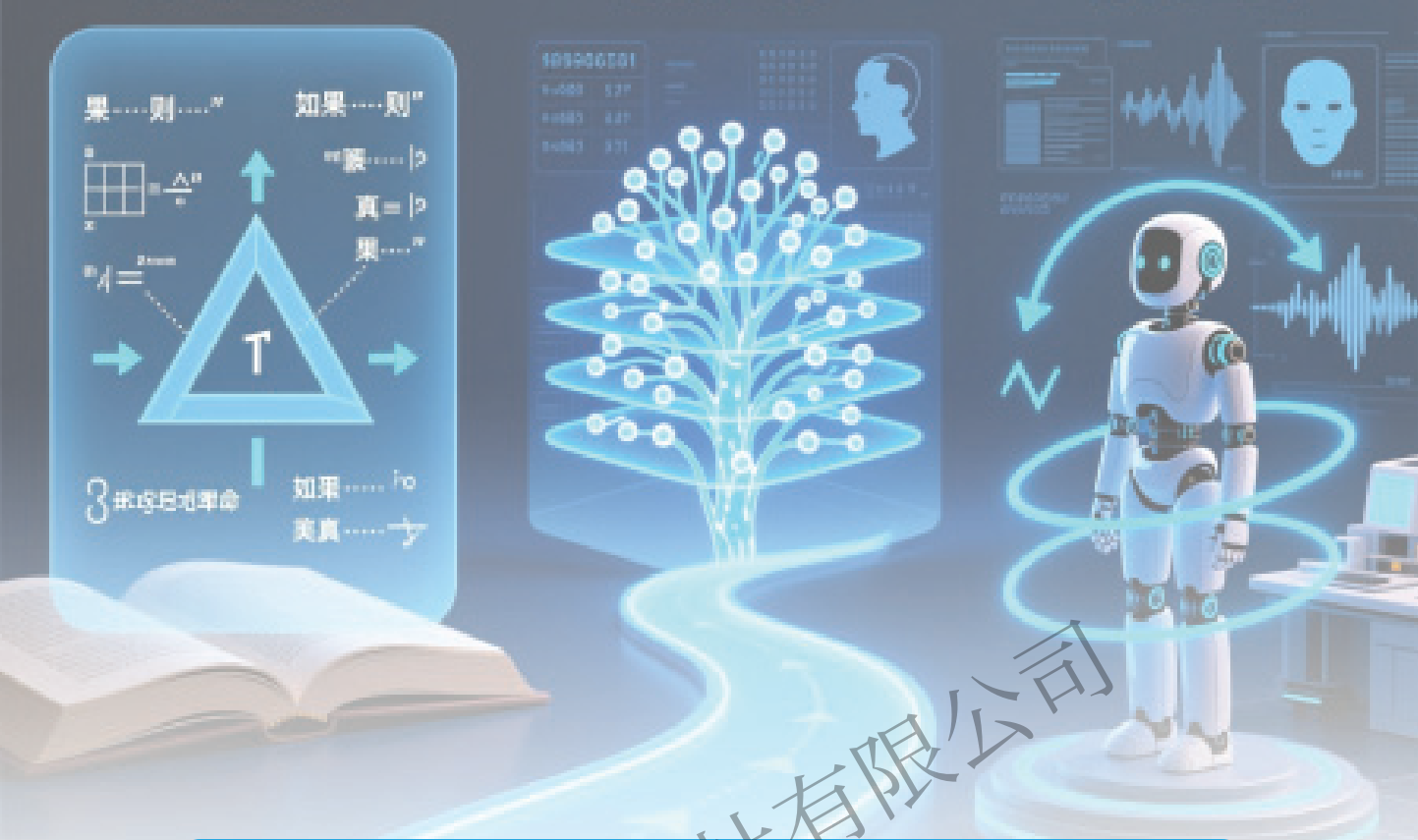


第 3 章
学会从数据中成长——典型机器学习模型应用

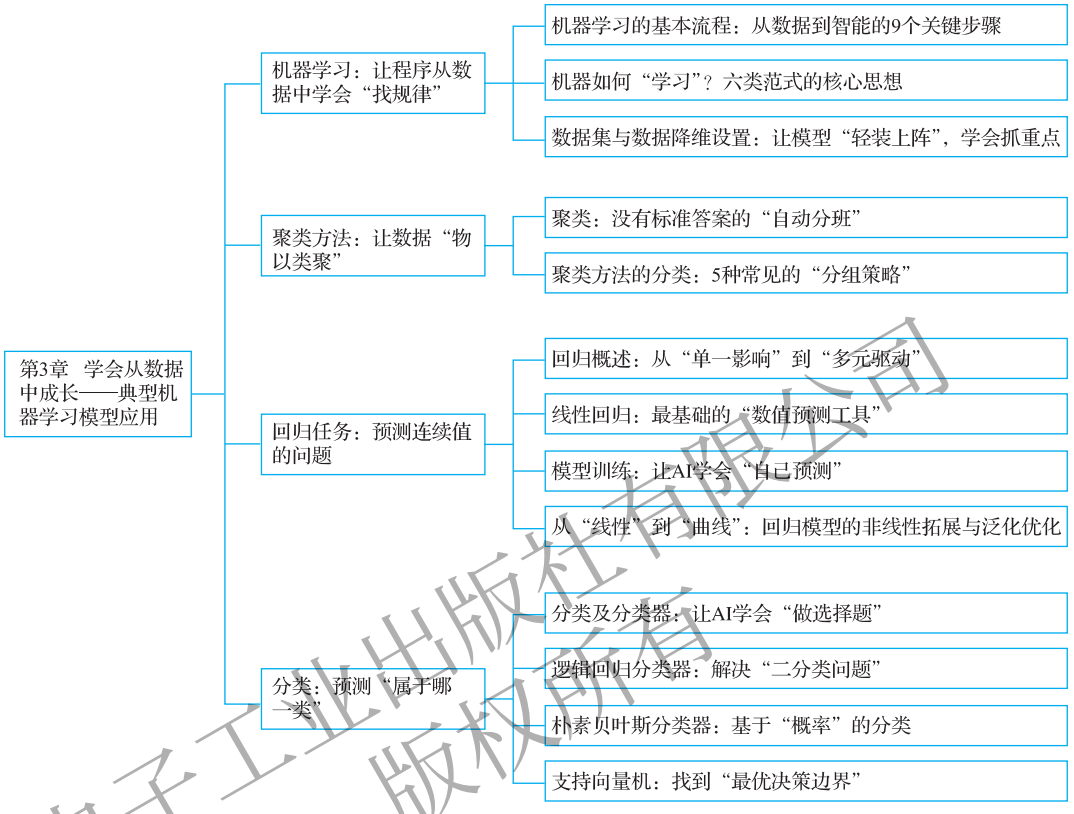
知识目标

（1）掌握机器学习的基本流程与六类范式的核心思想。
（2）熟悉从问题定义、数据预处理到模型部署的完整机器学习流程。
（3）理解线性回归、支持向量机、朴素贝叶斯等经典模型的原理与应用。
（4）了解分类与回归的本质区别、常用方法与应用领域。

思政目标

（1）增强运用 AI 技术服务农业、改善民生的社会责任感。
（2）弘扬科研人员扎根一线、攻坚克难的实干与奉献精神。
（3）树立突破“卡脖子”难题、推动核心技术自主可控的使命意识。
（4）培养尊重数据、科学决策的思维方式和可持续发展理念。

知识导图



导读案例

AI 赋能智慧养殖——破解家禽养殖难题的智能革命

家禽养殖是保障“菜篮子”工程、助力农民增收的核心产业，肉鸡与蛋鸡养殖更是关系民生的重要板块。但长期以来，传统养殖模式深陷“高成本、高风险、低效益”的困境，人工巡检与经验管理的局限性日益凸显：缺乏科学化标准支撑导致环境调控失准、疫病预警滞后、饲料投喂浪费等问题频发，种鸡死亡率普遍高达 5%，产蛋率仅 66.07%，其中 30% 以上的意外死亡源于温/湿度失控、通风不良等管理漏洞；单位面积养殖密度低、智能化不足，让每斤鸡肉的养殖成本居高不下至 6 元左右，严重压缩利润空间；小规模养殖户户均每批养殖约 2 万只、年养 3.5 批，高度依赖人力劳动，精细化管理无从谈起，一旦遭遇疫情或极端天气，便可能面临重大经济损失。如何降低死亡率、提升产蛋率与养殖效率，成为推动家禽养殖业转型升级的核心命题，而机器学习正是破解这一难题的“智能内核”。

为突破传统养殖瓶颈，自主可控的 AI 养鸡大模型应运而生，如图 3-1 所示，其核心支撑正是各类机器学习算法的协同发力。该模型采用“瘦设备+胖平台”创新架构，前端部署

低成本、易安装的数字装备，实时采集温度、湿度、氨气浓度、二氧化碳、光照强度、用水量、鸡群体重等多维度数据，为机器学习算法提供海量训练素材；后端依托统一数据中台，让机器学习算法成为养殖全流程的“智慧大脑”。监督学习算法通过标注的健康鸡群与患病鸡群数据，精准学习体征特征与疫病的映射关系，实现疫病早期预警；无监督学习算法自主挖掘环境参数间的潜在关联，识别温/湿度突变、氨气浓度超标等异常模式，为环境调控提供依据；强化学习算法则根据鸡群生长状态、饲料消耗与市场反馈，动态优化投喂策略与养殖密度，持续提升养殖效益。“种鸡销售决策模型”更是机器学习的典型应用，它融合了监督学习与回归分析，综合市场行情、日龄、体重、日增重趋势等多因素，动态推荐最佳出栏时机，打破传统单一化养殖决策逻辑。

这套机器学习驱动的智能养殖系统，背后是研发团队扎根产业、攻坚克难的坚守。为让模型适配不同地域、不同规模的养殖场景，收集了全国 20 余个省份的养殖数据，覆盖高温、高湿、严寒等多样环境；为提升疫病预警准确率，标注了上万组临床病例数据，反复优化监督学习模型的特征提取与分类逻辑；为降低中小养殖户使用门槛，通过迁移学习复用成熟养殖知识，让模型无须大量本地数据即可快速部署。研发人员深知，每个算法参数的微调，都关系到养殖户的实际收益；每次模型迭代，都承载着产业升级的期盼，他们用“科技助农”的初心，攻克了数据稀疏、场景复杂等诸多难题。最终，机器学习让家禽养殖从“经验驱动”彻底转向“数据驱动”，实现了养殖全流程的可视化、自动化与智能化。

图 3-1　AI 养鸡大模型

3.1 机器学习：让程序从数据中学会“找规律”

机器学习是 AI 领域的核心基石，既是连接理论模型与实际应用的关键桥梁，又是培养“数据思维”与“建模能力”的核心载体。从日常的智能推荐、语音识别到前沿的自动驾驶、医疗诊断，其思想与算法已广泛渗透于各类科技应用，掌握机器学习基础是深入 AI 领域的必备前提。本模块承接数学基础与编程能力，为后续深度学习等进阶内容铺垫，编写遵循

“从基础到应用、从理论到实践”原则，内容围绕“基础认知-经典模型-工具实战”层层递进，全方位讲解机器学习的基础知识，涵盖基本概念、算法划分、学习方式分类及模型训练等关键因素，介绍线性回归、支持向量机、朴素贝叶斯等经典模型原理与算法、Python 的 Scikit-learn 库等并结合实际应用案例，详细讲解上述模型调用方法并通过代码示例让学生掌握实现技巧。

与传统编程不同，机器学习的核心逻辑是让计算机从数据中自主学习规律，而非依赖人工编写的固定规则。在传统编程模式中，开发者需要明确输入到输出的映射规则（输入数据+映射规则→输出结果）；而在机器学习模式下，计算机通过分析大量标注数据，自动学习输入数据与输出结果的内在关联（输入数据+输出结果→学习规则），进而具备对新数据的预测能力。

3.1.1 机器学习的基本流程：从数据到智能的 9 个关键步骤

机器学习的本质，是从数据中自动发现规律，并利用这些规律对未来进行预测或决策。它不是魔法，而是一套严谨的科学方法论。一个典型的机器学习项目，往往遵循一条清晰的技术路径，如图 3-2 所示。理解这一完整流程，是掌握机器学习思维的第一步。

图 3-2 机器学习的基本流程

1. 问题定义：明确“我们要解决什么”

任何成功的机器学习应用都始于清晰的问题界定。例如，在智慧养殖场景中，问题是“如何降低鸡群死亡率”还是“何时出栏利润最高”，前者可能导向异常检测模型，后者则需要回归与决策优化。只有明确了任务类型（分类、回归、聚类等），才能选择合适的工具与评价标准。

2. 数据收集：让机器“见多识广”

数据是机器学习的“燃料”。无论是鸡舍中的温/湿度传感器记录，还是历史销售价格曲

线，原始数据构成了模型学习的基础。数据来源可以是数据库、API、物联网设备或人工采集。数据越丰富、覆盖场景越广，模型的泛化能力就越强。

3. 数据预处理：为数据“洗澡剪发”

现实世界的数据往往是“脏乱差”的：存在缺失值、异常点、单位不统一等问题。此阶段需要进行清洗（去除错误数据）、归一化（缩放到统一尺度）、编码（类别转数字）等操作，确保输入干净整洁，避免“垃圾进，垃圾出”（Garbage in, Garbage out）。

4. 特征工程：提炼数据中的“精华”

并非所有原始数据都适合直接喂给模型。特征工程是指从原始数据中构造更有意义的新变量。例如，仅知道“日龄”和“体重”不够，还需要计算“日增重趋势”；知道“温度”也不够，可结合“湿度”生成“体感温度指数”。高质量的特征能显著提升模型性能，甚至比模型本身更重要。

5. 模型选择：挑选合适的“学习者”

根据任务类型选择合适的学习范式与具体算法。

- 分类：逻辑回归、支持向量机、随机森林。
- 回归：线性回归、XGBoost。
- 聚类：K-Means、DBSCAN。不同的模型有不同的假设前提与适用范围，需要结合数据特点谨慎选择。

6. 模型训练：让模型“学会经验”

使用标注好的训练数据集，让算法自动调整内部参数，以最小化预测误差。这个过程类似学生通过做题来掌握解题方法。监督学习依赖标签指导，无监督学习则自行探索结构，强化学习通过奖励机制不断试错。

7. 模型评估：检验“学得怎么样”

不能只看训练表现，要测试模型在新数据上的泛化能力。常用方法包括划分训练集/验证集/测试集，使用准确率、精确率、召回率、AUC、均方误差等指标衡量性能。若过拟合严重（训练好但测试差），则需要重新调整模型复杂度。

8. 模型调优：精益求精的打磨

通过超参数搜索（如网格搜索、贝叶斯优化）、交叉验证等方式，进一步提升模型表现。例如，调整决策树的最大深度、神经网络的学习率等，找到最优配置组合。

9. 模型部署与监控：走向真实世界

当模型通过评估后，便可部署上线，用于实际预测。例如，将疫病预警模型接入管理系统，实时推送风险提示。但部署不是终点，还需要持续监控其表现，防止因环境变化导致性能下降（“模型漂移”），必要时重新训练更新。

机器学习不是一键生成的奇迹，而是一个环环相扣、反复迭代的工程过程。从问题出

发，以数据为基，经建模、验证、优化，最终服务于现实需求，这才是机器学习落地的真实图景。

3.1.2 机器如何“学习”？六类范式的核心思想

根据学习方式的不同，机器学习算法主要分为六大类，各类算法适用于不同的任务场景和数据条件。

1. 监督学习：在老师的指引下学会“看图说话”

监督学习（Supervised Learning）是最常见和基础的机器学习范式，核心特征是使用有标签数据进行训练。在训练过程中，模型明确知道每个输入特征对应的正确输出（标签），通过学习输入特征与输出标签之间的映射关系，实现对新数据的准确预测。

（1）核心思想：使用“输入特征-输出标签”成对的有标签数据进行训练，模型需要学习从输入到输出的明确映射关系，如同在“老师的指引”下学习。

（2）典型场景：适用于有明确预测目标、标签易获取的任务，如垃圾邮件识别（标签为垃圾/正常）、房价预测（标签为具体价格）、疾病诊断（标签为患病/健康）。

（3）常用算法：线性回归、逻辑回归、决策树、随机森林、支持向量机、深度学习模型等，是目前应用最广泛的学习范式。

2. 无监督学习：在黑暗中摸索数据的“隐藏乐章”

无监督学习（Unsupervised Learning）无须预先标记的目标输出作为指导，核心是让模型自主发现数据中的模式、结构和关系。

（1）核心思想：仅使用无标签数据进行训练，模型需要自主挖掘数据隐藏的结构、规律或关联，相当于“自学探索”。

（2）典型场景：适用于标签难以获取或无须明确预测目标的任务，如用户分群（按消费习惯聚类）、高维数据降维可视化、信用卡欺诈检测（发现异常模式）。

（3）常用算法：K-Means、层次聚类、主成分分析（PCA）、自编码器（Autoencoder）、关联规则挖掘（如购物篮分析）。

3. 半监督学习：善用少数“灯塔”照亮海量“迷雾”

半监督学习（Semi-supervised Learning）结合少量有标签数据和大量无标签数据进行训练，旨在利用有限标注信息引导模型理解更广泛的数据分布。

（1）核心思想：介于监督学习与无监督学习之间，结合少量有标签数据和大量无标签数据进行训练，既利用有标签数据的监督信息，又借助无标签数据的分布特征提升性能。

（2）典型场景：适用于标注成本高但无标签数据丰富的场景，如医学影像分析（专业医生标注 CT 片成本高）、语音识别（少量人工标注语音+大量未标注语音）、文本分类（仅部分文档有类别标签），能够在保证性能的同时显著降低数据标注负担。

（3）常用算法：自训练（模型使用有标签数据进行训练后，对无标签数据打“伪标签”

迭代优化）、一致性正则化（保证输入小扰动时预测结果稳定）。

4. 强化学习：智能体的“生存游戏”与“策略进化”

强化学习（Reinforcement Learning）关注智能体（Agent）与环境（Environment）的交互学习，其核心目标是让智能体通过试错积累经验，获得最大的累积奖励。在强化学习中，智能体根据当前环境状态选择行动，环境会对该行动给予奖励或惩罚反馈，智能体通过不断调整行为策略，优化长期收益。

（1）核心思想：不依赖静态数据集标签，通过“智能体与环境交互”学习——智能体采取动作后获得“奖励/惩罚”反馈，目标是优化策略以最大化长期收益，类似“试错成长”。

（2）典型场景：动态决策与控制任务，如游戏 AI（AlphaGo、Dota2 AI）、自动驾驶（实时应对路况）、智能机器人导航（适应复杂环境）。

（3）核心要素：包含状态（环境描述）、动作（智能体选择）、奖励（反馈信号）、策略（决策规则）四大核心组件。

5. 自监督学习：给机器一本“自我出题的练习册”

自监督学习（Self-supervised Learning）通过在无标签数据上构造伪监督任务来自动生成训练信号，使模型能够进行有效的预训练。例如，在自然语言处理中，BERT 模型采用掩码语言建模（Masked Language Model，MLM），即遮蔽部分文本并让模型预测被遮蔽词；在计算机视觉中，可通过图像补全、旋转预测等方式构建自监督任务。这一范式已成为大模型时代的基础性技术。

（1）核心思想：无监督学习的进阶衍生，通过构造“辅助任务”从无标签数据中自动生成监督信号，无须人工标注，核心是学习通用可迁移的特征表示。

（2）典型场景：大规模数据预处理与特征学习，如自然语言处理中 BERT 模型的 MLM、计算机视觉中的对比学习（MoCo）、语音识别的特征预训练。

（3）核心特点：与半监督学习不同，自监督学习不依赖少量有标签数据，而是通过数据本身构造监督信息，常用于大模型预训练阶段。

6. 迁移学习：站在巨人肩膀上的“知识跃迁”

迁移学习（Transfer Learning）旨在将在一个任务或领域上学到的知识迁移到另一个相关任务中，从而提升目标模型的学习效率与泛化能力。典型做法是使用在大规模数据集上预训练的模型（如 ResNet、BERT）作为起点，并在特定任务上进行微调（Fine-tuning）。迁移学习有效解决了小样本场景下的模型训练难题，被广泛应用于跨域识别、少样本学习等领域。

（1）核心思想：一种知识复用型学习范式，将在“源域/源任务”中训练好的模型参数与知识，迁移到相关的“目标域/目标任务”中，帮助新模型快速训练，降低对目标域有标签数据的依赖。

（2）典型场景：小样本、高标注成本的任务，如 ImageNet 预训练模型迁移到医学影像分类、BERT 微调处理小样本文本分类、低资源语言的机器翻译。

（3）核心特点：依托源任务与目标任务的相关性实现知识复用，能显著提升训练效率

与泛化能力，避免模型“从零开始学习”，是跨域任务与工程化应用的核心策略。

机器学习的核心范式根据学习方式的不同，呈现出多样化的知识获取路径。监督学习如同“有师之学”，依赖有标签数据建立输入与输出之间的精确映射，是当前应用最广泛的基础范式；无监督学习则像“自主探索”，在无标签数据中挖掘隐藏结构，揭示数据内在规律；半监督学习和自监督学习作为两者之间的桥梁与延伸，巧妙利用少量标注或自行构造监督信号，释放海量无标签数据的价值；强化学习模拟“试错进化”的决策过程，通过智能体与环境的持续交互优化长期策略；而迁移学习实现“知识跃迁”，将已有经验迁移到新任务中，显著提升学习效率。这六类范式各具特色、相辅相成，共同构成了现代机器学习的方法论基石，为不同场景下的智能建模提供了系统性的解决方案。

3.1.3 数据集与数据降维设置：让模型“轻装上阵”，学会抓重点

在机器学习项目中，我们常常面临一个现实矛盾：数据越多越好，但太复杂又会让模型“学不动”。就像一位新手养殖户面对几十个监控仪表——温度、湿度、氨气浓度、光照强度、饮水量、体重变化……信息密密麻麻，反而不知道该关注哪个指标。此时，我们需要做的不是收集更多数据，而是帮助模型“抓重点”，这就引出了本节的核心主题：数据集与数据降维设置。

1. 合理的数据集是成功的第一步

一个好的数据集，应当满足以下 3 个基本要求。

（1）代表性强：能覆盖实际应用场景。例如，在智慧养殖系统中，如果只采集夏季高温环境下的数据，模型就无法适应冬季寒冷地区的养殖情况。为此，研发团队专门收集了全国 20 多个省份、不同季节、不同规模鸡舍的数据，确保模型具备广泛的适用性。

（2）标签准确：对于监督学习任务（如疫病识别），每条数据都应有明确标签，如“这只鸡是否患病”“当天死亡原因是什么”。这些标签就像老师批改作业时的答案，是模型学习正确判断的基础。

（3）特征相关性强：输入特征必须与目标问题密切相关。例如，预测“何时出栏利润最高”时，日龄、体重、饲料消耗速率等是关键特征；而“鸡舍墙面颜色”这类无关特征应剔除，避免干扰模型判断。

2. 数据集的核心构成：喂给模型的“食材”要新鲜又全面

数据集是机器学习的“营养来源”，就像养鸡需要优质饲料一样，模型也需要高质量的数据才能“健康成长”。一个合理的数据集应由以下 3 部分组成。

（1）特征变量（Features）：描述样本的各种属性。例如，在智慧养殖系统中，每只鸡每天的体温、体重、采食量、环境温/湿度、光照时间等都属于特征。

（2）目标变量（Label/Target）：我们希望预测的结果，如“是否患病”“当日产蛋率”“最佳出栏日龄”等。

（3）样本数量与多样性：足够的数据量和丰富的场景覆盖，能让模型学会应对各种情

况。如果只用南方夏季的数据训练模型，它到了北方冬季就可能“水土不服”。

3. 为什么要进行数据降维？——给数据“瘦身”

随着传感器技术的发展，现代鸡舍可实时采集 10 多种环境与生理参数，形成高维数据空间。虽然信息丰富，但带来了“维度灾难”问题。

- 模型训练变慢，计算资源消耗大。
- 多个变量之间可能存在冗余（如温度升高常伴随湿度下降）。
- 过多无关或噪声特征会误导模型，降低预测准确性。

这时，就需要使用数据降维技术，把复杂的高维数据压缩成少数几个“核心特征”，既保留了关键信息，又提升了效率。常见的数据降维方法有两种，主成分分析（Principal Component Analysis，PCA）和线性判别分析（Linear Discriminant Analysis，LDA），如表 3-1 所示。

表 3-1 常见的数据降维方法

方法	原理简述	智慧养殖应用示例
主成分分析	将多个相关变量合并为少数几个不相关的“综合指标”	把温度、湿度、二氧化碳浓度合成“环境压力指数”，用于评估鸡群舒适度
线性判别分析	在分类任务中寻找最能区分不同类别的方向	区分健康鸡与患病鸡的关键体征组合，提升诊断准确率

通常先对数据进行标准化处理（消除单位差异），再应用主成分分析；可通过“累计方差贡献率”判断保留多少主成分合适（一般≥85%）。

4. 相关性分析：找出变量之间的“隐藏关系”

相关性分析是数据降维的前提，用于判断特征之间的关联程度，避免冗余。常用皮尔逊相关系数衡量，取值范围为[−1,1]。

- 取值接近 1：正相关强（如“氨气浓度”与“鸡群患病率”）。
- 取值接近−1：负相关强（如“通风时长”与“氨气浓度”）。
- 取值接近 0：几乎无关联（如“鸡的性别”与“光照强度”）。

常用的相关性分析工具是相关系数矩阵图（Heatmap）和散点图矩阵（Pair Plot）。

5. 如何设置才合理？——遵循“够用就好”原则

合理的数据降维设置需要把握以下 4 点。

（1）探索性数据分析（EDA）：画图观察各变量之间的关系，找出强相关的特征对，提前预判是否需要降维。

（2）保留足够信息量：通常要求降维后保留 85%～90% 的原始方差（适用于主成分分析）。

（3）结合业务解释性：尽量选择结果可解释的方法。例如，“综合健康指数”比抽象的“主成分 1”更容易被养殖户理解和信任。

（4）避免过度降维：一味追求低维度可能导致信息丢失。应通过交叉验证比较不同维度下的模型性能，找到最优平衡点。

很多初学者误以为“降维就是删掉不重要的列”。其实不然，真正的降维是信息的提炼与重组，如同把一本厚书浓缩成一份提纲，虽篇幅变短，但精华仍在。它不是简化问题，而是让模型更聪明地看待问题。

3.2 聚类方法：让数据“物以类聚”

在上一节中，我们了解到机器学习能帮助养殖场实现环境监控、疫病预警和出栏决策。但面对成千上万只鸡的个体差异，如何进行精细化管理？如哪些鸡生长快、吃得少？哪些鸡活动异常、可能生病？这时，我们就需要一种无须标签就能自动分组的技术——聚类（Clustering）。

3.2.1 聚类：没有标准答案的“自动分班”

想象一下，开学第一天，老师要把一群陌生的孩子分成几个小组，但她不知道谁成绩好、谁爱运动、谁性格内向，她只能根据孩子们的行为特征（说话声音大小、走路快慢、是否主动交流）来判断谁和谁更像。这正是聚类的核心思想。

- 在没有标签的情况下，把相似的数据样本归为一类（称为“簇”），做到“同类相近，异类相远”。
- 它属于无监督学习，不需要提前知道每个样本属于哪一类。
- 广泛应用于用户分群、异常检测、生物分类等领域。
- 在智慧养殖系统中可用于鸡群健康状态分组、采食行为模式识别、生长潜力评估等。

3.2.2 聚类方法的分类：5 种常见的“分组策略”

根据分组逻辑的不同，聚类方法可分为以下 5 类，如表 3-2 所示。

表 3-2 聚类方法的分类

类型	核心思想	典型算法	智慧养殖应用示例
划分式聚类	把数据划分为互不重叠的 K 个簇，每个样本只能属于一个组	K-Means、K-Medoids、CLARANS	按鸡群生长特征（体重、日增重、采食量）划分生长等级，为不同等级制定差异化投喂方案
层次聚类	构建树状结构，逐步合并或拆分簇	AGNES（自底向上）、DIANA（自顶向下）	分析不同批次鸡群的疫病易感性层次关系，识别高风险群体的共性特征
密度聚类	基于样本分布密度，发现任意形状的簇	DBSCAN、OPTICS	从鸡舍环境监测数据（温度、湿度、氨气浓度）中识别异常环境模式，预警疫病风险

续表

类型	核心思想	典型算法	智慧养殖应用示例
网格聚类	将数据空间划分为有限个单元格，在网格的基础上聚类	STING、CLIQUE	结合鸡群生理指标与环境数据，预测不同生长阶段的健康状态分布
模型式聚类	假设数据来自某种概率分布，拟合模型后进行归属判断	高斯混合模型（GMM）、自组织映射（SOM）	处理全国多省份养殖基地的海量传感器数据，快速划分环境适宜性区域

1. 划分式聚类（Partitioning Clustering）

（1）核心思想：将数据集一次性划分为互不重叠的 K 个簇，每个样本仅属于一个簇，通过迭代优化簇中心和样本归属，最小化簇内差异。

（2）典型算法：K-Means、K-Medoids、CLARANS。

（3）适用场景：针对全国 20 多个省份家禽养殖近 10 年的温/湿度、氨气浓度、鸡群体重及活动量等特征数据，该类数据簇结构紧凑、形状规则且数据量适配。通过聚类将环境条件、鸡群生长状态相近的区域归为一类，划分养殖适配性梯度，为智能养殖模型地域化适配、疫病防控差异化策略制定提供依据，提升养殖全流程智能化管控效率，助力家禽养殖产业转型升级。

K-Means 算法的核心特性如下。

K-Means 是划分式聚类中最常用的算法，属于硬聚类（每个样本仅属于一个簇），学习方式为无监督学习，无须先验标签即可完成数据分组。其核心目标是最小化所有样本到其所属簇中心（质心）的平方距离和（SSE，误差平方和），公式为 $\mathrm{SSE}=\sum_{k=1}^{K}\sum_{\boldsymbol{x}_i\in C_k}\|\boldsymbol{x}_i-\boldsymbol{\mu}_k\|^2$。其中，$K$ 为预设簇数，C_k 为第 k 个簇，$\boldsymbol{\mu}_k$ 为第 k 个簇的质心，$\boldsymbol{x}_i$ 为样本向量。

K-Means 算法的执行步骤如下。

（1）**初始化簇中心：**从数据集中随机选择 K 个样本作为初始质心。

（2）**分配样本归属：**计算每个样本与所有质心的距离（常用欧氏距离），将样本分配到距离最近的质心所属簇。

（3）**更新簇中心：**计算每个簇内所有样本的均值，将其作为新的质心。

（4）**迭代优化：**重复步骤（2）和步骤（3），直到质心不再变化（或变化小于预设阈值）、SSE 收敛，迭代停止。

2. 层次聚类（Hierarchical Clustering）

（1）核心思想：通过“自底向上合并”（凝聚式）或“自顶向下分裂”（分裂式）的方式，构建簇的层次结构（树状图），无须预先指定簇数 K。

（2）典型算法：AGNES、DIANA。

（3）适用场景：针对全国 20 多个省份的家禽养殖相关特征数据，当需要探索不同区域在环境条件、养殖基础、鸡群生长适配性上的层次关联时，通过该类算法构建簇的层次结构，清晰呈现区域间的养殖适配梯度关系，为分层级制定智能养殖方案、优化疫病防控差

异化策略提供支撑，助力家禽养殖产业转型升级。

3. 密度聚类（Density-based Clustering）

（1）核心思想：基于样本的局部密度进行聚类，将“密度相连”的样本划分为同一簇，能识别任意形状的簇，且可自动过滤噪声点。

（2）典型算法：DBSCAN、OPTICS。

（3）适用场景：当全国 20 多个省份的家禽养殖特征数据中混有噪声，且区域环境条件、鸡群生长适配性形成的簇形状不规则时，通过该类算法可精准识别任意形状的有效簇，自动过滤异常数据，为精准评估区域养殖潜力、优化智能养殖方案提供可靠支撑，助力家禽养殖产业转型升级。

4. 网格聚类（Grid-based Clustering）

（1）核心思想：将数据空间划分为固定大小的网格，基于网格的密度进行聚类，计算效率高，不受数据形状的影响。

（2）典型算法：STING、CLIQUE。

（3）适用场景：针对全国 20 多个省份家禽养殖近 10 年的高维特征大数据，该类算法计算高效且不受数据形状的限制。通过网格密度聚类快速处理温/湿度、氨气浓度、鸡群体重等海量监测数据，为智能养殖模型地域化部署、投喂策略动态优化提供高效支撑，助力提升养殖管理效率，推进家禽养殖产业转型升级。

5. 模型式聚类（Model-based Clustering）

（1）核心思想：假设数据由潜在的概率模型生成（如高斯分布混合），通过优化模型参数拟合数据分布，进而实现簇划分。

（2）典型算法：GMM、SOM。

（3）适用场景：针对全国 20 多个省份的家禽养殖特征数据，当需要基于概率分布假设分析区域养殖适配性与疫病防控潜力时，通过该类算法拟合数据潜在分布，兼具聚类效果与模型可解释性，为科学制定智能养殖发展策略、优化差异化疫病防控方案提供精准支撑，助力家禽养殖产业转型升级。

【聚类示例】 **K-Means 聚类**

假设“智能家禽养殖环境适配性分析系统”选取 6 个代表性养殖区域的核心特征（年平均温度、氨气浓度均值）数据，使用 K-Means 算法（K=2）进行聚类分析，如表 3-3 所示。

表 3-3 K-Means 聚类示例

养殖区域	年平均温度/℃	氨气浓度均值/ppm
广东佛山	26.5	18.2
内蒙古通辽	−2.1	8.5
湖北武汉	16.8	12.6
黑龙江哈尔滨	−5.3	7.8

续表

养殖区域	年平均温度/℃	氨气浓度均值/ppm
云南昆明	15.1	11.9
上海崇明	17.2	13.1

聚类过程如下。

（1）初始化质心：随机选择 K=2 个样本作为初始质心，假设选择广东佛山(26.5,18.2)和内蒙古通辽(−2.1,8.5)。

（2）分配样本：计算每个养殖区域与两个质心的欧氏距离，将区域分配到距离更近的簇。

① 簇 1（广东佛山）：广东佛山、湖北武汉、云南昆明、上海崇明（距离广东佛山更近）。

② 簇 2（内蒙古通辽）：内蒙古通辽、黑龙江哈尔滨（距离内蒙古通辽更近）。

（3）更新质心：计算两个簇的均值作为新质心。

① 簇 1 的新质心：((26.5+16.8+15.1+17.2)/4, (18.2+12.6+11.9+13.1)/4)=(18.90, 13.95)。

② 簇 2 的新质心：((−2.1+(−5.3))/2, (8.5+7.8)/2)=(−3.70, 8.15)。

（4）迭代收敛：再次计算每个养殖区域与新质心的欧氏距离，样本归属未发生变化，质心稳定，迭代停止。

（5）最终结果。

① 簇 1（高温高氨区）：{广东佛山、湖北武汉、云南昆明、上海崇明}，质心(18.90, 13.95)。

② 簇 2（低温低氨区）：{内蒙古通辽、黑龙江哈尔滨}，质心(−3.70, 8.15)。

K-Means 算法的 Python 实现

```
# 导入所需库
import numpy as np
from sklearn.cluster import KMeans

# 1. 准备数据：养殖区域、年平均温度、氨气浓度均值
regions = np.array(['广东佛山', '内蒙古通辽', '湖北武汉', '黑龙江哈尔滨', '云南昆明', '上海崇明'])
# 特征数据：对应上述养殖区域的[年平均温度, 氨气浓度均值]
features = np.array([[26.5, 18.2], [−2.1, 8.5], [16.8, 12.6],
                     [−5.3, 7.8], [15.1, 11.9], [17.2, 13.1]])

# 2. 构建 K-Means 模型并训练
kmeans = KMeans(n_clusters=2, random_state=0, n_init='auto')
labels = kmeans.fit_predict(features)

# 3. 输出聚类结果
print("聚类结果：")
for i in range(2):
    cluster_regions = regions[labels == i]
    centroid = kmeans.cluster_centers_[i]
```

```
    cluster_name = "高温高氨区" if i == 0 else "低温低氨区"
    print(f"{cluster_name}：{list(cluster_regions)}，质心：({centroid[0]:.2f}, {centroid[1]:.2f})")
```

【聚类示例】 AGNES 聚类

AGNES 是层次聚类中最常用的算法，其核心思想是“自底向上合并”——初始时每个样本为一个独立簇，通过迭代计算簇间相似度，将最相似的两个簇合并为新簇，直到所有样本合并为一个簇（或达到预设簇数），最终形成层状结构（树状图）。簇间相似度的计算方法如下。

- 单链法：以两个簇中最近样本的距离为簇间距离。
- 全链法：以两个簇中最远样本的距离为簇间距离。
- 平均链法：以两个簇中所有样本对的平均距离为簇间距离。
- 质心法：以两个簇的质心距离为簇间距离。

AGNES 算法的执行步骤如下。

（1）初始化：将每个样本视为一个独立簇（共 n 个簇，n 为样本数）。

（2）合并相似簇：计算簇间相似度，将相似度最高的两个簇合并为新簇。

（3）更新簇间相似度：重新计算新簇与其他所有簇的距离（按选定的簇间相似度的计算方法）。

（4）迭代合并：重复步骤（2）和步骤（3），直到所有样本合并为一个簇（或达到预设簇数）。

基于 K-Means 聚类示例中的 6 个代表性养殖区域特征数据（见表 3-3），使用单链法进行分析，聚类过程如下。

（1）初始化簇：6 个簇分别为{C1={广东佛山}, C2={内蒙古通辽}, C3={湖北武汉}, C4={黑龙江哈尔滨}, C5={云南昆明}, C6={上海崇明}}。

（2）计算簇间距离（欧氏距离）。

① C1 与 C3 距离：$\sqrt{(26.5-16.8)^2+(18.2-12.6)^2}\approx 11.20$。

② C3 与 C5 距离：$\sqrt{(16.8-15.1)^2+(12.6-11.9)^2}\approx 1.84$。

③ C3 与 C6 距离：$\sqrt{(16.8-17.2)^2+(12.6-13.1)^2}\approx 0.64$。

④ C2 与 C4 距离：$\sqrt{(-2.1-5.3)^2+(8.5-7.8)^2}\approx 3.28$。

（3）合并最近簇：选择距离最近的簇对{C3={湖北武汉}, C6={上海崇明}}合并为新簇 C7={湖北武汉,上海崇明}。

（4）更新簇间相似度：计算 C7 与其他簇的距离，重复合并步骤，依次合并{C3={湖北武汉}, C5={云南昆明}}为 C8={湖北武汉,云南昆明}（注：实际迭代中先合并 C7 与 C5），合并{C7={湖北武汉,上海崇明}, C8={湖北武汉,云南昆明}}为 C9={广东佛山,湖北武汉,云南昆明,上海崇明}，合并{C2={内蒙古通辽}, C4={黑龙江哈尔滨}}为 C10={内蒙古通辽,黑龙江哈尔滨}。

（5）最终结果：形成两个核心簇{C10={低温低氨区:内蒙古通辽,黑龙江哈尔滨},C9={高温高氨区:广东佛山,湖北武汉,云南昆明,上海崇明}}，树状图如图 3-3 所示。

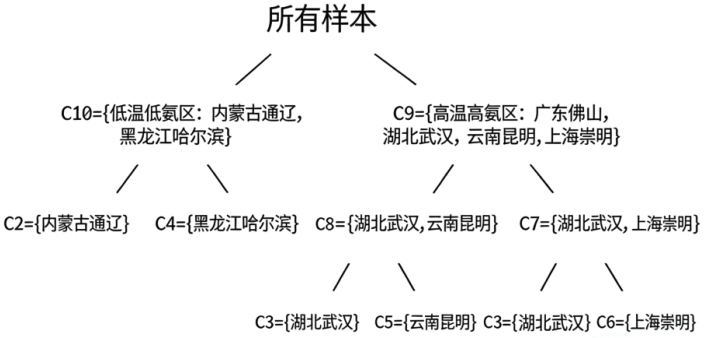


图 3-3　聚类结果树状图

AGNES 算法的 Python 实现

```
import numpy as np
from scipy.cluster.hierarchy import linkage, fcluster
from scipy.spatial.distance import pdist

# 养殖区域特征数据：[年平均温度，氨气浓度均值]
features = np.array([[26.5,18.2], [-2.1,8.5], [16.8,12.6],
                     [-5.3,7.8], [15.1,11.9], [17.2,13.1]])
regions = ['广东佛山','内蒙古通辽','湖北武汉','黑龙江哈尔滨','云南昆明','上海崇明']

# 单链法层次聚类（ward->single 为单链法）
Z = linkage(features, method='single')          # 构建树状图
labels = fcluster(Z, t=2, criterion='maxclust')  # 聚为 2 类

# 输出结果
for i in [1,2]:
    cluster = [regions[j] for j in range(6) if labels[j]==i]
    name = "高温高氨区" if i==1 else "低温低氨区"
    print(f"{name}：{cluster}")
```

3.3 回归任务：预测连续值的问题

回归是机器学习中最基础也最实用的技术之一，属于监督学习的范畴，其核心任务是“根据已知的输入特征，预测一个连续的数值结果”，如年龄、价格、温度、时间等。实现这一目标的方式，类似老师指导学生解题：先提供大量带有正确答案的样本（每个样本都包含“特征”和对应的“标签”），让模型从中学习规律；当面对新数据时，就能基于所学做出合理的数值预测。

在农业生产中，如何科学决策是提高效益的关键。例如，在现代化养鸡场中，养殖户最关心的一个问题是这批鸡什么时候卖最合适？卖得太早，体重不够，单价低；卖得太晚，饲料消耗大，成本上升，还可能影响下一批养殖节奏。理想的做法是根据每只鸡当前的生长状态，预测它未来达到理想体重的时间点——这就是一个典型的“回归任务”。

核心特点：输出是一个“数字”，而不是“类别”。

回归的典型应用场景如表 3-4 所示。

表 3-4　回归的典型应用场景

应用场景	预测目标（输出）	输入特征（依据）
智慧养鸡	最佳出栏时间/天	当前体重、日均增重、品种、饲养温度、饲料类型
智能温室	明日番茄产量/kg	光照时长、湿度、施肥记录、植株高度
物流调度	快递送达时间/h	距离、天气、交通状况、包裹质量
家庭健康	孩子成年身高/cm	父母身高、当前年龄身高、营养摄入

3.3.1　回归概述：从“单一影响”到“多元驱动”

回归的核心逻辑是通过分析海量有标签数据，挖掘特征与连续输出之间的隐藏规律，并用数学模型量化这种关系。根据不同的划分标准，回归可以分为不同的类型。

1. 按输入特征的个数分类

（1）一元回归：只关注一个特征，如仅根据鸡群日龄预测出栏效益。

（2）多元回归：同时考虑多个特征，如结合鸡群日龄、体重、市场价格一起预测出栏效益。

2. 按特征和结果的关系分类

（1）线性回归：特征和结果是“直线”关系，如鸡群日龄每增加 5 天，出栏效益匀速提升。

（2）非线性回归：特征和结果是“曲线”关系，如前期鸡群日龄增加，出栏效益增长较快，日龄超过阈值后，出栏效益增长放缓甚至下降。

3. 按预测结果的个数分类

（1）简单回归：只预测一个结果，如只预测鸡群的出栏效益。

（2）多重回归：同时预测多个结果，如同时预测鸡群的出栏效益、料肉比。

4. 应用场景

（1）智慧养殖：根据日龄、投喂量预测鸡群体重，结合市场行情确定最佳出栏时间，优化养殖成本。

（2）金融领域：预测明天的股票价格、评估贷款申请人是否会违约、估算保险理赔金额。

（3）市场分析：根据广告投入、产品定价预测销量，帮助商家制定最优定价策略。

（4）自然科学：根据温室气体排放数据预测未来气温变化，根据光照强度、浇水频率预测植物生长高度。

（5）医学健康：根据饮食习惯、运动频率预测患高血压的风险，根据药物剂量预测治疗效果。

举个贴合智慧养殖的例子：我们想确定鸡群日龄、体重、市场价格与出栏效益的关系，收集多组数据——鸡群不同日龄（如45日、55日、65日）、对应体重（如1.2kg、1.8kg、2.3kg）、同期市场价格（如8元/kg、7.5元/kg、7元/kg）及最终出栏效益（净利润12元/只、18元/只、15元/只）。回归会拟合出特征与出栏效益的最优关联模型，清晰呈现日龄、体重、市场价格如何共同影响出栏效益。后续只需输入实时数据，就能通过模型预测不同出栏时间的效益，精准锁定效益最高的出栏时间。

回归不是抽象的算法，而是实实在在解决问题的工具，在多个领域都有广泛应用。

3.3.2 线性回归：最基础的“数值预测工具”

线性回归是回归家族中最基础、最常用的模型，其核心思想是找到一条“最佳拟合直线”，用它来描述特征和结果之间的线性关系。其数学逻辑简单、可解释性强，是入门回归技术的最佳起点。

（1）一元线性回归：找到“最佳拟合直线”。

一元线性回归的核心是处理“单一特征与单一输出”之间的线性关系，目标是找到一条能最贴合所有数据点的直线，使得模型预测值与真实值的误差最小。一元线性回归的直线方程为

$$y=ax+b$$

式中，y为预测输出（因变量），即我们需要预测的连续数值（如鸡群出栏效益）；x为输入特征（自变量），即影响预测结果的单一因素（如鸡群日龄）；a为回归系数（直线斜率），表示输入特征x每变化一个单位，预测输出y平均变化的幅度；b为截距，表示当输入特征$x=0$时，预测输出y的基准值。

假设收集某养殖场20批肉鸡的“养殖日龄”（25天、30天、35天、…、120天）和对应的“单只出栏效益”（8元、12元、15元、…、32元），如图3-4所示，将这些数据绘制成散点图后，一元线性回归模型会通过算法找到一条最贴合所有散点的直线（“最佳拟合直线”），如图3-5所示。例如，最终得到的直线方程为$y=0.3141x+1.4752$，通过该方程，输入任意养殖日龄（如50天），即可快速预测出对应的单只出栏效益$0.3141\times50+1.4752\approx17.18$元，为判断出栏时间提供数据参考。

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	养殖日龄	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120
2	单只出栏效益	8	10	12	13	15	17	19	21	23	25	26	28	30	31	32	33	34	35	36	37

图3-4 一元线性回归模型预测结果示意图

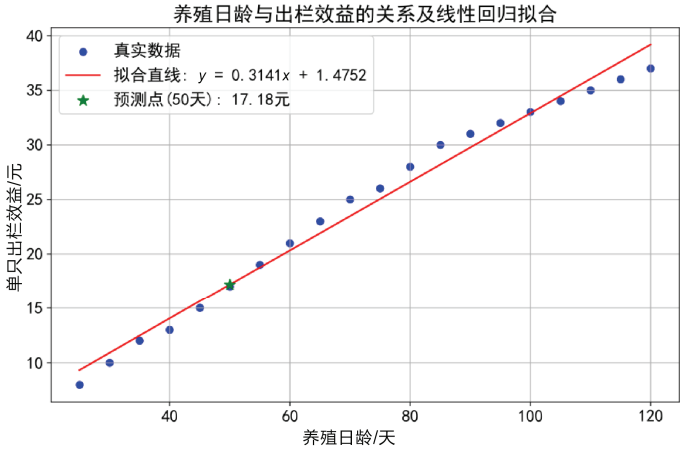


图 3-5　一元线性回归模型拟合直线示意图

（2）损失函数：判断“直线拟合得好不好”。

损失函数（Loss Function）是用来衡量模型预测值与真实值之间的差异或误差的函数，如图 3-6 所示。它的作用是帮助我们评估模型的性能，并为我们提供一个优化目标，以便通过调整模型参数来最小化误差。

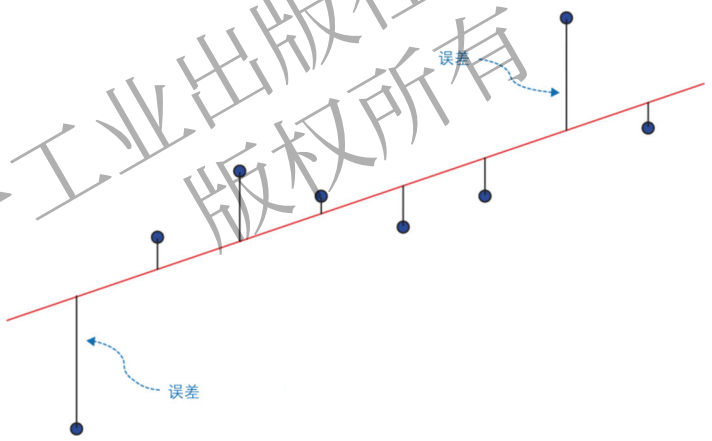


图 3-6　损失函数示意图

（3）梯度下降法：找到“最佳拟合直线”的方法。

找到“最佳拟合直线”的本质是求解使损失函数最小的参数 w（斜率）和 b（截距），而梯度下降法是实现这一目标的最常用算法。其核心思想可类比为“下山寻路”，结合智慧养殖场景理解如下。

① 初始位置：随机设定一组参数(w_0,b_0)，相当于我们先随意给出一个“鸡群日龄与出栏效益”的关联参数，站在误差“山顶”。

② 计算损失：根据当前参数计算损失函数，相当于用这组参数预测出栏效益，对比真实效益计算出整体误差（当前位置的“高度”）。

③ 梯度方向：计算损失函数在当前参数处的梯度（误差变化率最大的方向），相当于

找到能最快降低出栏效益预测误差的参数调整路径，如图 3-7 和图 3-8 所示。

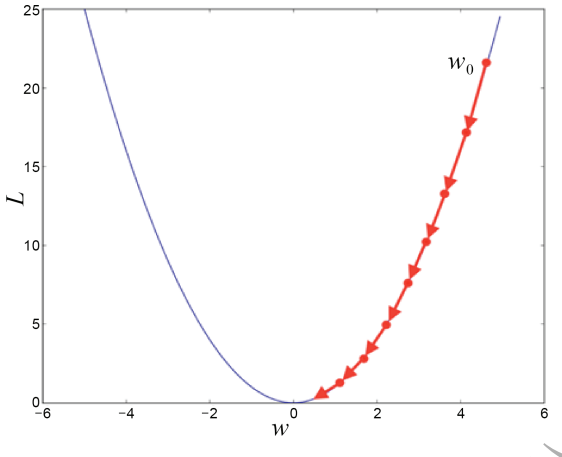


图 3-7　梯度下降法二维示意图

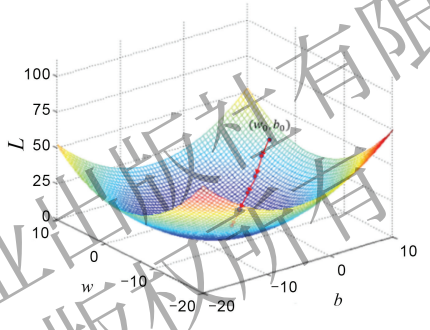


图 3-8　梯度下降法三维示意图

④ 迭代更新：沿着梯度的反方向（损失减小的方向），逐步调整参数 w 和 b，相当于朝着降低预测误差的方向，微调“日龄影响出栏效益”的参数。

⑤ 终止条件：重复上述步骤，直到损失函数不再明显减小（到达误差“山底”），此时的参数(w,b)即最优参数，对应的直线就是能精准预测鸡群出栏效益的最佳拟合直线。

图 3-7 和图 3-8 分别从二维和三维的角度展示了梯度下降法的优化过程，其坐标含义如下。

在图 3-7 中，横轴 w 表示模型参数（如“日龄影响出栏效益”的权重），纵轴 L 表示损失函数。红色路径代表从初始参数 w_0 出发，沿着损失下降的方向逐步迭代的过程，最终逼近最小值点。

在图 3-8 中，横轴 w 表示模型参数，纵轴 b 表示截距，竖轴 L 表示损失函数。曲面表示不同(w,b)下的损失，红色箭头从初始点(w_0,b_0)开始，沿梯度反方向向损失最小的“谷底”移动，直至收敛到最优参数。

其中，w_0 和 b_0 表示初始设定的参数，相当于优化过程的起点；随着迭代更新，参数逐渐趋近于最优解(w^*,b^*)。

（4）多元线性回归：考虑多个影响因素。

现实中，一个养殖结果往往受多个因素影响。例如，鸡群单只出栏效益不仅和养殖日龄有关，还和日投喂量、温/湿度控制精度、疫病防控效果等多种养殖因素相关，这时候就需要“多元线性回归”。多元线性回归的公式只是在一元线性回归的基础上增加了特征项：

$$y = a_1x_1 + a_2x_2 + \cdots + b$$

式中，x_1、x_2 为不同的养殖特征（如养殖日龄、日投喂量、温/湿度控制精度）；a_1、a_2 为每个养殖特征的“影响程度”（回归系数），系数越大，说明这个特征对出栏效益的影响越显著；b 为截距，即当所有养殖特征为 0 时，出栏效益的基准值。

多元线性回归的核心逻辑和一元线性回归的完全相同：找到一组最优的 $a_1, a_2, a_3, \cdots$ 和 b，让出栏效益预测值与真实值的差距（损失）最小。

3.3.3 模型训练：让 AI 学会“自己预测”

掌握了线性回归的原理后，我们需要通过“模型训练”让 AI 实际学会预测。训练的核心流程是准备数据→划分数据→训练模型→评估模型。

（1）数据集划分。

为了确保模型的泛化能力（对未知数据的预测能力），不能将所有数据都用于模型训练，需要将数据集划分为训练集、验证集和测试集 3 部分，如表 3-5 所示，各自承担不同的角色。

表 3-5 数据集划分

数据集类型	典型占比	核心作用	类比场景
训练集	70%～80%	用于模型学习特征与标签的关系，调整参数（如回归系数 a 和截距 b）	养殖户的“过往养殖记录”，用于总结出栏规律
验证集	10%～15%	用于调整模型超参数（如学习率、多项式阶次），优化模型结构，避免过拟合	养殖户的“近期小批量试养数据”，用于验证规律适用性
测试集	10%～15%	用于最终评估模型的泛化能力，模拟真实应用场景，不参与任何模型调整	养殖户的“新批次养殖数据”，用于检验模型实际预测效果

数据集划分的关键原则如下。

① 随机划分：打乱数据顺序后划分，避免数据的品种或养殖环境偏差（如所有优质种鸡样本都集中在训练集），确保数据集的代表性。

② 时间序列划分：按养殖时间顺序划分——用过去 1～2 年的养殖数据训练、近 3 个月的数据验证、最新 1 个月的数据测试，符合“用历史养殖经验预测未来出栏时间”的逻辑。

③ K 折交叉验证：当数据量较小时，可采用 K 折交叉验证（如 K=5），将数据分成 K 份，轮流用 K–1 份作为训练集、1 份作为验证集，最终取 K 次验证结果的均值，提升评估的可靠性。

（2）模型训练的完整流程。

① 数据读取与预处理。

数据读取：从文件（如 CSV、Excel）中读取种鸡的“鸡冠大小、脚距长度、日增重、

饲料消耗”等特征，以及对应的“最佳出栏时间”标签。

```
import pandas as pd
# 读取数据
data = pd.read_csv('chicken_slaughter_time.csv')
# 提取特征（鸡冠大小、脚距长度、日增重、饲料消耗）和标签（最佳出栏时间）
x = data[['鸡冠大小', '脚距长度', '日增重', '饲料消耗']].values  # 特征
y = data['最佳出栏时间'].values  # 标签
```

数据预处理：清洗数据（删除缺失值、异常值，如出栏时间为负数、日增重异常高的样本），确保数据质量。

② 划分数据集。

使用工具库将数据集按“8∶2”的比例划分为训练集和测试集（验证集可从训练集中进一步划分）。

```
from sklearn.model_selection import train_test_split
# 划分训练集（80%）和测试集（20%），random_state 确保划分结果可重复
X_train,X_test,y_train,y_test= train_test_split(X, y, test_size=0.2, random_state=42)
```

③ 模型训练与参数求解。

初始化线性回归模型，用训练集数据训练模型，通过梯度下降法求解最优参数（a 和 b）。

```
from sklearn.linear_model import LinearRegression
#初始化模型
model = LinearRegression()
#调整特征维度（适配模型输入要求）
X_train = X_train.reshape(-1, 1)
#用训练集训练模型
model.fit(X_train, y_train)
#输出最优参数
a = model.coef_[0]  #回归系数（斜率）
b = model.intercept_  #截距
print(f"最佳拟合直线方程：y = {a:.2f}x + {b:.2f}")
# 输出示例：最佳拟合直线方程：y = 16.06x + 9.58
```

④ 模型预测与可视化。

用训练好的模型对测试集数据进行预测，并通过可视化直观展示拟合效果。

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']                    # 设置中文字体
plt.rcParams['axes.unicode_minus'] = False
y_pred = model.predict(X_test)                                  # 对测试集进行预测
# 可视化：绘制真实出栏时间与预测出栏时间的对比散点图
plt.scatter(y_test, y_pred, color='blue', alpha=0.6, label='真实值 vs 预测值')
# 绘制理想拟合线（y=x）
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw=2, label='理想拟合线')
plt.xlabel('真实出栏时间（天）')
plt.ylabel('预测出栏时间（天）')
plt.title('种鸡最佳出栏时间线性回归拟合效果')
```

```
plt.legend()
plt.show()
```

⑤ 模型评估。

用均方误差（MSE）和决定系数（R^2）评估模型效果，判断模型的泛化能力。

```
from sklearn.metrics import mean_squared_error, r2_score
#计算均方误差
mse = mean_squared_error(y_test, y_pred)
#计算决定系数
r2 = r2_score(y_test, y_pred)
print(f"测试集 MSE：{mse:.2f}")
print(f"测试集 R²：{r2:.2f}")
#输出示例：测试集 MSE：19.09，测试集 R²：0.98
```

（3）模型评估指标：均方误差与决定系数的解读。

① 均方误差。

核心含义：衡量预测值与真实值的平均平方误差，其值越小，模型拟合效果越好。

解读示例：MSE=4.86 表示测试集中预测出栏时间与真实出栏时间的平均平方误差为4.86，实际平均误差为$\sqrt{4.86}\approx 2.21$天，误差较小，能满足养殖决策需求。

② 决定系数。

核心含义：衡量模型对数据规律的解释能力，取值范围为[0,1]，越接近 1，说明模型对数据的解释能力越强。

解读示例：R^2=0.96 表示模型能解释 96%的出栏时间变化，仅 4%的变化源于模型未考虑的其他因素（如突发疫病、市场临时调整），属于优秀模型。

3.3.4 从“线性”到“曲线”：回归模型的非线性拓展与泛化优化

线性回归仅能处理特征与出栏时间的线性关系，但现实中种鸡生长存在非线性规律（如前期日增重快，出栏时间对特征变化敏感；后期日增重放缓，敏感程度降低），此时需要用到多项式回归等非线性回归模型。同时，训练回归模型时需要避免欠拟合和过拟合问题，确保模型的泛化能力。

（1）多项式回归：处理“曲线关系”。

有时候，特征和出栏时间并不是直线关系。例如，“日增重”与“出栏时间”：日增重处于 0.05～0.1kg 阶段时，每提升 0.01kg，出栏时间可提前 3～4 天；但日增重超过 0.1kg 后，每提升 0.01kg，出栏时间仅提前 1 天左右——这时候用直线拟合会偏差很大，需要用“多项式回归”。公式为

$$y = a_0 + a_1x^1 + a_2x^2 + \cdots + a_nx^n$$

n 是多项式的“阶次”，n=2 时是二次多项式（拟合抛物线），n=3 时是三次多项式（拟合更复杂的曲线），如图 3-9 所示，其核心是把“非线性关系”转化为“线性关系”来处理。例如，对于二次多项式 $y=ax^2+bx+c$，我们可以把 x^2 看作一个新的特征 z，公式就变成

$y=az+bx+c$，这样就能用线性回归的方法求解参数了。

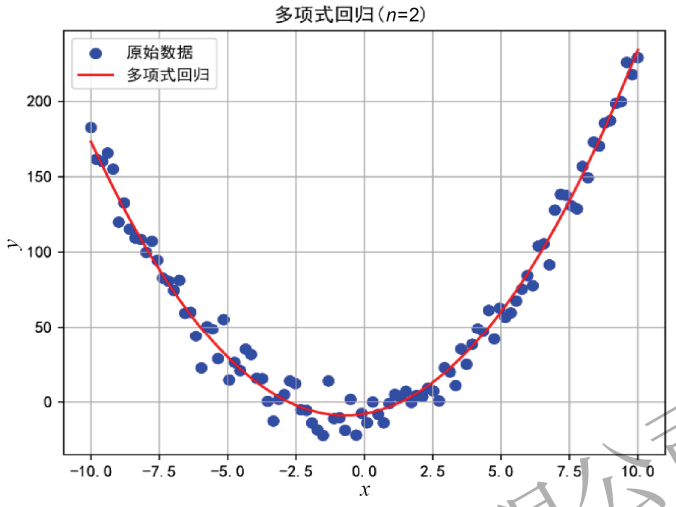


图 3-9　多项式回归示意图

（2）回归模型的常见问题：欠拟合与过拟合。

训练智慧养殖相关的回归模型时，最容易出现的两个问题是欠拟合与过拟合，这两个问题就像养殖户“凭经验养不好”和“照搬单一案例养不活”一样，会直接影响模型的泛化能力（如适配不同养殖场、不同批次鸡群），需要通过合理的方法规避。

① 欠拟合（学不会）。

表现：模型在训练数据上都预测不准，如用简单直线拟合“养殖日龄-出栏效益”的曲线关系，预测值与真实值误差很大。

原因：模型太简单，或者考虑的养殖特征太少。例如，只用“养殖日龄”预测出栏效益，却忽略了“日投喂量”“温/湿度控制”“疫病防控效果”等关键影响因素。

解决方法：补充更多养殖相关特征（如温/湿度、投喂量数据）、使用更复杂的模型（如把线性回归换成多项式回归）、调整模型参数提升拟合能力。

② 过拟合（学太死）。

表现：模型在某一个养殖场的训练数据上预测得特别准（如预测误差极小），但用到其他养殖场或新批次鸡群时，预测误差极大。

原因：模型太复杂，把训练数据中的“噪声”（如某批次鸡群偶然因天气突变导致的效益波动、个别只鸡的异常体重数据）当成了普遍的养殖规律。

解决方法：收集更多不同地域、不同规模养殖场的训练数据，覆盖多样养殖场景、简化模型（如把高次多项式回归换成简单线性回归）、使用“正则化”技术（相当于给模型“减负”，避免过度关注异常数据）、用交叉验证调整参数。

举例：用 15 阶多项式拟合某小型养殖场的“养殖日龄-出栏效益”数据，模型会在该养殖场的训练数据上拟合得近乎完美，但用到其他省份的规模化养殖场时，预测的出栏效益误差极大——这就是典型的过拟合，无法满足智慧养殖的规模化适配需求。

3.4 分类：预测“属于哪一类”

分类是 AI 监督学习的另一核心任务，与回归“预测连续值”的核心目标不同，分类聚焦预测离散类别——让 AI 通过学习事物的关键特征，将其精准划分到预先定义好的特定类别中。在智慧养殖场景中，这一技术有着极强的实用价值，如依据种鸡的鸡冠形态、体型、羽毛纹理、生长速率等多维度体征特征，让 AI 自动判断种鸡的公母归属。这种智能分类方式替代了传统人工鉴别效率低、易出错的痛点，为养殖场的精细化管理（如分群饲养、精准投喂、繁育规划）提供了高效支撑，而分类技术也凭借其广泛的场景适配性，成为 AI 赋能养殖产业、解决实际生产难题的重要工具。

3.4.1 分类及分类器：让 AI 学会“做选择题”

分类，通俗来讲就是让 AI 当“智能鉴别师”——先给它提供某只种鸡的一系列关键特征，如鸡冠大小与形态、体型肥瘦比例、羽毛纹理色泽、生长速率快慢等，再让专门的 AI 模型，如图 3-10 所示（我们称之为“分类器”），通过学习这些特征与性别间的映射关系，就像养殖老手凭借经验判断鸡的公母一样，最终精准输出这只种鸡对应的明确类别标签——“公鸡”或“母鸡”。

图 3-10　分类器示意图

分类和回归是 AI 监督学习的两大核心任务，其本质区别在于输出类型的不同。

3.4.2 逻辑回归分类器：解决“二分类问题”

逻辑回归是分类任务中最基础、最常用的模型之一，尽管名字中包含“回归”，但它本质上是一种二分类模型，核心优势是原理简单、计算快速、可解释性强，适合处理二分类问题（如“是/否”、“好/坏”和种鸡的“公/母”）。它的核心思路是“将线性回归的输出转化为分类概率”。它的工作流程如下。

（1）先像线性回归一样，计算一个线性结果（如根据肿瘤的特征、种鸡的体征计算一个数值）。

（2）用一个“逻辑函数”（也叫作 Sigmoid 函数）把这个数值转换成 0 到 1 之间的概率。

（3）设定一个阈值（如 0.5）：概率大于 0.5，就分到“正类”（如公鸡）；概率小于 0.5，就分到“负类”（如母鸡）。

Sigmoid 函数的图像像一个“S”形，如图 3-11 所示：当输入很大时，输出接近 1；当输入很小时，输出接近 0；当输入为 0 时，输出正好是 0.5——完美适配二分类的需求。

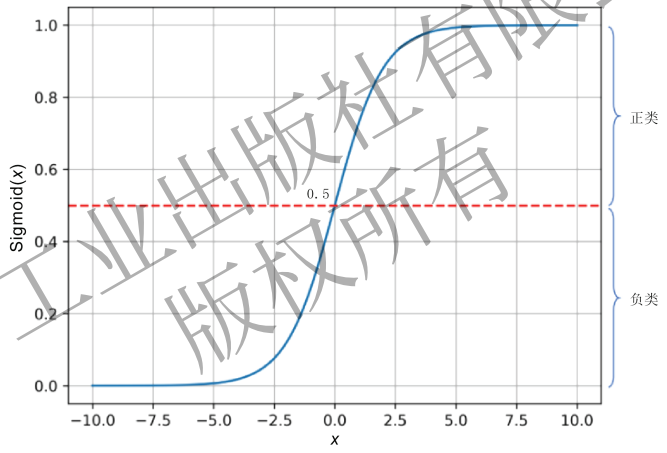


图 3-11 Sigmoid 函数的图像

我们用智慧养殖场景中的真实数据集来演示：该数据集包含数百个种鸡样本，每个种鸡样本都记录了多项关键体征特征，如鸡冠大小与形态、体型肥瘦比例、羽毛纹理色泽、生长速率快慢等，样本标签明确分为“公鸡”和“母鸡”两类。训练步骤如下。

（1）加载数据：读取种鸡的体征特征与“公/母”标签。

（2）划分数据集：训练集 80%，测试集 20%。

（3）训练模型：让模型学习“哪些特征是公鸡/母鸡的标志”（如公鸡的鸡冠特征更明显）。

（4）预测测试：用训练好的模型判断测试集中的种鸡是公鸡是母鸡。

（5）计算准确率：最终模型在测试集上的准确率达到非常高的水平——这意味着当该模型面对新的种鸡样本时，能精准识别其公母归属，判断正确率非常高，完全满足养殖场的实际应用需求。

```
# 导入工具库
import pandas as pd
```

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

# 1. 模拟加载部分种鸡体征数据集（特征+公母标签）
data = pd.DataFrame({
    '鸡冠大小': [9.2, 5.8, 8.8, 6.1, 9.5, 5.9],   # 体征特征 1
    '体型肥瘦比例': [0.81, 0.63, 0.79, 0.65, 0.83, 0.62],   # 体征特征 2
    '生长速率': [1.3, 0.7, 1.2, 0.8, 1.4, 0.75],   # 体征特征 3
    '性别标签': [1, 0, 1, 0, 1, 0]   # 1=公鸡 0=母鸡
})

# 2. 拆分特征/标签，按 8∶2 划分训练集/测试集
X = data[['鸡冠大小', '体型肥瘦比例', '生长速率']]
y = data['性别标签']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 3. 训练分类模型并预测
model = DecisionTreeClassifier()   # 简单易理解的分类模型
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

# 4. 计算并输出准确率
acc = accuracy_score(y_test, y_pred)
print(f"种鸡公母识别准确率：{acc*100:.2f}%")
print("模型可精准识别新样本，满足养殖场实际需求")
```


3.4.3 朴素贝叶斯分类器：基于“概率”的分类

朴素贝叶斯分类器的核心是“贝叶斯定理”。我们举一个种鸡养殖场景的例子：假设在某养殖场的种鸡群体中，公鸡占比 30%（100 只种鸡里有 30 只公鸡）；通过特征检测判断公母，其中对公鸡的识别准确率是 92%（真正的公鸡中 92%能被正确识别），对母鸡的识别准确率是 94%（真正的母鸡中 94%能被正确识别）。现在检测到一只种鸡符合公鸡的特征，它实际是公鸡的概率是多少？贝叶斯定理就是帮我们计算这个“后验概率”——在“检测符合公鸡特征”这个条件下，“实际是公鸡”的概率。简单来说，贝叶斯定理的逻辑是根据新的特征证据（如鸡冠大小、脚距长度等），更新我们对种鸡公母的概率判断。

朴素贝叶斯里的“朴素”是指一个简单的假设：每个特征之间都是相互独立的。例如，判断种鸡公母时，我们假设“鸡冠大小”“脚距长度”“体重”“羽色”这些特征互不影响——这个假设虽然不完全符合现实，但能让计算变得简单，且实际应用中效果往往较好。

假设有一个种鸡数据集，如表 3-6 所示，我们要预测一只种鸡的性别（公/母），观察到的特征为鸡冠大小=大、脚距长度=长、体重=重、羽色=鲜艳。

表 3-6 种鸡数据集示例

编号	鸡冠大小	脚距长度	体重	羽色	性别
1	大	长	重	鲜艳	公
2	大	长	重	暗淡	公
3	大	短	重	鲜艳	公
4	中	长	中	鲜艳	公
5	小	短	轻	暗淡	母
6	中	短	中	暗淡	母
7	小	长	轻	鲜艳	母

（1）计算先验概率——未观察特征时，种鸡是公鸡或母鸡的概率。

① 公鸡有 4 只，总样本有 7 个，所以 P（公鸡）=4/7≈0.57。

② 母鸡有 3 只，总样本有 7 个，所以 P（母鸡）=3/7≈0.43。

（2）计算条件概率——已知是公鸡/母鸡时，出现某个特征的概率。

① 公鸡中“鸡冠大小=大”的有 3 只，所以 P(大|公鸡)=3/4=0.75。

② 母鸡中“鸡冠大小=大”的有 0 只，所以 P(大|母鸡)=0/3=0。

采用同样的方法计算出其他特征的条件概率。

（3）计算后验概率——结合所有特征，种鸡是公鸡或母鸡的概率。

① P（公鸡|所有特征）=P（公鸡）×P（大|公鸡）×P（长|公鸡）×P（重|公鸡）×P（鲜艳|公鸡）≈0.57×0.75×0.75×0.75×0.75≈0.18。

② P（母鸡|所有特征）=P（母鸡）×P（大|母鸡）×P（长|母鸡）×P（重|母鸡）×P（鲜艳|母鸡）≈0.43×0×0.33×0×0.33≈0。

（4）分类——哪个概率大，就分到哪一类。

针对步骤（3）中的①，0.18>0，所以这只种鸡是公鸡。

【朴素贝叶斯预测种鸡公母示例】

我们用某养殖场的种鸡数据集（包含鸡冠大小、脚距长度、体重、羽色等特征，标签是性别）进行分类。

（1）读取数据：加载种鸡的特征和标签。

（2）划分数据集：训练集 80%，测试集 20%。

（3）训练模型：让模型学习“哪些特征组合对应种鸡性别”。

（4）预测评估：最终准确率是 83.2%——能较好地预测种鸡性别。

```
# 导入工具库
import pandas as pd
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
```

```
    # 读取数据
    data = pd.read_csv('chicken_gender.csv')
    # 特征编码：将类别特征转换为数值（鸡冠大小：小=0、中=1、大=2；脚距长度：短=0、中=1、长=2；体重：轻=0、中=1、重=2；羽色：暗淡=0、鲜艳=1）
    le = LabelEncoder()
    data['鸡冠大小'] = le.fit_transform(data['鸡冠大小'])
    data['脚距长度'] = le.fit_transform(data['脚距长度'])
    data['体重'] = le.fit_transform(data['体重'])
    data['羽色'] = le.fit_transform(data['羽色'])
    # 提取特征和标签
    X = data[['鸡冠大小', '脚距长度', '体重', '羽色']]  # 特征
    y = data['性别']  # 标签：公=1，母=0
    # 划分数据集
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    # 训练朴素贝叶斯模型
    model = GaussianNB()
    model.fit(X_train, y_train)
    # 预测测试集
    y_pred = model.predict(X_test)
    # 计算准确率
    accuracy = accuracy_score(y_test, y_pred)
    print("准确率：", accuracy)  # 输出：准确率：0.832model.fit(X_train, y_train)
    # 预测测试集
    y_pred = model.predict(X_test)  #计算准确率
    accuracy = accuracy_score(y_test, y_pred)
    print("准确率：", accuracy)  #输出：准确率：0.8185model.fit(X_train, y_train)
    # 预测测试集
    y_pred = model.predict(X_test)  #计算准确率
    accuracy = accuracy_score(y_test, y_pred)
    print("准确率：", accuracy)  #输出：准确率：0.8185
```

3.4.4 支持向量机：找到“最优决策边界”

支持向量机的核心思想很直观：在两类样本之间，找到一条“最优决策边界”，让这条边界离两类样本的距离都尽可能远——这样分类时误差最小，模型更稳定。如图 3-12 所示，我们要区分“大熊猫”和“金丝猴”的样本，在坐标图上，这两类样本各占一边。可以画很多条线把它们分开，但支持向量机要找的是“正中间”的那条线——它到两类样本的距离最大。

这条“最优决策边界”也叫作“最大间隔超平面”：在二维数据（如两个特征）中，它是一条直线；在三维数据中，它是一个平面；在更高维度中，它是“超平面”。

（1）线性可分：两类样本能通过一条直线（或超平面）完全分开，如上面“大熊猫”和“金丝猴”的例子。

（2）非线性可分：两类样本无法通过直线分开，如像“圆形”一样分布的样本——这时

候支持向量机会用到“核函数”这个神奇的工具。

核函数的作用：把低维空间中“非线性可分”的数据映射到高维空间，让它们变成“线性可分”的。就像把一张揉皱的纸展开，原本缠绕在一起的点，在展开后就能用直线分开，如图 3-13 所示。

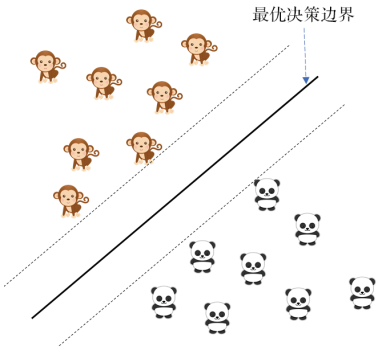


图 3-12　最优决策边界示意图

图 3-13　核函数示意图

在实际应用中，我们使用 Scikit-learn 库中的 SVC 类来实现支持向量机分类器，以种鸡性别预测为例，使用种鸡的鸡冠大小、脚距长度、体重、羽色 4 个特征，标签为性别，进行分类训练与预测。

```
from sklearn import datasets
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
# 读取种鸡数据集（模拟数据，实际可替换为真实数据集路径）
data = pd.read_csv('chicken_gender.csv')
# 特征编码：将类别特征转换为数值
le = LabelEncoder()
data['鸡冠大小'] = le.fit_transform(data['鸡冠大小'])  # 小=0、中=1、大=2
data['脚距长度'] = le.fit_transform(data['脚距长度'])  # 短=0、中=1、长=2
data['体重'] = le.fit_transform(data['体重'])  # 轻=0、中=1、重=2
data['羽色'] = le.fit_transform(data['羽色'])  # 暗淡=0、鲜艳=1
# 提取特征和标签
X = data[['鸡冠大小', '脚距长度', '体重', '羽色']]  # 4 个特征
y = le.fit_transform(data['性别'])  # 标签：公=1，母=0
# 查看数据集信息
print("数据集特征：", ['鸡冠大小', '脚距长度', '体重', '羽色'])
print("数据集标签：", ['母', '公'])
print("数据集大小：", X.shape)
print("数据集大小：", iris.data.shape)
```

常用的核函数如下。

（1）线性核函数：直接找直线边界，如图 3-14 所示。

（2）多项式核函数：处理轻微非线性数据，如图 3-15 所示（不同数据效果不同）。

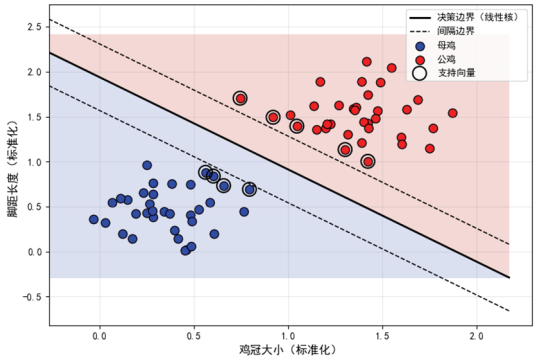


图 3-14　线性核函数示意图

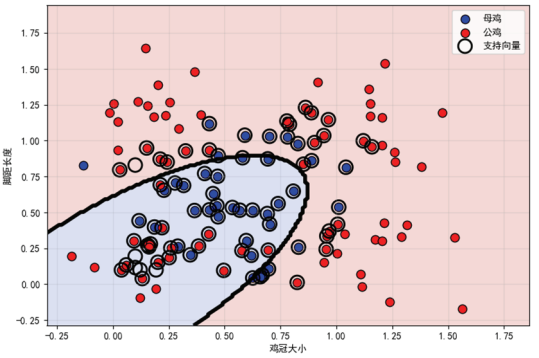


图 3-15　多项式核函数示意图

（3）高斯核函数（RBF）：处理复杂非线性数据；高斯核函数有 2 个重要参数（C 和 gamma），调整它们能改变模型的分类效果：C 越大，模型越“较真”，越容易过拟合；gamma 越大，模型越关注局部样本，也越容易过拟合，如图 3-16 所示。

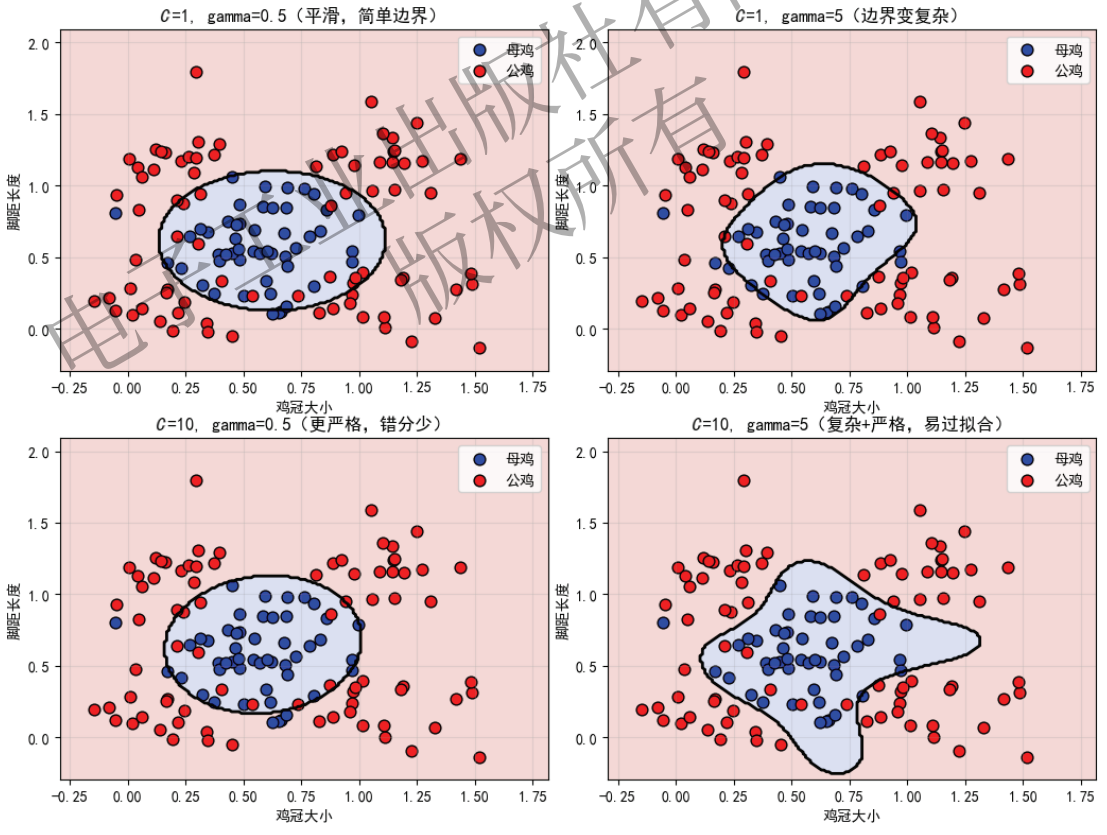


图 3-16　高斯核函数示意图

3.5 本章小结

本章围绕“基础认知–经典模型–工具实战”逻辑，系统构建了机器学习的核心知识体系：先明确机器学习“数据驱动”的本质，区分于传统编程的“规则驱动”，介绍其“训练–应用”两大阶段及“数据收集–特征提取–模型选择–训练评估–优化部署”标准化流程，并按学习方式将算法分为监督学习、无监督学习、强化学习 3 类；接着讲解数据集“样本–特征–标签”的构成及开放资源，重点阐述通过相关性分析和主成分分析实现数据降维的方法；随后聚焦无监督学习的聚类任务，详解 K-Means、AGNES 等算法的原理与实现；最后深入监督学习的回归与分类两大核心任务，前者以线性回归为核心，延伸多项式回归及欠/过拟合解决方案，后者系统介绍逻辑回归、朴素贝叶斯、支持向量机分类器的机制与案例。全章结合 Scikit-learn 库实战，兼顾理论与应用，既培养“数据思维”与“建模能力”，又为后续深度学习铺垫，所学知识可广泛应用于金融、医疗、环境监测等领域。

3.6 习题与测试

本章数字资源

1. 单选题

（1）在机器学习中，用于描述样本属性的信息被称为（　　）。

A. 标签　　B. 数据集　　C. 特征　　D. 模型

（2）以下哪项属于监督学习任务？（　　）

A. 聚类　　B. 降维　　C. 回归　　D. 关联规则挖掘

（3）主成分分析主要用于（　　）。

A. 分类　　B. 数据降维　　C. 模型评估　　D. 参数调优

（4）下列关于开放数据集的说法，正确的是（　　）。

A. 仅由企业私有发布　　B. 需要付费才能使用

C. 通常免费且可下载　　D. 不包含现实世界数据

（5）在聚类算法中，基于“密度相连”思想的算法是（　　）。

A. K-Means　　B. AGNES　　C. DBSCAN　　D. STING

（6）下列任务属于回归问题的是（　　）。

A. 预测邮件是否为垃圾邮件　　B. 预测用户年龄

C. 预测图像中的数字类别（0～9）　　D. 预测明天是否下雨

（7）回归问题的目标是预测（　　）。

A. 离散类别　　B. 连续值　　C. 概率　　D. 聚类中心

（8）线性回归的损失函数通常是（　　）。

A. 交叉熵损失　　B. 均方误差

C. Hinge 损失　　D. KL 散度

（9）训练集的主要作用是（　　）。

A. 训练模型　　B. 调优模型　　C. 最终评估　　D. 以上所有

（10）朴素贝叶斯分类器的“朴素”假设是（　　）。

A. 特征相互独立　　B. 特征权重相等

C. 类别先验概率相同　　D. 特征服从高斯分布

（11）以下哪种评估指标用于分类任务的精度衡量？（　　）

A. 均方误差　　B. 准确率　　C. 决定系数　　D. 调整决定系数

（12）回归与分类的共同点是（　　）。

A. 均属于无监督学习　　B. 均预测连续值

C. 均属于监督学习　　D. 均不需要标签

（13）支持向量机的核心思想是（　　）。

A. 寻找最大间隔超平面　　B. 最小化损失函数

C. 聚类相似样本　　D. 最大化分类准确率

（14）支持向量机在非线性可分数据中通过（　　）实现分类。

A. 线性回归　　B. 核技巧　　C. 决策树　　D. 随机森林

（15）机器学习中的“过拟合”主要表现为（　　）。

A. 模型在训练集上表现太好　　B. 模型在训练集上表现太差

C. 模型在新数据上表现太好　　D. 模型在新数据上表现太差

2. 判断题

（1）监督学习不需要有标签数据即可训练模型。（　　）

（2）样本是构成数据集的基本单位，包含特征和可能的标签。（　　）

（3）皮尔逊相关系数的取值范围为[0,1]，表示变量间的正相关程度。（　　）

（4）主成分分析的第一步是对数据进行标准化处理。（　　）

（5）层次聚类只能采用自顶向下的分裂方式构建层状结构。（　　）

（6）监督学习的分类算法需要有标签的训练数据。（　　）

（7）逻辑回归只能用于二分类问题。（　　）

（8）支持向量机的核函数可以将低维数据映射到高维空间。（　　）

（9）过拟合可以通过增加数据量完全解决。（　　）

（10）梯度下降法一定可以找到全局最优解。（　　）

3. 填空题

（1）在监督学习中，用于预测连续值的任务称为_____。

（2）在主成分分析中，选取主成分的依据通常是累计方差贡献率达到_____%以上。

（3）热图常用于可视化_____矩阵。

（4）K-Means 属于_____式聚类算法。

（5）在强化学习中，智能体通过最大化累积_____来优化行为策略。

（6）回归分析的主要任务是预测_____值（如房价、温度、销售额），其输出具有连续性和可细分性。

（7）当数据非线性可分时，支持向量机使用_____函数进行映射。

（8）在监督学习中，回归和分类任务均需要使用_____数据（训练集包含特征和对应的标签）。

（9）逻辑回归通过_____函数将线性回归的输出映射到(0,1)区间，从而表示概率。

（10）在模型评估中，决定系数越接近_____，表示回归模型拟合效果越好。