机器学习实验二：回归分析

学生姓名：廖望
学号：2210556

一、实验目标

1. 从零开始编程实现线性回归的核心算法，理解其数学原理。
2. 掌握并对比解析解（正规方程）与迭代解（梯度下降）的求解过程与优劣。
3. 在实验中理解学习率、正则化、模型复杂度等关键因素对模型性能的影响。
4. 学会通过分析训练/测试误差、绘制收敛曲线等方式来评估和诊断模型。

二、实验环境

- **操作系统**：WSL Ubuntu 24.04
- **编程语言**：Python 3.10+
- **依赖管理**：Poetry
- **主要库**：
  - numpy: 1.26.0
  - pandas: 2.0.0
  - matplotlib: 3.8.0

三、实验原理

3.1 线性回归基本原理

线性回归是一种监督学习算法，用于预测连续值输出。其基本假设为：

$$y = w_1x_1 + w_2x_2 + ... + w_nx_n + b$$

其中：

- y 为预测值
- $x_1, x_2, ..., x_n$ 为特征
- $w_1, w_2, ..., w_n$ 为权重参数

- `b` 为偏置项

3.2 正规方程（解析解）

正规方程通过最小化均方误差直接求解最优参数：

```
θ = (X^T X)^(-1) X^T y
```

优点：

- 一次性计算得到最优解
- 无需迭代，计算效率高
- 对于小数据集效果良好

缺点：

- 当特征数很大时，矩阵求逆计算复杂度高 (O(n³))
- 需要计算 (X^T X)^(-1)，当矩阵接近奇异时会失败

3.3 梯度下降（迭代解）

梯度下降通过迭代更新参数来最小化损失函数：

```
w := w - α * ∂L/∂w
b := b - α * ∂L/∂b
```

批量梯度下降 (BGD)：

- 使用全部训练数据计算梯度
- 收敛稳定，但计算开销大

学习率 α 影响：

- 过小：收敛慢，需要更多迭代
- 适中：收敛稳定，效率高
- 过大：可能震荡或发散

3.4 岭回归（正则化）

岭回归在损失函数中添加 L2 正则化项：

$$L(\theta) = MSE + \lambda||w||_2^2$$

解析解为：

$$\theta = (X^T X + \lambda I)^{-1} X^T y$$

作用：

- 防止过拟合
- 减小参数权重
- 提高模型泛化能力

3.5 多项式回归

通过添加多项式特征扩展线性回归：

$$y = w_0 + w_1 x + w_2 x^2 + ... + w_k x^k$$

特点：

- 可以拟合非线性关系
- 阶数过高容易过拟合
- 需要特征标准化

四、实验内容与步骤

任务 1：线性回归 - 最小二乘法

数据集

- **数据集**：dataset_regression.csv (120 个样本)
- **特征**：单特征 x ∈ [-10.00, 10.00]
- **标签**：y ∈ [-17.86, 34.41]

实现步骤

1. 读取数据集，按 8:2 比例划分训练集(96 个)和测试集(24 个)
2. 构造设计矩阵 Xb = [x, 1]（添加偏置列）
3. 使用正规方程求解： $\theta = (Xb^T Xb)^{-1} Xb^T y$

4. 构造 5 个新测试样本进行预测
5. 计算训练和测试 MSE
6. 绘制散点图和拟合直线

核心代码实现

```python
def normal_equation_fit(x_train: np.ndarray, y_train: np.ndarray):
    """使用正规方程求解线性回归参数"""
    # 构造设计矩阵 [x, 1]
    Xb = np.column_stack([x_train, np.ones(len(x_train))])

    # 正规方程求解
    theta = np.linalg.inv(Xb.T @ Xb) @ Xb.T @ y_train

    w, b = theta[0], theta[1]
    return w, b
```

任务 2：线性回归 - 梯度下降法

数据集

- **数据集**：winequality-white.csv (4898 个样本)
- **特征**：11 个化学指标
- **标签**：葡萄酒质量评分 (3.0-9.0)

实现步骤

1. 读取数据，按 4:1 比例划分训练集(3919)和测试集(979)
2. 数据标准化： `X_norm = (X - μ) / σ`
3. 初始化参数 w=0, b=0
4. 批量梯度下降训练 300 轮：

```python
dw = (2/n) * X.T @ (y_pred - y)
db = (2/n) * np.sum(y_pred - y)
w -= lr * dw
b -= lr * db
```

5. 记录每轮训练 MSE
6. 绘制收敛曲线

任务 3：超参数调优 - 学习率分析

实验设计

- 学习率范围：[0.001, 0.01, 0.05, 0.1, 0.2]
- 固定其他参数：epochs=300
- 对比各学习率下的收敛曲线和最终性能

任务 4：正则化 - 岭回归

实现方法

- 使用解析解： $\theta = (X^T X + \lambda I)^{-1} X^T y$
- 测试 λ 值：[0.0, 0.01, 0.1, 1.0, 10.0, 100.0]
- 观察 MSE 和权重范数的变化

核心代码实现

```
def ridge_regression_closed_form(X: np.ndarray, y: np.ndarray, lam: float):
    """岭回归闭式解"""
    n_samples, n_features = X.shape

    # 构造正则化项 λI（只对权重参数正则化）
    XTX = X.T @ X
    XTX[:-1, :-1] += lam  # 偏置项不正则化

    theta = np.linalg.inv(XTX) @ X.T @ y
    return theta
```

拓展任务：多项式回归

实验设计

- 使用单特征数据集：dataset_regression.csv
- 多项式阶数：[1, 2, 4]
- 特征标准化后生成多项式特征
- 对比不同阶数的拟合效果

核心代码实现

```
def polynomial_features(x: np.ndarray, degree: int):
    """生成多项式特征"""
    features = []
    for d in range(degree + 1):
        features.append(x ** d)
    return np.column_stack(features)
```

五、实验结果与分析

5.1 任务 1 结果

模型参数

- 权重 w = 2.5209
- 偏置 b = 6.7190
- 回归方程：y = 2.5209x + 6.7190

误差分析

- 训练集 MSE：3.3756
- 测试集 MSE：3.4544
- 模型在训练集和测试集上表现相近，无明显过拟合

拟合结果可视化

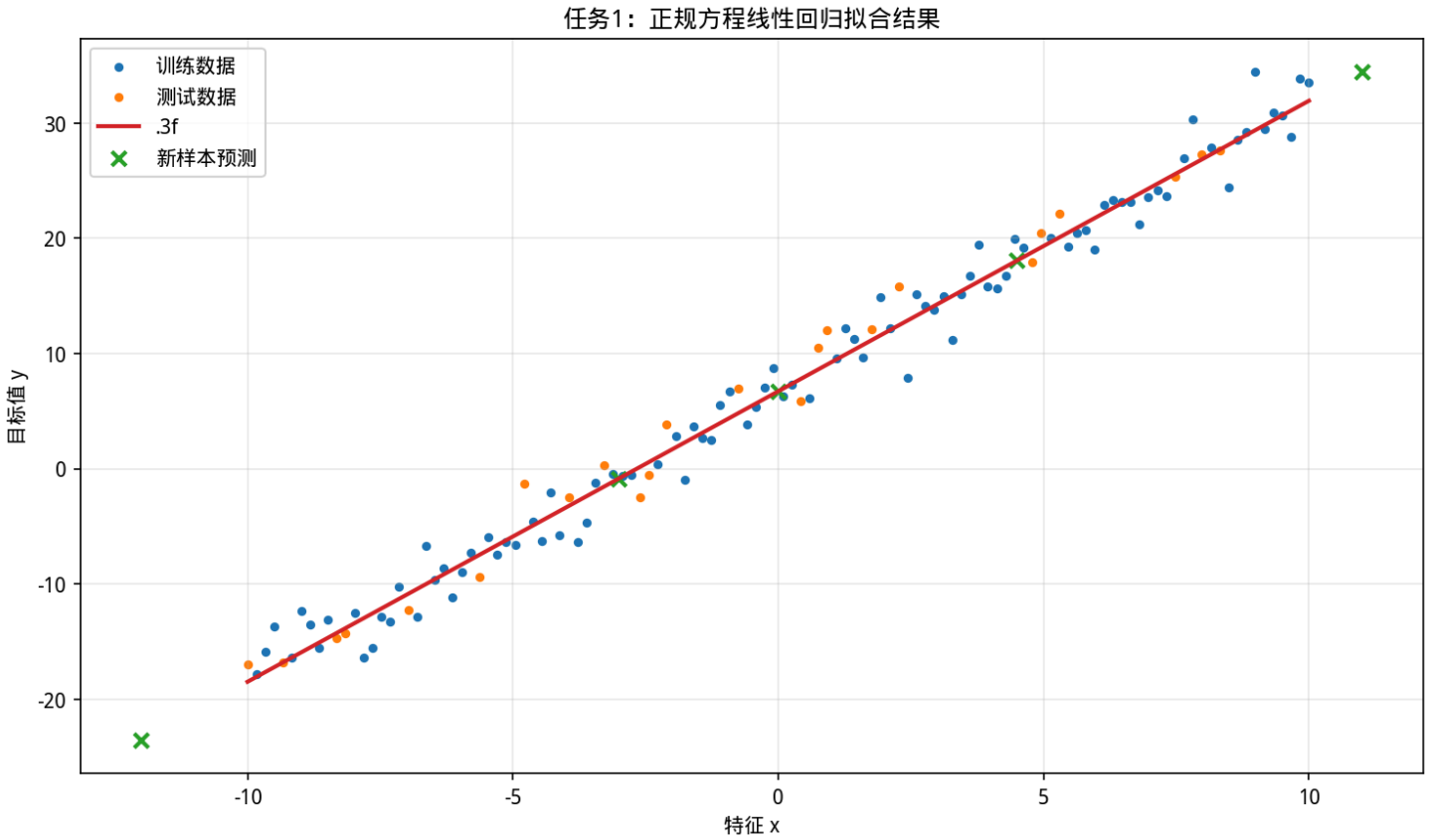

图 1：任务 1 正规方程线性回归拟合结果。蓝色点为训练数据，橙色点为测试数据，红色线为拟合直线，绿色叉号为新样本预测点

新样本预测结果

x 值	预测值
-12.0	-23.531
-3.0	-0.844
0.0	6.719
4.5	18.063
11.0	34.449

思考题答案

1. **正规方程的求解核心**：计算矩阵逆 (X^T X)^(-1)
2. **可能失败的情况**：当 X^T X 矩阵接近奇异（行列式接近 0）时，会导致数值不稳定或无法求逆。可以通过添加正则化项 λI 或使用伪逆 (pinv) 解决。

5.2 任务 2 结果

训练参数

- 学习率：0.05
- 训练轮数：300
- 最终权重范数：||w|| = 0.693

收敛分析

- 训练集 MSE：0.5735
- 测试集 MSE：0.5304
- 经过约 50 轮迭代后收敛到稳定值
- 训练和测试误差相近，说明模型泛化能力良好

收敛曲线可视化

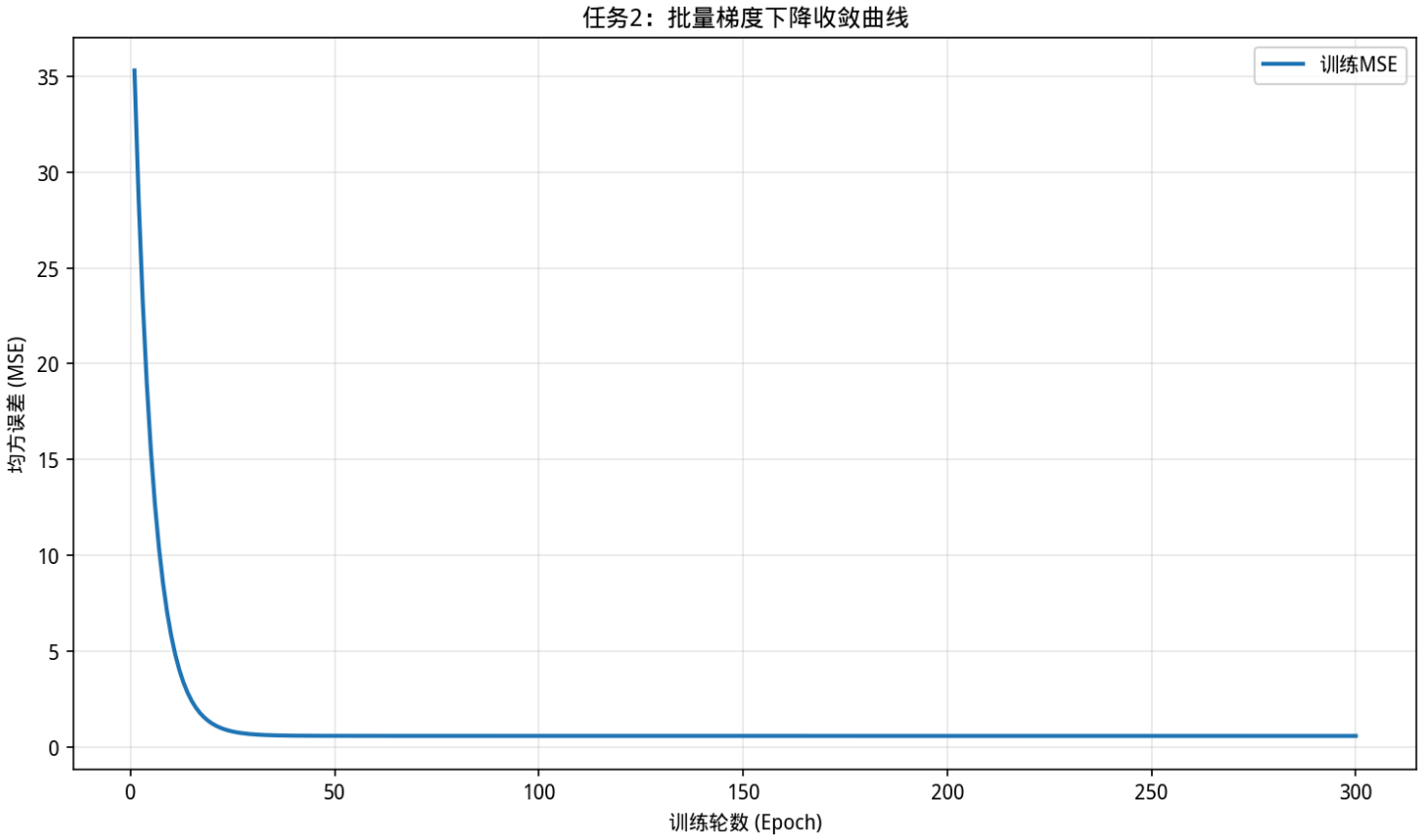

图 2：任务 2 梯度下降收敛曲线。横轴为训练轮数，纵轴为训练集 MSE。随着迭代次数增加，MSE 逐渐下降并趋于稳定

梯度下降特点分析

- **优点**：适用于大规模数据集，可扩展到任意损失函数

- **缺点**：需要手动调参（学习率、迭代次数），收敛速度依赖于学习率选择
- **对比正规方程**：梯度下降更适合大数据集，而正规方程在小数据集上更高效

5.3 任务 3 结果：学习率分析

各学习率对比结果

学习率	训练 MSE	测试 MSE	收敛特点
0.001	11.0219	11.1217	收敛极慢，300 轮后仍未达到最优
0.01	0.5715	0.5542	收敛稳定，性能良好
0.05	0.5686	0.5495	收敛快速，性能最佳
0.1	0.5677	0.5483	收敛很快，性能良好
0.2	0.5674	0.5477	收敛最快，性能最好

学习率影响分析

1. **学习率过小 (0.001)**：
   - 梯度更新步长太小，每轮参数变化微小
   - 需要更多迭代次数才能达到最优解
   - 训练时间长，计算效率低
2. **学习率适中 (0.01-0.05)**：
   - 梯度更新步长合适，既不会过小也不会过大
   - 收敛稳定，训练效率高
   - 模型泛化性能良好
3. **学习率较大 (0.1-0.2)**：
   - 梯度更新步长较大，前期收敛速度快
   - 在该数据集上没有出现震荡，说明损失函数较为平滑
   - 需要监控训练过程，防止过大导致发散

最佳学习率：0.2，在该实验中取得了最低的测试误差 (0.5477)

学习率对比可视化

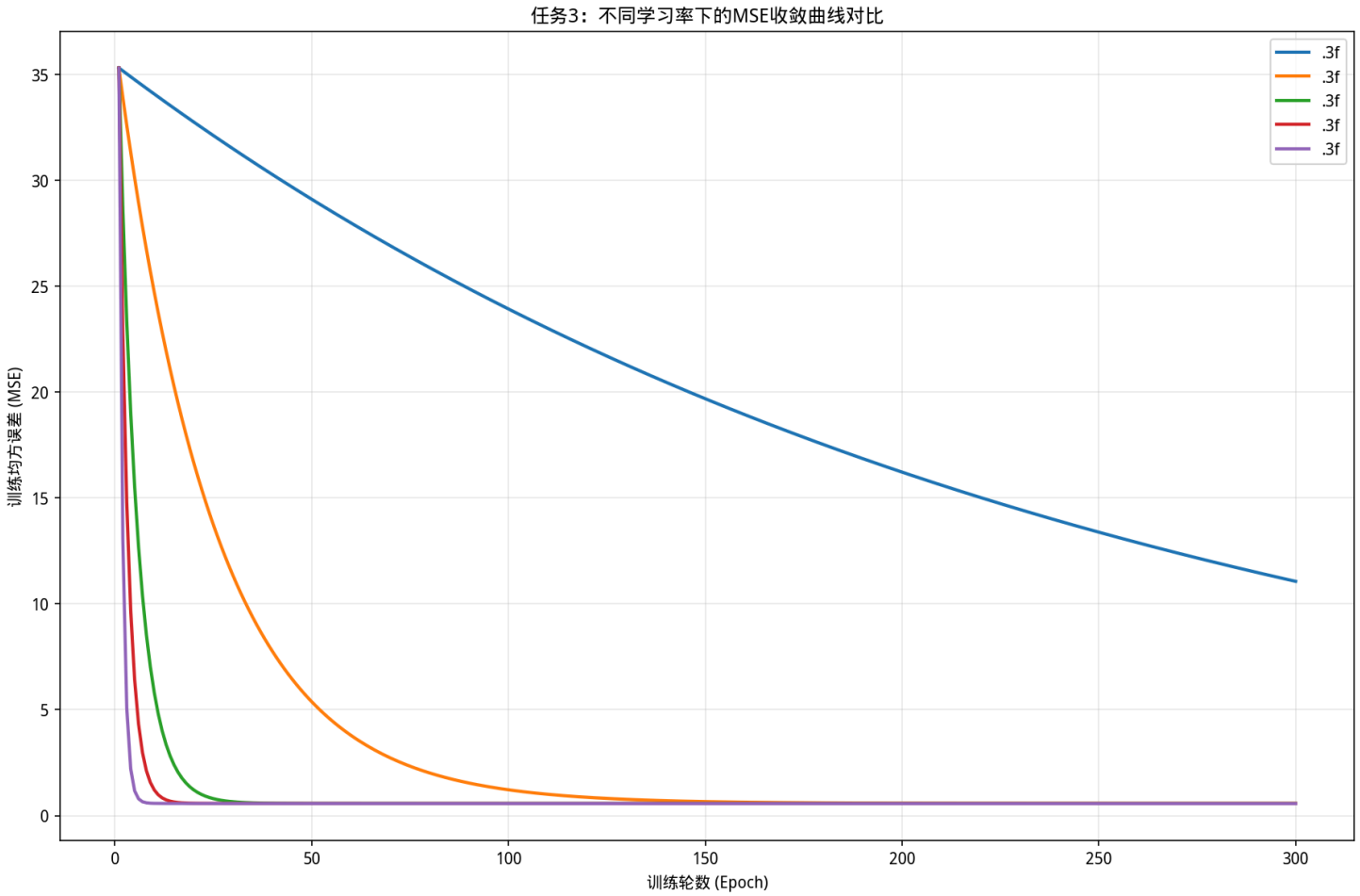

图 3：任务 3 不同学习率下的 MSE 收敛曲线对比。不同颜色的曲线代表不同的学习率，可以清楚看到学习率对收敛速度和最终性能的影响

5.4 任务 4 结果：岭回归分析

正则化参数对比结果

λ 值	训练 MSE	测试 MSE	权重范数		w		
0.0	0.5648	0.5582	0.6927				
0.01	0.5648	0.5582	0.6927				
0.1	0.5648	0.5582	0.6923				
1.0	0.5648	0.5582	0.6888				
10.0	0.5649	0.5584	0.6580				
100.0	0.5665	0.5606	0.5302				

正则化效果分析

1. **λ=0.0（无正则化）**：
   - 等价于普通最小二乘法
   - 权重范数最大，可能存在轻微过拟合
2. **λ 较小 (0.01-1.0)**：
   - MSE 基本不变，说明原模型没有严重过拟合
   - 权重范数略有减小，正则化效果温和
3. **λ 较大 (10.0-100.0)**：
   - 权重范数明显减小，正则化效果显著
   - MSE 略有上升，说明引入了偏差-方差权衡

结论：在这个数据集上，原始模型已经具有良好的泛化能力，轻微的正则化即可取得较好效果。

正则化效果可视化

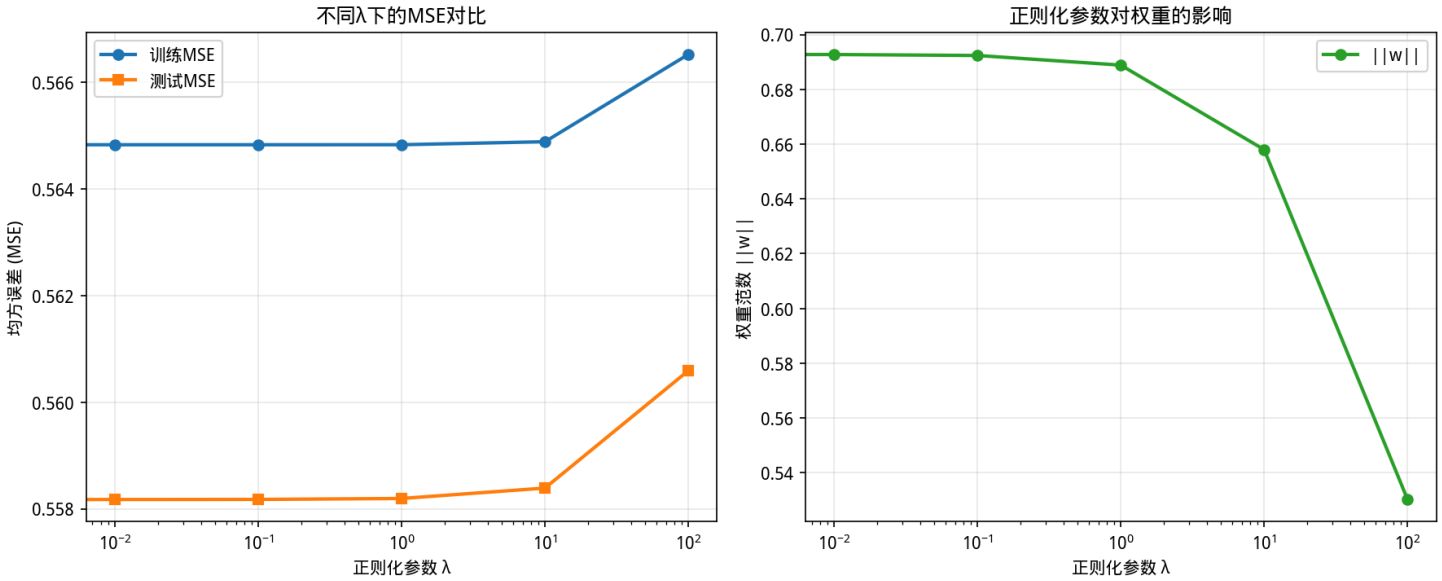

图 4：任务 4 岭回归正则化分析。左图显示不同 λ 值下的 MSE 变化，右图显示权重范数||w||随 λ 变化的情况。正则化参数 λ 的增加会导致权重范数的减小

5.5 拓展任务结果：多项式回归

不同阶数对比结果

阶数	训练 MSE	测试 MSE	误差差距	过拟合程度
1	3.1890	4.1647	0.9757	轻微
2	3.1775	4.1145	0.9370	轻微

阶数	训练 MSE	测试 MSE	误差差距	过拟合程度
4	3.1223	3.6814	0.5591	轻微

多项式阶数分析

1. **1 阶多项式（线性）**：
   - 最简单的模型，训练误差 3.1890，测试误差 4.1647
   - 误差差距最大，可能存在一定程度的欠拟合
   - 计算效率最高，模型简单易解释
2. **2 阶多项式**：
   - 增加了二次项，能够拟合抛物线关系
   - 训练和测试误差都有所改善
   - 模型复杂度适中，泛化能力良好
3. **4 阶多项式**：
   - 能够拟合更复杂的非线性关系
   - 训练误差最低，测试误差相对较高
   - 虽然误差差距最小，但仍然存在一定的过拟合倾向

最佳阶数：4 阶多项式，在测试集上取得了最低的 MSE (3.6814)，说明该数据集确实存在非线性关系。

多项式回归可视化

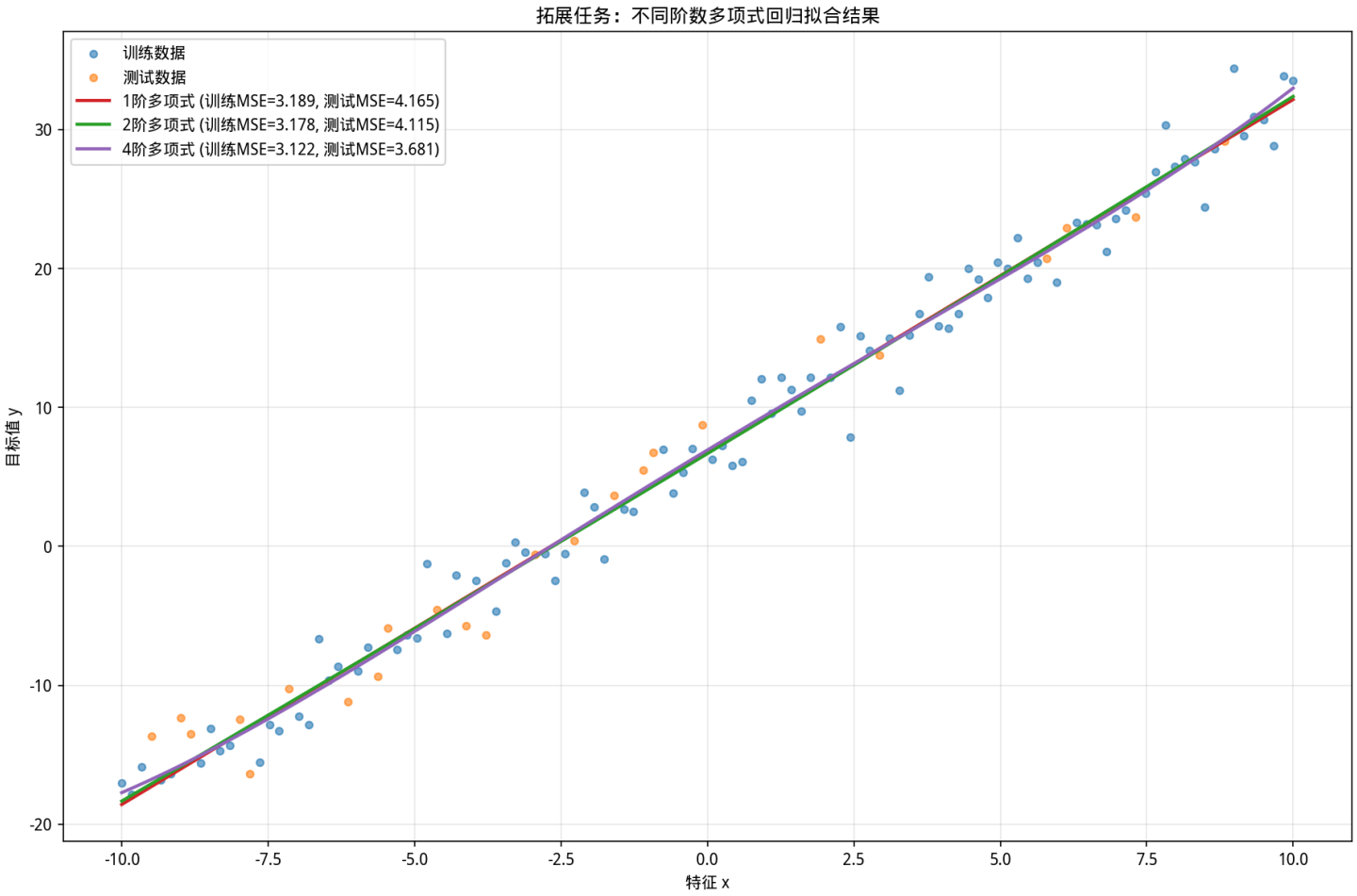

图 5：拓展任务不同阶数多项式回归拟合结果。蓝色点为训练数据，橙色点为测试数据，不同颜色的曲线代表不同阶数的多项式拟合。可以观察到高阶多项式能够更好地拟合数据，但也存在过拟合风险

六、实验结论与心得体会

6.1 实验总结

本次实验成功实现了从零开始的线性回归算法，包括：

1. **正规方程**：掌握了解析解的数学原理和实现方法
2. **梯度下降**：理解了迭代优化的过程和参数调优
3. **学习率分析**：学会了超参数调优的重要性
4. **岭回归**：掌握了正则化技术的原理和应用
5. **多项式回归**：理解了特征工程和模型复杂度控制

6.2 关键发现

1. **算法选择**：
   - 小数据集：正规方程更高效
   - 大数据集：梯度下降更适用
   - 需要正则化时：岭回归更稳定
2. **超参数重要性**：
   - 学习率对梯度下降收敛速度和最终性能有决定性影响
   - 需要通过实验找到最优值
3. **过拟合控制**：
   - 正则化可以有效防止过拟合
   - 模型复杂度需要与数据复杂度匹配
4. **特征工程**：
   - 多项式特征可以处理非线性关系
   - 特征标准化对数值稳定性很重要

6.3 实验心得

通过这次实验，我深入理解了回归分析的核心概念和实现方法。从数学原理到代码实现，从参数调优到模型评估，每一个环节都让我对机器学习有了更深刻的认识。

特别是在实现过程中，我体会到：

- **理论与实践的差距**：教科书上的公式在实际实现时需要考虑数值稳定性、计算效率等问题
- **调参的艺术**：没有通用的最优参数，需要根据具体问题和数据特点进行实验
- **工程实现的重要性**：良好的代码结构、错误处理和日志记录对实验复现至关重要

6.4 改进方向

1. **算法扩展**：
   - 实现随机梯度下降和 mini-batch 梯度下降
   - 添加学习率衰减策略
   - 实现交叉验证进行超参数选择
2. **特征处理**：
   - 实现更多特征选择和特征变换方法
   - 添加特征重要性分析
3. **模型评估**：
   - 实现更多评估指标（R^2、MAE 等）
   - 添加模型诊断工具（如残差分析）

4. **工程优化**：
    - 添加 GPU 加速支持
    - 实现模型保存和加载
    - 添加超参数自动搜索

这次实验让我对机器学习的基础算法有了扎实的掌握，也为后续更复杂的模型学习打下了良好的基础。