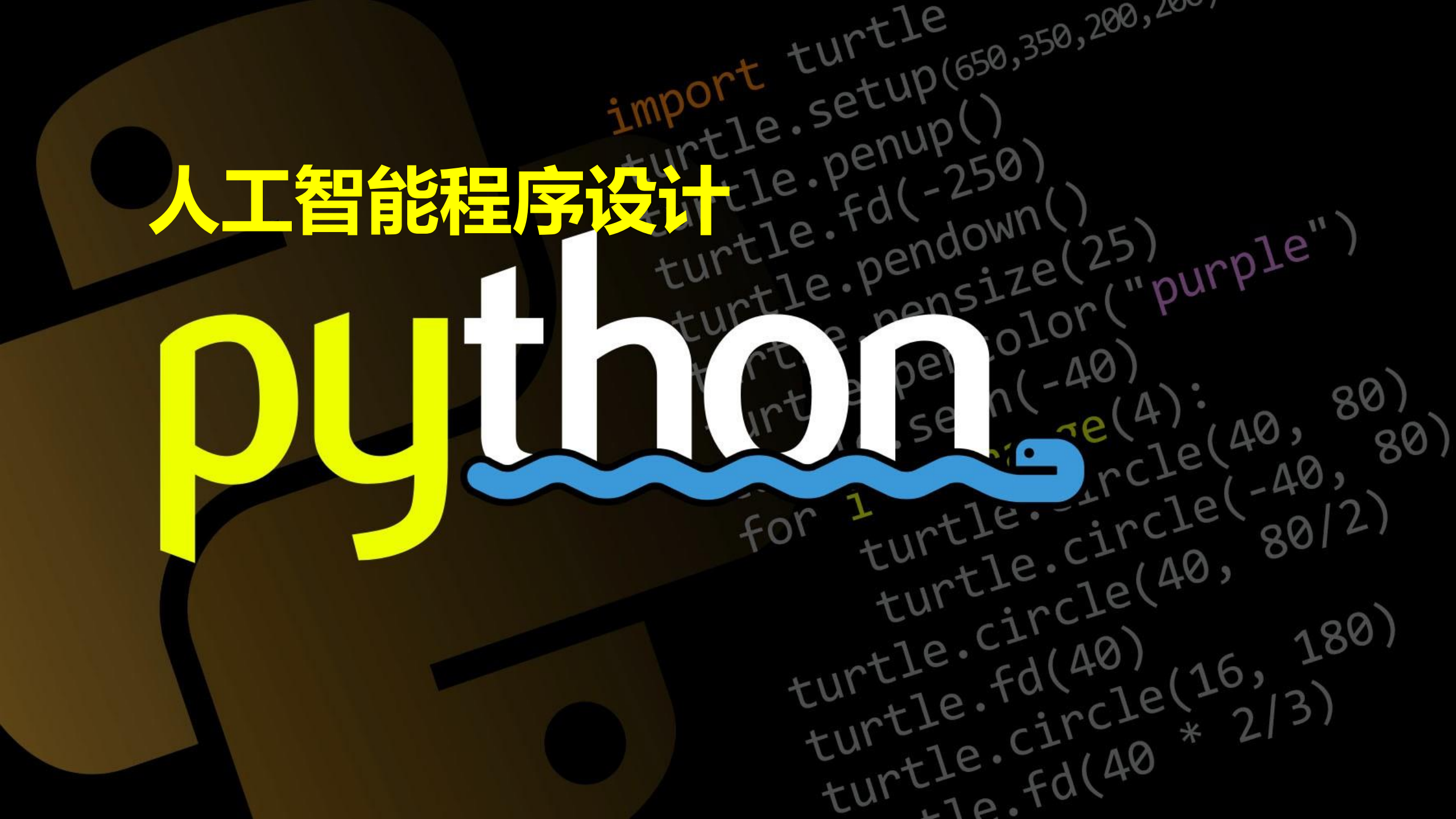
人工智能程序设计


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

11.2 机器学习工具库SCIKIT-LEARN

北京石油化工学院 人工智能研究院

刘 强

11.2.1 Scikit-learn简介

Scikit-learn（简称**sklearn**）是Python生态中最流行的传统机器学习库：

- 集成了几乎所有经典机器学习算法
- 提供统一的API接口
- 是学习和应用机器学习的首选工具

```
pip install scikit-learn
```

核心特点

Scikit-learn 的核心优势：

- **算法全面**：包含分类、回归、聚类、降维等各类算法
- **接口统一**：所有模型都使用相同的 `fit()`、`predict()`、`score()` 方法
- **工具完善**：提供数据预处理、模型选择、性能评估等完整工具链
- **易于使用**：API设计直观，文档丰富，适合初学者

11.2.2 机器学习工作流程

1. **数据收集**：获取相关的特征和标签数据
2. **数据预处理**：清洗数据，处理缺失值和异常值
3. **数据划分**：分为训练集和测试集
4. **模型训练**：选择算法，用训练集学习规律
5. **模型评估**：用测试集验证性能
6. **预测应用**：对新数据进行预测

示例 11.2.1：学生挂科预测系统

收集学生的相关信息作为特征，以及他们的考试结果作为标签：

```
import pandas as pd

## 创建学生数据
data = pd.DataFrame({
    '平时成绩': [85, 60, 92, 45, 78],
    '出勤次数': [28, 15, 30, 8, 25],
    '作业完成率': [0.95, 0.70, 1.0, 0.30, 0.85],
    '是否挂科': [0, 1, 0, 1, 0]  # 0=不挂科, 1=挂科
})
```

步骤2：数据预处理

处理缺失值、异常值，分离特征和标签：

```
## 分离特征和标签
X = data[['平时成绩', '出勤次数', '作业完成率']]  # 特征
y = data['是否挂科']  # 标签
```

步骤3：划分数据集

将数据分为训练集和测试集，训练集用于学习规律，测试集用于验证效果：

`test_size=0.2` 表示 20% 的数据用于测试，80% 用于训练。

```
from sklearn.model_selection import train_test_split

## 划分训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

步骤4：选择和训练模型

选择合适的机器学习算法，用训练数据让机器学习规律：

`fit()` 方法是 Scikit-learn 的统一训练接口。

```
from sklearn.tree import DecisionTreeClassifier

## 创建并训练模型
model = DecisionTreeClassifier()
model.fit(X_train, y_train)
```

步骤5：评估模型性能

用测试集验证模型的预测能力：

```
from sklearn.metrics import accuracy_score

## 预测并评估
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("模型准确率: {:.3f}".format(accuracy))

## 或者直接使用模型的score方法
accuracy = model.score(X_test, y_test)
print("准确率: {:.3f}".format(accuracy))
```

步骤6：预测新数据

使用训练好的模型对新学生进行预测：

```
## 新学生数据：[平时成绩，出勤次数，作业完成率]
new_student = [[88, 28, 0.95]]

## 进行预测
prediction = model.predict(new_student)
result = '不挂科' if prediction[0] == 0 else '挂科'
print("预测结果：{}".format(result))
```

11.2.3 机器学习数学基础

在机器学习中，"距离"帮助我们量化数据点之间的相似性，距离越小表示越相似。

生活中的例子：推荐系统中，如果两个用户喜欢的电影类型越相似，我们就说他们的"距离"越近。

欧几里得距离

最常用的距离计算方法是欧几里得距离，就是两点之间的直线距离：

计算示例：

- 学生A的特征：[85, 28]
- 学生B的特征：[82, 26]
- 距离 = $\sqrt{[(85-82)^2 + (28-26)^2]} = \sqrt{[9 + 4]} = \sqrt{13} \approx 3.6$

应用场景：K近邻算法、聚类算法、推荐系统

误差：衡量预测准确性

误差用来衡量模型预测结果与真实值之间的差距。误差越小，说明模型预测越准确。

常用误差指标：

- **平均绝对误差（MAE）**：预测值与真实值差的绝对值的平均数
- **均方误差（MSE）**：预测值与真实值差的平方的平均数
- **均方根误差（RMSE）**：MSE的平方根，单位与原数据相同

误差计算示例

假设预测房价 [95, 125, 80] 万，实际房价 [100, 120, 85] 万：

MAE = (|95-100| + |125-120| + |80-85|) ÷ 3 = (5 + 5 + 5) ÷ 3 = 5万元

误差的重要性：

- **模型训练**：机器学习的目标就是最小化预测误差
- **模型选择**：通过比较不同模型的误差来选择最佳模型
- **性能评估**：误差是评估模型性能的重要指标

11.2.4 内置数据集使用

Scikit-learn 提供了多个经典数据集：

- **无需下载**：直接调用即可使用
- **格式标准**：数据已经预处理，格式统一
- **经典案例**：都是机器学习领域的经典问题

常用内置数据集

分类任务数据集：

- **鸢尾花数据集（Iris）**：150个样本，4个特征，3种类别
- **手写数字数据集（Digits）**：1797个样本，64个特征，10个类别
- **乳腺癌数据集**：569个样本，30个特征，2个类别

回归任务数据集：

- **加州房价数据集**：20640个样本，8个特征
- **糖尿病数据集**：442个样本，10个特征

实践练习

练习 11.2.1：数据集探索

1. 加载糖尿病数据集（`load_diabetes`），分析其基本信息
2. 比较鸢尾花、手写数字、乳腺癌数据集的特征数量和样本数量
3. 观察不同数据集的数据分布特点

实践练习

练习 11.2.2：距离和误差计算

1. 实现曼哈顿距离计算函数
2. 对比欧几里得距离和曼哈顿距离在相同数据上的结果
3. 使用不同误差指标评估同一个预测结果

本节小结

- Scikit-learn 是 Python 最流行的机器学习库
- 统一接口：`fit()`、`predict()`、`score()`
- 机器学习工作流程：数据收集→预处理→划分→训练→评估→预测
- 距离：量化数据相似性
- 误差：衡量预测准确性