人工智能实验 第九周

一、实验题目

利用感知机算法在给定数据集完成购房预测训练，选择合适的损失函数，画出数据可视化图，$Loss$曲线图

二、实验内容

1.算法原理

本实验使用多层感知机(MLP)实现，包含了一个隐藏层，用于学习输入特征与输出目标之间的非线性映射关系(房价与收入房龄的关系不一定是线性)。使用反向传播和梯度下降对损失函数最小化从而完成任务。其中损失函数除了使用一般的均方误差(MSE)外，加入了正则误差防止过拟合。同时本实验对输入X以及最后的输出y进行了归一化，避免特征值差异过大。

模型结构：输入层(2维) -> 隐藏层(16个神经元，ReLu激活) -> 输出层(1维，线性输出)

模型具体计算，X为输入的数据 **1. 第一层线性变换，前向传播部分：**

$$Z_1 = XW_1 + b_1$$

2. ReLu激活，增加非线性表达能力：

$$A_1 = max\left(0, Z_1\right)$$

3. 输出层，隐藏层输出乘以第二层权重得到预测值：

$$Z_2 = A_1W_2 + b_2 \quad \hat{y} = Z_2$$

4. 损失函数，利用均方误差加正则损失：

$$MSE = \frac{1}{2n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2$$

$$L_2 = \lambda\left(\|W_1\|^2 + \|W_2\|^2\right)$$

$$L = MSE + L_2$$

5. 反向传播部分:

- 输出层误差项：

$$\frac{\partial L}{\partial Z_2} = \frac{1}{n}(Z_2 - y)$$

- 输出层权重与偏置梯度：

$$\frac{\partial L}{\partial W_2} = A_1^\top \cdot \frac{\partial L}{\partial Z_2} + \lambda W_2$$

$$\frac{\partial L}{\partial b_2} = \sum_{i=1}^{n} \frac{\partial L}{\partial Z_2^{(i)}}$$

- 反传到隐藏层：

$$\frac{\partial L}{\partial A_1} = \frac{\partial L}{\partial Z_2} \cdot W_2^\top$$

$$\frac{\partial L}{\partial Z_1} = \frac{\partial L}{\partial A_1} \odot \mathrm{ReLU}'(Z_1)$$

- 第一层权重与偏置梯度：

$$\frac{\partial L}{\partial W_1} = X^\top \cdot \frac{\partial L}{\partial Z_1} + \lambda W_1$$

$$\frac{\partial L}{\partial b_1} = \sum_{i=1}^{n} \frac{\partial L}{\partial Z_1^{(i)}}$$

- 更新公式：

$$W = W - \eta \frac{\partial L}{\partial w}$$

$$b = b - \eta \frac{\partial L}{\partial b}$$

具体代码位于函数 `train_mlp` 中，接受参数 `X` 输入数据，`y` 预测值，`hidden_dim` 隐藏层维度，`lr` 学习率，`epochs` 训练轮数。该函数循环 `epochs` 次，根据前面给出的公式进行计算，最终得到权重并以字典 `model` 形式返回。

2.关键代码

下面给出计算的核心代码函数 `train_mlp`

In [1]:
```
def train_mlp(X, y, hidden_dim=16, lr=0.01, epochs=1000, reg_lambda=0.001):
    n_samples, n_features = X.shape
    W1 = np.random.randn(n_features, hidden_dim) * 0.1
    b1 = np.zeros((1, hidden_dim))
    W2 = np.random.randn(hidden_dim, 1) * 0.1
    b2 = np.zeros((1, 1))
    losses = []

    for i in range(epochs):
        Z1 = X @ W1 + b1
        A1 = relu(Z1)
        Z2 = A1 @ W2 + b2
        y_pred = Z2
```

```
        mse_loss = np.mean((y_pred - y) ** 2) / 2
        reg_loss = reg_lambda * (np.sum(W1**2) + np.sum(W2**2))
        loss = mse_loss + reg_loss
        losses.append(loss)

        dZ2 = (y_pred - y) / n_samples
        dW2 = A1.T @ dZ2 + reg_lambda * W2
        db2 = np.sum(dZ2, axis=0, keepdims=True)

        dA1 = dZ2 @ W2.T
        dZ1 = dA1 * relu_derivative(Z1)
        dW1 = X.T @ dZ1 + reg_lambda * W1
        db1 = np.sum(dZ1, axis=0, keepdims=True)

        W1 -= lr * dW1
        b1 -= lr * db1
        W2 -= lr * dW2
        b2 -= lr * db2

    return {"W1": W1, "b1": b1, "W2": W2, "b2": b2, "losses": losses}
```

在实验过程中发现有一系列的等于于500000的房价不随房龄改变而改变，可以认为是数据出现了问题，故考虑舍去。舍去代码如下：

```
mask = y_raw.flatten() < price_upper_limit
X_raw, y_raw = X_raw[mask], y_raw[mask]
```

三、实验结果分析

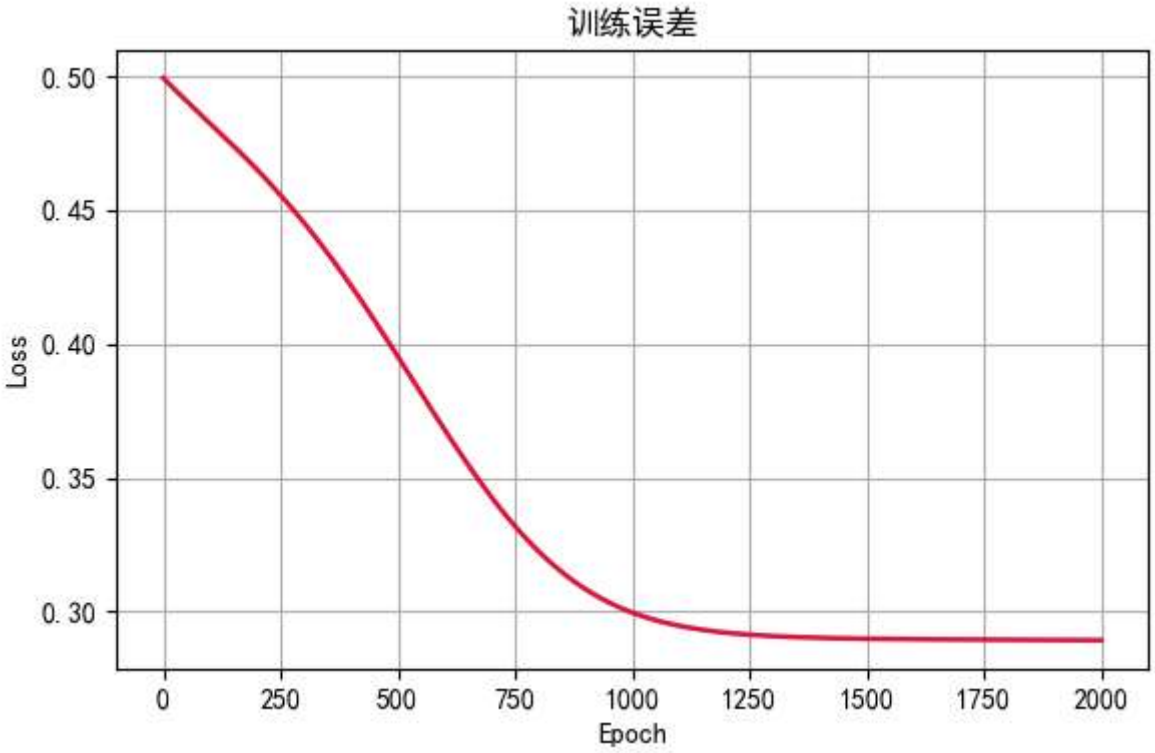
1.实验结果展示

实验每隔100次训练输出$Loss$值，同时最后可视化输出：1房价收入图，2训练误差图，3预测房价-收入图，下面给出一个测试代码，参数如下：

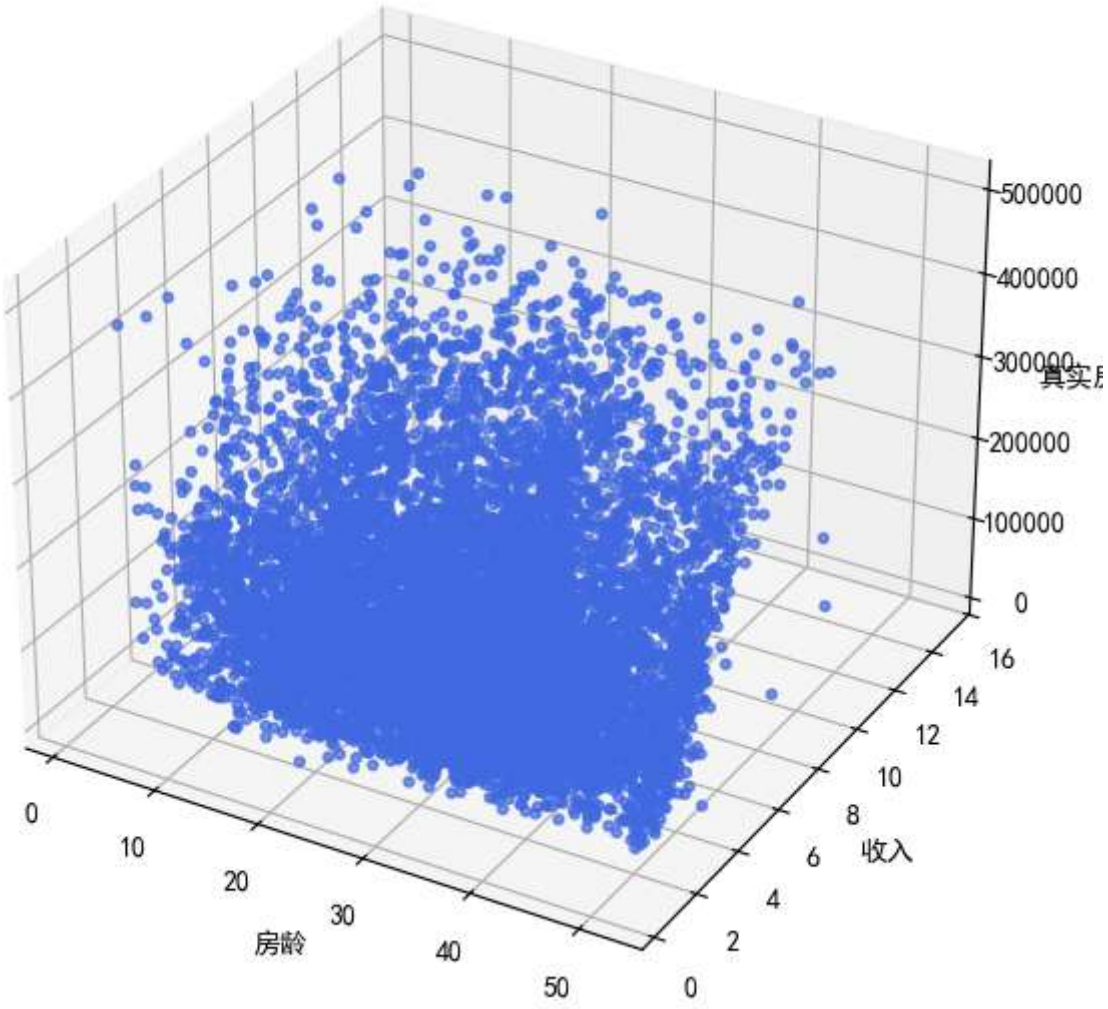
```
def run_mlp(csv_path="MLP_data.csv",
                feature_cols=(2, 3),
                target_col=4,
                hidden_dim=16,
                lr=0.005,
                epochs=1500,
                reg_lambda=0.005,
                price_upper_limit=500000,
                fig_path="mlp_result.png"):
```

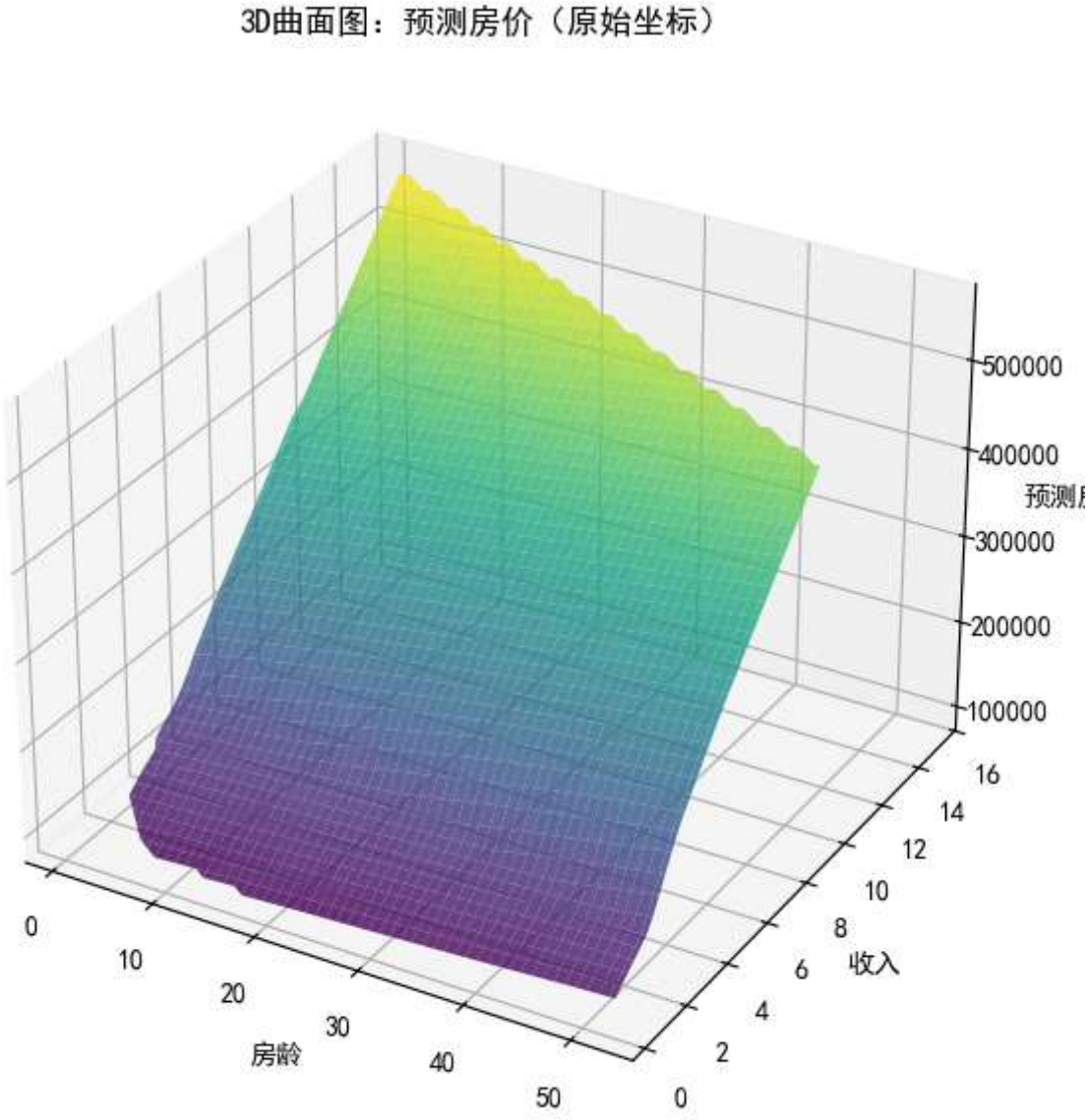
In [2]:
```
import sys
import os
import numpy as np

sys.path.append(r"D:\Desktop\人工智能\实验\Homework\homework5")

from predict_price_new import run_mlp

model = run_mlp(csv_path=r"D:\Desktop\人工智能\实验\Homework\homework5\MLP_data.
```

```
Epoch 0 | Loss 0.4993
Epoch 100 | Loss 0.4823
Epoch 200 | Loss 0.4650
Epoch 300 | Loss 0.4452
Epoch 400 | Loss 0.4218
Epoch 500 | Loss 0.3952
Epoch 600 | Loss 0.3678
Epoch 700 | Loss 0.3427
Epoch 800 | Loss 0.3225
Epoch 900 | Loss 0.3084
Epoch 1000 | Loss 0.2997
Epoch 1100 | Loss 0.2948
Epoch 1200 | Loss 0.2922
Epoch 1300 | Loss 0.2909
Epoch 1400 | Loss 0.2903
Epoch 1500 | Loss 0.2899
Epoch 1600 | Loss 0.2898
Epoch 1700 | Loss 0.2896
Epoch 1800 | Loss 0.2895
Epoch 1900 | Loss 0.2895
```



3D散点图：真实房价（原始坐标）





2.性能分析

由上图的预测房价-收入图以及损失函数$Loss$可以看到。预测值大部分接近实际值，并且$Loss$下降平滑最终达到0.28。故可以认为其的预测效果达标。由于单隐藏层的限制，预测图仍存在一定的锯齿状波动。