



# 第二章 机器学习基础

## 2.3节机器学习常见模型

课程名称: 人工智能通识教程

主讲人:

日期:

单位: 华南农业大学



# 目录

## C O N T E N T S

01 回顾与新知识

02 分类模型

03 聚类模型

04 回归模型

05 课堂总结

# 知识重点

分类模型（解决 “属于哪一类” 的定性问题 ）：

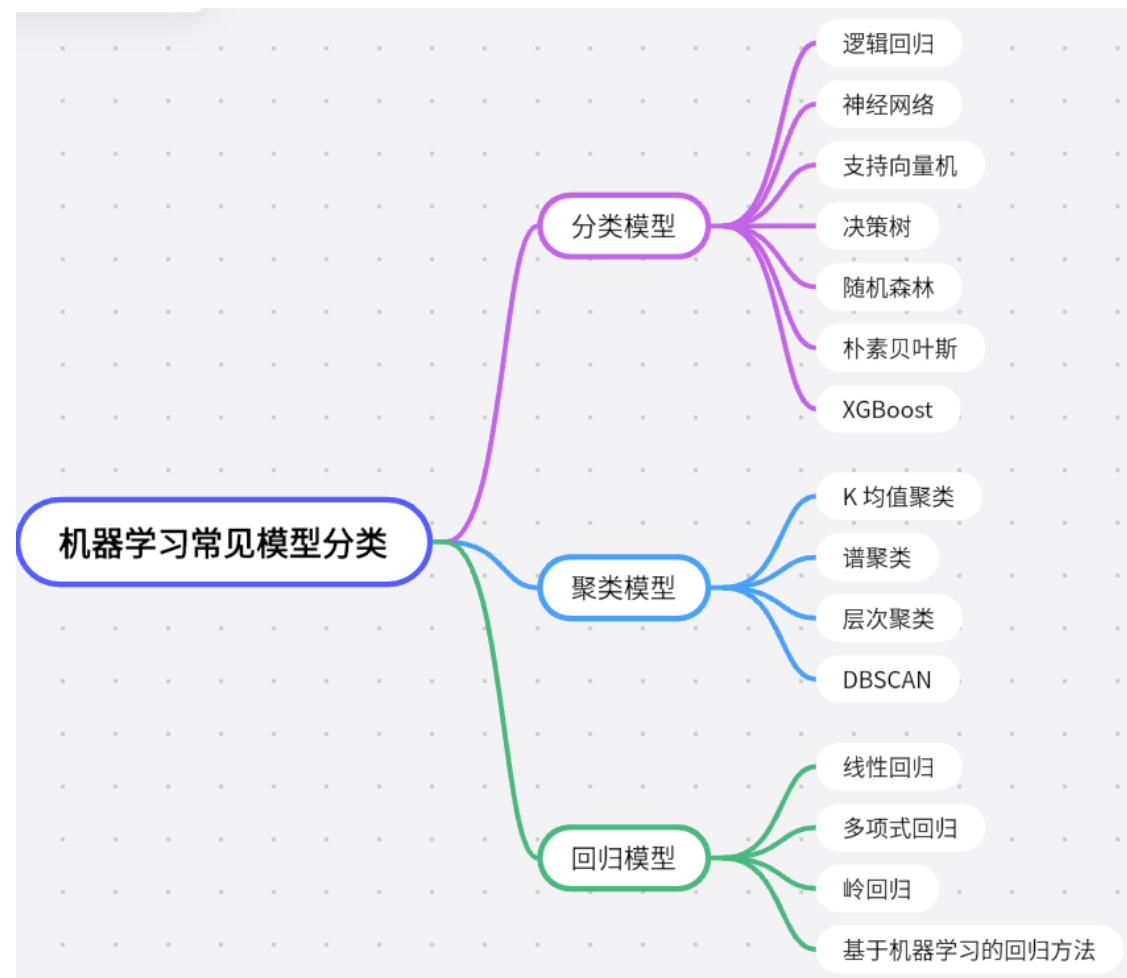
包含逻辑回归、神经网络、支持向量机、决策树、随机森林、朴素贝叶斯、XGBoost 等算法。

聚类模型（解决 “如何分组” 的无监督问题 ）：

包含 K 均值聚类、谱聚类、层次聚类、DBSCAN 等算法。

回归模型（解决 “预测连续值” 的定量问题 ）：

包含线性回归、多项式回归、岭回归、基于机器学习的回归方法（如随机森林回归 ）等。





# 01

## 回顾与思考

# 机器学习的三要素



- 模型（Model）- 模型是对 “输入与输出关系” 的假设。本质上，模型是一个 “参数化的函数”，参数就是我们要通过数据学习的 “未知数”。
- 学习准则 - 有了模型，怎么判断它好不好？学习准则就是 “打分标准”，通常转化为 “目标函数”。
- 优化算法 - 优化算法的核心是：如何高效找到最优参数。

# 思考

---



- (1) 如果你的手机相册里有1000张照片，如何让手机自动把'猫的照片'和'狗的照片'分开？
- (2) 如果给你500个陌生人的购物小票，如何自动发现哪些人购物习惯相似？
- (3) 如果知道一个学生每天的学习时间，如何预测他期末考试能考多少分？



## 02 分类模型

# 举例引入



例如 你的电子邮箱自动把邮件分成 正常邮件 和 垃圾邮件。

简单描述：

输入：每封邮件的特征（比如：发件人、主题关键词、内容中的特殊词、链接数量等）

处理：分类模型已经学习过成千上万封标记好的邮件（哪些是垃圾邮件，哪些不是）

输出：当你收到新邮件时，模型会判断“这看起来像垃圾邮件吗？”，然后决定放进收件箱还是垃圾箱  
就像一个经验丰富的邮局分拣员，看过很多信件后，能快速判断哪些是广告传单（垃圾邮件），哪些是重要信件（正常邮件）。只不过这个“分拣员”是电脑程序，而且永远不会累！



# 分类模型的定义

分类（Classification）模型用于预测离散的类别标签，比如判断一封邮件是垃圾邮件还是正常邮件，或者识别一张图片中的动物是猫还是狗。

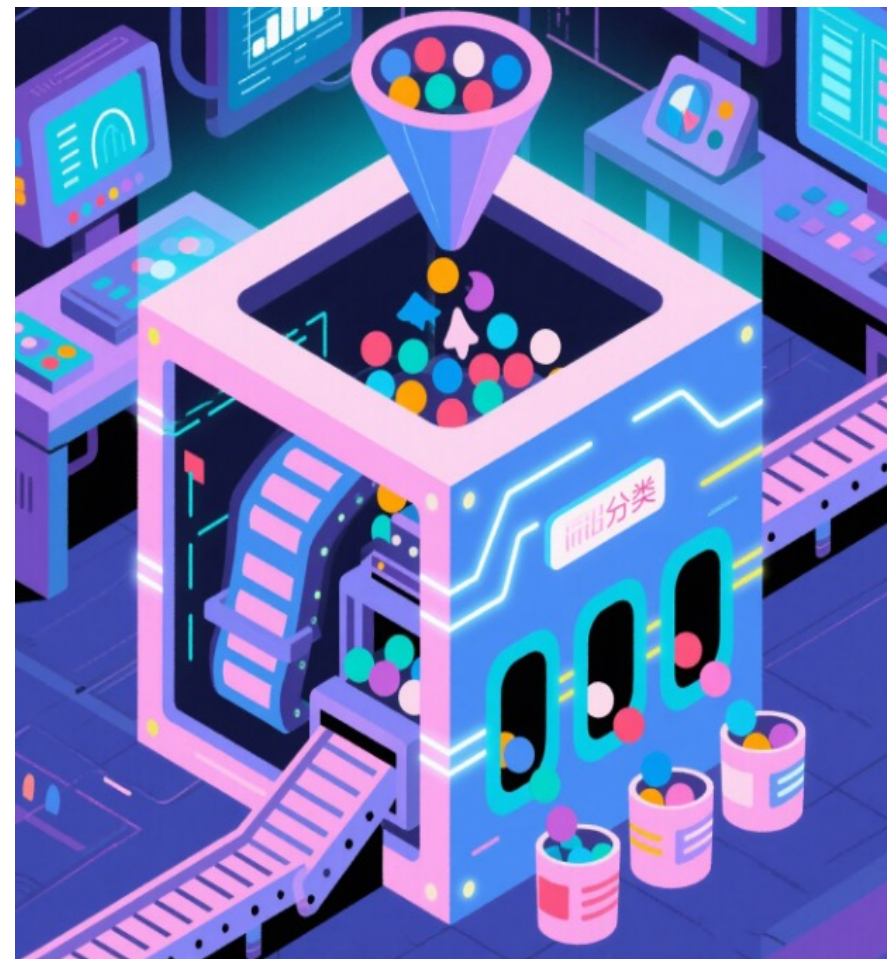
分类模型是机器学习中用于将输入数据划分到预定义类别或标签的一种算法。简单来说，它就像一个“分类器”，能够根据数据的特征自动判断它属于哪个类别。

输入：特征（features） - 描述数据点的各种属性

例如：判断水果时，特征可以是颜色、重量、大小等

输出：类别标签（class label） - 预先定义的可能类别

例如：水果分类中的“苹果”、“香蕉”、“橙子”等



# 分类模型核心



核心任务：让计算机学会自动把新邮件分成“正常邮件”和“垃圾邮件”！

关键问题：计算机又不是人，它怎么知道哪些邮件是垃圾？它需要一套“判断规则”！

解决方案：教它！用“带答案的练习题”来训练它！

第一步：学习规则（模型训练）—— 制作“分类规则手册”

（1）准备“带答案的练习题”：我们收集成千上万封历史邮件。

每封邮件都人工贴好了正确答案（标签）： “正常邮件” 或  “垃圾邮件”。

这些就是计算机要学习的“带答案的练习题”。

（2）告诉计算机“看什么”（特征）：

我们告诉计算机每封邮件有哪些特征值得关注（就像破案线索）：

邮件标题： 有没有“免费”、“恭喜中奖”、“限时特惠”、“紧急通知”等词？

发件人地址： 是不是陌生地址？看起来乱不乱（比如 asdfg12345@weirddomain.com）？

邮件正文： 有没有大量感叹号（!!!）？有没有可疑链接？有没有让你点链接或输密码的要求？

邮件格式： 排版是不是特别花哨混乱？图片特别多？

# 分类模型核心



(3) 计算机自己“找规律”：


计算机开始疯狂“刷题”！

它分析每一封“练习题”邮件：看看它的特征是什么，再看看它对应的正确答案（标签）是什么。

它拼命在特征和标签之间找关联模式（规则）：

“咦？标题里带‘免费’这个词的邮件，90% 都被标为  垃圾邮件！”

“发件人地址特别奇怪的邮件，几乎100% 都是  垃圾邮件！”

“标题是‘项目进度’ 的邮件，95% 都是  正常邮件。”

“正文里有‘妈妈周末炖了汤’ 这种话的，99% 是  正常邮件。”

（计算机就是在做统计和模式识别：什么特征组合出现时，最可能对应哪个标签）

(4) 形成“分类规则手册”（模型）：

计算机把找到的所有规律、模式、关联性，总结归纳成一套内部的“判断规则”。

这套规则就像一本《垃圾邮件识别指南》 或者一个自动决策机器。

这就是训练好的“分类模型”！ 它的核心就是这本学到的“规则手册”。

# 分类模型核心



第二步：使用规则进行预测 —— 用“规则手册”判断新邮件

现在，来了一封全新的邮件，谁也没见过，也没贴标签。计算机怎么判断它是正常还是垃圾？

(1) 观察特征：

计算机立刻提取这封新邮件的特征：

标题：“免费领取最新iPhone！”

发件人：“prize.winner@superluckymail.biz”

正文：“\*\*恭喜您中奖！点击链接立即领取>> [可疑链接] \*\*” （有很多感叹号）

(2) 查阅“规则手册”：

计算机拿出它训练时辛苦总结的那本《垃圾邮件识别指南》（模型）。

它一条条对照新邮件的特征，去“手册”里查：

规则手册说：“标题带‘免费’ -> 极大概率是垃圾邮件。”

规则手册说：“发件人地址奇怪 -> 几乎肯定是垃圾邮件。”

规则手册说：“正文有‘点击链接’ + 很多感叹号 -> 高概率是垃圾邮件。”

# 分类模型核心



(3) 综合判断，贴上标签（预测）：

计算机综合所有匹配的规则，计算出一个总的判断。

根据“手册”里的经验（学到的模式），它得出结论：这封新邮件符合非常多的垃圾邮件特征。

预测结果： **✗** 垃圾邮件！

动作： 自动把这封邮件扔进“垃圾邮件箱”。

总结核心思想：

学习规则（训练）： 用大量带标签的例子（特征+正确答案） 教计算机，让它自己总结出特征和标签之间的关联规律，形成一本内部的 “分类规则手册”（模型）。

应用规则预测： 遇到新东西（新邮件）时：

提取它的特征。

拿特征去查“规则手册”。

根据手册里的规律，预测它最可能属于哪个类别（正常 or 垃圾），并贴上标签。

# 常见的分类模型



01 逻辑回归

02 支持向量机

03 决策树

04 随机森林

05 朴素贝叶斯

06 XGBoost

07 神经网络



# 举例引入



银行用逻辑回归判断你的信用卡交易是否是欺诈行为。

- (1) 输入特征：交易金额（突然的大额消费？）
- (2) 消费地点（和你平时活动区域差很远？）
- (3) 消费时间（半夜3点买珠宝？）
- (4) 消费频率（1分钟内连续多笔交易？）

处理过程：

逻辑回归会给每个特征分配“可疑程度权重”，计算这些特征的加权总和，通过“S型函数”把结果压缩到0-1之间（像概率一样）

输出结果：

接近0（比如0.1）：“正常交易”；接近1（比如0.9）：“可能是欺诈”

银行设个阈值（比如0.7），超过就冻结卡片确认

可以使用逻辑函数（S型曲线）把线性计算的结果“压扁”成概率值，就像用滤镜把强光调成柔和的亮度。

# 逻辑回归-定义

逻辑回归是一种经典的分类模型，尽管名字中有“回归”，但它主要用于二分类问题（比如判断是/否）。逻辑回归的核心思想是将线性回归的输出通过一个逻辑函数

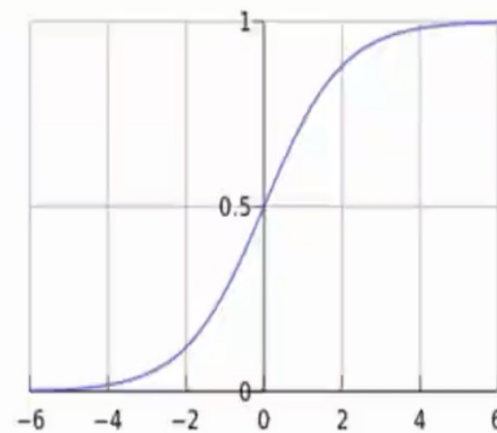
（sigmoid函数）转化为概率值。给定输入特征向量  $x = [x_1, x_2, \dots, x_n]$ ，逻辑回归模型可以表示为：
$$z = \beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \dots + \beta_n * x_n$$
这里， $\beta_0$ 是截距项， $\beta_1 \dots \beta_n$ 是特征对应的权重。然后通过 sigmoid 函数将  $z$  转化为概率：
$$h(x) = \sigma(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-\beta^T x}}$$
其中， $h(x)$  表示给定输入特征  $x$  预测为正类的概率。

✓ Sigmoid 函数

✎ 公式：
$$g(z) = \frac{1}{1+e^{-z}}$$

✎ 自变量取值为任意实数，值域 $[0,1]$

✎ 解释：将任意的输入映射到了 $[0,1]$ 区间  
我们在线性回归中可以得到一个预测值，再将该值映射到Sigmoid 函数中这样就完成了由值到概率的转换，也就是分类任务



CSDN @goTshgo



# sigmoid函数例子

单个特征的情况假设逻辑回归模型为:

1. 计算  $z$ :

$$z = 0.5 * 3 = 1.5$$

2. 代入 sigmoid 函数:  $\sigma(1.5) = 1 / (1 + e)^{-1.5}$

3. 计算  $e^{-1.5}$ :

$$e^{-1.5} \approx 0.2231 \text{ (使用自然常数 } e \approx 2.7183 \text{)}$$

4. 计算分母

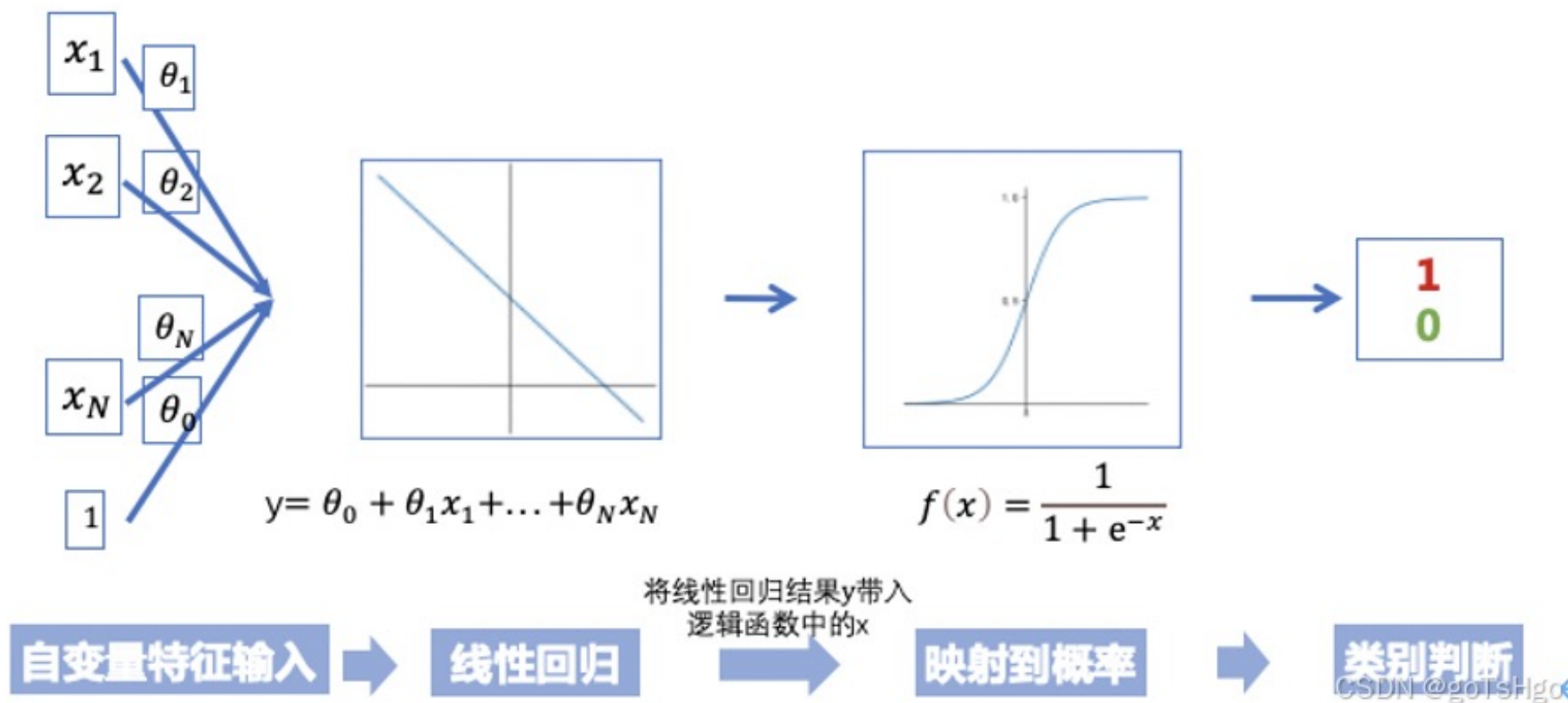
$$1 + 0.2231 = 1.2231$$

5. 计算最终概率

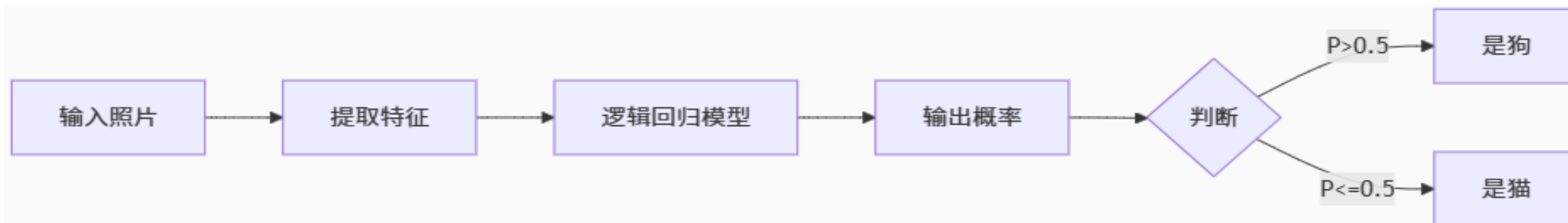
$$\sigma(1.5) = 1 / 1.2231 \approx 0.8175$$

当  $x = 3$  时, Sigmoid 函数输出约为 0.8175, 表示事件发生的概率为 81.75%。

# 整体流程



# 逻辑回归算法流程



1. 输入照片
2. 提取特征：从照片中筛选关键信息
3. 逻辑回归模型：基于历史标注数据（已分类的猫狗照片）学习 “特征→类别” 的映射规律；对新输入的 “提取特征”； 计算概率，输出照片属于 “狗” 的概率（0~1 之间的数值）
4. 输出概率：逻辑回归模型的计算结果，比如输出  $P=0.7$ ，代表这张照片是 “狗” 的概率为 70%；
5. 判断：若  $P > 0.5 \rightarrow$  判定为 “是狗”（概率更高，倾向狗的类别）；若  $P \leq 0.5 \rightarrow$  判定为 “是猫”（概率更低，倾向猫的分类）

# 由二分类到多分类

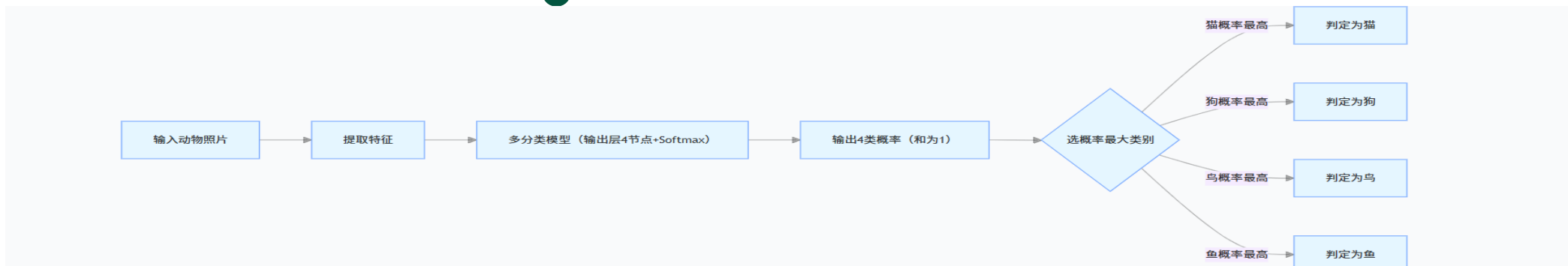


拓展到多分类：区分 “猫、狗、鸟、鱼”（动物识别）

现在要教模型区分 4 种动物，输出可能是：猫（1）、狗（2）、鸟（3）、鱼（4）

维度	二分类（猫 vs 狗）	多分类（猫、狗、鸟、鱼）
输出形式	1 个概率（ $0 \sim 1$ ，代表是猫的概率）	4 个概率（和为 1，分别代表 4 类概率）
激活函数	Sigmoid（压缩到 $0 \sim 1$ ）	Softmax（压缩到多分类概率，和为 1）
损失函数	二元交叉熵（BCELoss）	多分类交叉熵（CrossEntropyLoss）

# 多分类例子



数据准备：收集猫、狗、鸟、鱼的照片，分别打标签 1/2/3/4。

模型调整：输出层从 “1 个节点” 改成 “4 个节点”（对应 4 类），用 Softmax 激活，让输出 4 个概率和为 1。

训练逻辑：模型学习 “这 4 类动物的特征差异”（比如鸟有翅膀、鱼没脚…），损失函数衡量 “预测的 4 个概率” 和 “真实标签” 的差距，反向传播调参数。

预测判断：模型输出 4 个概率（比如猫 0.6、狗 0.2、鸟 0.1、鱼 0.1），选 概率最大的类别作为结果（这里判定为猫）。

# 举例引入



你拿着一个饮料瓶站在智能垃圾桶前，垃圾桶要判断这是“可回收塑料”还是“不可回收垃圾”。

收集特征：（1）瓶子的反光程度（塑料反光 vs 纸盒哑光）（2）敲击声音的频率（塑料清脆 vs 玻璃清脆）（3）红外线检测的材质特征

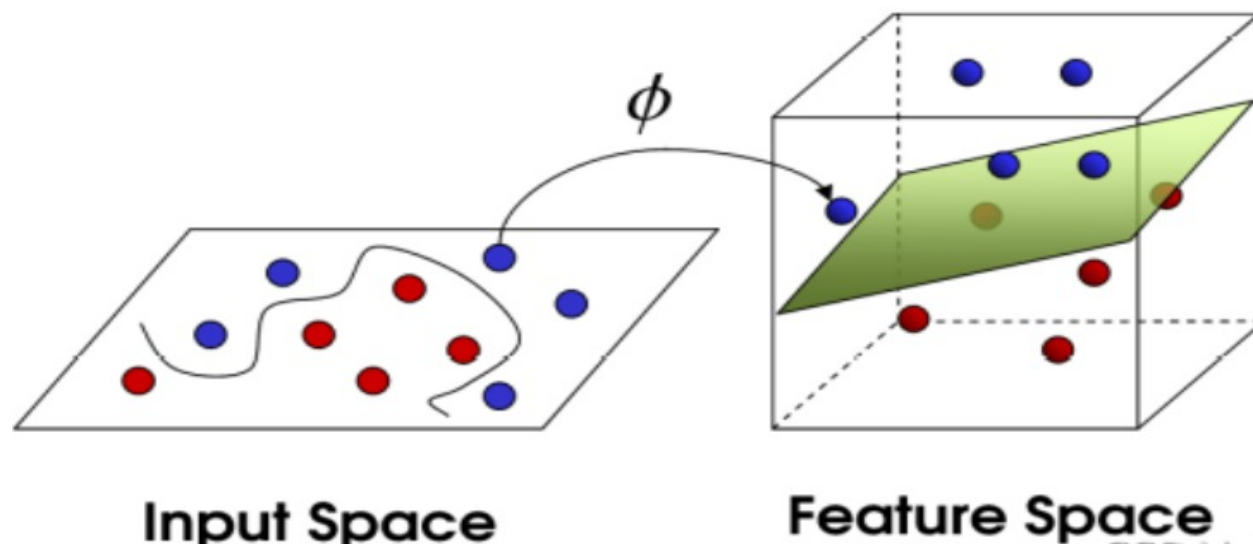
找最佳分界线：（1）想象把这些特征画在图上，塑料瓶和纸盒会形成两团点（2）找到一条最宽的“隔离带”分开这两类垃圾（3）特别关注那些容易混淆的样本（比如哑光塑料瓶）

做出判断：新垃圾的特征落在隔离带哪边，就属于哪类

给出分类结果：“这是可回收塑料，请投入蓝色垃圾桶”

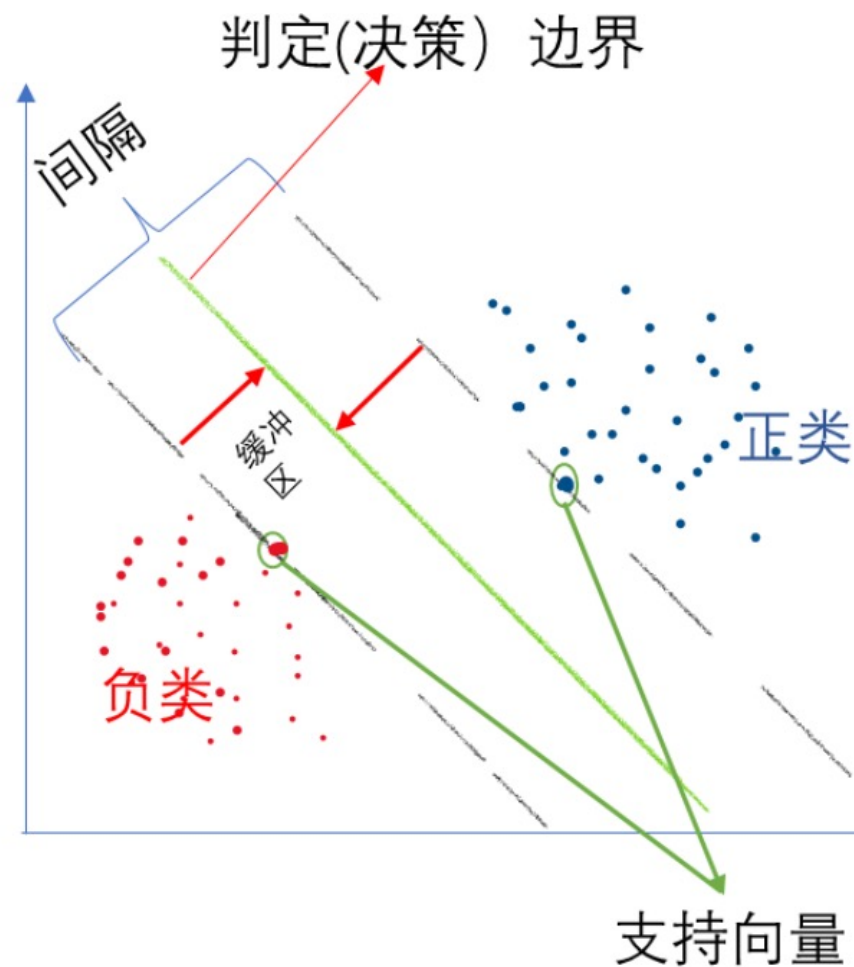
# 支持向量机-定义

支持向量机是一种非常常用的分类模型，其目标在于找到一个“超平面”将不同类别的数据分开。



# 支持向量机-详解

以一个二维平面为例，判定边界是一个超平面（在本图中其实是一条线，但是可以将它想象为一个平面乃至更高维形式在二维平面的映射），它是由支持向量所确定的（支持向量是离判定边界最近的样本点，它们决定了判定边界的位置）。间隔的正中就是判定边界，间隔距离体现了两类数据的差异大小

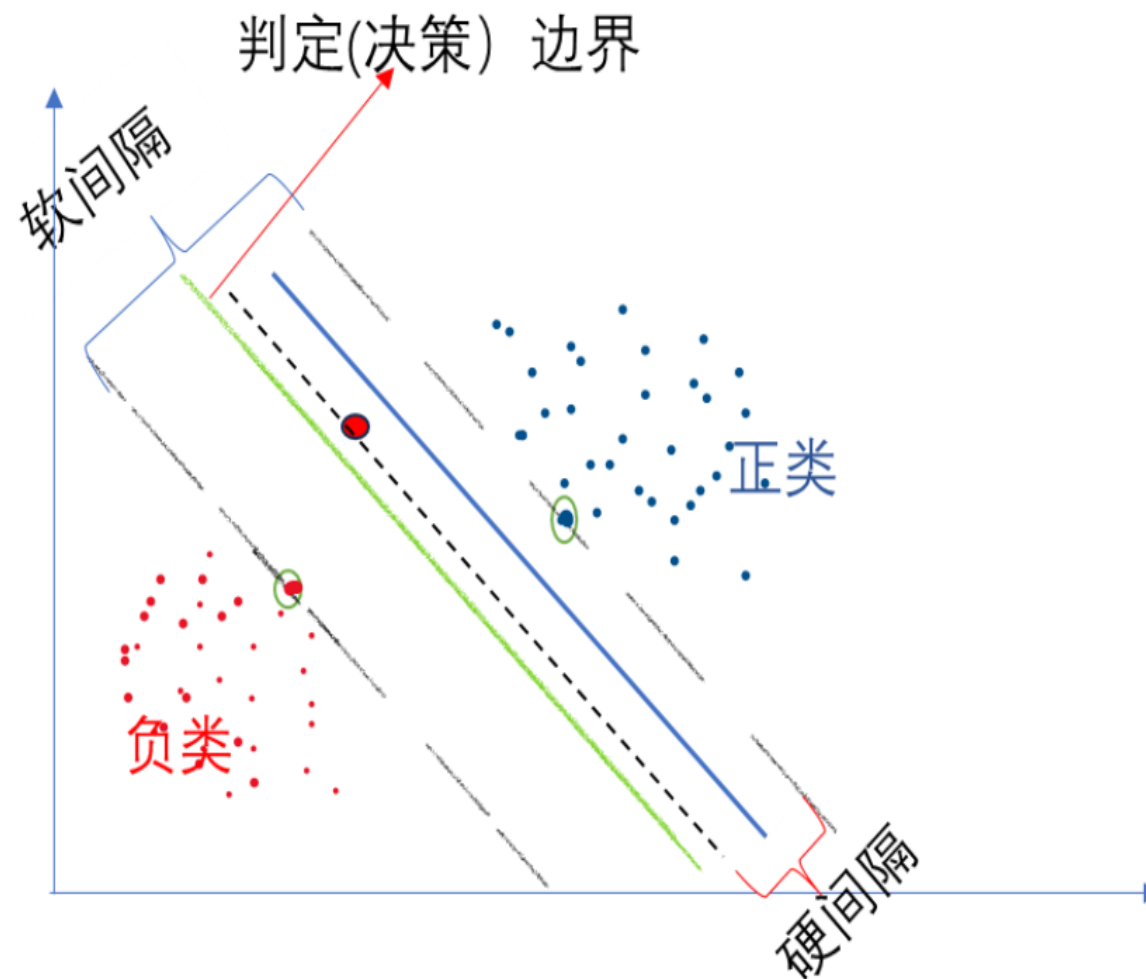




# 支持向量机-详解

若严格地规定所有的样本点都不在“缓冲区”，都正确的在两边，称为硬间隔分类；但是在一般情况下，不易实现，这里有两个问题：第一，它只对线性可分的数据起作用。第二，有异常值的干扰。

为了避免这些问题，可使用软间隔分类：在保持“缓冲区”尽可能大和避免间隔违规之间找到一个良好的平衡，在sklearn中的SVM类，可以使用超参数  $C$  (惩罚系数)，控制了模型的复杂度和容错能力。



# 举例引入

你的运动APP帮你决定今天要不要去跑步

决策树如何帮你做决定：

第一个问题：现在在下雨吗？（1）是 → “今天别跑了”（直接结束）（2）否 → 继续问...

第二个问题：空气质量指数(AQI)大于150吗？（1）是 → “今天不适合户外运动”（结束）  
（2）否 → 继续问...

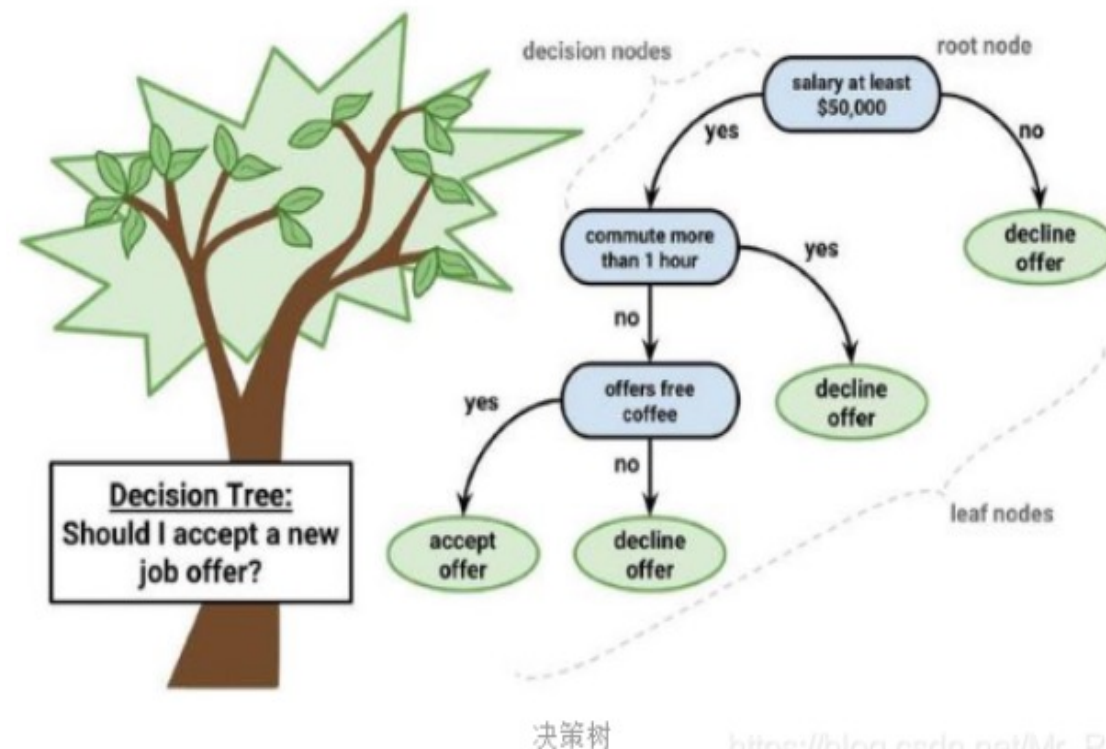
第三个问题：气温在15-28度之间吗？（1）是 → “完美！快去跑步吧！”（2）否 → “今天天气太极端，改室内运动吧”

像玩“20个问题”游戏，通过一系列是/否问题得出结论

每个判断点都把问题简单二分

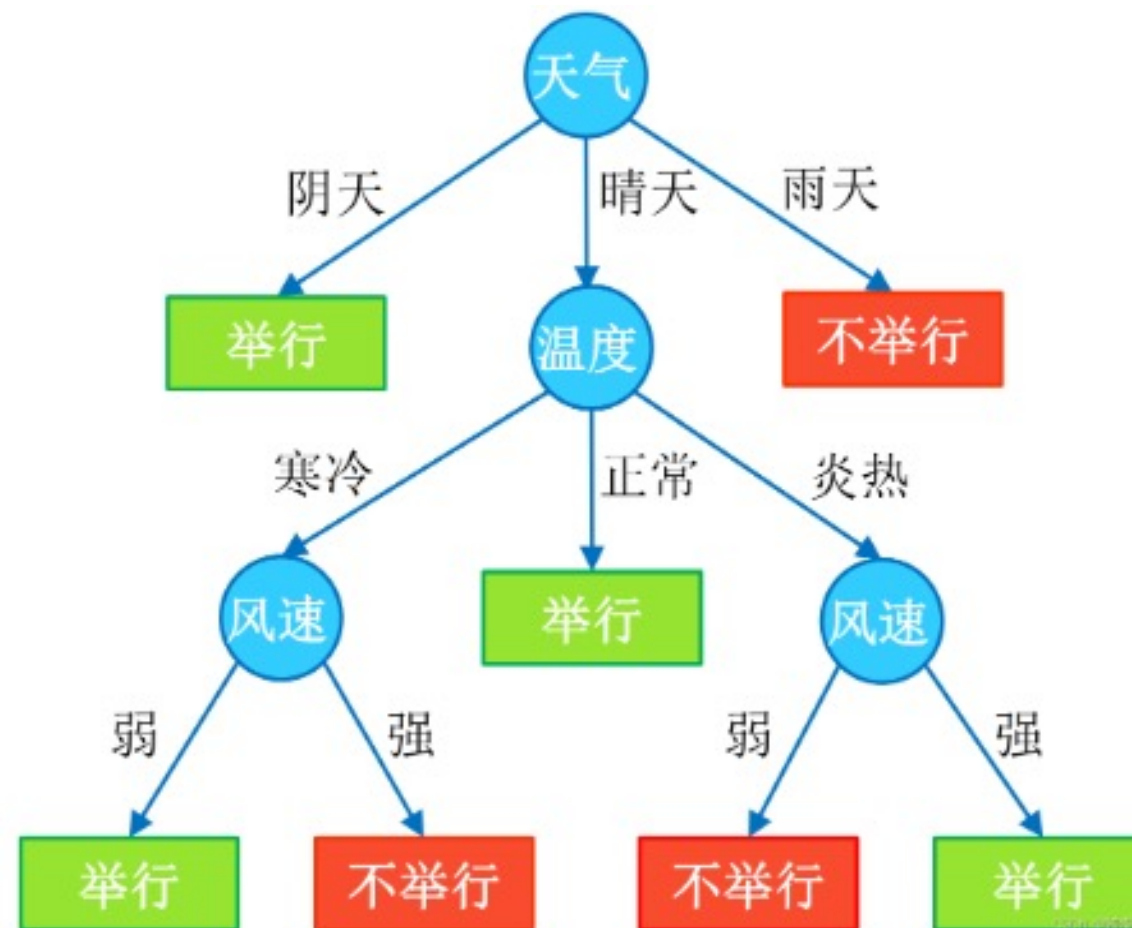
# 决策树-定义

决策树（Decision Tree），又称判断树，它是一种以树形数据结构来展示决策规则和分类结果的模型，作为一种归纳学习算法，其重点是将看似无序、杂乱的已知实例，通过某种技术手段将它们转化成可以预测未知实例的树状模型，每一条从根结点（对最终分类结果贡献最大的属性）到叶子结点（最终分类结果）的路径都代表一条决策的规则。



# 决策树-详解

由于这种决策分支画成图形很像一棵树的枝干，故称决策树。它的运行机制非常通俗易懂，因此被誉为机器学习中，最“友好”的算法。下面通过一个简单的例子来阐述它的执行流程。假设根据大量数据（含 3 个指标：天气、温度、风速）构建了一棵“可预测学校会不会举办运动会”的决策树（如右图所示）。



# 决策树-详解

接下来，当我们拿到某个数据时，就能做出对应预测。在对任意数据进行预测时，都需要从决策树的根结点开始，一步步走到叶子结点（执行决策的过程）。如，对下表中的第一条数据（[ 阴天，寒冷，强 ]）：首先从根结点出发，判断“天气”取值，而该数据的“天气”属性取值为“阴天”，从决策树可知，此时可直接输出决策结果为“举行”。这时，无论其他属性取值为什么，都不需要再执行任何决策（类似于“短路”现象）。

天气	温度	风速	预测结果
阴天	寒冷	强	举行
晴天	炎热	弱	不举行
晴天	寒冷	弱	举行
雨天	正常	弱	不举行

# 举例引入



多位专家会诊判断患者是否患有糖尿病

组建专家团队：（1）邀请10位不同的医生（就像10棵决策树）（2）每位医生关注不同的指标组

合：王医生重点看血糖和年龄；李医生关注BMI和家族史；张医生特别注意血压和妊娠次数

独立诊断：每位医生根据自己的经验给出判断（是/否）；有的医生可能误判，但没关系

民主投票：统计10位医生的诊断结果，7位说“是”，3位说“否” → 最终诊断“患有糖尿病”

避免“偏科专家”（单一决策树容易过度关注某些特征）

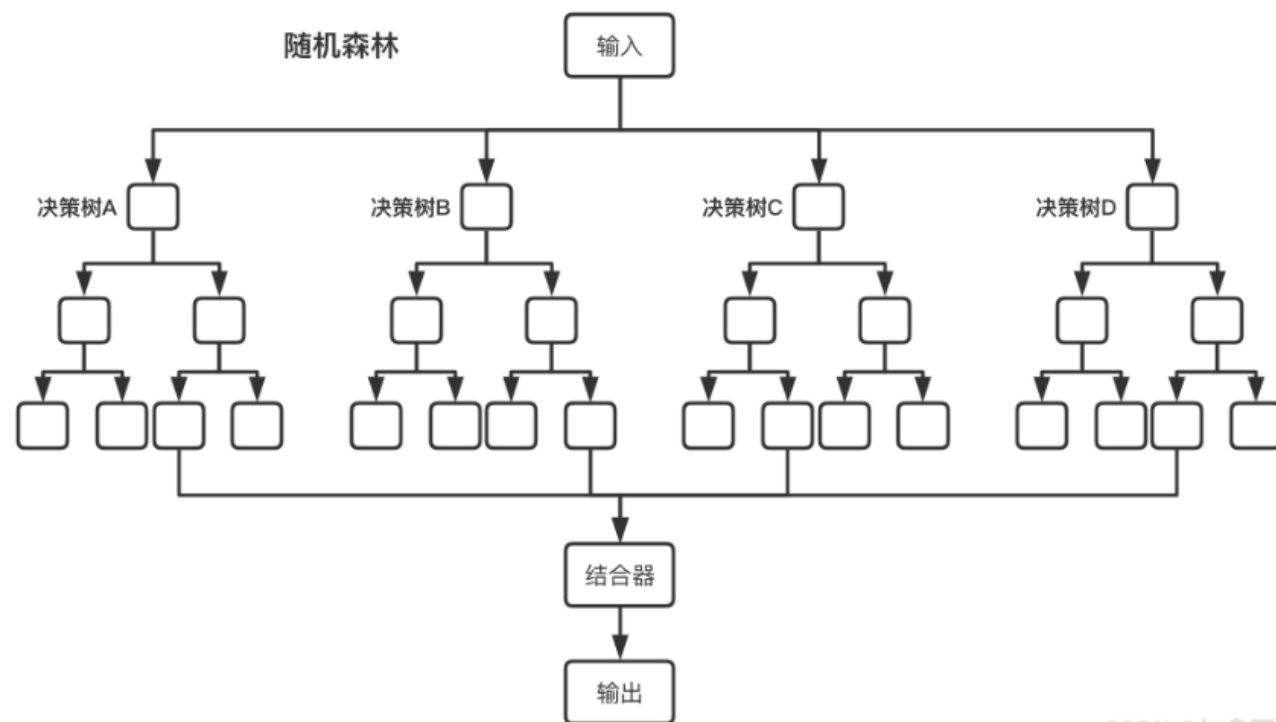
综合多角度判断更可靠

即使个别医生判断失误，整体结果仍然准确



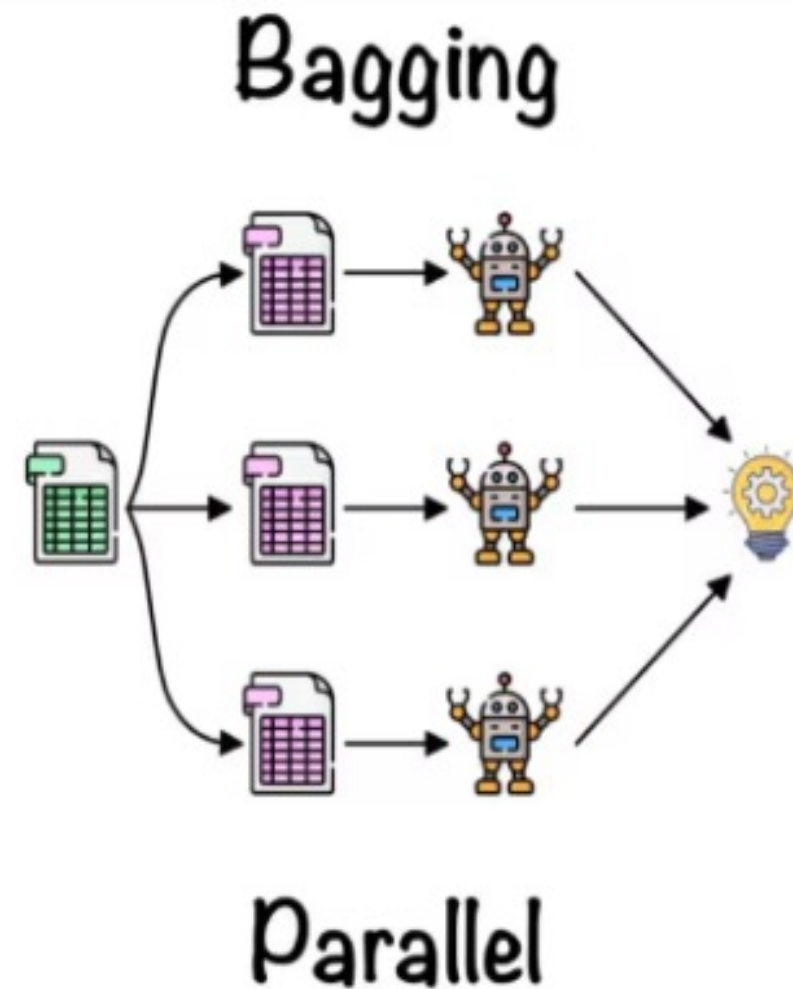
# 随机森林-定义

随机森林是一种集成学习模型，由多个决策树组成。每棵决策树对数据进行分类，最终通过投票或平均的方式得出最终结果。随机森林的优点是准确性高，抗过拟合能力强，但训练时间较长。



# 随机森林-核心思想

将若干个弱分类器的分类结果进行投票选择，从而组成一个强分类器，这就是随机森林 bagging 的思想。这种方法的核心思想是“众人拾柴火焰高”，即多个模型的预测结果通过某种方式结合起来，往往比单一模型的预测结果更加准确和稳定。





# 举例引入

你的电子邮箱自动判断“促销邮件”还是“重要邮件”

学习阶段：先看1000封已知分类的邮件（500封促销/500封重要）

统计关键词出现的概率：“优惠”在促销邮件出现概率：85%；“会议”在重要邮件出现概率：72%

收到新邮件时：发现邮件包含：“优惠”、“限时”、“折扣”

计算： $*是促销邮件的概率 = 85\% \times 90\% \times 80\% = 61.2\%$

$*是重要邮件的概率 = 5\% \times 2\% \times 3\% = 0.003\%$

比较后判断：这肯定是促销邮件！

简单地假设每个关键词之间互不影响（虽然实际上“限时”和“折扣”经常一起出现）

就像天真的小朋友认为“红色”和“圆形”完全没关系

## 朴素贝叶斯

朴素贝叶斯是基于贝叶斯定理和特征条件独立假设的算法，比如，判断一封邮件是否是垃圾邮件，朴素贝叶斯会计算邮件中出现某些关键词的概率，并根据这些概率做出预测。它的优点是计算速度快，适合处理高维数据（如文本分类），但对特征独立性假设较强，可能影响准确性。

## 贝叶斯

贝叶斯公式又被称为贝叶斯规则，是概率统计中的应用所观察到的现象对有关概率分布的主观判断（先验概率）进行修正的标准方法。如果你看到一个人总是做一些好事，则那个人多半会是一个好人。这就是说，当你不能准确知悉一个事物的本质时，你可以依靠与事物特定本质相关的事件出现的多少去判断其本质属性的概率。用数学语言表达就是：支持某项属性的事件发生得愈多，则该属性成立的可能性就愈大。

# 条件概率与后验概率

**条件概率：**记事件A发生的概率为 $P(A)$ ，事件B发生的概率为 $P(B)$ ，则在B事件发生的前提下，A事件发生的概率即为条件概率，记为 $P(A|B)$ 。如左图：

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

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

**后验概率：**在贝叶斯统计中，后验概率是在考虑新信息之后事件发生的修正或更新概率。后验概率通过使用贝叶斯定理更新先验概率来计算。用统计术语来说，后验概率是假设事件B已经发生的情况下事件A发生的概率。如右图：

# 贝叶斯公式

贝叶斯公式（Bayes' theorem）：也称为贝叶斯规则或贝叶斯定理，是概率论中的一个基本定理，描述了在已知先验概率的情况下，如何更新我们对事件的概率。

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

$P(A)$  这是概率中最基本的符号，表示  $A$  出现的概率。

$P(B|A)$  是条件概率的符号，表示事件  $A$  发生的条件下，事件  $B$  发生的概率，条件概率是“贝叶斯公式”的关键所在，它也被称为“似然度”。

$P(A|B)$  是条件概率的符号，表示事件  $B$  发生的条件下，事件  $A$  发生的概率，这个计算结果也被称为“后验概率”。

$P(B)$  是边缘概率，表示观测到 $B$ 的概率。

# 朴素贝叶斯公式

朴素贝叶斯分类器的原理基于贝叶斯定理，即根据已知类别的数据来估计特征与类别之间的概率分布，然后使用这些概率来对新样本进行分类。

朴素贝叶斯算法是假设各个特征之间相互独立，也是朴素这词的意思。那么贝叶斯公式中 $P(X|Y)$ 可写成：

$$P(X|Y) = P(x_1|Y) * P(x_2|Y) * P(x_3|Y) \cdots * P(x_i|Y)$$

具体地，设特征向量为 $X=(X_1, X_2, \dots, X_i)$ ，类别集合为 $Y=(Y_1, Y_2, \dots, Y_i)$ ，我们的目标是计算在给定特征向量 $X$ 的条件下，属于每个类别的概率 $P(x_i|Y)$ ，然后选择具有最大后验概率的类别作为样本的分类结果。

朴素贝叶斯公式：

$$P(Y|X) = \frac{P(x_1|Y)*P(x_2|Y)*P(x_3|Y)\cdots*P(x_i|Y)*P(Y)}{P(X)}$$

其中 $P(Y)$ 是类别 $Y$ 的先验概率， $P(x_i|Y)$ 是在类别 $Y$ 下特征向量 $X$ 的条件概率， $P(X)$ 是特征向量 $X$ 的边缘概率。

由于 $P(X)$ 对所有类别都是相同的，因此可以忽略掉。

# 举例引入



淘宝预测你下次会买什么商品

组建超级侦探团队：（1）不像随机森林的“普通医生”，XGBoost是“明星侦探小组”（2）每个侦探（决策树）都特别擅长发现不同线索：侦探A专注你的浏览历史；侦探B研究你的购物车变化；侦探C分析你最近搜索关键词

接力破案模式：第一轮：侦探A找出80%的线索 第二轮：侦探B针对剩下的20%线索深入调查 第三轮：侦探C查漏补缺，纠正前两位的小失误，经过多轮“破案”，预测越来越精准

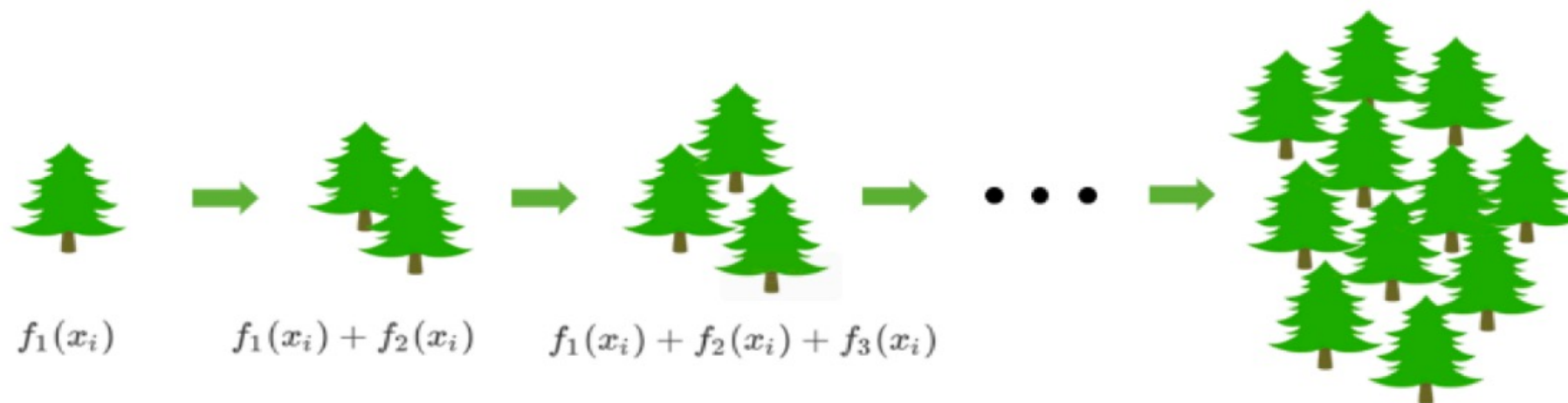
智能奖惩机制：

预测正确的侦探获得“奖金”（权重增加）；常出错的侦探被“扣钱”（权重降低）；新加入的侦探都专门针对之前没解决好的案例

像学霸刷题：会重点练习错题（关注预测错误的样本）像团队协作：后加入的成员专门补强薄弱环节 自带“防过拟合”机制：不会因为太关注细节而失去大局观

# XGBoost-定义

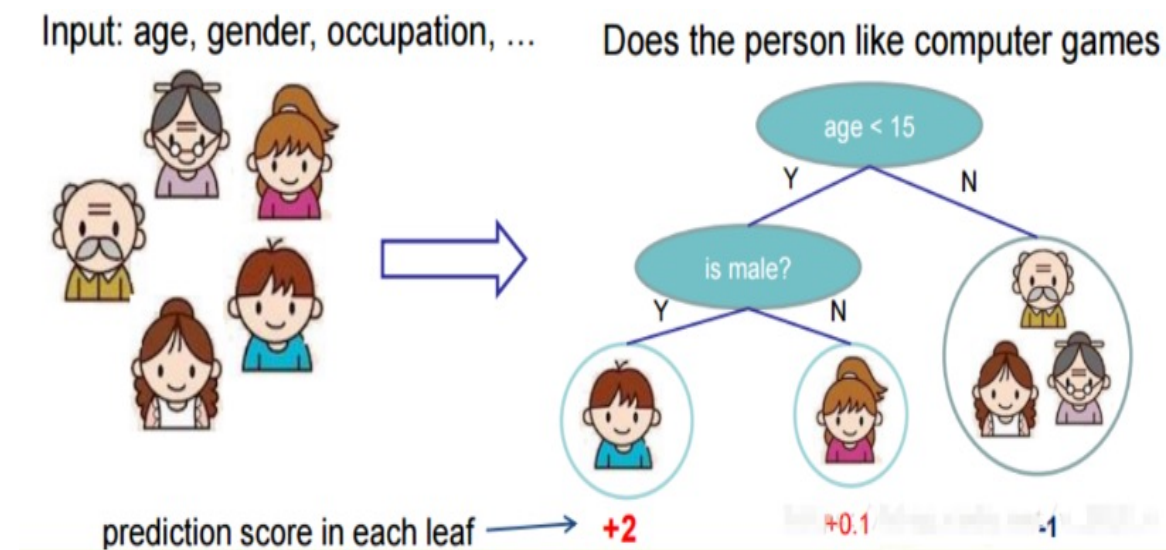
XGBoost 是一种基于梯度提升树的集成学习模型，通过逐步添加树模型来拟合残差（即 模型预测值与真实值之间的差异）。它的优点是预测精度高，训练速度快，广泛应用于各种 机器学习竞赛和实际场景。





# XGBoost-详解

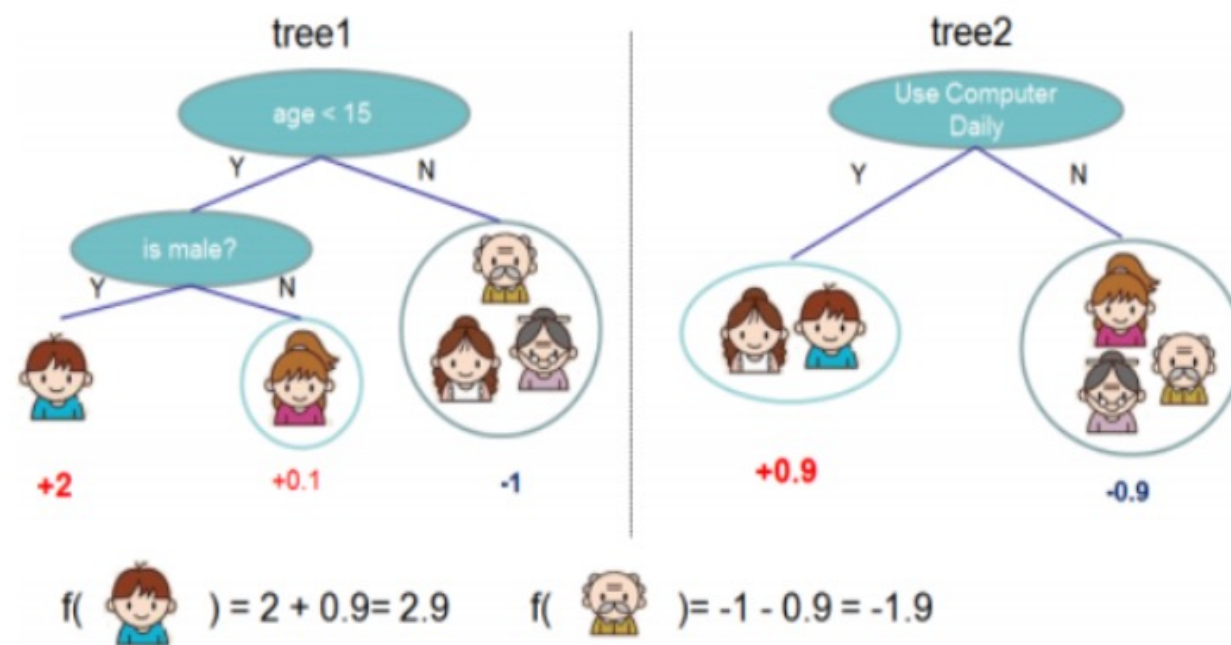
先来举个例子，我们要预测一家人对电子游戏的喜好程度，考虑到年轻和年老相比，年轻更可能喜欢电子游戏，以及男性和女性相比，男性更喜欢电子游戏，故先根据年龄大小区分小孩和大人，然后再通过性别区分开是男是女，逐一给各人在电子游戏喜好程度上打分，如右图所示。





# XGBoost-详解

就这样，训练出了2棵树tree1和tree2，两棵树的结论累加起来便是最终的结论，所以小孩的预测分数就是两棵树中小孩所落到的结点的分数相加： $2+0.9=2.9$ 。爷爷的预测分数同理： $-1+(-0.9)=-1.9$ 。具体如右图所示：



# 举例引入



你的手机相册能自动识别照片里的内容，把“猫”、“食物”、“风景”分开存放。

输入：一张照片（对神经网络来说，其实就是一堆数字，代表每个像素的颜色）

处理过程（像多层次的“猜谜游戏”）：（1）第一层：识别基础元素（比如边缘、颜色块）“这张图左下角有一些弯曲的线条，可能是圆形？”

（2）中间层：组合特征（把线条组合成物体部分）“弯曲线条+毛茸茸的纹理+三角形耳朵=可能是猫头”

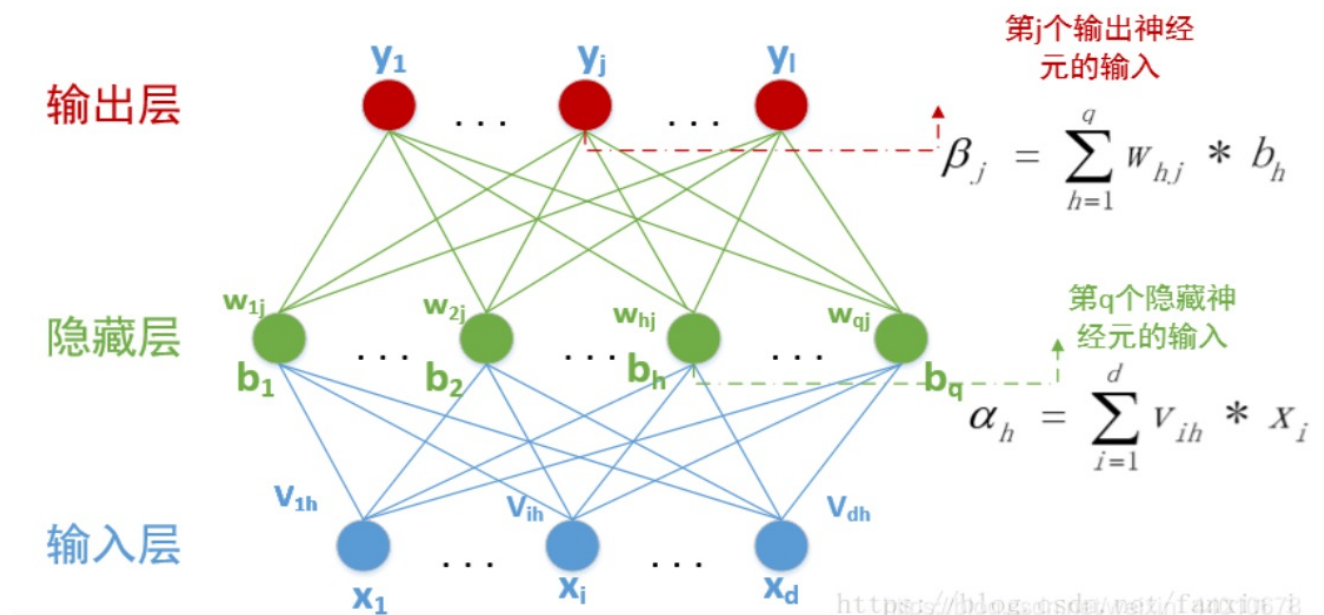
（3）输出层：综合判断“有猫头+胡须+尾巴=这是猫的照片！”

输出：给照片打上“猫”的标签，自动归类到宠物相册

像这种通过多层处理的分类模型称作-神经网络

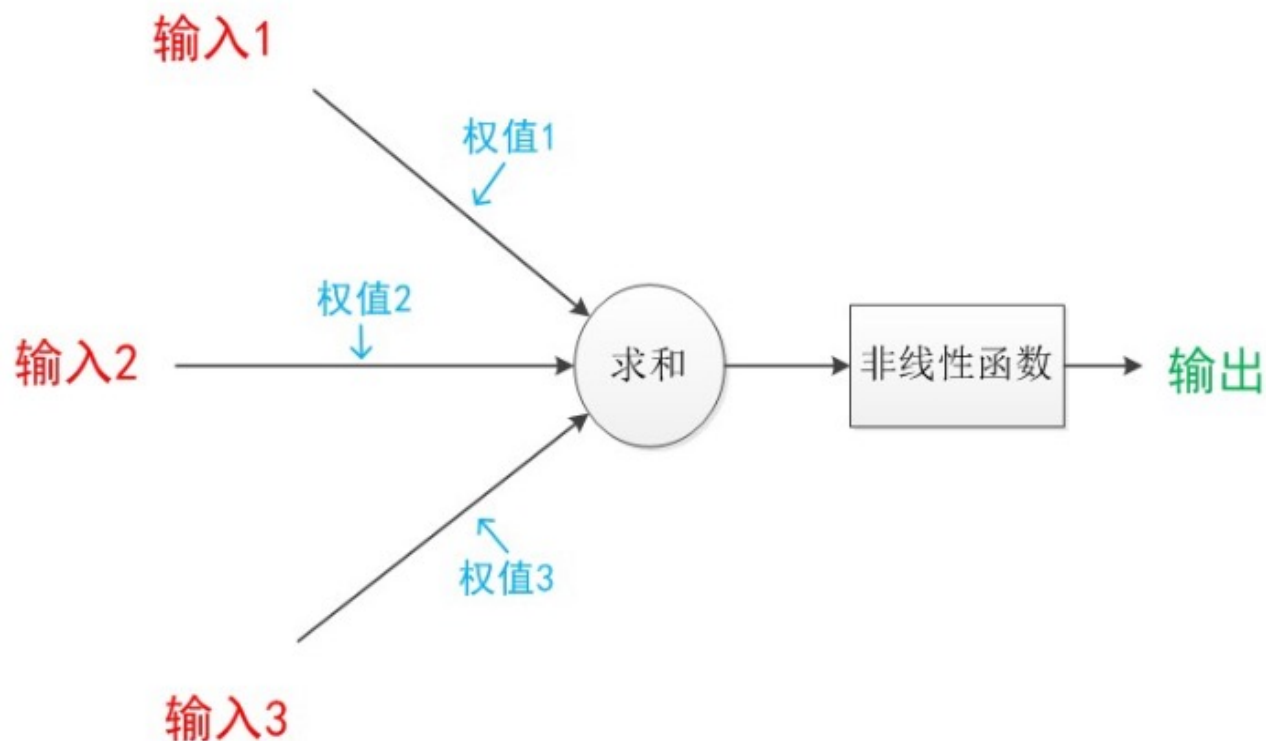
# 神经网络-定义

神经网络是一种模拟人脑神经元工作方式的分类模型，它由多个“层”组成，每一层包含多个“神经元”，这些神经元通过权重和激活函数连接起来



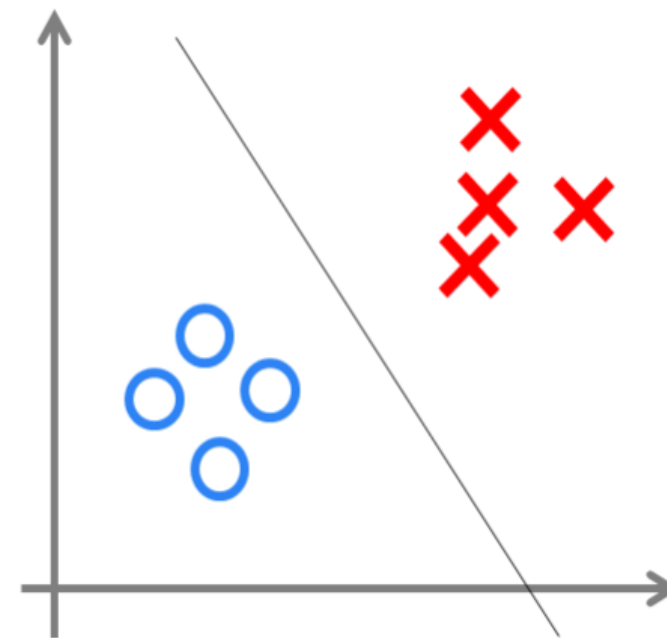
# 神经元

神经元模型是一个包含输入，输出与计算功能的模型。输入可以类比为神经元的树突，而输出可以类比为神经元的轴突，计算则可以类比为细胞核。



# 神经网络优点

神经网络的优点在于可以处理非常复杂的非线性关系，适合高维数据（如图像、文本）；但往往需要较为大量的数据和计算资源，训练时间较长，且模型的可解释性较差。第 2.4 节将进一步介绍从传统神经网络到深度神经网络（即深度学习）的发展与应用。



单层神经网络（决策分界）



# 03

## 聚类模型

# 举例引入



沃尔玛用聚类分析找出不同类型的购物者

收集购物特征：（1）每周购物频率（每天都来 vs 周末突击采购）（2）购物篮商品组合（生鲜占比 vs 零食占比）（3）消费金额区间（小额高频 vs 大额低频）（4）优惠券使用习惯

自动发现相似群体：（1）算法发现500个特征相似的家庭主妇：1. 特征：工作日下午采购、买大量生鲜、常用优惠券 2. 标签：“精打细算型主妇”（2）识别出200个相似的年轻白领：1. 特征：周末晚上购物、进口食品多、几乎不用优惠券 2. 标签：“品质生活型白领”

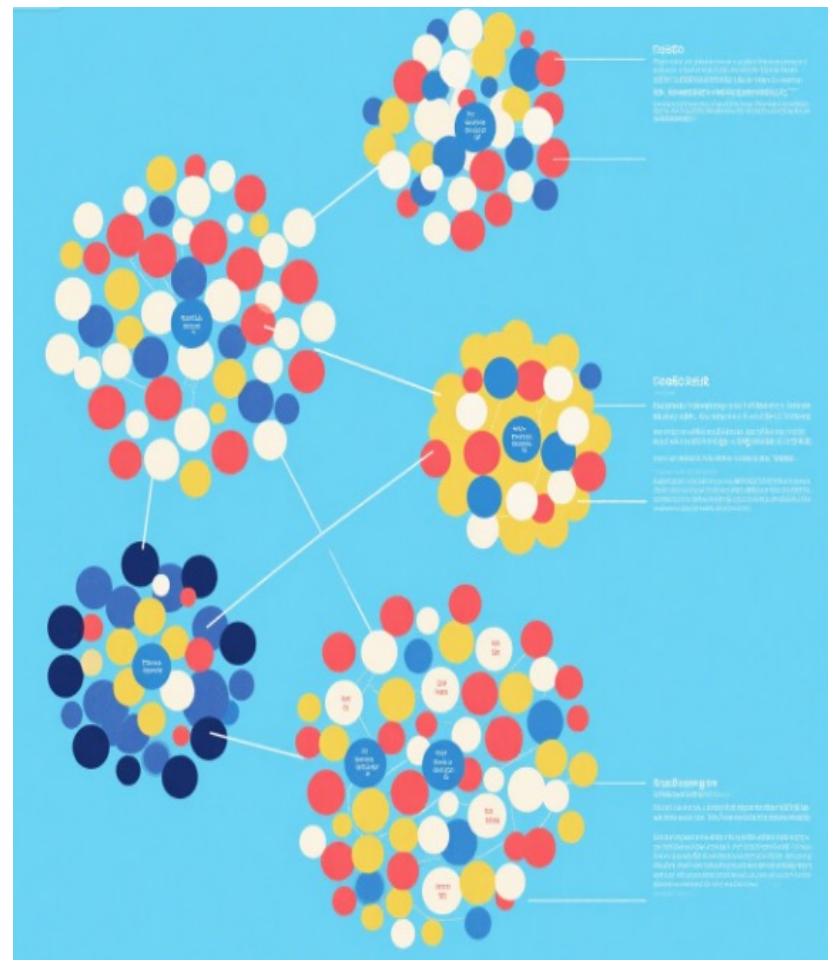
制定精准策略：（1）给“精打细算型”推送折扣信息（2）为“品质生活型”推荐新品试吃（3）对“周末囤货型”提供大包装商品

就像自动把散落的弹珠按颜色/大小分成不同罐子；不需要预先知道有哪些类型（与分类模型的关键区别）；完全基于数据本身的相似性来分组



# 聚类模型的定义

聚类（Clustering）模型的目标是在无监督（即没有真实标签）的情况下，将一组数据样本划分为不同的（Clusters），以使得每个簇内的数据尽可能相似，而不同簇之间的数据差异较大。





# 聚类模型的核心



核心任务：让计算机自动发现“隐藏的自然分组”！

关键问题：我们没有预先告诉计算机任何“正确答案”（比如谁是“高价值客户”、谁是“家庭主妇”），但计算机能不能自己在一堆数据里发现“物以类聚，人以群分”的规律？

解决方案：让计算机自己观察数据点之间的相似度，把相似的聚在一起，不相似的分开！这就是聚类！

第一步：学习分组规则（模型训练）—— 发现“朋友圈”

（1）准备“观察对象”（只有特征，没有标签！）：

假设超市有成千上万名顾客的购物记录。

我们给计算机看每个顾客的特征（描述他们的数据点）：1. 购物频率：每周来几次？（1次？5次？）2. 平均消费金额：每次花多少钱？（50元？500元？）3. 常购商品类型：买很多母婴用品？爱买进口零食？只买打折商品？经常买生鲜？4. 购物时间：喜欢早上来？还是晚上来？周末来？

重要区别：这里没有预先告诉计算机任何顾客的“类别标签”（比如“家庭主妇”、“上班族”、“学生党”）！计算机完全不知道“正确答案”是什么。

# 聚类模型的核心



## （2）计算机的“观察”原则（核心规则）：

计算机只有一个核心任务：计算顾客之间的“相似度”。

规则手册的核心条款：

“长得像的，放一起！”：如果两个顾客的特征非常接近（比如：都每周来5次、都花500元左右、都爱买进口零食、都周末下午来），那他们很可能属于同一个“朋友圈”（簇）。

“差别大的，分开站！”：如果两个顾客的特征差别很大（比如：一个每周来1次花50元只买打折品，一个每周来5次花500元买进口零食），那他们肯定不属于同一个“朋友圈”。

计算机怎么判断“像不像”？它内部会用数学方法（比如计算距离）来衡量特征之间的接近程度。简单理解就是：各方面习惯越接近的顾客，越相似。

# 聚类模型的核心



(3) 计算机自己“找朋友”（发现分组）：

计算机开始扫描所有顾客的数据点。

它不断地：

找相似： 把那些特征高度相似的顾客拉拢到一起，初步形成一个“小圈子”。

分不同： 确保特征差异大的顾客待在各自的圈子里，不强行拉到一起。

调圈子： 看看圈子里的成员是不是都足够相似？圈子之间是不是足够不同？不断调整优化，目标是让圈子内部成员尽可能相似，同时不同圈子之间成员尽可能不同。

经过一番计算和调整，计算机最终找到了几个比较稳定的“顾客朋友圈”。

# 聚类模型的核心



## （4）形成“分组结果”（聚类模型）：

计算机的成果是：给每个顾客打上一个“组号”（簇标签），比如 Customer A 属于“组1”，Customer B 属于“组2”...

更重要的是，它定义了每个“组”的特征：

组1（高频高消美食家）： 特征：每周来4-5次，平均消费>400元，常购进口零食/精品生鲜，多在周末下午。  
(这个组的人购物习惯很相似)

组2（精打细算家庭客）： 特征：每周来2-3次，平均消费100-200元，常购打折日用品/粮油米面，多在工作日晚上。（这个组内部也相似，但和组1明显不同）

组3（偶尔应急上班族）： 特征：每周来<1次，平均消费<100元，常购速食/饮料/少量水果，时间不固定。  
(又一个不同的组)

这就是训练好的“聚类模型”！ 它的核心成果是：发现的分组结果 以及 每个组的特征画像。

# 聚类模型的核心



第二步：使用规则进行“归类”（模型应用）—— 新顾客找“朋友圈”

现在，超市来了一位新顾客，计算机怎么知道TA可能属于哪个组？

（1）观察新顾客的特征：

计算机提取这位新顾客的购物特征：1. 购物频率：每周来4次 2. 平均消费：每次约450元 3. 常购商品：经常买进口牛排、精酿啤酒、有机蔬菜 4. 购物时间：主要在周六下午（和训练时看的特征类型完全一样）

（2）查阅“分组画像”（对比组特征）：

计算机拿出它之前发现的那几个“顾客朋友圈”的特征画像（组1、组2、组3...的描述）。

它把新顾客的特征，挨个去和每个组的典型特征画像做比较：

比较组1（高频高消美食家）画像： 每周4-5次， >400元，进口零食/精品生鲜，周末下午。 哇！新顾客特征和这个画像高度匹配！

比较组2（精打细算家庭客）画像： 每周2-3次，100-200元，打折日用品，工作日晚上。 嗯... 频率、金额、商品类型、时间都对不上。

# 聚类模型的核心



(3) 找到最相似的组（预测归属）：

计算机判断：新顾客的特征和“组1”的特征画像最相似！

预测结果： 这位新顾客很可能属于“高频高消美食家”组（组1）。

总结核心思想：

学习分组规则（训练 - 无监督）：

输入： 只有特征数据，没有预设标签/答案！

核心规则： 计算数据点之间的相似度，把相似的聚在一起（簇内相似度高），把不相似的分开（簇间差异大）。

输出： 发现隐藏的自然分组（簇）以及每个组的特征画像（模型结果）。

应用规则归类（预测）：

输入： 一个新数据点的特征。

过程： 将这个新点的特征，与训练得到的各个组的特征画像进行相似度比较。

输出： 预测新点属于哪个已发现的组（簇标签），即找到它最相似的“朋友圈”

# 常见的聚类模型

---



01 K均值聚类

02 谱聚类

03 层次聚类

04 DBSCAN

# 举例引入



外卖用K-Means算法优化骑手配送区域划分

数据准备阶段：（1）收集城市所有外卖订单的配送位置（就像在地图上撒豆子）（2）每个订单用经纬度坐标表示（如[116.404, 39.915]）

聚类过程三步走：（1）第一步：随机选3个点作为初始“中心配送站”（比如西单、国贸、中关村）

（2）第二步：把每个订单分配给最近的配送站 1. 朝阳门的订单→国贸站 2. 北大附近的订单→中关村站（3）第三步：重新计算每个站的“中心位置”（像调整基站位置）

重复这个过程直到配送站位置不再明显移动

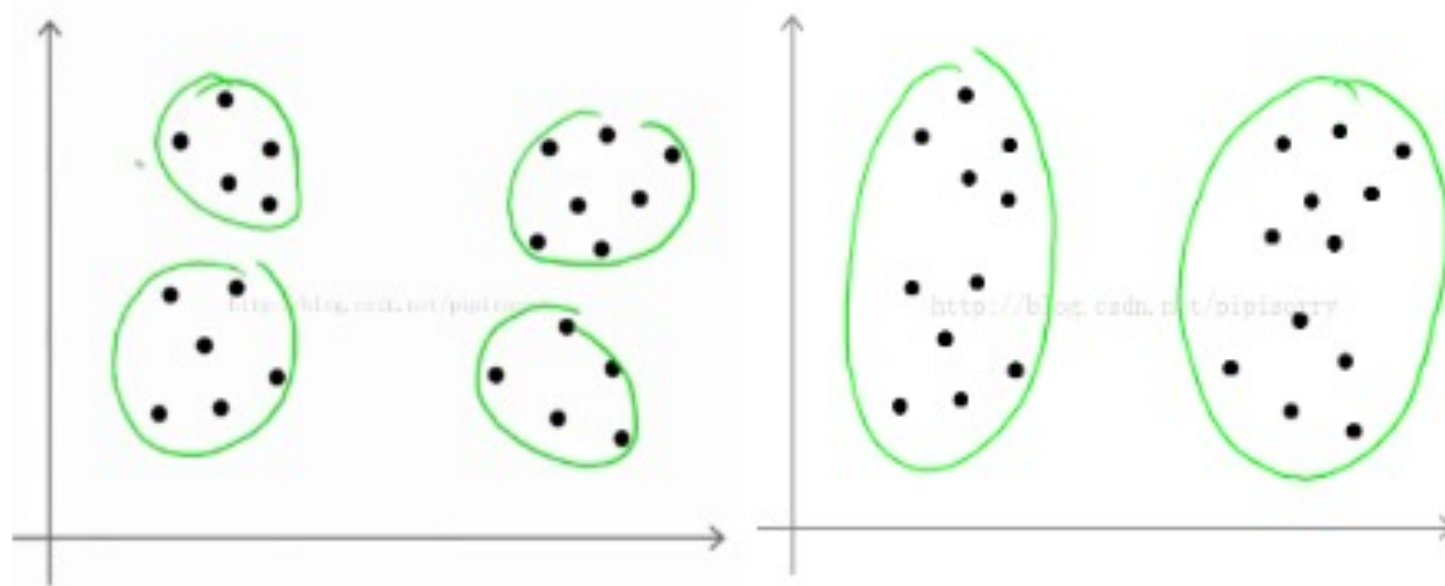
最终结果：（1）自动划分出最优的5个配送区域（2）每个区域有明确的中心点和覆盖范围（3）骑手数量按区域订单量智能分配

像玩“套圈游戏”：不断调整圈的位置，让每个圈罩住的点最紧凑；自动发现城市外卖热力中心（不依赖人工）；能随订单分布变化动态调整（每月重新计算一次）



# K均值聚类-定义

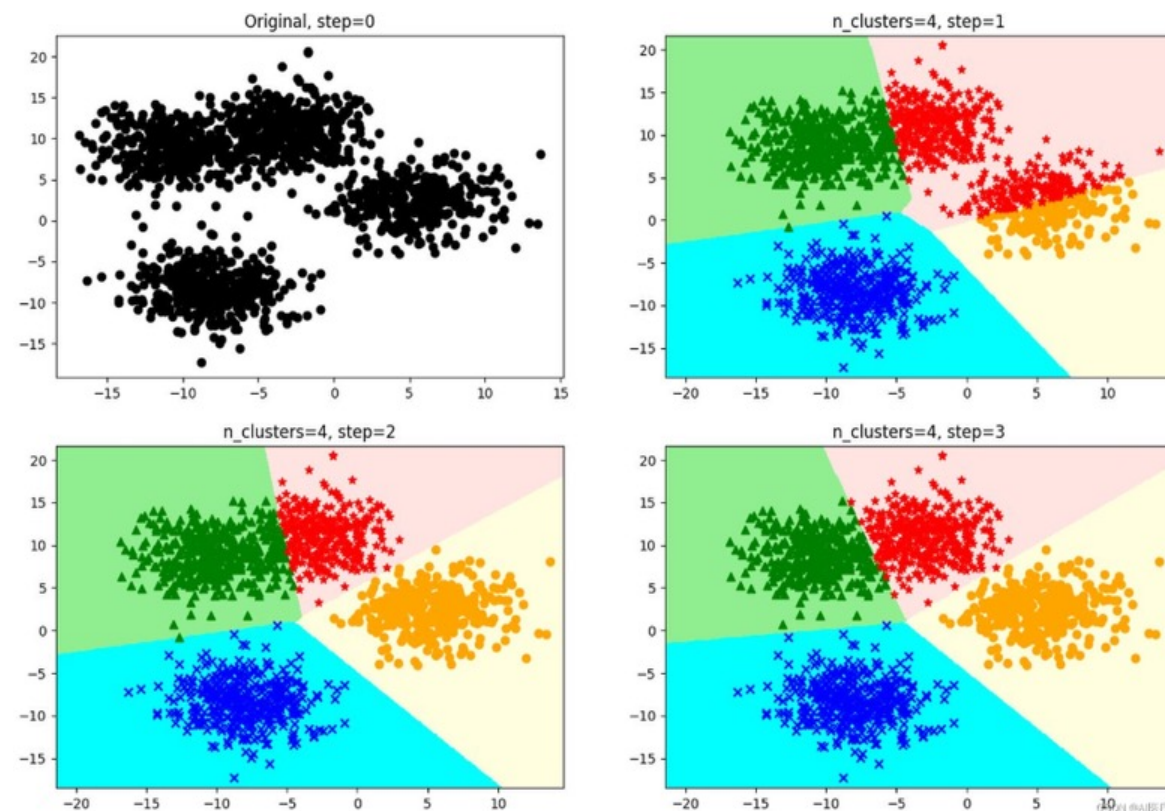
K-means 是一种经典的聚类法，它将数据划分为  $K$  个簇，每个簇的中心由簇内数据的平均值决定。K-means 的优点是简单高效，但需要预先指定簇的数量  $K$ 。



# K均值聚类-详解

K-Means聚类算法步骤实质是EM算法（最大期望算（Expectation-Maximization algorithm, EM））的模型优化过程，具体步骤如下：

- （1）随机选择 $k$ 个样本作为初始簇类的均值向量；
- （2）将每个样本数据集划分离它距离最近的簇；
- （3）根据每个样本所属的簇，更新簇类的均值向量；
- （4）重复（2）（3）步，当达到设置的迭代次数或簇类的均值向量不再改变时，模型构建完成，输出聚类算法结果



# 欧式距离

在数学中，欧氏距离或欧几里德度量是欧几里得空间中两点之间的“普通”直线距离。通过这个距离，欧几里德空间成为度量空间。通常，对于n维空间来说，欧几里得距离可以表示为，如

右图所示：
$$d(\mathbf{p}, \mathbf{q}) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}$$

比如在一个二维平面中，有数据点 A 和聚类中心 B，通过欧氏距离公式可以计算出它们之间的距离。距离越小，说明数据点 A 与聚类中心 B 在特征空间中的位置越接近。

# K均值聚类解决问题



针对问题（2）如果给你500个陌生人的购物小票，如何自动发现哪些人购物习惯相似？给出一下K均值算法解决方法：初始化聚类中心：

1. 随机从 500 个样本中选  $K$  个点，作为初始的  $K$  个簇中心（比如  $K=4$ ，就随机挑 4 个人的购物特征当中心）。
2. 分配样本到簇：对每个陌生人的购物特征样本，计算其与  $K$  个簇中心的欧氏距离（K-Means 默认常用欧氏距离），把样本分配到距离最近的簇中心所属的簇。
3. 更新簇中心：对每个簇，重新计算其所有样本特征的均值（如消费金额均值、各类商品购买占比均值等），作为新的簇中心。
4. 迭代优化：重复“分配样本→更新簇中心”步骤，直到簇中心变化小于设定阈值（如中心坐标变化小于 0.001），或达到最大迭代次数（如迭代 100 次），算法停止。

# 举例引入



微信“你可能认识的人”推荐背后的谱聚类

构建好友关系网：（1）把每个用户看作一个点（2）用户之间的聊天频率、共同好友数作为“连接强度”（3）画出整个社交网络的关系图（像星座连线图）

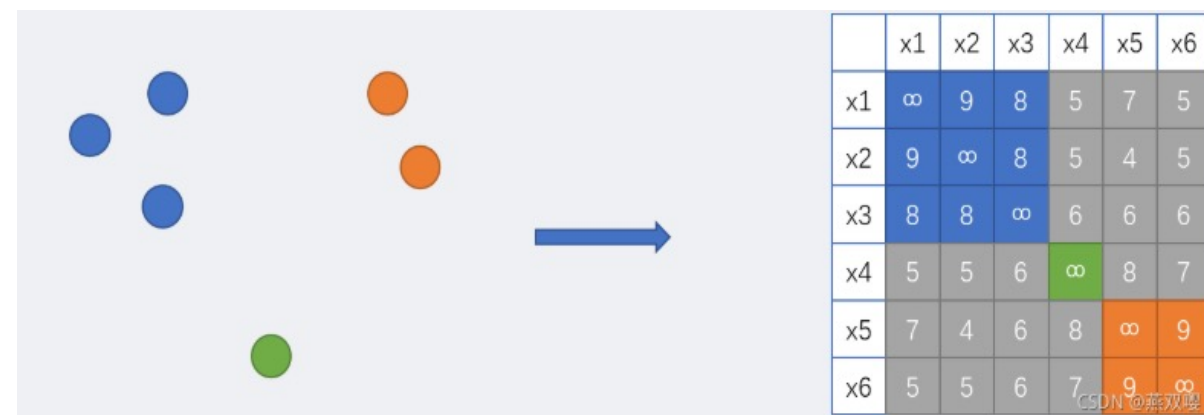
关键洞察：（1）发现三个明显“抱团”的群体：（2）大学同学群：互相之间联系紧密，但很少联系外人（3）公司同事群：内部频繁互动，形成独立模块（4）健身俱乐部群：自成一个圈子

切割关系网：（1）不是简单按距离划分，而是找到“连接最稀疏处”下刀，像精准解开一团乱麻，沿着最脆弱的地方分开

谱聚类像按交友关系重组兴趣小组、像用化学方法分解材料

# 谱聚类-定义

谱聚类是一种基于图论的聚类方法，它通过数据的相似性矩阵来构建图结构，并利用图的谱（即图的特征值和特征向量）进行聚类。谱聚类的核心思想是将数据点看作图中的节点，数据点之间的相似性看作边的权重，然后通过对图的拉普拉斯矩阵进行特征分解来划分数据，其显著优势在于具备复杂非线性数据的聚类能力。

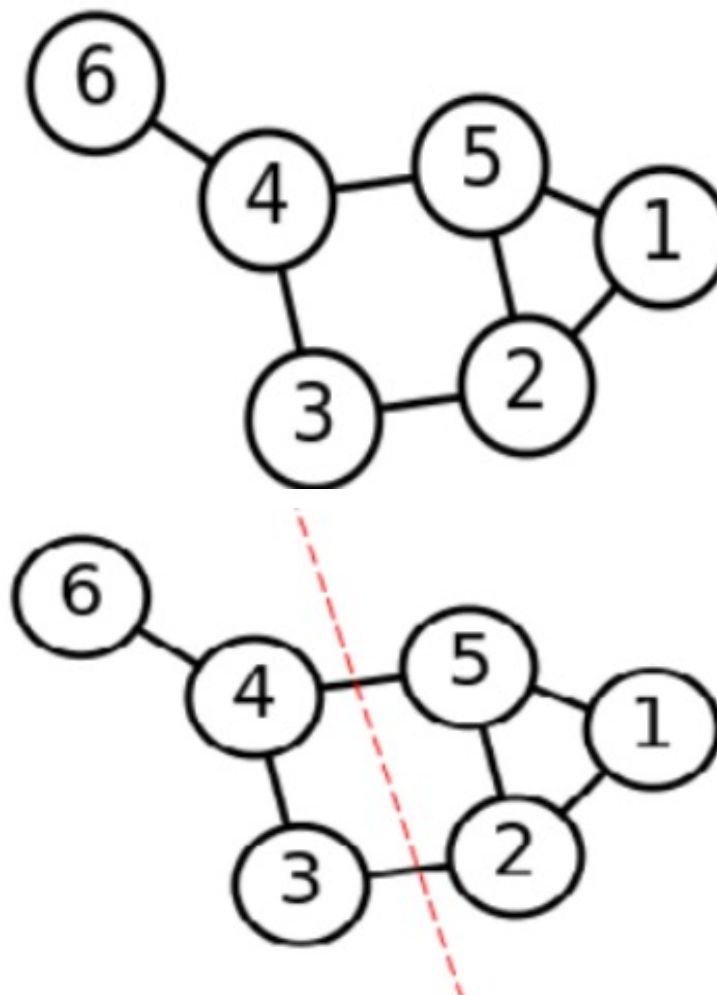




# 谱聚类-详解

谱聚类过程主要有两步，第一步是构图，将采样点数据构造成一张网图，表示为 $G(V, E)$ ， $V$ 表示图中的点， $E$ 表示点与点之间的边，如右上图：

第二步是切图，即将第一步构造出来的按照一定的切边准则，切分成不同的图，而不同的子图，即我们对应的聚类结果，举例如右下图：



# 拉普拉斯矩阵

拉普拉斯矩阵 (Laplacian matrix)，也称为基尔霍夫矩阵，是表示图的一种矩阵。给定一个有 $n$ 个顶点的图，其拉普拉斯矩阵被定义为： $L = D - W$ （其中  $D$  为图的度矩阵， $W$ 为图的邻接矩阵）；

以前图为例， $W$  如右上图所示；

把的每一列元素加起来得到个数，然后把它们放在对角线上（其它地方都是零），组成一个 $N \times N$  对角矩阵，记为度矩阵  $D$ ，如右下图所示：

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$



# 拉普拉斯矩阵

根据拉普拉斯矩阵的定义  $L = D - W$ ，可得拉普拉斯矩阵  $L$  为：

$$\begin{pmatrix} 2 & -1 & 0 & 0 & -1 & 0 \\ -1 & 3 & -1 & 0 & -1 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & -1 & 0 & -1 & 3 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{pmatrix}$$

# 举例引入



某集团公司用层次聚类重组全国200家门店

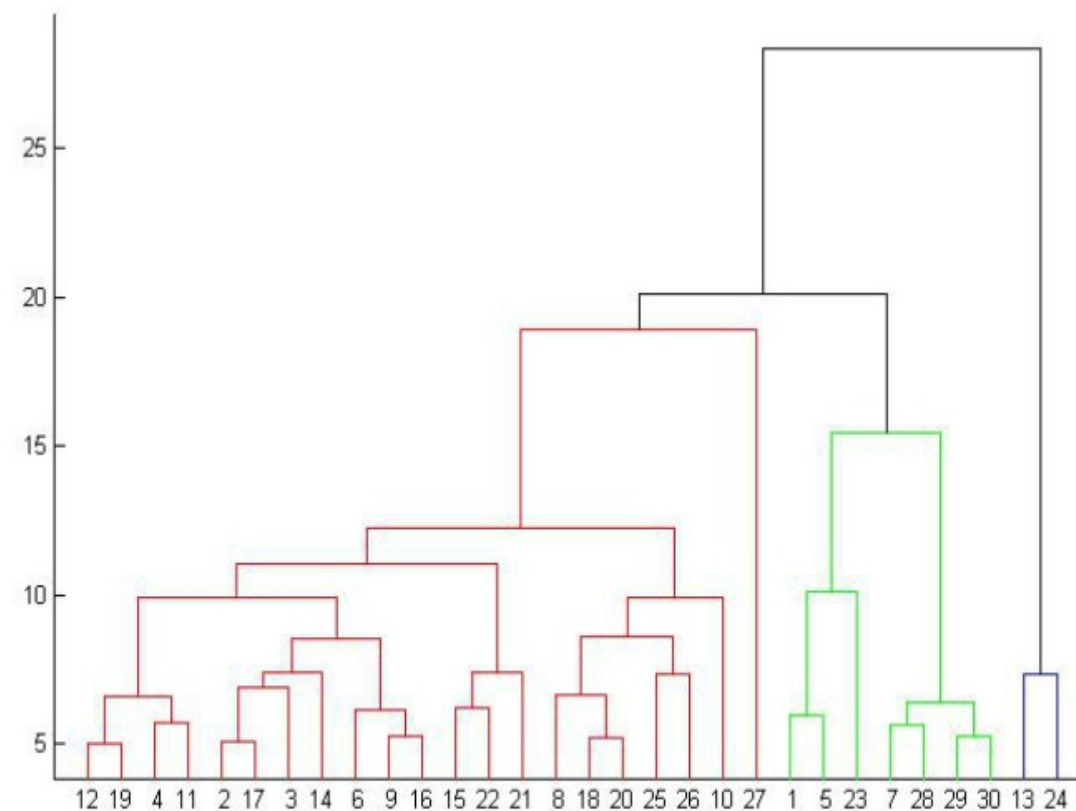
数据准备阶段：（1）收集每家门店的特征数据：（2）月营业额、客流量、面积大小（3）商品品类结构、会员复购率（4）所在城市消费水平

聚类过程（自底向上）：（1）第一步：每家门店先自成一组（200个微型群组）（2）第二步：找到最相似的两家门店合并（比如北京朝阳店和海淀店）（3）第三步：继续合并最相似的群组先合并同城市门店→再合并同省份门店→最后形成大区

生成聚类树状图：（1）像家族族谱一样展示合并过程（2）可以自由选择切割层级：  
切在高层→形成5个大区 切在中层→得到20个省级分部 切在底层→保留200家独立门店  
层次聚类像先同桌→再小组→最后分班级；层次聚类像绘制家谱树  
企业组织图就是天然的层次聚类结果

# 层次聚类-定义

层次聚类通过构建树状结构将数据逐步合并或分裂为不同的簇。层次聚类的优点是不需要预先指定簇的数量，但计算复杂度较高。

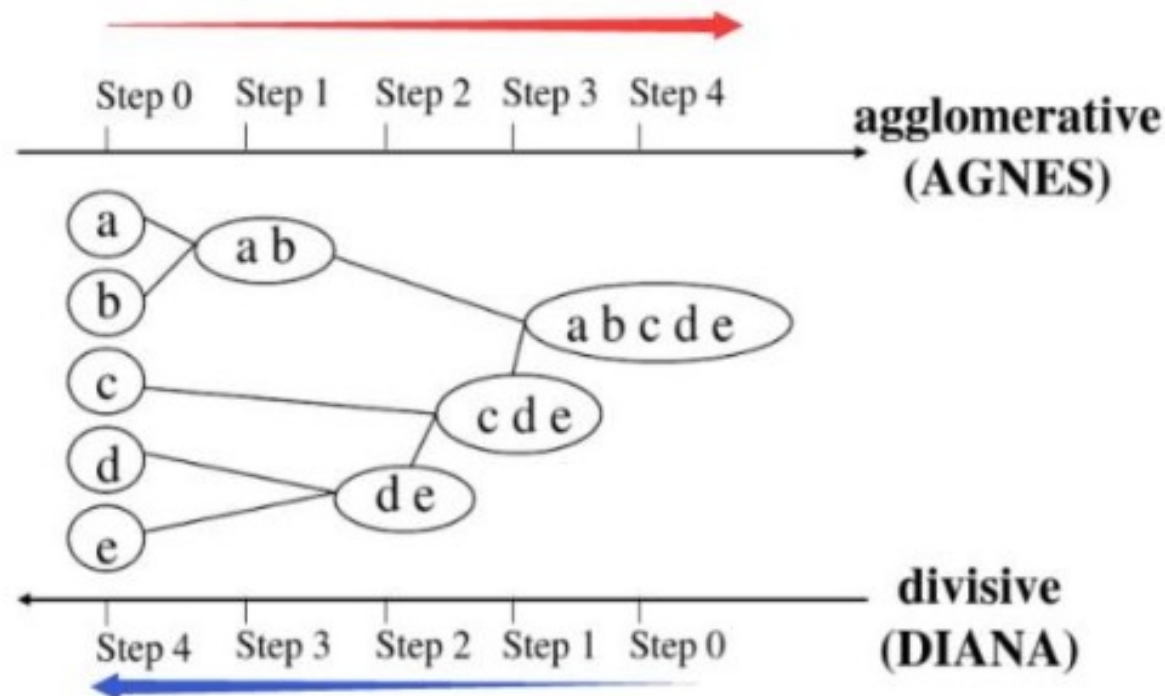


# 层次聚类-详解

简单的说层次聚类的合并算法是通过计算每一个类别的数据点与所有数据点之间的距离来确定它们之间的相似性，距离越小，相似度越高。并将距离最近的两个数据点或类别进行组合，生成聚类树。

在实际应用中一般有两种层次聚类方法：凝聚层次聚类方法&分裂层次聚类方法。

凝聚的(agglomerative)和分裂的(divisive)层次聚类图示



# 举例引入



滴滴出行用DBSCAN发现城市实时打车热点

数据观察：（1）把每个打车订单的定位点撒在城市地图上（2）有些地方点很密集（如商场/写字楼）（3）有些地方点很稀疏（如郊区公园）

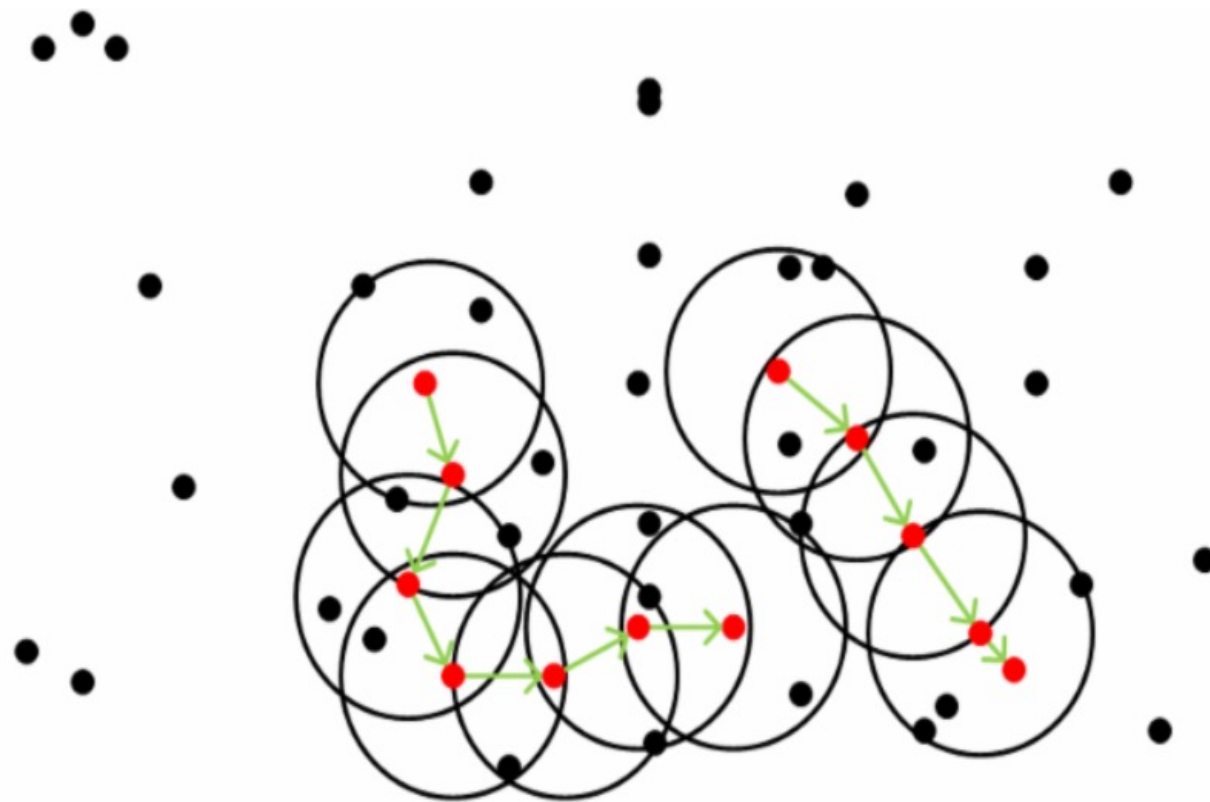
核心原理应用：（1）判断热点：如果一个地点周围500米内有至少15个打车点（核心点），就把这片区域标记为“打车热点”（2）边界处理：周边零星的点也划入热点（如商场侧门的几个订单）（3）噪声过滤：单独的几个偏远订单不划入任何区域（视为随机噪声）

动态结果：（1）晚高峰时国贸区域自动扩展成一个大热点（2）凌晨时段拆分成几个小热点（酒吧区、宵夜区）（3）居民区在工作日早晨自动浮现为临时热点

DBSCAN像根据鱼群密度智能下网

# DBSCAN-定义

DBSCAN 是一种基于密度的聚类算法，它将高密度区域的数据点划分为同一簇，并将低密度区域的数据点标记为噪声。DBSCAN 的优点是可以发现任意形状的簇，且对噪声数据不敏感。



# DBSCAN-详解



假设样本集是 $*D=(x_1, x_2, \dots, x_m)*$ , 则DBSCAN具体的密度描述如下:

- 1)  $\epsilon$ -邻域: 对于 $x_j \in D$ , 其 $\epsilon$ -邻域包含样本集 $D$ 中与 $x_j$ 的距离不大于 $\epsilon$ 的子样本集, 这个子样本集的个数记为 $|N_\epsilon(x_j)|$
- 2) 核心对象: 对于任一样本 $x_j \in D$ , 如果其 $\epsilon$ -邻域对应的 $N_\epsilon(x_j)$ 至少包含 $MinPts$ 个样本, 则 $x_j$ 是核心对象。
- 3) 密度直达: 如果 $x_i$ 位于 $x_j$ 的 $\epsilon$ -邻域中, 且 $x_j$ 是核心对象, 则称 $x_i$ 由 $x_j$ 密度直达。注意反之不一定成立, 即此时不能说 $x_j$ 由 $x_i$ 密度直达, 除非且 $x_i$ 也是核心对象。
- 4) 密度可达: 对于 $x_i$ 和 $x_j$ , 如果存在样本序列 $p_1, p_2, \dots, p_T$ , 满足 $p_1=x_i, p_T=x_j$ , 且 $p_{t+1}$ 由 $p_t$ 密度直达, 则称 $x_j$ 由 $x_i$ 密度可达。也就是说, 密度可达满足传递性。此时序列中的传递样本 $p_1, p_2, \dots, p_{T-1}$ 均为核心对象, 因为只有核心对象才能使其他样本密度直达。注意密度可达也不满足对称性, 这个可以由密度直达的不对称性得出。
- 5) 密度相连: 对于 $x_i$ 和 $x_j$ , 如果存在核心对象样本 $x_k$ , 使 $x_i$ 和 $x_j$ 均由 $x_k$ 密度可达, 则称 $x_i$ 和 $x_j$ 密度相连。注意密度相连关系是满足对称性的。



# DBSCAN-详解

右图基于密度的聚类中的密度可达和密度相  
连性

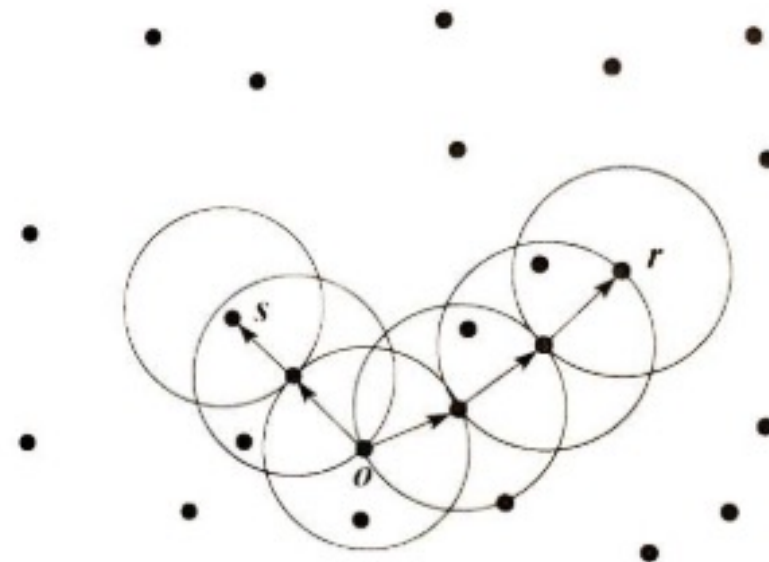
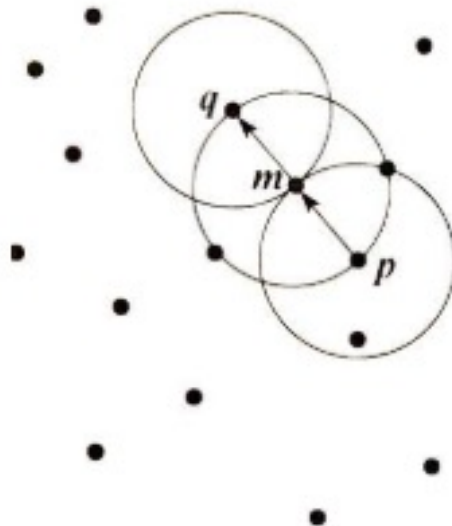
由右图可看出m, p, o, r 都是核心对象, 因为  
他们的内都只是包含3个对象。

1. 对象q是从m直接密度可达的。对象m从p直  
接密度可达的。

2. 对象q是从p（间接）密度可达的, 因为q从  
m直接密度可达, m从p直接密度可达。

3. r和s是从o密度可达的, 而o是从r密度可  
达的, 所有o, r和s都是密度相连的。

半径为 $\epsilon$ ,  $MinPts = 3$







# 04

## 回归模型

# 举例引入



回归模型预测北京二手房成交价

输入特征：（1）房子特征：面积（85m<sup>2</sup>）、房龄（10年）、楼层（中层）（2）位置特征：到地铁距离（500米）、学区质量（一类学区）（3）小区配套：绿化率（35%）、停车位比例（1:0.8）

模型计算：（1）每个特征都有影响权重：1.\*面积每增加1m<sup>2</sup> → 房价+5000元\* 2.\*房龄每增加1年 → 房价-3000元\* 3.\*一类学区 → 房价+30万\*

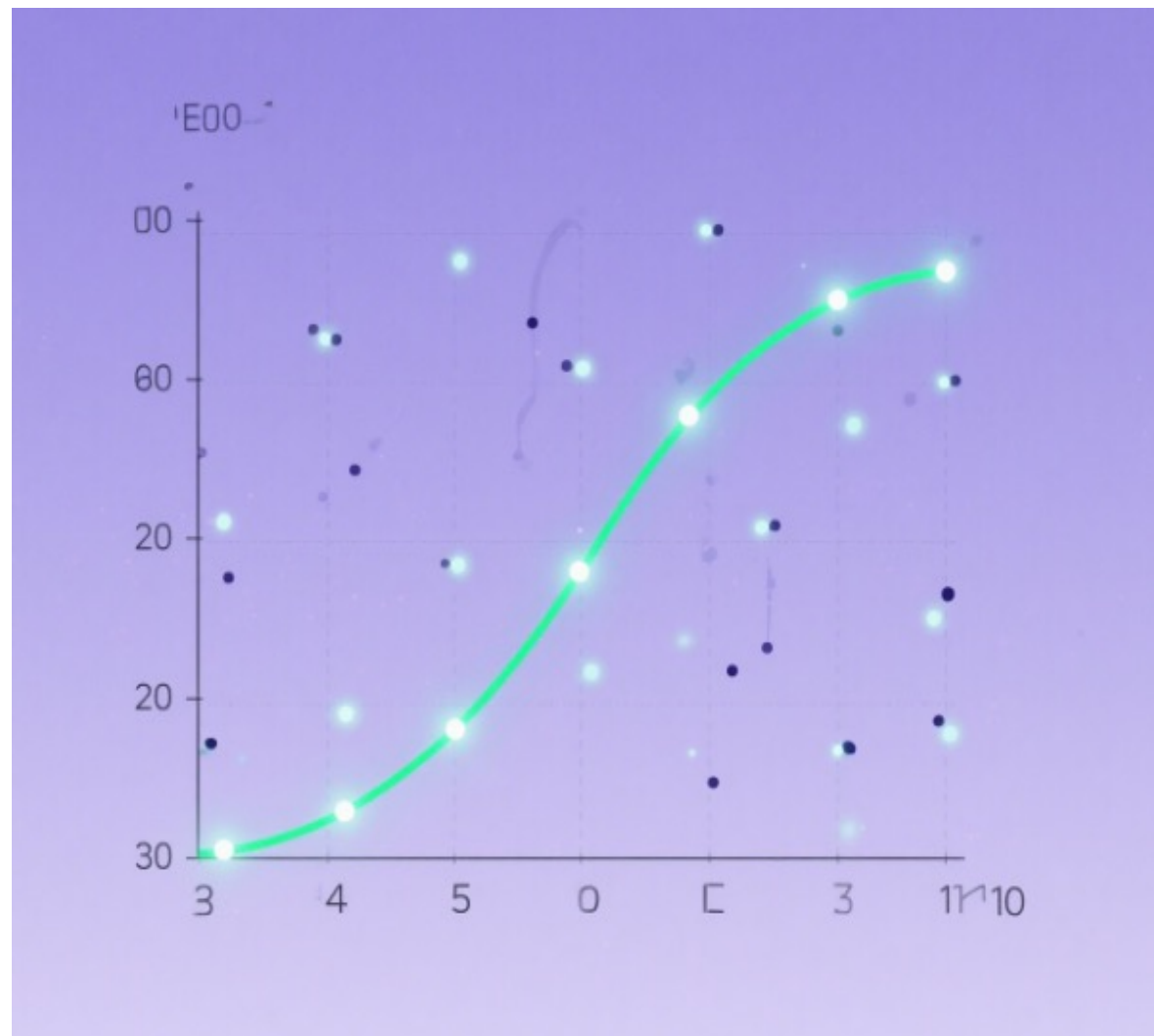
（2）综合计算： $85 \times 5000 - 10 \times 3000 + 30\text{万} + \dots = \text{预估总价}$

输出结果：这套房预估成交价：625万（实际成交620万），同时给出价格区间：610-640万

回归模型像计算器（直接输出数值结果），把模型比作老练的中介：看一眼房子就能估出合理价位，能说清为什么这样估价

# 回归模型的定义

回归模型（Regression Model）是一种用于预测或分析变量之间关系的统计与机器学习模型，核心目标是通过已知的输入数据（自变量）来预测一个或多个连续型的输出结果（因变量），或揭示自变量对因变量的影响规律。



# 回归模型的核心



核心任务：让计算机学会“估算”一个具体的数值！

关键问题：我们不是想让计算机分门别类（像分类），也不是想让它自动分组（像聚类），而是想让它根据一些信息预测一个具体的、连续的数字，比如：

根据房子的大小、地段、房龄... 预测这房子大概卖多少钱？

根据广告投入的金额... 预测产品大概能卖出去多少件？

根据学习时长、复习次数... 预测这次考试大概能考多少分？

解决方案：让计算机学习特征（X）和我们要预测的目标数值（Y）之间的关联趋势，然后根据这个趋势线来估算！这就是回归！

# 回归模型的核心



第一步：学习估算规则（模型训练）—— 找“最佳趋势线”

（1）准备“带答案的练习题”（监督学习）：

我们收集很多套房子的历史数据。

每套房子的数据包含两部分：

特征（X）： 1. 描述房子的信息。 2. 房子大小（平方米） 3. 地段（比如离市中心距离，公里） 4. 房龄（年） 5. 房间数量（间）...（其他可能相关的信息）

目标数值（Y）： 我们想预测的那个具体数字（正确答案）！

这套房子实际卖了多少钱？（比如：350万， 480万， 210万...）

这些就是计算机要学习的“带数字答案的练习题”。

# 回归模型的核心



(2) 计算机的“任务”：找到X和Y的关系模式（核心规则）：

计算机的核心任务：分析这些“练习题”，找出特征（X）的变化是如何影响目标数值（Y）的。

规则手册的核心条款：

“特征变，目标跟着变”：当某个特征（比如房子大小）增大时，目标数值（房价）通常是增大还是减小？变化的幅度大概是多少？

“综合考量，找出最佳趋势”：计算机要综合考虑所有特征（大小、地段、房龄...）对房价的共同影响，试图找出一条最能代表这些数据点分布整体趋势的线（或面）。想象在散点图上画一条最贴合所有点的线。计算机怎么找？它内部会用数学方法（比如最小化预测值和真实值的差距之和），不断调整这条“趋势线”的位置和斜率，让它尽可能准确地穿过或靠近大多数数据点。目标是让预测值接近真实值。

# 回归模型的核心



## （3）形成“估算公式”（回归模型）：

经过学习，计算机总结出了它认为最能描述特征（X） 和 目标数值（Y） 之间关系的规律。

这个规律通常体现为一个数学公式（函数） 或者一个“估算器”：

例如（简化概念）：预测房价  $\approx$  （基础价格） + （每平方米单价 \* 面积） - （年折旧率 \* 房龄） + （地段加成系数 \* 地段得分） ...

这个公式或估算器，就是训练好的“回归模型”！ 它的核心就是学到的那个关联规则—— 知道了房子的特征，就能按这个规则套算出大概的房价。

# 回归模型的核心



第二步：使用规则进行数值预测（模型应用）—— 估算新房子价格

现在，有一套新房源上市了，我们不知道它最终会卖多少钱。计算机怎么用学到的模型来预测？

（1）观察新房子的特征（X）：

计算机提取这套新房子的特征值：1. 面积：120平方米 2. 地段（离市中心）：5公里 3. 房龄：8年  
4. 房间数：3间（和训练时看的特征类型完全一样）

（2）套用“估算公式”：

计算机拿出它训练时辛苦找到的那个“房价估算公式”（回归模型）。

它把新房子的具体特征值代入公式中的 X：

把 面积 = 120 代入公式

把 地段距离 = 5 代入公式

把 房龄 = 8 代入公式

把 房间数 = 3 代入公式



# 回归模型的核心



## (3) 计算预测值 (Y) :

计算机按照公式里定义好的数学规则（加、减、乘、系数...），进行一系列计算。

输出结果： 计算出一个具体的预测数值，比如：预测房价  $\approx$  318.5万元。

解读： 根据模型从历史数据中学到的规律，这套具有上述特征的房子，预计售价大约在318.5万元左右。

## 总结核心思想：

学习估算规则（训练 - 监督学习）：

输入： 特征 (X) + 目标数值 (Y)（带数字答案的练习题）。

核心规则： 分析X和Y的关系，找出最能描述数据整体变化趋势的数学模式/公式（通常是找一条“最佳拟合线”）。目标是让预测值尽可能接近真实值。

输出： 一个可以用来估算的“公式”或“规则”（回归模型）。

应用规则预测数值（预测）：

输入： 一个新事物的特征值 (X)。

过程： 将这些特征值代入训练好的“估算公式”中。

输出： 计算出一个预测的具体数值 (Y)（房价、销量、分数...）。

# 常见的回归模型

---



01 线性回归

02 多项式回归

03 岭回归

04 基于机器学习的回归方法

# 举例引入

用线性回归预估你的外卖送达时间

关键影响因素：（1）餐厅距离：3公里（2）当前时段：晚高峰（18:30）（3）天气状况：

小雨（4）骑手数量：周边有15个空闲骑手

简单计算公式：

预估时间 = 基础时间(10分钟) + 距离 $\times$ 0.8分钟/公里 + 高峰时段 $\times$ 5分钟 - 骑手多 $\times$ 0.2分钟/人 + 雨天 $\times$ 3分钟

代入计算：10 + 3 $\times$ 0.8 + 5 - 15 $\times$ 0.2 + 3 = 19分钟

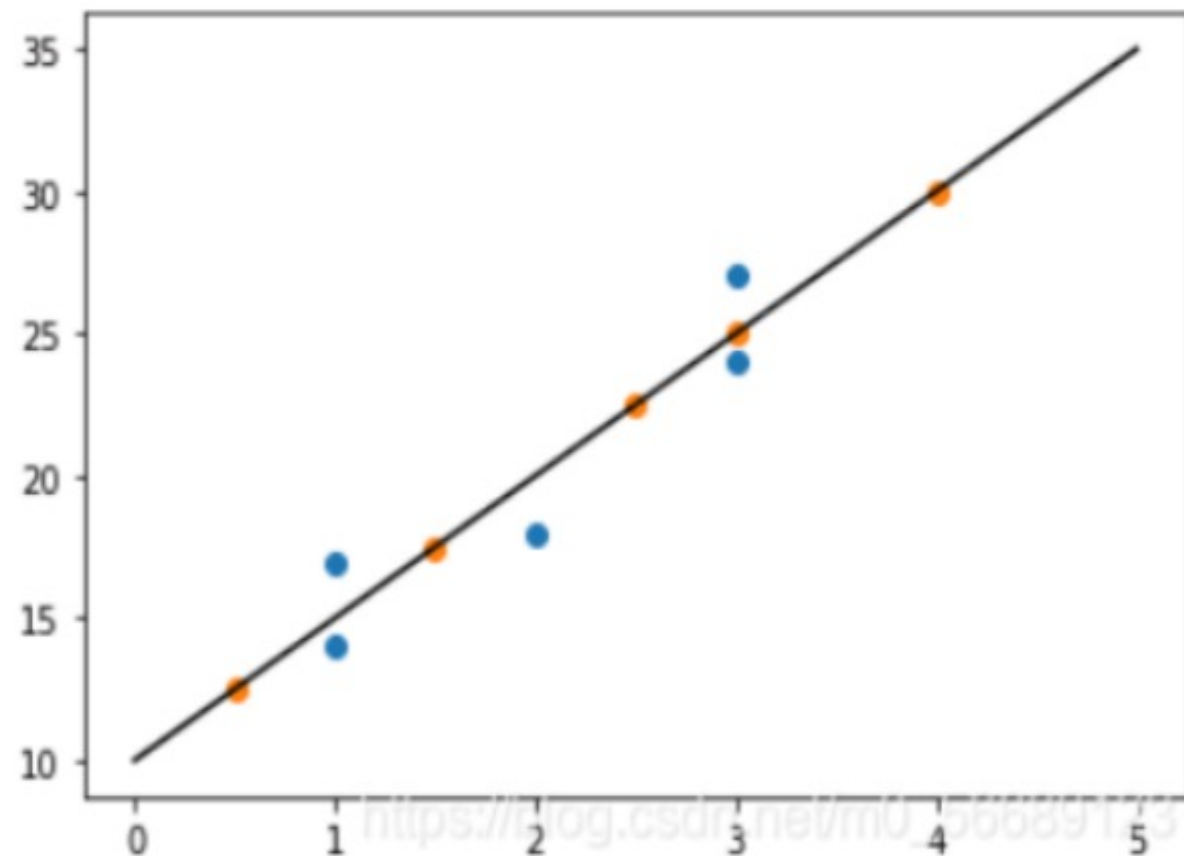
实际显示：APP显示：“预计19-23分钟送达”，最终送达时间：21分钟（在预测范围内）

线性回归像看路标（距终点500米，速度100米/分钟）

把模型比作老司机：“3公里晴天白天大概15分钟，现在下雨加堵车，再加5分钟”

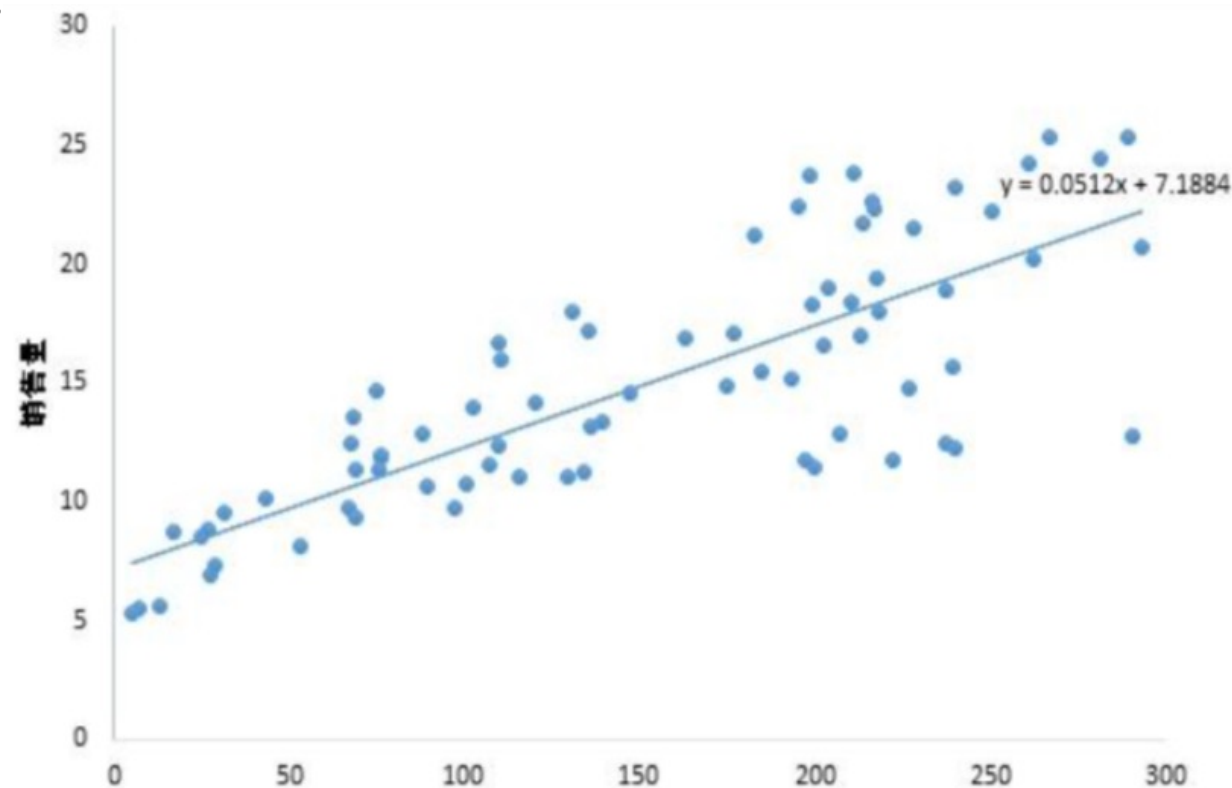
# 线性回归-定义

线性回归是一种简单的回归模型，它假设输入特征与输出值之间存在线性关系。线性回归的优点是简单直观，但对非线性关系拟合效果较差。



# 线性回归-详解

线性回归是一种非常简单易用且应用广泛的算法，其原理也非常容易理解，因此非常适合作为机器学习的入门算法。当我们在中学学习二元一次方程时，我们通常将 $y$ 视为因变量， $x$ 视为自变量，从而得到方程： $y = a * x + b$  其中， $a$  是斜率，表示自变量  $x$  对因变量  $y$  的影响程度； $b$  是截距，表示当  $x = 0$  时  $y$  的值。线性回归的目标就是找到最佳的参数  $a$  和  $b$ ，使得预测值  $y$ ，与实际值  $y$  之间的差异最小。当我们只用一个 $x$ 来预测 $y$ ，就是一元线性回归，也就是在找一个直线来拟合数据。



# 线性回归解决问题



针对问题(3)“如果知道一个学生每天的学习时间，如何预测他期末考试能考多少分？”，给出解决方法：

数据采集：记录学习时间和分数。

模型训练：用线性回归计算  $w$ （影响系数）和  $b$ （截距）。

预测应用：输入新学习时间，用  $(y = w * x + b)$  算分数。

可视化：用散点图 + 直线展示模型，直观呈现“学习时间 - 分数”关系。

# 举例引入

用多项式回归预测各站点的车辆需求量

问题复杂性：（1）线性回归假设用车量随时间均匀变化（不符合实际）（2）真实情况：早晨需求骤增，午间平缓，晚高峰又激增

曲线拟合过程：（1）基础线性项：时间  $\times$  系数（2）加入时间<sup>2</sup>项：捕捉早晚高峰的“抛物线”效应（3）加入时间<sup>3</sup>项：更精细地刻画午间低谷

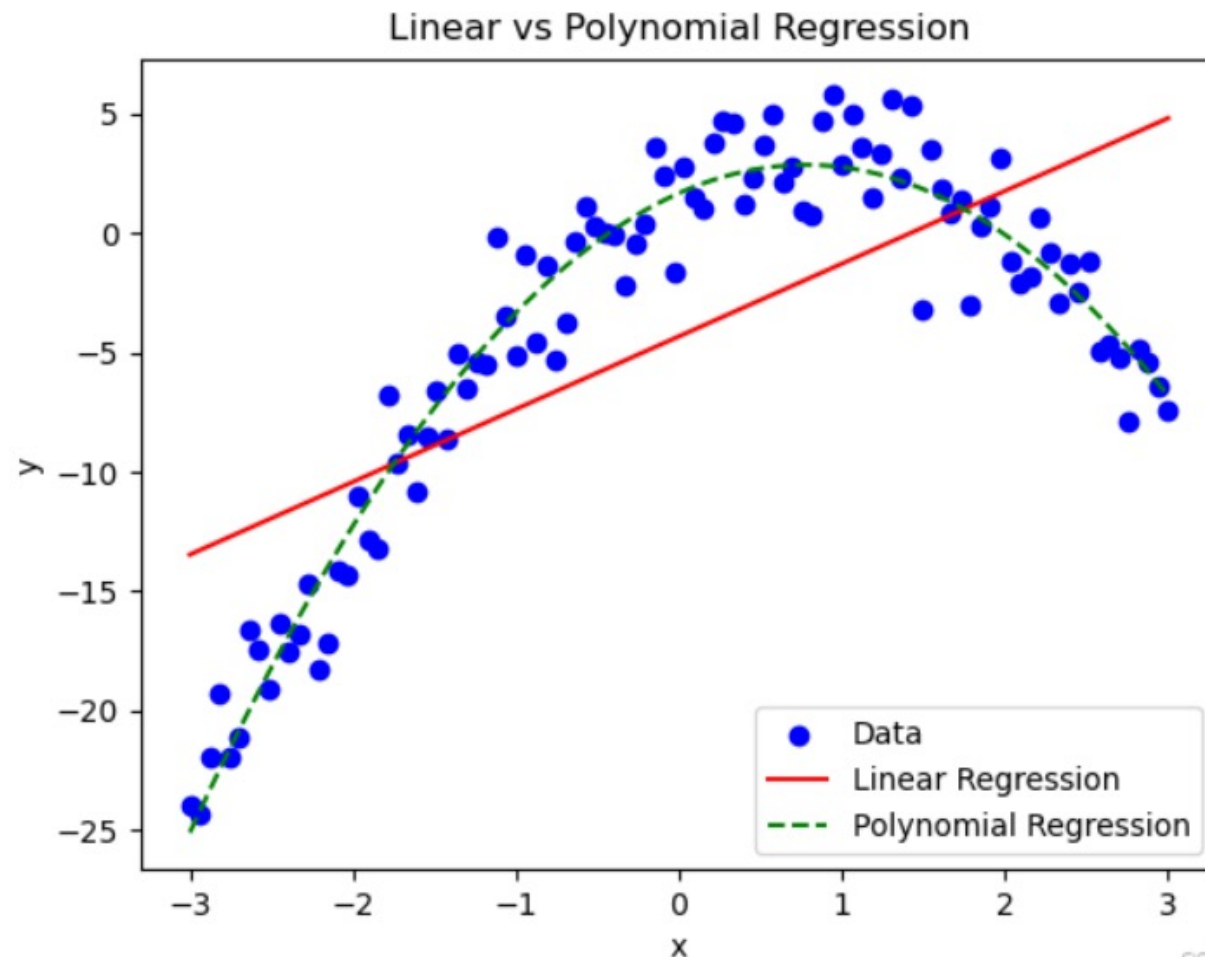
预测用车量 =  $a \times \text{时间} + b \times \text{时间}^2 + c \times \text{时间}^3 + \text{基础量}$ （4）像用可弯曲的尺子拟合数据点，而不是僵硬的直尺

实际效果：（1）早7点：预测需求量120%（实际115%）（2）下午2点：预测65%（实际70%）（3）晚6点：预测135%（实际130%）（4）比线性回归准确率提升40%

线性关系像匀速行驶的汽车；多项式回归像有加速/减速的真实路况；多项式回归像用曲线描绘真实地形

# 多项式回归-定义

多项式回归是在原始特征的基础上引入多阶项（如平方项、立方项）来拟合非线性关系。其优点是可表示非线性关系，形式简单，易于可视化，但也容易过拟合，特别是在阶数过高时对噪声较敏感。





# 多项式回归-详解



线性回归：线性回归假设目标变量和特征变量之间存在线性关系，即： $y = \beta_0 + \beta_1 x + \epsilon$

这种方法在特征与目标变量呈线性关系时效果很好，但在处理复杂的非线性关系时表现较差。

多项式回归：多项式回归通过引入高次项来拟合数据的非线性关系。通过增加多项式的阶数，可以捕捉到更多复杂的模式，但同时也增加了模型的复杂性和过拟合的风险。多项式回归适用于以下场景：

- (1) 数据中的非线性关系显著，如某些时间序列预测、经济数据分析等。
- (2) 需要通过模型捕捉复杂的模式和趋势。
- (3) 有足够的數據支持模型训练，避免过拟合风险。

# 多项式回归-详解

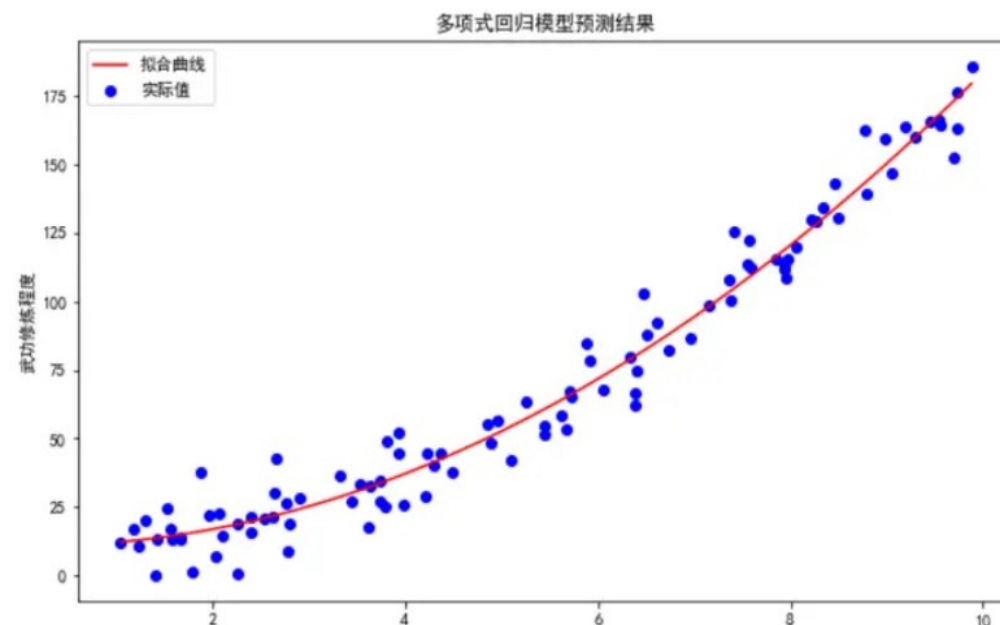
多项式回归的基本方程是通过在线性回归模型中加入多项式特征来构建的。其一般形式

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \cdots + \beta_n x^n + \epsilon$$

其中， $y$ 是目标变量， $x$ 是特征变量

$\beta_0, \beta_1, \beta_2, \cdots, \beta_n$ 是待估参数， $\epsilon$ 是误差项。

通过这种方式，多项式回归可以拟合出更加复杂的曲线，而不仅仅是直线。



# 举例引入



证券公司用岭回归预测明日股价走势

数据挑战：（1）影响股价的因素太多（100+个指标）（2）许多指标高度相关（如GDP增长率与PMI指数）（3）传统线性回归会出现“系数爆炸”（某个系数变得极大）

岭回归的解决方案：（1）给所有影响因素的权重系数“系上安全带”（2）防止某些系数变得过大而主导预测结果（3）平衡预测准确性和模型稳定性，调整后的权重 = 原权重  $\times$  (1 - 惩罚力度)

实际预测效果：（1）避免给出“明天股价会涨300%”的荒谬预测（2）当多个指标同时指向上涨时，给出稳健的+2.5%预测（3）在市场波动期尤其可靠（防止过度反应）

像在数学优化的山峰间“筑起山岭”，防止模型滑向悬崖般的极端解

把模型比作自动驾驶：普通回归可能突然猛打方向，岭回归会平稳修正路线

**岭回归：**当特征之间存在多重共线性（即多个变量高度相关）时，普通线性回归模型会变得不稳定。岭回归通过在损失函数中加入 L2 正则项，约束参数大小，有效防止过拟合，提升模型泛化能力。

**L2正则化：**在机器学习中，正则化（regularization）是防止模型过拟合（overfitting）、降低模型泛化误差的一类方法。过拟合，简单来说就是模型对当前数据拟合得非常好，训练集上的误差

损失甚至可以等于0，但是在实际应用中的泛化误差却很大。L2正则化：
$$J(\theta) = \frac{1}{m} \sum_{i=1}^m \left( \theta^T \cdot x^{(i)} - y^{(i)} \right)^2 + \alpha \sum_{i=1}^n \theta_i^2$$

其本质是通过权重衰减，弱化不显著的特征所占权重。

# 举例引入



用机器学习模型预测iPhone新品发售首月销量

复杂因素处理：（1）传统方法无法处理的非线性关系：（2）社交媒体讨论热度与销量的指数级关系（3）竞品价格变动影响的阈值效应

海量特征自动筛选：（1）\*从500+潜在特征中识别出30个关键指标\*（2）自动发现“小红书种草笔记数”比“微博热搜次数”更重要

多模型协作预测：（1）XGBoost处理结构化数据（价格、配置参数）（2）神经网络分析非结构化数据（用户评论情感倾向）（3）集成模型综合各子模型结果

动态调整机制：（1）实时吸收预售数据修正预测（2）自动识别异常模式（如黄牛抢购干扰）（3）预测区间随不确定性动态变化

实际预测效果：输入：新品参数+市场环境+历史数据

输出：\*基础预测：首月销量58-62万台 \*概率分布：超过65万台的可能性23% 实际销量：61.4万台（在预测区间内）

传统统计方法像老式收音机，机器学习回归像智能音箱。把系统比作资深买手：不仅看参数表格，更能感知市场情绪波动

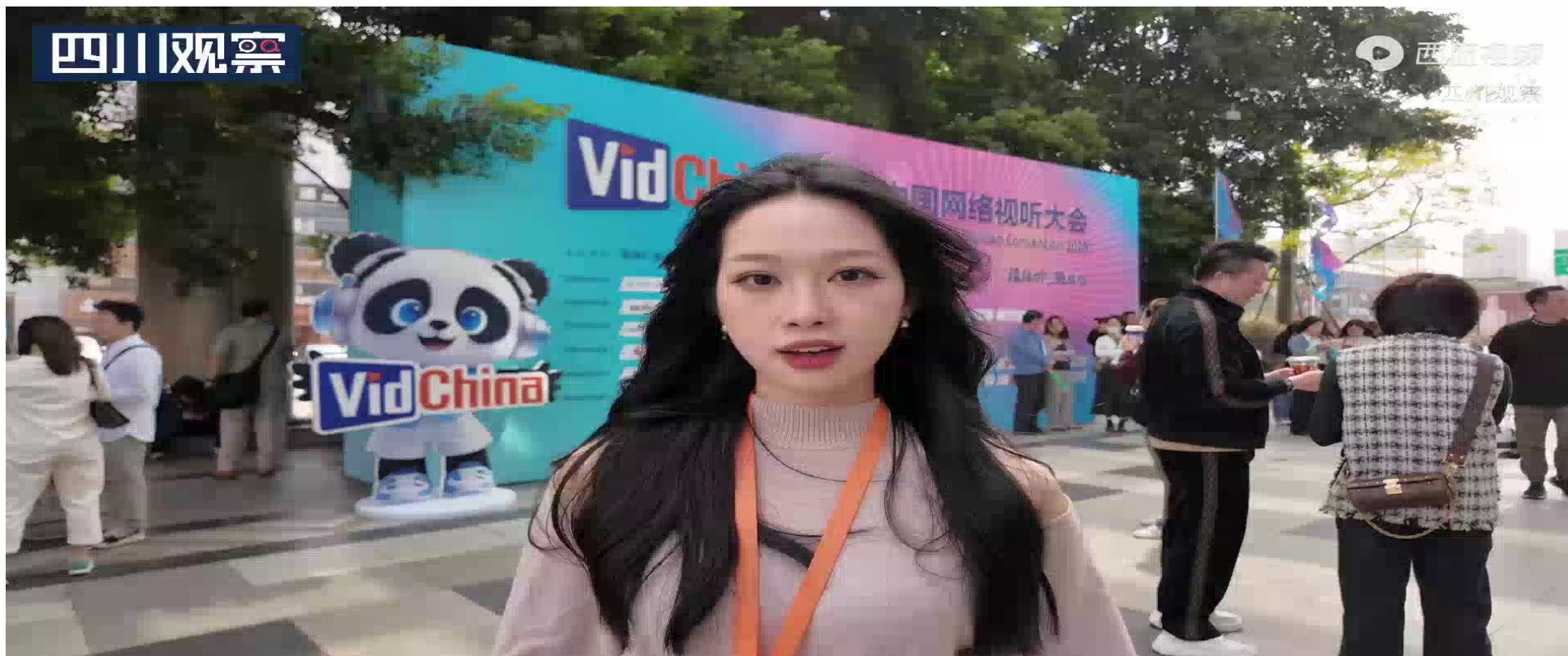
# 基于机器学习的回归方法



传统回归模型通常需要人为设定函数形式，且对特征之间的关系假设较为强烈。而基于机器学习的回归方法则通过数据驱动的方式自动建模，能够处理非线性、复杂的特征交互关系，具有更强的适应性与灵活性。常见的机器学习回归模型包括：支持向量回归（Support Vector Regression）、随机森林回归（Random Forest Regression）等。关于基于机器学习的回归方法的深入内容，特别是其在时间序列预测中的应用及其深度学习扩展，将在第 4.2.4 节“预测分析”中详细讲解。



## 《打开新“视”界大门！网络视听大会探展vlog来啦～》





# 05

## 总结



# 分类模型总结

模型名称	核心特点	主要优点
逻辑回归	基于 Sigmoid 函数输出概率，适用于二分类，线性决策边界。	解释性强（系数反映特征影响），训练快，数据量要求低，输出概率可解释。
神经网络	多层网络结构，通过非线性激活函数拟合复杂关系，适配非结构化数据。	非线性拟合能力强，自动提取高阶特征，大规模数据下表现优异，衍生模型场景广。
支持向量机（SVM）	寻找最大间隔超平面，核函数处理非线性，适用于高维小样本。	抗过拟合能力强，对高维数据稳定，在基因、文本分类等领域效果显著。
决策树	以“if-then”规则构建树状结构，通过特征阈值递归划分数据，可视化程度高。	可解释性极强（决策路径清晰），无需特征归一化，支持混合类型数据。
随机森林	多棵决策树集成，Bagging 抽样 + 特征随机选择降低方差，投票 / 均值输出结果。	抗过拟合能力强，对异常值不敏感，训练效率高，泛化性能稳定。
朴素贝叶斯	基于贝叶斯定理，假设特征条件独立，通过先验 / 似然概率计算后验概率。	训练速度极快，对高维稀疏数据（如文本）表现优异，内存占用低。
XGBoost	梯度提升树集成，残差迭代优化 + 正则化控制复杂度，竞赛常用模型。	预测精度高，支持并行计算，对缺失值和异常值鲁棒性强，适用多场景分类任务。

# 聚类模型总结

模型名称	核心特点	主要优点
K 均值聚类	预设 K 值，随机初始化中心后迭代分配样本（最小欧氏距离），形成球形簇。	算法简单高效，可扩展性强（适合大规模数据），聚类结果直观易懂。
谱聚类	基于图论，数据转化为相似度图后特征值分解降维，再用 K-means 聚类。	能发现任意形状簇，对高维数据降维效果好，适合图像分割、社区检测等场景。
层次聚类	聚合式 / 分裂式构建树状图（Dendrogram），无需预设 K 值。	无需手动设定簇数，聚类结果可通过树状图展示层次关系，适合探索数据结构。
DBSCAN	基于密度定义簇（核心点、边界点、噪声点），通过 $\epsilon$ 和最小点数控制聚类，识别噪声。	无需预设 K 值，能发现任意形状簇，对噪声数据不敏感，适用于空间数据或异常值场景。

# 回归模型总结

模型名称	核心特点	主要优点
线性回归	假设线性关系 $y = w_1 \cdot x_1 + \dots + b$ ，最小二乘法求解参数。	模型简单易解释（系数反映影响程度），训练快，可作为基准模型分析数据趋势。
多项式回归	引入自变量高次项（如 $x^2$ ）拟合非线性关系，本质是线性模型扩展。	捕捉曲线趋势，灵活性高于线性回归，实现简单（仅需特征转换）。
岭回归	线性回归 + L2 正则化（惩罚系数平方和），限制参数绝对值。	解决多重共线性问题，降低过拟合风险，保留全部特征（适合特征均重要场景）。
基于机器学习的回归方法	如随机森林回归、SVR，基于树集成或核函数的非线性回归，无需假设数据分布。	拟合能力强（捕捉非线性关系），对异常值不敏感（树模型），泛化性能优异。

我们知道人类通过看书、练习‘学习’知识，那机器想‘学会’识别猫狗、预测天气，得怎么给它‘喂知识’？深度学习里的‘学习’和人类学习有啥相似又不同的地方？



参考：

<https://blog.csdn.net/>