{1}{n} \sum_{i=1}^n R_i ,$$

$$\text{macro-}F1 = \frac{2 \times \text{macro-}P \times \text{macro-}R}{\text{macro-}P + \text{macro-}R} .$$

**微(micro-)**查准率、查全率、F1

$$\text{micro-}P = \frac{\overline{TP}}{\overline{TP} + \overline{FP}} ,$$

$$\text{micro-}R = \frac{\overline{TP}}{\overline{TP} + \overline{FN}} ,$$

$$\text{micro-}F1 = \frac{2 \times \text{micro-}P \times \text{micro-}R}{\text{micro-}P + \text{micro-}R} .$$



# 性能评价指标-分类



**对数损失** (Log loss) 亦被称为逻辑回归损失 (Logistic regression loss) 或交叉熵损失 (Cross-entropy loss)。

**二分类问题:**  $y \in \{0, 1\}$  且  $p = \Pr(y = 1)$  则对每个样本的对数损失为

$$L_{\log}(y, p) = -\log \Pr(y|p) = -(y \log(p) + (1 - y) \log(1 - p))$$

**多分类问题:** 设Y为指示矩阵, 即当样本i的分类为k,  $y_{i,k} = 1$  设P为估计的概率矩阵,  $p_{i,k} = \Pr(t_{i,k} = 1)$  则对每个样本的对数损失为

$$L_{\log}(Y_i, P_i) = -\log \Pr(Y_i|P_i) = \sum_{k=1}^K y_{i,k} \log p_{i,k}$$



# 性能评价指标-回归

**平均绝对误差：**平均绝对误差MAE (Mean Absolute Error) 又被称为l1范数损失 (l1-norm loss)

$$\text{MAE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} |y_i - \hat{y}_i|$$

**平均平方误差：**平均平方误差MSE (Mean Squared Error) 又被称为l2范数损失 (l2-norm loss) :

**均方根差RMSE:** 
$$\text{MSE}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=1}^{n_{\text{samples}}} (y_i - \hat{y}_i)^2$$

**R Squared:**是将预测值跟只使用均值的情况下相比，看能好多少。

$$R^2 = 1 - \frac{(\sum_i (\hat{y}_i - \bar{y})^2) / m}{(\sum_i (y_i - \bar{y})^2) / m} = 1 - \frac{\text{MSE}(\hat{y}, y)}{\text{Var}(y)}$$

# 性能评价指标-聚类



**外部指标**对数据集  $D = \{x_1, x_2, \dots, x_m\}$ , 假定通过聚类给出的簇划分为  $C = \{C_1, C_2, \dots, C_k\}$  参考模型给出的簇划分为  $C^* = \{C_1^*, C_2^*, \dots, C_k^*\}$ , 通过比对  $C$  和  $C^*$  来判定聚类结果的好坏。

**Jaccard系数, FM 指数, Rand 指数, 纯度purity, 熵 entropy, 互信息, Adjusted Rand Index (ARI), F-measure, Probabilistic Rand Index (PRI)**

**内部指标**对聚类数据结构上的描述, 类内距离小, 类间距离大较好。

**DB 指数(Davies-Bouldin Index, 简称DBI):** 衡量同一簇中数据的紧密性, 越小越好。

**Dunn 指数(Dunn Index 简称DI):** 衡量同一簇中数据的紧密性, 越大越好。 **Silhouette:** 衡量同一簇中数据的紧密性, 越大越好。

**Modurity:** 衡量模块性, 越大越好。







# 目录



1

## 概论

学科定义、发展历程、  
机器学习方法、应用场景与挑战

2

## 机器学习准备

鸢尾花分类任务简介  
数据预处理、特征工程

3

## 机器学习方法

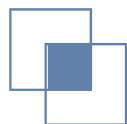
学习方法：分类、回归、聚类、其他  
机器学习模型评估

4

## 课程实践

实践：鸢尾花分类



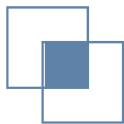


# 课程实践



实践：鸢尾花分类





# 感谢聆听



有什么问题吗？

