机器学习核心理论与应用

监督学习 · 无监督学习 · 强化学习 · 特征工程 · 模型评估

从理论到实践，全面掌握机器学习方法

[scale=0.9]

[circle, draw, minimum size=0.6cm, fill=blue!20, line width=1pt] (x1) at (0, -1.8) ; [circle, draw, minimum size=0.6cm, fill=blue!20, line width=1pt] (x2) at (0, -0.6) ; [circle, draw, minimum size=0.6cm, fill=blue!20, line width=1pt] (x3) at (0, 0.6) ; [circle, draw, minimum size=0.6cm, fill=blue!20, line width=1pt] (x4) at (0, 1.8) ;

[circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h11) at (3, -4.2) ; [circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h12) at (3, -3.0) ; [circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h13) at (3, -1.8) ; [circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h14) at (3, -0.6) ; [circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h15) at (3, 0.6) ; [circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h16) at (3, 1.8) ; [circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h17) at (3, 3.0) ; [circle, draw, minimum size=0.5cm, fill=green!20, line width=1pt] (h18) at (3, 4.2) ;

[circle, draw, minimum size=0.5cm, fill=orange!20, line width=1pt] (h21) at (6, -1.8) ; [circle,

draw, minimum size=0.5cm, fill=orange!20, line width=1pt] (h22) at (6, -0.6) ; [circle, draw, minimum size=0.5cm, fill=orange!20, line width=1pt] (h23) at (6, 0.6) ; [circle, draw, minimum size=0.5cm, fill=orange!20, line width=1pt] (h24) at (6, 1.8) ;

[circle, draw, minimum size=0.6cm, fill=red!20, line width=1pt] (y) at (9, 0) ;

ıin 1,...,4 ȷin 1,...,8 [->, gray!50, line width=0.3pt] (xı) – (h1ȷ);

ıin 1,...,8 ȷin 1,...,4 [->, gray!50, line width=0.3pt] (h1ı) – (h2ȷ);

ıin 1,...,4 [->, gray!50, line width=0.3pt] (h2ı) – (y);

机器学习核心理论与应用

2026 年 1 月 6 日

目录

Part I

第一部分：机器学习基础

1 引言

机器学习（Machine Learning, ML）是人工智能的核心分支，通过从数据中自动学习模式和规律，使计算机系统能够完成特定任务而无需显式编程。机器学习在图像识别、自然语言处理、推荐系统、医疗诊断、金融风控等领域取得了显著成功。

机器学习的独特价值：

- **处理高维特征**：能够处理包含数千甚至数万个特征的数据，自动发现重要特征
- **捕捉非线性关系**：通过非线性模型（如神经网络、核方法）捕捉数据中的复杂模式
- **自适应学习**：能够从新数据中持续学习，适应数据分布的变化
- **端到端学习**：直接从原始数据学习到最终输出，减少人工特征工程的需求

本文档系统性地介绍机器学习的核心理论、经典算法和实际应用，涵盖监督学习、无监督学习、强化学习等主要范式，以及特征工程、模型评估、集成学习等重要技术。

2 机器学习基础

机器学习根据学习方式可以分为三大类：监督学习、无监督学习和强化学习。每种范式适用于不同类型的问题和场景。

2.1 监督学习

定义 2.1 (监督学习). *监督学习（Supervised Learning）是从带标签的训练数据中学习一个映射函数 $f : \mathcal{X} \to \mathcal{Y}$，使得对于新的输入 $\mathbf{x} \in \mathcal{X}$，能够预测其标签 $y \in \mathcal{Y}$。*

给定训练数据集 $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$，其中 $\mathbf{x}_i \in \mathbb{R}^d$ 是特征向量，y_i 是对应的标签，目标是学习函数 f 使得 $f(\mathbf{x}_i) \approx y_i$。

通俗解释：监督学习就像有老师指导的学习。老师提供大量” 题目-答案” 对（训练数据），学生（模型）通过学习这些例子，掌握解题方法，然后能够解答新的题目。

2.1.1 分类问题

分类问题的目标是预测离散的类别标签。根据类别数量，可以分为：

- **二分类**：只有两个类别，如垃圾邮件分类（垃圾/正常）、疾病诊断（患病/健康）
- **多分类**：有多个类别，如图像分类（猫/狗/鸟等）、手写数字识别（0-9）

典型应用：

例 2.1 (垃圾邮件分类)**. 问题**：自动识别邮件是否为垃圾邮件。

数据：

- 特征：邮件内容中的词频、发件人信息、邮件标题等
- 标签：垃圾邮件（*1*）或正常邮件（*0*）

方法：使用朴素贝叶斯、逻辑回归或支持向量机等分类算法。

实际应用：*Gmail* 使用机器学习模型过滤垃圾邮件，准确率超过 *99%*。

2.1.2 回归问题

回归问题的目标是预测连续的数值。

典型应用：

例 2.2 (房价预测)**. 问题**：根据房屋特征预测房价。

数据：

- 特征：面积、卧室数、位置、建造年份等
- 标签：房价（连续值）

方法：使用线性回归、决策树回归或神经网络等算法。

实际应用：*Zillow* 的 *Zestimate* 使用机器学习模型预测房价，帮助用户了解房产价值。

2.1.3 监督学习的优势与局限性

优势：

- 有明确的优化目标（最小化预测误差）

- 可以使用大量标注数据进行训练
- 模型性能可以通过测试集准确评估
- 理论基础相对完善

局限性:

- 需要大量标注数据，标注成本高
- 对标注质量要求高，错误标注会影响模型性能
- 难以处理标注数据稀缺的场景
- 模型可能过拟合训练数据

2.2 无监督学习

定义 2.2 (无监督学习)**.** *无监督学习（Unsupervised Learning）是从无标签的数据中学习数据的内在结构和模式，不需要人工标注。*

给定数据集 $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$，其中只有特征向量，没有标签，目标是发现数据中的隐藏模式、结构或分布。

通俗解释：无监督学习就像没有老师指导的自学。学生只能看到大量数据，需要自己发现其中的规律和模式，比如哪些数据相似、数据如何分组等。

2.2.1 聚类

聚类是将相似的数据点分组的过程。

典型应用:

例 2.3 (客户细分)**. 问题**：*根据客户的购买行为将其分为不同的群体。*

数据：*客户的购买历史、浏览记录、消费金额等特征（无标签）。*

方法：*使用 K-means、层次聚类等算法将客户分为若干群体。*

实际应用：*电商平台使用聚类分析进行客户细分，为不同群体提供个性化推荐和营销策略。*

2.2.2 降维

降维是将高维数据映射到低维空间，保留主要信息。

典型应用：

例 2.4 (数据可视化). **问题**：将高维数据（如 *1000* 维）可视化到 *2D* 或 *3D* 空间。

方法：使用 *PCA*（主成分分析)、*t-SNE*、*UMAP* 等降维算法。

实际应用：在生物信息学中，使用 *t-SNE* 将基因表达数据降维到 *2D*，可视化不同细胞类型的分布。

2.2.3 无监督学习的优势与局限性

优势：

- 不需要标注数据，数据获取成本低
- 可以发现人类未预见的模式和结构
- 适用于探索性数据分析
- 可以作为监督学习的预处理步骤

局限性：

- 没有明确的优化目标，评估困难
- 结果可能难以解释和验证
- 对参数选择敏感（如聚类数量）
- 可能发现无意义的模式

2.3 强化学习

定义 2.3 (强化学习). 强化学习（*Reinforcement Learning, RL*）是智能体（*Agent*）通过与环境的交互来学习最优策略，通过试错和奖励信号来指导学习过程。

强化学习问题可以建模为马尔可夫决策过程（*MDP*)：$(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$，其中：

- $\mathcal{S}$：状态空间
- $\mathcal{A}$：动作空间

- $\mathcal{P}$：状态转移概率
- $\mathcal{R}$：奖励函数
- γ：折扣因子

智能体的目标是学习策略 $\pi : \mathcal{S} \to \mathcal{A}$，最大化累积奖励：

$$R = \sum_{t=0}^{\infty} \gamma^t r_t$$

通俗解释：强化学习就像训练宠物。宠物（智能体）做出动作（如坐下），主人（环境）给出奖励（如给食物）或惩罚（如不给食物）。通过反复尝试，宠物学会在什么情况下做什么动作能获得最多奖励。

2.3.1 强化学习的核心概念

- **状态（State）**：环境在某个时刻的完整描述
- **动作（Action）**：智能体可以执行的操作
- **奖励（Reward）**：环境对智能体动作的反馈信号
- **策略（Policy）**：从状态到动作的映射
- **价值函数（Value Function）**：评估状态或动作的长期价值

2.3.2 典型应用

例 2.5 (游戏 AI)**. 问题**：训练 *AI* 在游戏中达到人类或超人类水平。

方法：

- *Deep Q-Network (DQN)*：使用深度神经网络近似 Q 函数
- *Policy Gradient*：直接优化策略函数
- *Actor-Critic*：结合价值函数和策略梯度

实际应用：

- ***AlphaGo***：*DeepMind* 开发的围棋 *AI*，*2016* 年击败世界冠军李世石
- ***AlphaStar***：在《星际争霸 *II*》中达到大师级水平

- ***OpenAI Five***：在 *Dota 2* 中击败职业战队

例 2.6 (机器人控制)**. 问题**：训练机器人完成复杂任务，如抓取物体、行走等。

方法：使用强化学习让机器人在仿真或真实环境中通过试错学习。

实际应用：*Boston Dynamics* 的机器人使用强化学习进行运动控制，实现复杂的平衡和移动能力。

2.3.3 强化学习的优势与局限性

优势:

- 适用于序列决策问题
- 可以通过试错学习，不需要大量标注数据
- 能够学习长期策略
- 适用于动态环境

局限性:

- 训练过程可能很慢，需要大量交互
- 奖励函数设计困难
- 探索与利用的平衡问题
- 安全性问题（在关键应用中）

Part II

第二部分：经典算法与特征工程

3 经典算法

本节介绍机器学习中的经典算法，包括线性回归、决策树、随机森林和支持向量机。这些算法虽然相对简单，但在许多实际问题中仍然非常有效。

3.1 线性回归

线性回归是监督学习中最基础的算法之一，用于解决回归问题。

定义 3.1 (线性回归). *线性回归假设目标变量 y 与特征向量 $\mathbf{x}$ 之间存在线性关系：*

$$y = \mathbf{w}^T\mathbf{x} + b + \epsilon$$

其中 $\mathbf{w} \in \mathbb{R}^d$ 是权重向量，$b \in \mathbb{R}$ 是偏置项，ϵ 是误差项（通常假设 $\epsilon \sim \mathcal{N}(0, \sigma^2)$）。

给定训练数据 $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$，线性回归的目标是找到参数 $\mathbf{w}$ 和 b，使得预测误差最小。

通俗解释：线性回归就像用一条直线去拟合数据点。例如，用一条直线拟合” 房屋面积-房价” 的关系，直线的斜率和截距就是学到的参数。

3.1.1 最小二乘法

最常用的方法是最小二乘法（Least Squares），最小化平方误差：

$$L(\mathbf{w}, b) = \sum_{i=1}^{n}(y_i - (\mathbf{w}^T\mathbf{x}_i + b))^2$$

通过求导并令导数为零，可以得到闭式解：

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

其中 $\mathbf{X} \in \mathbb{R}^{n\times d}$ 是特征矩阵，$\mathbf{y} \in \mathbb{R}^n$ 是标签向量。

3.1.2 正则化

为了防止过拟合，可以加入正则化项：

- **Ridge 回归（L_2 正则化）**：

$$L_{\text{Ridge}} = \sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 + \lambda\|\mathbf{w}\|_2^2$$

  其中 $\lambda > 0$ 是正则化系数。Ridge 回归倾向于产生较小的权重。

- **Lasso 回归（L_1 正则化）**：

$$L_{\text{Lasso}} = \sum_{i=1}^{n}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 + \lambda\|\mathbf{w}\|_1$$

  Lasso 回归可以产生稀疏解（许多权重为 0），实现特征选择。

3.1.3 应用场景

例 3.1 (房价预测). 使用线性回归预测房价，特征包括面积、卧室数、位置等。*Ridge* 回归可以防止过拟合，*Lasso* 回归可以自动选择重要特征。

例 3.2 (股票价格预测). 使用历史价格、交易量等特征预测未来股价。虽然股价受多种因素影响，线性回归可以作为基准模型。

3.1.4 优势与局限性

优势:

- 简单易懂，计算效率高
- 有闭式解，训练快速
- 可解释性强（权重表示特征重要性）
- 适合作为基准模型

局限性:

- 只能捕捉线性关系
- 对异常值敏感
- 假设特征之间相互独立
- 需要特征工程（如多项式特征）来捕捉非线性

3.2 决策树

决策树是一种基于树结构的分类和回归算法，通过一系列规则进行决策。

定义 3.2 (决策树). 决策树（*Decision Tree*）是一个树形结构，其中:

- 内部节点: 表示特征测试
- 分支: 表示测试结果
- 叶节点: 表示类别标签或数值

从根节点到叶节点的路径对应一条决策规则。

通俗解释: 决策树就像医生诊断疾病的流程。先问" 是否发烧？ "，如果" 是" 再问" 是否咳嗽？ "，根据一系列问题的答案，最终得出诊断结果。

3.2.1 构建决策树

决策树的构建是一个递归过程：

Algorithm 1 构建决策树

Require: 训练数据集 $\mathcal{D}$，特征集 $\mathcal{F}$
Ensure: 决策树 T
  if $\mathcal{D}$ 中所有样本属于同一类别 **then**
    返回叶节点，标记为该类别
  else if $\mathcal{F}$ 为空或 $\mathcal{D}$ 为空 **then**
    返回叶节点，标记为 $\mathcal{D}$ 中多数类
  else
    选择最优特征 $f^* = \arg\max_{f \in \mathcal{F}} \mathrm{IG}(\mathcal{D}, f)$
    为 f^* 的每个可能值创建分支
    for 每个分支 **do**
      $D_v = \mathcal{D}$ 中 $f^* = v$ 的样本子集
      if D_v 为空 **then**
        添加叶节点，标记为 $\mathcal{D}$ 中多数类
      else
        递归构建子树：$\mathrm{BuildTree}(D_v, \mathcal{F} \setminus \{f^*\})$
      end if
    end for
  end if

3.2.2 特征选择准则

常用的特征选择准则包括：

- **信息增益（Information Gain）**：

$$\mathrm{IG}(\mathcal{D}, f) = H(\mathcal{D}) - \sum_{v} \frac{|\mathcal{D}_v|}{|\mathcal{D}|} H(\mathcal{D}_v)$$

  其中 $H(\mathcal{D})$ 是数据集的熵。

- **基尼不纯度（Gini Impurity）**：

$$G(\mathcal{D}) = 1 - \sum_{k} p_k^2$$

  其中 p_k 是类别 k 在数据集中的比例。

- **基尼增益**:

$$\text{GiniGain}(\mathcal{D}, f) = G(\mathcal{D}) - \sum_v \frac{|\mathcal{D}_v|}{|\mathcal{D}|} G(\mathcal{D}_v)$$

3.2.3 剪枝

为了防止过拟合，需要对决策树进行剪枝:

- **预剪枝**：在构建过程中提前停止（如限制树深度、最小样本数）
- **后剪枝**：构建完整树后，自底向上删除节点，用叶节点替代

3.2.4 应用场景

例 3.3 (医疗诊断). 使用决策树根据症状（发烧、咳嗽、头痛等）诊断疾病。决策树的可解释性使得医生可以理解诊断依据。

例 3.4 (信用评估). 银行使用决策树评估贷款申请人的信用风险，根据收入、工作年限、信用历史等特征做出决策。

3.2.5 优势与局限性

优势:

- 可解释性强，决策过程清晰
- 可以处理数值和类别特征
- 不需要特征缩放
- 可以捕捉非线性关系

局限性:

- 容易过拟合
- 对数据的小变化敏感（不稳定）
- 倾向于选择具有更多取值的特征
- 难以处理特征之间的交互

3.3 随机森林

随机森林（Random Forest）是决策树的集成方法，通过组合多个决策树来提高性能。

定义 3.3 (随机森林). 随机森林由 B 棵决策树组成，每棵树在训练时：

1. 使用自助采样（*Bootstrap Sampling*）从训练集中采样
2. 在每个节点分裂时，随机选择 k 个特征（通常 $k = \sqrt{d}$）进行考虑

预测时，对于分类问题使用投票，对于回归问题使用平均。

通俗解释：随机森林就像一群专家投票做决策。每个专家（决策树）根据自己的经验（不同的训练数据）给出意见，最终综合所有专家的意见做出决策。这样比单个专家更可靠。

3.3.1 算法流程

Algorithm 2 随机森林训练

Require: 训练数据集 $\mathcal{D}$，树的数量 B，特征采样数 k
Ensure: 随机森林 $\{T_1, T_2, \ldots, T_B\}$
  for $b = 1$ to B **do**
    使用自助采样从 $\mathcal{D}$ 中采样得到 $\mathcal{D}_b$
    使用 $\mathcal{D}_b$ 训练决策树 T_b，在每个节点随机选择 k 个特征
  end for

3.3.2 随机性的作用

随机森林通过两种随机性提高性能：

- **样本随机性**：每棵树使用不同的训练样本（自助采样）
- **特征随机性**：每棵树在每个节点考虑不同的特征子集

这种随机性使得各棵树之间具有多样性，减少过拟合，提高泛化能力。

3.3.3 应用场景

例 3.5 (图像分类). 在 *CIFAR-10* 数据集上，随机森林可以作为基准模型，虽然不如深度学习模型，但训练速度快，可解释性强。

例 3.6 (特征重要性分析). 随机森林可以计算特征重要性，帮助理解哪些特征对预测最重要。这在特征工程和模型解释中很有价值。

3.3.4 优势与局限性

优势:

- 性能通常优于单棵决策树
- 对过拟合有较强的抵抗力
- 可以处理高维特征
- 可以计算特征重要性
- 训练可以并行化

局限性:

- 模型可解释性不如单棵决策树
- 需要更多内存和计算资源
- 对于某些问题，可能不如梯度提升方法（如 XGBoost）

3.4 支持向量机

支持向量机（Support Vector Machine, SVM）是一种强大的分类算法，基于最大间隔原理。

定义 3.4 (支持向量机). *对于线性可分的二分类问题，SVM 寻找一个超平面 $\mathbf{w}^T\mathbf{x}+b=0$，使得两类样本之间的间隔（margin）最大。*

间隔定义为两类样本到超平面的最小距离:

$$margin = \frac{2}{\|\mathbf{w}\|}$$

SVM 的优化目标是:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 \quad s.t. \quad y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \forall i$$

通俗解释: SVM 就像在两个群体之间画一条” 最宽的路”。这条路要尽可能宽，同时要确保两边的群体都在路的正确一侧。支持向量就是那些” 站在路边” 的样本点。

3.4.1 软间隔 SVM

对于线性不可分的情况，引入松弛变量 ξ_i，允许一些样本分类错误：

$$\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

约束条件：

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

其中 $C > 0$ 是惩罚参数，控制对误分类的惩罚程度。

3.4.2 核技巧

对于非线性问题，使用核函数将数据映射到高维空间，在高维空间中线性可分：

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$$

常用的核函数包括：

- **多项式核**：$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T\mathbf{x}_j + 1)^d$
- **径向基函数（RBF）核**：$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2)$
- **Sigmoid 核**：$K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\alpha\mathbf{x}_i^T\mathbf{x}_j + c)$

3.4.3 对偶形式

SVM 的对偶形式为：

$$\max_{\boldsymbol{\alpha}} \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

约束条件：

$$\sum_{i=1}^{n}\alpha_i y_i = 0, \quad 0 \leq \alpha_i \leq C$$

对偶形式的优势：

- 只需要计算核函数，不需要显式映射到高维空间
- 支持向量（$\alpha_i > 0$）的数量通常远小于样本数
- 更容易扩展到大规模问题

3.4.4 应用场景

例 3.7 (文本分类). *SVM 在文本分类任务中表现优异，特别是在小样本情况下。使用 TF-IDF 特征和 RBF 核，可以在新闻分类、情感分析等任务中取得良好效果。*

例 3.8 (图像分类). *在图像分类任务中，SVM 可以作为特征分类器。例如，使用 CNN 提取特征，然后用 SVM 进行分类，这在某些情况下比端到端的 CNN 更有效。*

3.4.5 优势与局限性

优势:

- 在中小规模数据集上表现优异
- 通过核函数可以处理非线性问题
- 理论基础完善（基于统计学习理论）
- 对过拟合有较好的控制
- 支持向量提供了模型的稀疏表示

局限性:

- 对大规模数据集计算成本高
- 对特征缩放敏感
- 核函数和参数选择需要经验
- 可解释性不如决策树
- 概率输出需要额外处理（Platt scaling）

4 特征工程

特征工程是机器学习中至关重要的一环，好的特征可以显著提升模型性能。特征工程包括特征选择、特征变换和特征编码等。

4.1 特征选择

特征选择是从原始特征中选择最有用的特征子集，减少维度，提高模型性能和可解释性。

4.1.1 过滤方法（Filter Methods）

过滤方法基于特征的统计特性进行选择，独立于具体的学习算法：

- **方差选择**：移除方差很小的特征（几乎不变的特征）
- **相关系数**：选择与目标变量相关性高的特征
- **互信息**：选择与目标变量互信息大的特征
- **卡方检验**：用于分类问题，检验特征与标签的独立性

4.1.2 包装方法（Wrapper Methods）

包装方法使用学习算法来评估特征子集：

- **前向选择**：从空集开始，逐步添加最有用的特征
- **后向消除**：从完整特征集开始，逐步移除最无用的特征
- **递归特征消除（RFE）**：递归地移除最不重要的特征

4.1.3 嵌入方法（Embedded Methods）

嵌入方法在模型训练过程中进行特征选择：

- **Lasso 回归**：L_1 正则化自动产生稀疏解
- **决策树**：通过特征重要性进行选择
- **随机森林**：通过特征重要性排序

4.2 特征变换

特征变换是对特征进行数学变换，改变特征的分布或关系。

4.2.1 标准化和归一化

- **标准化（Z-score）**：

$$z = \frac{x - \mu}{\sigma}$$

将特征转换为均值为 0、标准差为 1 的分布。

- **最小-最大归一化**：

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

将特征缩放到 $[0, 1]$ 区间。

- **Robust 缩放**：使用中位数和四分位距，对异常值更鲁棒。

为什么需要特征缩放：

- 许多算法（如 SVM、K-means、神经网络）对特征尺度敏感
- 梯度下降算法在特征尺度不一致时收敛慢
- 距离-based 算法（如 KNN）受特征尺度影响大

4.2.2 多项式特征

通过创建特征的多项式组合来捕捉非线性关系：

$$(x_1, x_2) \to (x_1, x_2, x_1^2, x_1 x_2, x_2^2)$$

应用场景：线性回归无法捕捉非线性关系时，可以使用多项式特征。

4.2.3 对数变换

对偏态分布进行对数变换，使其更接近正态分布：

$$x' = \log(x + 1)$$

应用场景：处理价格、收入等右偏分布的数据。

4.3 特征编码

特征编码是将类别特征转换为数值特征的过程。

4.3.1 独热编码（One-Hot Encoding）

将类别特征转换为二进制向量：

例 4.1. 颜色特征：$[\text{"红"}, \text{"绿"}, \text{"蓝"}] \to$

- 红：$[1, 0, 0]$
- 绿：$[0, 1, 0]$
- 蓝：$[0, 0, 1]$

优势：不引入类别间的顺序关系。

局限性：类别数量多时会产生高维稀疏特征。

4.3.2 标签编码（Label Encoding）

将类别映射为整数：

例 4.2 (" 低", " 中", " 高"). → *[0, 1, 2]*

注意：只适用于有序类别，否则会引入虚假的顺序关系。

4.3.3 目标编码（Target Encoding）

使用目标变量的统计量对类别进行编码：

$$x' = \frac{\sum_{i:x_i=x} y_i}{|\{i : x_i = x\}|}$$

即用该类别的平均目标值作为编码。

优势：可以捕捉类别与目标的关系。

注意：需要防止过拟合（如使用交叉验证）。

Part III

第三部分：模型评估与高级方法

5 模型评估与验证

模型评估是机器学习流程中的关键步骤，用于衡量模型性能、选择最佳模型和防止过拟合。

5.1 评估指标

5.1.1 分类问题指标

- **准确率（Accuracy）**：
$$\text{Accuracy} = \frac{\text{正确预测数}}{\text{总样本数}}$$
- **精确率（Precision）**：
$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$
预测为正例中真正为正例的比例。
- **召回率（Recall）**：
$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$
真正例中被正确预测的比例。
- **F1 分数**：
$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$
精确率和召回率的调和平均。
- **ROC 曲线和 AUC**：ROC 曲线以假正例率为横轴，真正例率为纵轴。AUC（曲线下面积）衡量分类器的整体性能。

5.1.2 回归问题指标

- **均方误差（MSE）**：
$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$
- **均方根误差（RMSE）**：
$$\text{RMSE} = \sqrt{\text{MSE}}$$
- **平均绝对误差（MAE）**：
$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$
- **决定系数（R^2）**：
$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$
衡量模型解释的方差比例。

5.2 交叉验证

交叉验证是评估模型泛化能力的重要方法。

5.2.1 K 折交叉验证

将数据集分为 K 折，每次使用 $K-1$ 折训练，剩余 1 折测试，重复 K 次：

Algorithm 3 K 折交叉验证

Require: 数据集 $\mathcal{D}$，折数 K，学习算法 $\mathcal{A}$

Ensure: 平均性能指标

将 $\mathcal{D}$ 随机分为 K 折：$\mathcal{D}_1, \ldots, \mathcal{D}_K$

for $k = 1$ to K **do**

训练集：$\mathcal{D}_{\text{train}} = \mathcal{D} \setminus \mathcal{D}_k$

测试集：$\mathcal{D}_{\text{test}} = \mathcal{D}_k$

使用 $\mathcal{D}_{\text{train}}$ 训练模型 $M_k = \mathcal{A}(\mathcal{D}_{\text{train}})$

在 $\mathcal{D}_{\text{test}}$ 上评估性能 s_k

end for

return $\bar{s} = \frac{1}{K}\sum_{k=1}^{K} s_k$

优势：

- 充分利用数据
- 提供性能估计的方差
- 减少对数据划分的依赖

常见选择：$K = 5$ 或 $K = 10$。

5.2.2 留一法交叉验证（LOOCV）

$K = n$ 的特殊情况，每次留一个样本作为测试集。计算成本高，但无偏估计。

5.3 过拟合与欠拟合

定义 5.1 (过拟合). 过拟合（*Overfitting*）是指模型在训练集上表现很好，但在测试集上表现较差的现象。模型过度学习了训练数据的噪声和细节，导致泛化能力差。

定义 5.2 (欠拟合). 欠拟合（*Underfitting*）是指模型在训练集和测试集上都表现较差的现象。模型过于简单，无法捕捉数据中的基本模式。

通俗解释：

- **过拟合**：就像学生死记硬背了所有练习题，但遇到新题目就不会了
- **欠拟合**：就像学生只学了基础知识，连练习题都做不好

5.3.1 识别过拟合和欠拟合

- **过拟合的迹象**：
  - 训练误差很小，但验证误差很大
  - 模型复杂度高（如深度很深的决策树）
  - 训练集和验证集性能差距大
- **欠拟合的迹象**：
  - 训练误差和验证误差都很大
  - 模型复杂度低（如线性模型处理非线性问题）
  - 模型无法捕捉数据的基本模式

5.3.2 解决方法

解决过拟合：

- 增加训练数据
- 减少模型复杂度
- 正则化（L_1、L_2）
- Dropout（神经网络）
- 早停（Early Stopping）

解决欠拟合：

- 增加模型复杂度
- 增加特征
- 减少正则化
- 增加训练时间

5.4 偏差-方差权衡

定义 5.3 (偏差和方差).

- **偏差（*Bias*）**：模型的期望预测与真实值的差异，衡量模型的拟合能力
- **方差（*Variance*）**：模型在不同训练集上预测的差异，衡量模型的稳定性

总误差可以分解为：

$$Error = Bias^2 + Variance + Irreducible\ Error$$

通俗解释：

- **高偏差**：模型太简单，无法捕捉数据模式（欠拟合）
- **高方差**：模型太复杂，对训练数据的小变化敏感（过拟合）

5.4.1 偏差-方差权衡

- **简单模型**（如线性回归）：
  - 高偏差，低方差
  - 可能欠拟合
- **复杂模型**（如深度神经网络）：
  - 低偏差，高方差
  - 可能过拟合
- **理想模型**：
  - 低偏差，低方差
  - 需要合适的模型复杂度和正则化

6 集成学习方法

集成学习通过组合多个基学习器来提高预测性能，是机器学习中的重要技术。

6.1 Bagging

Bagging（Bootstrap Aggregating）通过训练多个模型并平均预测结果来减少方差。

定义 6.1 (Bagging)**.** *Bagging 算法*:

1. *使用自助采样从训练集中生成 B 个不同的训练集*
2. *在每个训练集上训练一个基学习器*
3. *对于分类问题使用投票，对于回归问题使用平均*

通俗解释: Bagging 就像多个专家独立给出意见，然后综合所有意见做决策。每个专家看到的数据略有不同，但综合起来更可靠。

6.1.1 随机森林

随机森林是 Bagging 的特例，基学习器是决策树，并在特征选择时引入随机性。

6.1.2 优势与局限性

优势:

- 减少方差，提高泛化能力
- 可以并行训练
- 对过拟合有抵抗力

局限性:

- 不能减少偏差
- 需要足够的计算资源

6.2 Boosting

Boosting 通过顺序训练多个弱学习器，每个学习器关注前一个学习器的错误。

定义 6.2 (Boosting)**.** *Boosting 算法*:

1. *初始化样本权重*

2. 对于 $t = 1, \ldots, T$：

- 使用当前权重训练弱学习器 h_t
- 计算 h_t 的误差
- 更新样本权重（增加错误样本的权重）
- 计算 h_t 的权重 α_t

3. 最终预测：$H(\mathbf{x}) = \sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})$

通俗解释：Boosting 就像学生做错题后，老师重点讲解错题，学生反复练习直到掌握。每个弱学习器专注于前一个学习器的薄弱环节。

6.2.1 AdaBoost

AdaBoost（Adaptive Boosting）是经典的 Boosting 算法：

Algorithm 4 AdaBoost

Require: 训练集 $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$，弱学习器 $\mathcal{A}$，迭代次数 T

Ensure: 集成模型 H

初始化样本权重：$w_i^{(1)} = 1/n, \forall i$

for $t = 1$ to T **do**

使用权重 $\mathbf{w}^{(t)}$ 训练弱学习器：$h_t = \mathcal{A}(\mathcal{D}, \mathbf{w}^{(t)})$

计算加权误差：$\epsilon_t = \sum_{i: h_t(\mathbf{x}_i) \neq y_i} w_i^{(t)}$

计算学习器权重：$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$

更新样本权重：$w_i^{(t+1)} = \frac{w_i^{(t)}}{Z_t} \exp(-\alpha_t y_i h_t(\mathbf{x}_i))$

end for

return $H(\mathbf{x}) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(\mathbf{x})\right)$

6.2.2 梯度提升

梯度提升（Gradient Boosting）将 Boosting 视为优化问题，使用梯度下降来最小化损失函数。

核心思想：每个新学习器拟合前一个模型的负梯度（残差）。

6.2.3 XGBoost 和 LightGBM

- **XGBoost**：优化的梯度提升实现，支持并行计算、正则化、处理缺失值

- **LightGBM**：更快的梯度提升实现，使用基于直方图的算法和 Leaf-wise 树生长策略

应用场景：在 Kaggle 等数据科学竞赛中，XGBoost 和 LightGBM 经常是获胜方案的核心组件。

6.2.4 优势与局限性

优势：

- 通常比单个模型性能更好
- 可以减少偏差和方差
- 可以处理各种类型的数据

局限性：

- 训练时间较长（顺序训练）
- 对异常值敏感
- 可解释性较差

6.3 Stacking

Stacking（堆叠）使用元学习器来组合多个基学习器的预测。

定义 6.3 (Stacking)**.** *Stacking 算法*：

1. 使用交叉验证训练多个基学习器
2. 使用基学习器的预测作为特征，训练元学习器
3. 最终预测使用元学习器

通俗解释：Stacking 就像有一个” 超级裁判”，它不直接看原始数据，而是看各个” 专家”（基学习器）的意见，然后综合这些意见做出最终判断。

6.3.1 算法流程

Algorithm 5 Stacking

Require: 训练集 $\mathcal{D}$，基学习器 $\{\mathcal{A}_1, \ldots, \mathcal{A}_M\}$，元学习器 $\mathcal{A}_{\text{meta}}$

Ensure: 集成模型

使用 K 折交叉验证训练基学习器

for $m = 1$ to M **do**

for 每折 k **do**

在折 k 的训练集上训练 $h_{m,k}$

在折 k 的验证集上生成预测 $p_{m,k}$

end for

基学习器 m 的完整预测：$\mathbf{p}_m = [p_{m,1}, \ldots, p_{m,K}]$

end for

使用 $\{(\mathbf{p}_1, \ldots, \mathbf{p}_M), \mathbf{y}\}$ 训练元学习器

return 集成模型（基学习器 + 元学习器）

6.3.2 优势与局限性

优势：

- 可以捕捉基学习器之间的互补性
- 通常性能优于单个模型
- 灵活性高，可以使用不同类型的基学习器

局限性：

- 计算成本高
- 需要仔细设计，避免过拟合
- 可解释性差

7 在线学习与增量学习

在线学习和增量学习使模型能够从新数据中持续学习，适应数据分布的变化。

7.1 在线学习

定义 7.1 (在线学习). *在线学习（Online Learning）是一种学习范式，模型逐个处理样本，每处理一个样本就更新模型参数，不需要存储所有历史数据。*

给定数据流 $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots\}$，在线学习算法在时刻 t：

1. 接收样本 $(\mathbf{x}_t, y_t)$
2. 使用当前模型 f_t 进行预测
3. 根据预测误差更新模型：$f_{t+1} = Update(f_t, (\mathbf{x}_t, y_t))$

通俗解释：在线学习就像实时学习，每来一个新例子就立即学习，不需要等所有数据都收集完。就像学生每做一道题就立即知道答案并学习，而不是等所有题目做完再统一学习。

7.1.1 在线梯度下降

在线梯度下降是随机梯度下降的在线版本：

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla_{\mathbf{w}} \ell(f(\mathbf{x}_t; \mathbf{w}_t), y_t)$$

其中 η_t 是学习率，通常随时间衰减。

7.1.2 应用场景

例 7.1 (推荐系统). 在线推荐系统需要实时响应用户行为，根据用户的实时反馈更新推荐模型。例如，用户点击了某个商品，系统立即更新该用户的兴趣模型。

例 7.2 (广告投放). 在线广告系统需要根据实时点击率调整广告投放策略，快速适应市场变化。

7.1.3 优势与局限性

优势：

- 内存效率高（不需要存储所有数据）
- 可以快速适应数据分布变化
- 适合大规模数据流
- 可以实时更新模型

局限性：

- 对异常值敏感

- 可能遗忘历史信息
- 需要仔细设计学习率
- 难以评估模型性能

7.2 增量学习

定义 7.2 (增量学习). 增量学习（*Incremental Learning*）是模型在已有知识的基础上，从新数据中学习新知识，同时保留或整合旧知识的过程。

与在线学习的区别：增量学习通常处理批量新数据，并且需要处理”灾难性遗忘”问题。

通俗解释：增量学习就像人类学习新知识。学习新内容时，不会完全忘记旧知识，而是将新旧知识整合在一起。但机器学习模型容易”遗忘”旧知识，需要特殊技术来解决。

7.2.1 灾难性遗忘

灾难性遗忘（Catastrophic Forgetting）是指模型在学习新任务时，会大幅降低在旧任务上的性能。

原因：神经网络参数在训练新任务时被更新，可能破坏对旧任务的记忆。

7.2.2 解决方法

- **弹性权重巩固（EWC）**：在更新参数时，对重要参数施加惩罚，防止大幅改变
- **渐进式神经网络**：为每个任务添加新的网络分支，保留旧网络不变
- **回放机制**：存储部分旧数据，与新数据一起训练
- **知识蒸馏**：使用旧模型指导新模型学习

7.2.3 应用场景

例 7.3 (持续学习系统). 智能助手需要不断学习新技能，但不能忘记已有技能。例如，学习新语言时不能忘记已掌握的语言。

例 7.4 (个性化推荐). 推荐系统需要适应用户兴趣的变化，同时保留对用户长期偏好的理解。

8 总结

本文档系统性地介绍了机器学习的核心理论与应用，包括：

- **机器学习基础**：监督学习、无监督学习和强化学习三大范式，每种范式适用于不同类型的问题
- **经典算法**：线性回归、决策树、随机森林和支持向量机等基础算法，虽然简单但在许多场景中仍然有效
- **特征工程**：特征选择、变换和编码等技术，是提升模型性能的关键
- **模型评估**：交叉验证、过拟合/欠拟合识别、偏差-方差权衡等评估技术
- **集成学习**：Bagging、Boosting 和 Stacking 等方法，通过组合多个模型提高性能
- **在线与增量学习**：使模型能够从新数据中持续学习，适应动态环境

机器学习的核心价值：

- 能够从数据中自动学习模式，减少人工规则设计
- 可以处理高维、复杂的现实世界数据
- 通过持续学习适应环境变化
- 在多个领域取得了突破性进展

未来发展方向：

- 自动化机器学习（AutoML）：减少人工调参和特征工程
- 可解释性：提高模型的可解释性和可信度
- 持续学习：更好地处理数据分布变化和任务演化
- 小样本学习：在数据稀缺场景下仍能有效学习
- 联邦学习：在保护隐私的前提下进行分布式学习

机器学习作为人工智能的核心技术，将继续在各个领域发挥重要作用，推动技术进步和社会发展。