中山大学数据科学与计算机学院

移动信息工程专业-人工智能

本科生实验报告

（2017-2018 学年秋季学期）

课程名称：**Artificial Intelligence**

教学班级	**15M1**	专业（方向）	**移动互联网**
学号	**15352048**	姓名	**陈潇**

一、 实验题目

1. 实现 PLA 原始算法和口袋算法
2. 采用 4 种指标评价并分析你的实验结果
3. 尝试优化，并对优化后的结果进行分析

二、 实验内容

1. 算法原理

1） 感知机学习算法在本次实验的内容中是以二元样本展开的。即样本中的训练数据只包含正样本和负样本两种，我们需要通过算法拟合一条直线通过这个二元样本平面，使得其中的正负样本在二元平面内一分为二，从而帮助我们对测试样例进行标签的预测，即 PLA。

2） 由于以上一分为二的条件满足条件比较复杂，通常情况下不容易达成，即不收敛，所以需要寻求得出全局最优的策略，即原始算法和口袋算法。

3） 原始算法下，即使最后得出的结果不收敛，也直接将其作为全局最优的结果。

4） 口袋算法下，通过设置迭代次数，得到一定范围内的【对于训练数据】的全局最优，这个结果并不一定比原始算法得出的结果在测试样例上得出的结果更好。

5） 由于 PLA 在更加复杂的情况下会出现过拟合的情况，所以迭代次数和全局最优的策略是一个重要的衡量标准。

2. 伪代码

```
void PLA()
{
    E[i] = trainSet[i].num = {Xi1, Xi2, Xi3.....Xin};
    L[i] = trainSet[i].label = {Yi1, Yi2, Yi3.....Yin};
    vector w0 = {1, ,1 ,1 ...., 1};
    for i0 = 0 : 10
        for i = 0 : trainSet.size
            signW = 0;
            for j = 0 : E[i].size
                signW += E[i][j] * w0[j];
            if sign(signW) != L[i]
                for j = 0 : w0.size
                    w0[j] = w0[j] + L[i] * E [i];

    T[i] = testSet[i].num = {Xi1, Xi2, Xi3,.....,Xin}
    T[i].label = T[i][j] *= w0[j]
}
```

```
void PLA_Pocket()
{
    E[i] = trainSet[i].num = {Xi1, Xi2, Xi3.....Xin};
    L[i] = trainSet[i].label = {Yi1, Yi2, Yi3.....Yin};
    vector w0 = w1 = {1, ,1 ,1 ...., 1};
    for i0 = 0 : 10
        for i = 0 : trainSet.size
            signW = 0;
            for j = 0 : E[i].size
                signW += E[i][j] * w0[j];
            if sign(signW) != L[i]
                for j = 0 : w0.size
                    w0[j] = w0[j] + L[i] * E [i];
                if accuracy(w0,T) > accuracy(w1,T)
                    w1 = w0;

    T[i] = testSet[i].num = {Xi1, Xi2, Xi3,.....,Xin}
    T[i].label = T[i][j] *= w1[j]
}
```

3. 关键代码截图（带注释）

本次实验的难度较小，关键代码部分就是 PLA 的计算实现部分。

大部分的变量我都设置为全局变量，主要数据（训练数据和测试数据）都使用结构相同的 struct 存储。在 TA 所给训练样本的条件下内存开销为 6MB

PLA 部分

```
//设置总体迭代次数
    for (i0 = 0; i0 < 10; i0++)
    {
        //遍历所有训练数据
        for (i = 0; i < trainSet.size(); i++)
        {
            signW = 0;
            for (j = 0; j < trainSet[i].num.size(); j++)
            {
                signW += trainSet[i].num[j] * w0[j];
            }
            //若预测不正确，则更新w的值
            if (sign(signW) != trainSet[i].label)
            {
                for (j = 0; j < w0.size(); j++)
                {
                    w0[j] = w0[j] + trainSet[i].label * trainSet[i].num[j];
                }
            }
        }
    }
```

1）原始算法的情况下，不需要存储 w0 的全局最优情况，只需要在设置的迭代次数的条件下进行修正向量角度的修正，这个修正是由预测标签错误而发生的。最理想的状态是在迭代次数内获得了一个 w0，使得该 w0 预测每一个测试数据都是正确的。

PLA 口袋算法部分

```
//设置总体迭代次数
    for (i0 = 0; i0 < 10; i0++)
    {
        //遍历所有训练数据
        for (i = 0; i < trainSet.size(); i++)
        {
            signW = 0;
            for (j = 0; j < trainSet[i].num.size(); j++)
            {
                signW += trainSet[i].num[j] * w0[j];
            }
            //若预测不正确，则更新w的值
            if (sign(signW) != trainSet[i].label)
            {
                for (j = 0; j < w0.size(); j++)
                {
                    w0[j] = w0[j] + trainSet[i].label * trainSet[i].num[j];
                }
                //由于已经更新w的值，需要判断w的错误率是否小于全局最优
                double c1 = checktrainSet(w0), c2 = checktrainSet(w1);
                if (c1 > c2)
                {
                    for (j = 0; j < w1.size(); j++)
                    {
                        w1[j] = w0[j];
                    }
                }
```

2）PLA 口袋算法部分相比于 PLA 原始算法部分只多出了一个存储 w0 计算过程中得出的最好的准确率的结果的部分代码。

4. 创新点&优化（如果有）

三、 实验结果及分析

1. 实验结果展示示例（可图可表可文字，尽量可视化）

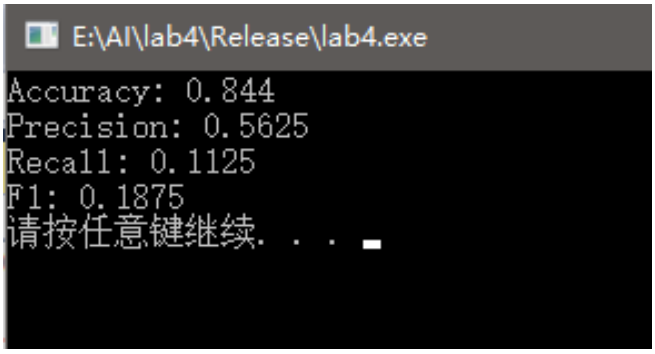

图 1 PLA 原始算法验证集（迭代 10 次）

1） 迭代 10 次的最终 w0 的准确率为 0.844。

编号	特征1	特征2	标签
Train1	-4	-1	+1
Train2	0	3	-1
Test1	-2	3	？ 原本为-1

图 2 PLA 原始算法小数据集（迭代 10 次）

2） 准确率为 1，因为在验证集中设置为-1 为正确，预测的结果也是-1，故准确率为 100%。

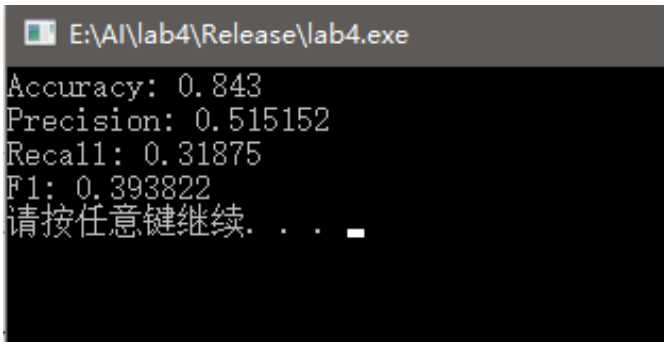

图 3 PLA 口袋算法（迭代 50 次）验证集

3） 可以看到口袋算法算出的向量在测试样例中预测得到的正确率并不一定比原始算法来得高，这也充分说明了其脆弱性以及二元数据的极强的不稳定性（衡量指标太少）。

四、 思考题

1. 有什么其他的手段可以解决数据集非线性可分的问题？

支持向量机（SVM）：

可以解决小样本情况下的数据集非线性可分。把向量映射到更高阶以解决问题。

2.请查询相关资料，解释为什么要用这四种评测指标，各自的意义是什么。

Accuracy 准确率是最直观表现模型预测的准确程度的指标。它的意义是展示模型对于整体样本中算法的预测准确程度。

Precision 精确率是衡量预测为 1 且正确的样本占所有预测为 1 的样本的个数。他缩小了样本的个数，精确到必须要为 1 的情况。

Recall 召回率是表现判定为正确的正样本占所有被判为正样本的个数。对于某些特定情况（如正样例数量极少）的情况下，可以辅助 Pecision 进行模型的评估。

F1 是综合了召回率和准确率的综合评判指标，目的就是总结召回率和准确率。

在实际模型使用中，通过这四者的表现能更好地评判模型的效果，另外，还有漏警概率、虚警概率等用于评判错误判断的后果的指标。

|---------- 如有优化，重复 1，2 步的展示，分析优化后结果 ----------|

PS：可以自己设计报告模板，但是内容必须包括上述的几个部分，不需要写实验感想