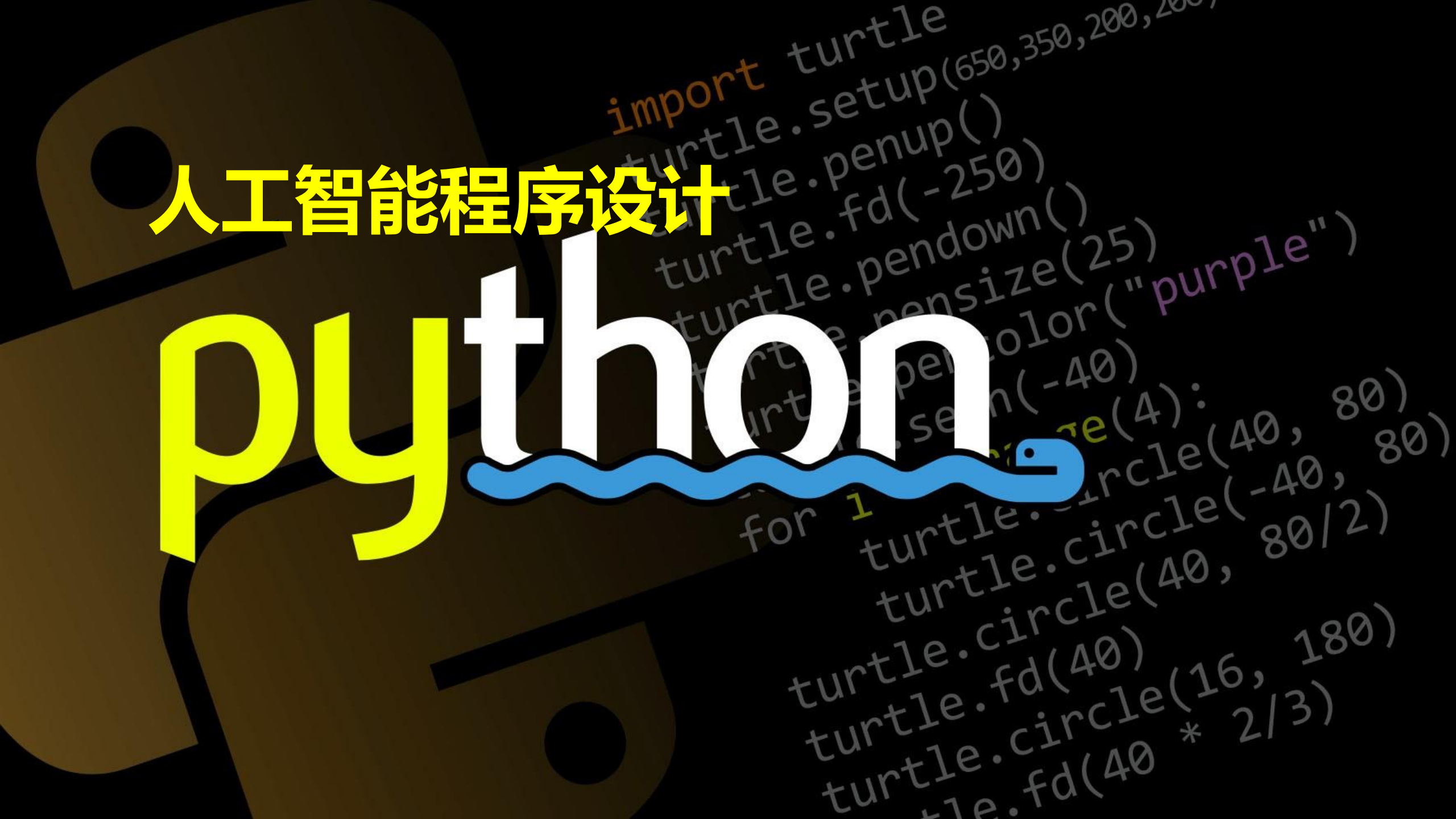
人工智能程序设计


python
import turtle
turtle.setup(650,350,200,200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for 1 in range(4):
turtle.circle(40, 80)
turtle.circle(-40, 80)
turtle.fd(40)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)

人工智能程序设计

11.3 机器学习模型

北京石油化工学院 人工智能研究院

刘 强

11.3.1 决策树模型

决策树的工作方式就像我们日常生活中的决策过程。决定是否出门打球：

1. 首先看天气：如果下雨，就不去；如果晴天，继续考虑
2. 再看温度：如果太热（>35°C），就不去；如果适宜，继续考虑
3. 最后看时间：如果是工作日，就不去；如果是周末，就去

决策树的工作原理

决策树通过一系列"是/否"问题来对数据进行分类或预测：

- **内部节点**：代表一个特征上的测试
- **分支**：代表测试的结果
- **叶节点**：代表一个类别标签或预测值

决策树具有很强的可解释性，能够清楚地看到模型的决策过程。

决策树的优势

决策树具有以下显著优势：

- **可解释性强**：能够清楚地看到模型的决策过程
- **无需复杂预处理**：可以直接处理数值型和类别型特征
- **处理非线性关系**：能够捕捉特征间的复杂交互

决策树的应用场景

分类任务：

- 医疗诊断：分析患者症状和检查结果诊断疾病
- 信用评估：根据个人信息评估贷款风险
- 客户分类：根据购买行为对客户进行分群

回归任务：

- 房价预测：分析房屋特征预测价格
- 销售预测：根据历史数据预测未来销量

决策树的局限性

决策树也存在一些局限性：

- **容易过拟合**：树过于复杂时，在训练数据上表现好，但新数据上表现差
- **不稳定性**：数据的小变化可能导致完全不同的树结构
- **偏向多值特征**：倾向于选择取值较多的特征进行分割

11.3.2 K近邻算法

K近邻（K-Nearest Neighbors, **KNN**）算法的思想：

要预测一个新样本的类别，就看它周围最近的K个邻居都是什么类别，然后根据多数投票来决定。

生活类比：在一个新社区买房时，想知道这个地区的安全性，最简单的方法就是问问周围几个最近的邻居。

KNN算法的工作原理

KNN算法的核心步骤：

1. **计算距离**：计算新样本与训练集中所有样本的距离
2. **找到邻居**：选择距离最近的K个样本
3. **进行预测**：

- 分类任务：根据K个邻居的类别进行多数投票
- 回归任务：计算K个邻居标签值的平均值

KNN算法的特点

优势:

- **简单易懂**：算法逻辑直观，容易理解和实现
- **无需训练**：懒惰学习算法，不需要显式的训练过程
- **适应性强**：能够处理多分类问题和非线性数据分布

劣势:

- **计算复杂度高**：每次预测都需要计算与所有训练样本的距离
- **存储需求大**：需要保存所有训练数据
- **对噪声敏感**：异常值可能显著影响预测结果

KNN算法的应用场景

分类任务:

- 推荐系统：分析用户行为相似性推荐商品
- 图像识别：根据像素特征相似性识别图像
- 文本分类：根据词频特征进行文档分类

回归任务:

- 房价预测：根据相似房屋的价格预测新房价格
- 股价预测：根据历史相似情况预测走势

距离度量方法

KNN算法的关键是如何计算样本间的距离：

- **欧几里得距离**：两点间的直线距离，适用于连续数值特征
- **曼哈顿距离**：各维度差值的绝对值之和，更加鲁棒
- **余弦距离**：关注向量的方向而非大小，适用于高维稀疏数据

K值的选择

K值的选择对算法性能有重要影响：

- **K值过小**：模型复杂度高，容易过拟合，对噪声敏感
- **K值过大**：模型过于简单，可能欠拟合，丢失重要模式

实践建议：

- 通过交叉验证选择最优K值
- 倾向于选择奇数以避免二分类问题中的平票

11.3.3 模型对比与评估

方面	决策树	K近邻
可解释性	高，规则清晰	低，黑盒模型
训练效率	需要时间构建	无需训练
预测效率	快	慢（大数据时）
内存需求	小	大
数据质量要求	鲁棒	需要特征缩放

模型选择指导

选择决策树的情况：

- 需要模型具有良好的可解释性
- 数据中包含大量类别型特征
- 内存资源有限
- 数据存在缺失值等质量问题

选择KNN的情况：

- 数据分布不规则或呈现复杂非线性关系
- 不需要解释模型的决策过程
- 有充足的计算和存储资源

实践练习

练习 11.3.1：模型理解

1. 解释决策树和KNN算法的核心思想差异
2. 分析两种算法各自适用的场景
3. 讨论如何选择合适的K值

实践练习

练习 11.3.2：算法应用

1. 设计一个使用决策树的实际应用场景
2. 设计一个使用KNN的实际应用场景
3. 比较两种方法在你设计场景中的优缺点

本节小结

- **决策树**：通过一系列"是/否"问题进行决策，可解释性强
- **K近邻**：根据最近邻居的类别投票决定，简单直观
- 两种算法各有优劣，根据实际场景选择
- 模型选择需要考虑：可解释性、效率、数据质量