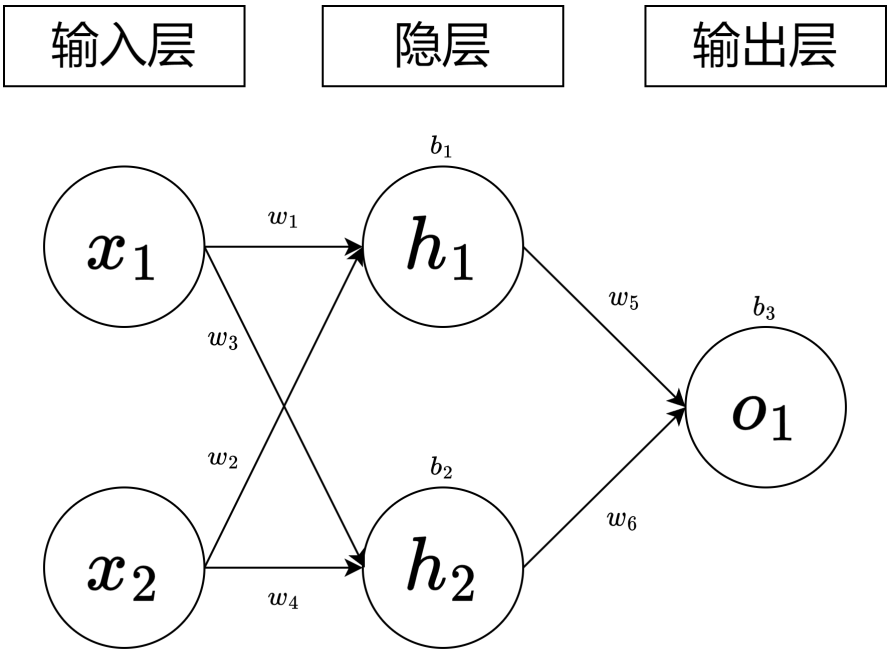
[Problem 3] 编程题说明

在本题中，我们尝试基于 `numpy` 实现一个简单的前馈神经网络，用作二分类模型。神经网络的结构如下图所示：



其中，$\{x_1, x_2\}, \{h_1, h_2\}, \{o_1\}$ 分别表示输入层、隐层和输出层神经元；w_i, b_i 为各节点的权重与偏置参数；$\{h_1, h_2, o_1\}$ 均使用 Sigmoid 作为激活函数。

> 在本次实验中，为了便于初学者对神经网络 BP 算法的理解，我们基于拆分后的标量表示（如 $\{w_1, w_3\}$）而不是更常规的向量表示（如 $\mathbf{w}$）来实现神经网络。但是，在实际工作中，我们一般直接使用 `numpy` 等库提供的向量级运算操作 —— 这会极大幅度提升模型的运行效率。

我们提供的数据集文件 `X_train.txt` 与 `y_train.txt`，分别包含 512 条样本的**特征**（二维）与**标签**（0/1 值）。

我们提供了两个代码模板文件：`p3_models.py` 与 `p3_main.py`，其中：

- `p3_models.py` 包含神经网络分类器模型的代码，**需要在压缩包中提交**；
- `p3_main.py` 提供了关于如何加载数据集并调用分类器模型的样例，**不强制提交**，你可以在示例代码的基础上进行超参数调节等流程。

[3.3 - 15pts] 实现神经网络

> 在本题的实现过程中，请不要使用除 `numpy` 以外的第三方 `Python` 库

需要实现的内容包括：

- [3pts] 实现 Sigmoid 函数 `sigmoid(x)`，及 Sigmoid 函数的导数 `deriv_sigmoid(x)`
- [2pts] 实现均方误差函数 `mse_loss(y_true, y_pred)`
- [7pts] 实现神经网络的训练过程 `NeuralNetworkClassifier.fit(self, X, y)`，主要包括*前向传播，反向传播（梯度计算）及使用梯度下降法更新参数*这三部分
- [3pts] 实现神经网络的预测过程 `NeuralNetworkClassifier.predict_item(self, x)`，即：给定样本特征 `x`，输出其属于 `1` 类的概率

请确保你的代码可以正常运行，我们将使用如下方式测试你的代码：

```python
from p3_models import NeuralNetworkClassifier
# Loading train/test datasets ...
clf = NeuralNetworkClassifier()
clf.fit(X_train, y_train)
score = clf.score(X_test, y_test)
```

只要你的实现正确，且满足基本的精度要求，即可获得本题的全部分数。

[3.4 - 5pts] 超参数调节

你可以参考 `p3_main.py`，基于已有数据自由设计合理的超参数搜索与评估方法，挑选在该数据集上最适合模型的超参数 `learning_rate` 与 `max_epoch`。为了便于评估，选用的 `max_epoch` 请尽量不要超过 100。

> 你可以在 `p3_main.py` 中使用任何第三方库 —— 我们不强制要求提交 `main.py`（当然交了也不会扣分），也不会运行你提交的 `p3_models.py` 之外的脚本文件。

在完成超参数调节后，请修改 `NeuralNetworkClassifier.__init__()` 函数，**使得调用模型时默认使用你选取的最优超参数**（即：我们会通过 `clf = NeuralNetworkClassifier()` 调用你的模型，并使用这种情况下的分类精度评定成绩）。此外，**请按照题目要求，在作业文档中简要汇报你的超参数调节流程。**

我们会在另一个未公开的测试数据集 `X_test.txt` 与 `y_test.txt` 上测试你的模型在你设定的默认参数下的表现。只要汇报的超参数调节流程合理，且在你挑选的最优超参数下，模型在测试集上的准确率高于我们设定的基线值，即可获得本题的全部分数。