



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

(2016 学年秋季学期)

课程名称: Artificial Intelligence

教学班级	M2	专业 (方向)	移动互联网
学号	14353205	姓名	刘万里

一、 实验题目

感知机算法实现

二、 实验内容

1. 算法原理

对于二元分类问题, 向量的每一维对应一个权重, 我们尽可能地找出这样的一个权重向量, 使得可以完美地 (至少也是错误率最低的) 划分二元分类, 通过迭代的算法每次更新权重向量, 直至每一个训练样本都可以正确划分 (至少也是错误率最低), 再使用该权重向量去划分测试机得到分类。

2. 伪代码

1) Init-PLA

```
output.open("myAnswer.txt");
makeUpMatrix("train_data.txt"); //将train.data数据存储在matrix这个二维矩阵
readLabels(); //读取train,test的标签
findW(); //完善W数组了
seeW(); //看看找到的W数组和迭代次数
makeUpMatrix("test_data.txt"); //将test.data数据存储在matrix这个二维矩阵
test(); //对test进行分类
output.close();
```

2) Pocket-PLA

```
output.open("myansWer.txt");
makeUpMatrix("train_data.txt"); //将train.data数据存储在matrix这个二维矩阵
readLabels(); //读取train,test的标签
findW(); //完善W数组了
seeW(); //看看找到的W数组和迭代次数
makeUpMatrix("test_data.txt"); //将test.data数据存储在matrix这个二维矩阵
test(); //对test进行分类
output.close();
```

3) 优化 (如果有)

3. 关键代码截图 (带注释)

1) Init-PLA



封装好的便于使用的点乘函数：

```
84 int dotProduct(int *a, int *b, int length) // (二维数组, w数组)
85 {
86     int ans = 0;
87     for(int i = 0 ; i < length ; i++)
88     {
89         ans += (*a) * (*b);
90         a++;
91         b++;
92     }
93     return ans;
94 }
```

找到正确划分的 W 向量：

```
112 void findW()
113 {
114     bool ok = false; // 是否找到w数组了
115     while(!ok) // 还没完善
116     {
117         for(int i = 0 ; i < ROWS; i++)
118         {
119             if(sign( dotProduct(matrix[i], W, COLUMNS) ) != trainY[i])
120             {
121                 updateW(trainY[i], matrix[i]);
122                 iterTimes++; // 迭代一次
123                 cout<<iterTimes<<endl;
124                 break;
125             }
126             if(i == ROWS - 1) ok = true; // 找到了!
127         }
128     }
129 }
```

对测试集进行划分：

```
147 void test()
148 {
149     int TP = 0, FN = 0, FP = 0, TN = 0;
150     for(int i = 0 ; i < ROWS; i++)
151     {
152         myY[i] = sign(dotProduct(matrix[i], W, COLUMNS));
153         if(testY[i] == 1) // 正确答案为1
154         {
155             if(myY[i] == 1) TP++;
156             else FN++;
157         }
158         else // 正确答案为-1
159         {
160             if(myY[i] == -1) FP++;
161             else TN++;
162         }
163     }
164     cal4(TP, FN, FP, TN);
165 }
```



2) Pocket-PLA

都和原始的 PLA 是一样的，不过寻找 W 数组的方法不一样： $bestW$ 就是找到的最好的权重向量

```

104 void findW()
105 {
106     bool ok = false; // 是否找到W数组了
107     while((!ok) && (iterTimes < MAXITERTIMES)) // 还没完善
108     {
109         ok = true;
110         for(int i = 0 ; i < ROWS; i++)
111         {
112             if(sign( dotProduct(matrix[i],W,COLUMNS) ) != trainY[i])
113             {
114                 ok = false;
115                 updateW(trainY[i],matrix[i]);
116                 iterTimes++; // 迭代一次
117                 cout<<iterTimes<<endl;
118                 if(isBetter(trainY[i],matrix[i]))
119                 {
120                     for(int i = 0 ; i < COLUMNS;i++)
121                     {bestW[i] = W[i];} // 更新bestW 数组
122                 }
123                 break;
124             }
125         }
126     }
127 }
    
```

其中比较函数 isBetter:

```

92 bool isBetter(int label,int *a)
93 {
94     int originRightTimes = 0, newRightTimes = 0;
95     for(int i = 0 ; i < ROWS;i++)
96     {
97         if(sign(dotProduct(matrix[i],W,COLUMNS)) == trainY[i]) originRightTimes++;
98         // 计算原来的W得到的正确次数
99         if(sign(dotProduct(matrix[i],bestW,COLUMNS)) == trainY[i]) newRightTimes++;
100     }
101     printf("原正确次数= %d, 后正确次数= %d\n",originRightTimes,newRightTimes);
102     if(newRightTimes < originRightTimes) return true;
103     else return false;
104 }
    
```

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）

（原始算法结果）

\\请截图你们的运行结果，包含四种指标

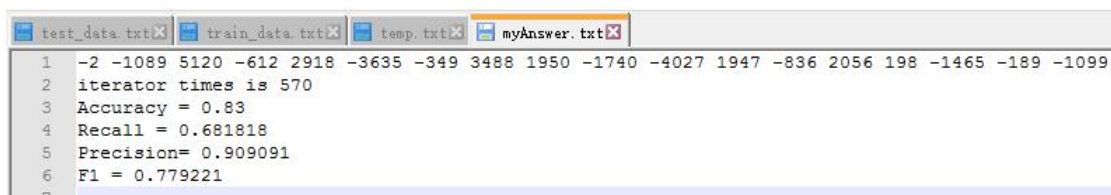
1) Init-PLA

```

1 -2 -1089 5120 -612 2918 -3635 -349 3488 1950 -1740 -4027 1947 -836 2056 198 -1465
2 iterator times is 570
3 Accuracy = 0.83
4 Recall = 0.681818
5 Precision= 0.909091
6 F1 = 0.779221

```

2) Pocket-PLA



```

1 -2 -1089 5120 -612 2918 -3635 -349 3488 1950 -1740 -4027 1947 -836 2056 198 -1465 -189 -1099
2 iterator times is 570
3 Accuracy = 0.83
4 Recall = 0.681818
5 Precision= 0.909091
6 F1 = 0.779221
7

```

2. 评测指标展示即分析

	Accuracy	Precision	Recall	F-1
Init	0.83	0.909091	0.681818	0.779221
pocket	0.83	0.909091	0.681818	0.779221

|-----如有优化，请重复 1，2，分析优化后的算法结果-----|

四、 回答问题

1. 请查询相关资料，记录这四种评测指标有什么意义，尤其是在信息检索中

Accuracy 准确度 is the difference between the measured value and the true value of a tested material.

Precision 精确度 is the repeatability of successive measurements under the same conditions.

测量的准确度高，是指系统误差较小，这时测量数据的平均值偏离真值较少，但数据分散的情况，即偶然误差的大小不明确。

测量精确度（也常简称精度）高，是指偶然误差与系统误差都比较小，这时测量数据比较集中在真值附近。

召回率(recall)的公式是,它计算的是所有"正确被检索的 item(TP)"占有"应该检索到的 item(TP+FN)"的比例。

F1 值就是精确值 precision 和召回率 recall 的调和均值

通俗地说没，准确率就是找得对，召回率就是找得全

准确率和召回率是互相影响的，理想情况下肯定是做到两者都高，但是一般情况下准确率高、召回率就低，召回率低、准确率高



2. 思考迭代数对于原始 PLA 算法有什么影响

如果 train 的迭代次数增加，有助于提高正确率，对于线性不可分的数据也能找到一个较为理想的犯错误最少的权重 w 。

3. 有什么其他的手段可以解决数据集非线性可分的问题？

PS：如果不喜欢我们提供的模板，可以自行重新设计，唯一的要求就是这个模板里面提到的内容大纲不能更改，另，本学期的实验不需要写实验感想。

命名格式为：学号_姓名拼音.pdf