你的报告应该有良好的结构。对于每一项任务，解释你应该做什么，你实际做了什么，结果是什么，最重要的是，得出一些结论。

1 Introduction

在该任务中，我们通过构建单层感知器在MINST数据集上实现多分类任务。为实现该功能，我们对数据进行了与处理，并在不同weights区间上进行了实验，以期找到最快的收敛速度和准确率。最后，通过交叉验证的方式对模型的鲁棒性进行了验证。

2 What we do

2.1 Data preprrcessing

在数据预处理阶段，我们首先通过Numpy加载数据集，并通过sklearn提供的model_selection.train_test_split方法将数据集划分为Training sets和Validation sets，每个sets中均包含用与训练的inputs和用于验证的targets。

此外，我们为每个input set添加了额外bais以用于神经网络训练，将每个target转换为dummy matrix以便在训练过程中运用矩阵乘法。

Basic datasets	Shape	preprocessing datasets	Shape
Input_train	1365, 256	X_train	1365, 257
Target_train	1365	y_train_dummy	1365, 10
Input_validation	342, 256	X_validation	342, 257
Target_validation	342	y_validation_dummy	342,10

2.2 Training process

在模型训练阶段，我们循环读取Input set的每行数据，使用sigmoid作为激活函数完成模型的向前传播过程。随后使用□w = □x(d - out)*out'(x)作为梯度下降函数更新模型权重。最终以模型正确预测数量与target总数的比值作为误差函数判断循环终点。

2.3 Weights optimization & Cross validation

在模型优化阶段，我们尝试了多种不同weights区间对模型准确性带来的影响，在weights为完全随机数，【-1，1】随机数，【-0.5，0.5】随机数和weights初始值为0中进行实验。

为generation的展示模型性能，我们采用5-flod交叉验证检验模型准确性。在该方法中我们通过sklearn提供的model_selection.KFold方法随机划分数据集，其中每个train set和test set的数据大小如下表所示

Each fold datasets	Shape
Input_train[fold]	1092, 257
Test_train[fold]	273, 257

2.4 Result

通过上述方法，我们在交叉验证中实验不同weights区间对模型准确性的影响，将模型误差小于0.03或迭代次数达到1000作为训练终点，得到结果如下表所示：

Weights section	Accuracy Train	Accuracy Validation	Accuracy Test
Completely random	0.188	0.216	0.228
(-1, 1)	0.797	0.798	0.727
(-0.5, 0.5)	0.973	0.956	0.877
(-0.3, 0.3)	0.970	0.961	0.886
(-0.1, 0.1)	0.972	0.950	0.882
0	0.970	0.956	0.879

可以很清楚的发现在weights取值区间为（-0.3，0.3）时，模型取得了最佳表现，将该weight运用在交叉验证中的得到：

K-fold	Accuracy Train	Accuracy Validation	Accuracy Test
1	0.941	0.950	0.872
2	0.956	0.947	0.882
3	0.970	0.956	0.883
4	0.981	0.947	0.881
5	0.981	0.950	0.879

测试集上平均准确率为0.879，其混淆矩阵为：

```
array([[215,   0,   3,   3,   2,   3,   2,   0,   3,   0],
       [  0, 113,   0,   0,   1,   0,   0,   0,   2,   1],
       [  1,   0,  80,   2,   0,   0,   0,   1,   3,   0],
       [  1,   0,   2,  67,   0,  11,   1,   1,   4,   0],
       [  3,   2,   5,   1,  76,   2,   1,   3,   1,   3],
       [  1,   0,   0,   1,   2,  35,   0,   0,   2,   2],
       [  1,   3,   2,   0,   2,   0,  86,   0,   0,   0],
       [  0,   2,   2,   1,   1,   2,   0,  56,   4,   2],
       [  1,   0,   7,   3,   0,   0,   0,   0,  73,   1],
       [  1,   1,   0,   1,   2,   2,   0,   3,   0,  79]])
```

3 Conclution

通过weights对比试验发现，模型在weight区间为（-1～0，0～1 ）时都有较好表现，在（-0.3，0.3）时表现最优。通过分析模型训练过程中的各项参数变化得出这一现象的猜测为，在初次计算output时，全随机权重易导致output值为接近于0的极小数，通过该□w = □*x*(d - out)*out（1-out）公式更新梯度时会导致□w为0而无法完成梯度下降。