Report of Linear Models

Yansong He

2830791284@qq.com

Abstract

本报告探讨了深度学习在自然语言处理中的应用，重点分析了三种线性模型：普通最小二乘法、梯度下降法和牛顿-拉夫森法。通过对不同模型的参数估计和均方误差（MSE）的比较，我们展示了每种方法在处理数据中的有效性和局限性。实验结果表明，虽然线性模型提供了基本的拟合能力，但在复杂任务中的性能不尽理想。

Introduction

ChatGPT的流行引发了广泛讨论，但在很多情况下，讨论并未转化为实际编程实现。为了解决实际问题，本报告基于三种经典线性模型来探索它们在数据拟合中的表现。这些模型的基本理论和实际应用都将进行深入分析，提供更清晰的视角来理解线性回归在机器学习中的价值。

M1: Least Squares

我们可以使用经典的普通最小二乘（OLS）方法来估计参数。OLS的目标是通过最小化观测值和模型预测值之间的垂直距离（残差）的平方和来找到最适合一组数据点的线（或超平面）。还有对LMS的几何解释并使用以下直接解析解：

$$\theta = (\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T\mathbf{y}$$

给定一个训练数据

$$D = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$$

如果我们学习的模型是线性模型：

$$h_\theta(\mathbf{x}^{(i)}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

我们需要学习的函数是映射：

$$f_\theta(\mathbf{x}^{(i)}) \to y^{(i)}$$

我们需要对参数进行调优：

$$\theta = [\theta_0, \theta_1, \ldots \theta_n]^T$$

以满足以下方程：

$$\begin{cases} \theta_0 + \theta_1 x_1^{(1)} + \theta_2 x_2^{(1)} + \cdots + \theta_n x_n^{(1)} = y^{(1)} \\ \theta_0 + \theta_1 x_1^{(2)} + \theta_2 x_2^{(2)} + \cdots + \theta_n x_n^{(2)} = y^{(2)} \\ \vdots \\ \theta_0 + \theta_1 x_1^{(N)} + \theta_2 x_2^{(N)} + \cdots + \theta_n x_n^{(N)} = y^{(N)} \end{cases}$$

这可以写入矩阵表示：

$$\begin{bmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \cdots & x_n^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \cdots & x_n^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & x_2^{(N)} & \cdots & x_n^{(N)} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$$

然而，很有可能我们无法解决方程。我们可以找到最佳近似：

$$X\theta \approx Y$$
$$\min \|X\theta - Y\|_2^2$$

M2: Gradient Descent

损失函数及其最小值求法

$$L(\theta) = \frac{1}{2N} \sum_{i=1}^{N} [f_\theta(x^{(i)}) - y^{(i)}]^2$$

$$\begin{cases} \frac{\partial L}{\partial \theta_1} = \frac{1}{N} \sum_{i=1}^{N} [f_\theta(x^{(i)}) - y^{(i)}] * x_1^{(i)} \\ \\ \frac{\partial L}{\partial \theta_2} = \frac{1}{N} \sum_{i=1}^{N} [f_\theta(x^{(i)}) - y^{(i)}] * x_2^{(i)} \\ \\ \ldots\ldots \end{cases}$$

学习率为α，则梯度下降更新公式为：

$$\theta_i = \theta_i - \alpha \cdot \frac{\partial L}{\partial \theta_i}$$

若将以上内容用矩阵表示，则满足：

$$\nabla L(\theta) = \frac{1}{N} \mathbf{X}^T (\mathbf{X}\theta - \mathbf{Y})$$
$$\theta = \theta - \alpha \cdot \frac{1}{N} \mathbf{X}^T (\mathbf{X}\theta - \mathbf{Y})$$

M3: Newton-Raphson Method

求根：

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

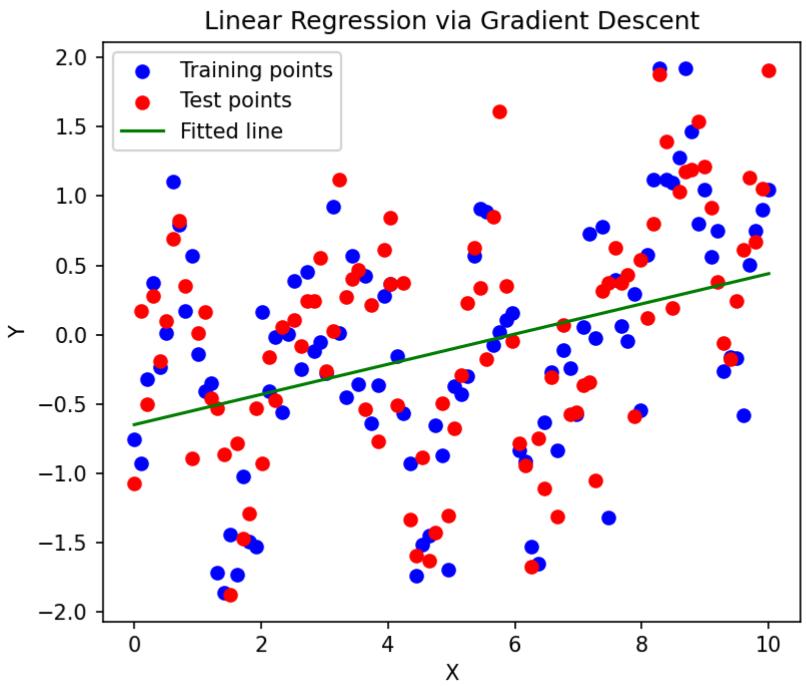
优化：

$$x_{n+1} = x_n - \frac{f'(x_n)}{f''(x_n)}$$

Experimental Studies

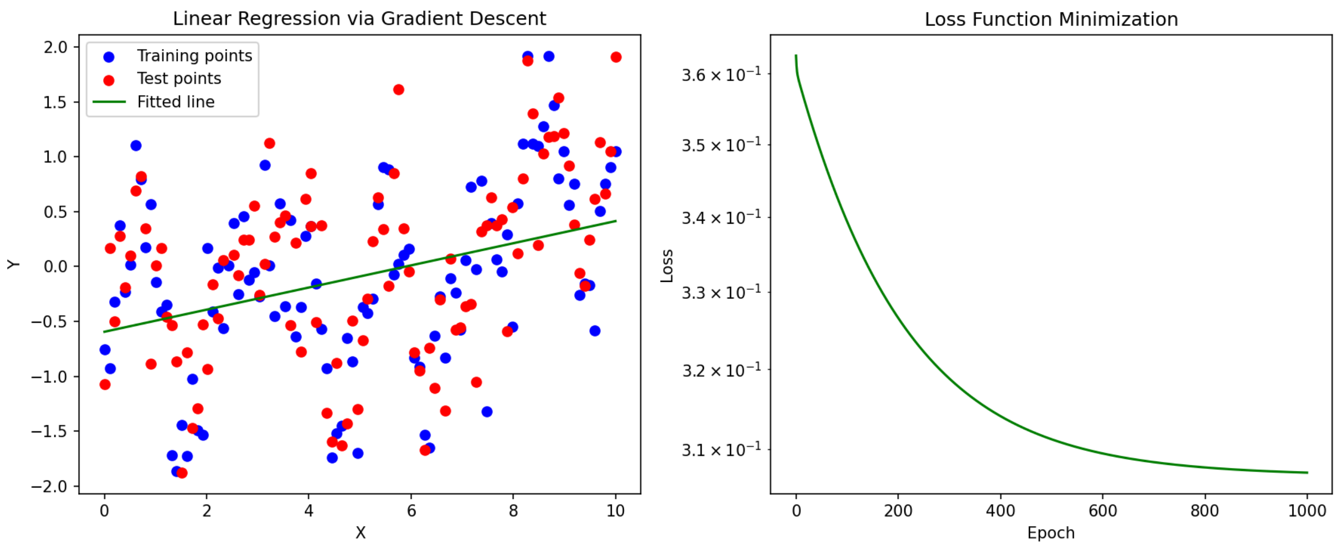
Models	θ_0	θ_1	MSE
Least Squares	-0.5945539505142685	0.10079895562357076	0.5934116636978033
Gradient Descent	-0.6487466967301827	0.10894738685803657	0.5950433861467296
Newton-Raphson Method	-0.6487466967301855	0.10894738685803709	0.5950433861467296

(1) Least Squares



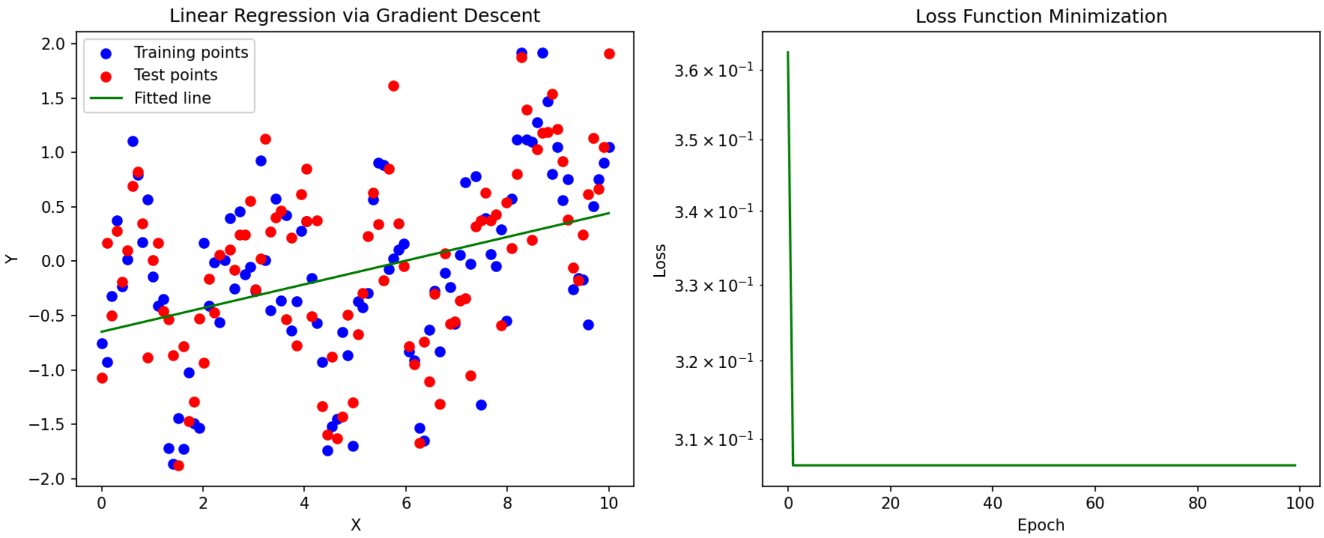
- 结果显示该方法的拟合能力较弱，尽管是线性模型的经典方法，其性能仅能应付较为简单的数据关系。

(2) Gradient Descent



- 梯度下降法的收敛速度较慢，且对学习率的选择敏感，导致最终的MSE略高于普通最小二乘法，表明在特定情况下可能造成不必要的误差累积。

(3) Newton-Raphson Method



- 尽管该方法通常比梯度下降更快收敛，但在本实验中，结果与梯度下降法相似，显示了该特定数据集的挑战性。

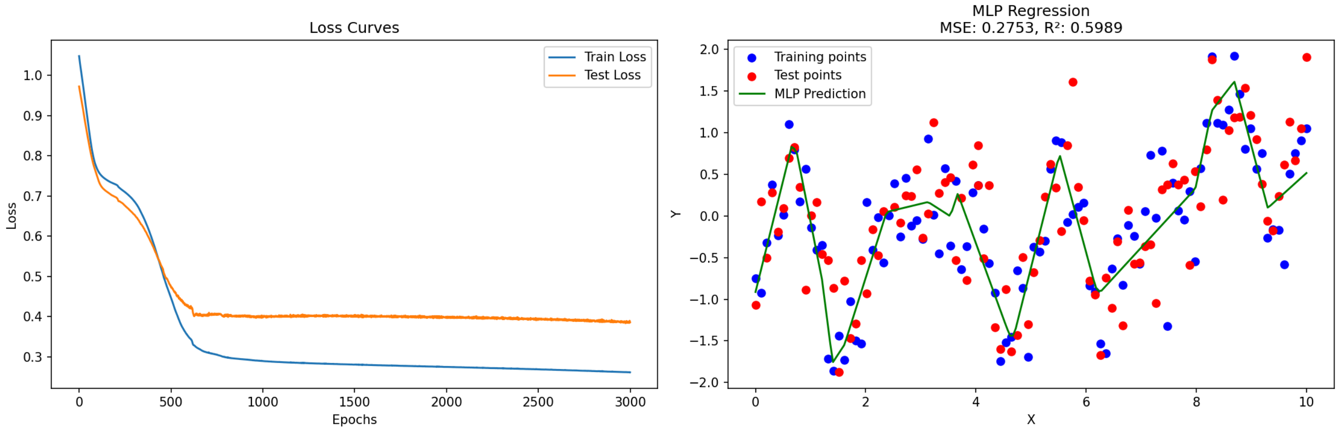
线性拟合的效果并不理想，故可以采用MLP前馈神经网络的方式进行拟合，分别设置不同的隐藏层结构和学习率配置：

```
# 要尝试的网络配置
hidden_layer_configs = [


    (20, 10),
    (20, 20, 10),
    (30, 20, 10),
]
learning_rates = [0.01, 0.001]
```

训练结果如下，最佳模型配置为：

$$hidden\_layers : (20, 20, 10), learning\_rate : 0.001$$
$$MSE : 0.27525009659436783$$
$$R^2 : 0.598891641097498$$



这一结果显示，通过设置适当的隐含层配置与学习率，MLP模型在数据拟合中显著优于传统的线性模型。具体而言，MSE的显著降低表明模型能够更好地捕捉数据的复杂关系，同时R^2值接近0.6，表明模型解释了近60%的变异性，进一步验证了其有效性。