单层感知机实验报告

2020.9.21 高泽文，李志康，崔庆元，李锦洋

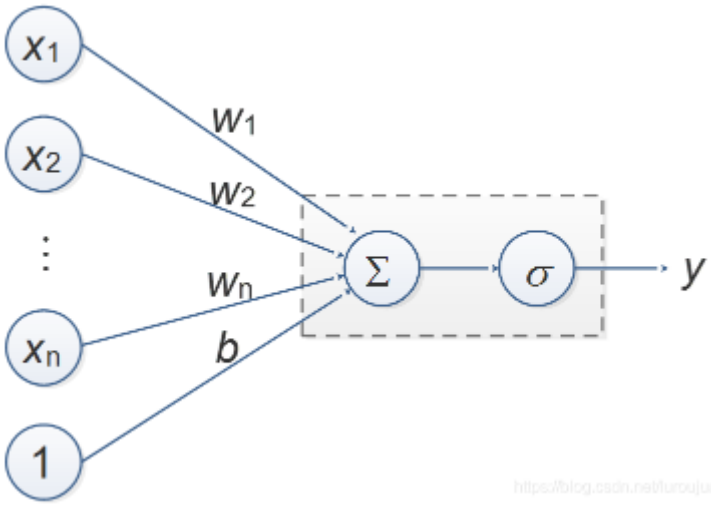
实验代码链接：github.com/xuzichi/2020cv/week1

一、实验原理

1、概述

感知机（perceptron），是人工神经网络中最基础的网络结构（perceptron一般特指单层感知器，而多层感知器一般被称为MLP）。

2、模型展示



可以用公式表示为$y=\delta(W^TX)$

其中X代表向量$[x_1,x_2,\dots,x_n,1]$，W代表向量$[w_1,w_2,\dots,w_n,b]$，δ代表激活函数。

为了达到预期的目的，需要对权值W进行调整，因此需要设计算法，通过训练集自动调整它的权值。

二、实验参数定义与设置

1、感知机的学习速率`r`

习速率越大，权值的稳定性会越差；速率太小，则会导致迭代的次数增多，消耗更多的时间和计算机资源。

经过权衡，我们小组将 r 值设为0.01 。

2、错误率阈值`s`

每一次迭代后，错误率都会产生变化，大多数情况下是在减小的，这也非常符合实际需求，因此需要设置一个阈值，当错误率小于这个值，就停止迭代。错误率自然是越低越好，但也要考虑到过高的精度所需要的迭代次数。

经过权衡，我们小组将 s 值设为 0.004 ，也就是 4e-3 。

3、初始权重值`w`

这其实是一个矩阵，里面的元素代表权值。初始的各个权值其实比较重要，如果给得科学，可以显著减少迭代调整权值的次数。我们小组选择采用一组标准正态分布。语句如下：

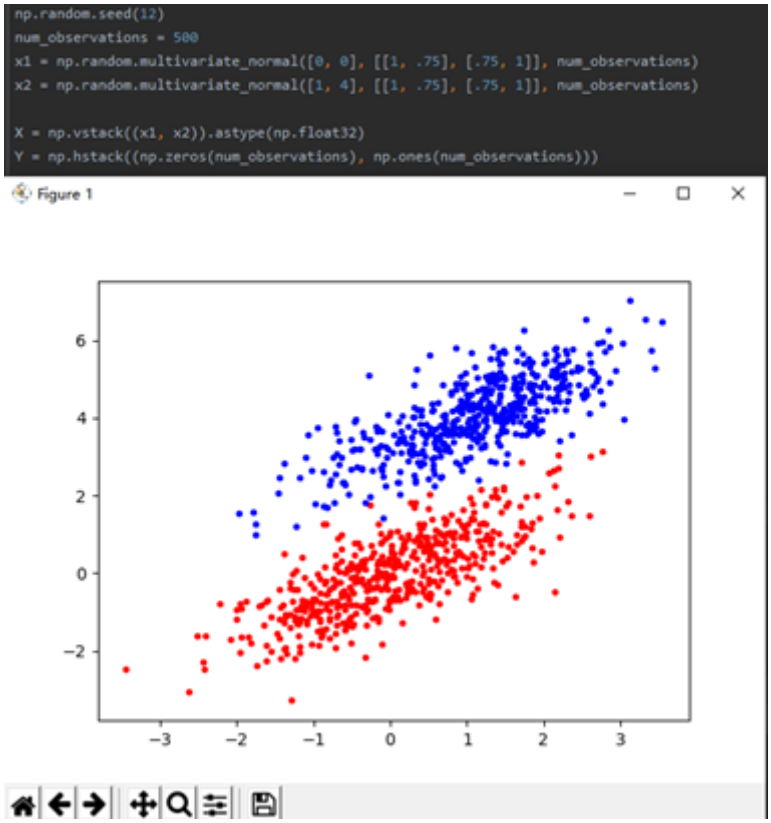
```
W = np.random.randn(1, 3)
```

`randn`函数返回一个或一组样本，具有标准正态分布。

三、实验算法

1、

导入待分割、分类的数据集，将其可视化。



2、

初始化权重和阈值，设定学习速率。初始权重w预设为一组随机标准正态分布；错误率阈值s设置为0.0004，也就是4e-3；学习速率r设置为0.01。

3、

对于训练集中每个示例，通过已知输入，计算出实际输出，将其和预期输出对比，然后通过算法对权重进行更新，使得新权重更加精确地分割数据。

Tips：第三步后得到的新权重应当立刻用于训练集中，随后进行进一步优化更新，而不是等到训练集中的所有数据都完成这些步骤。

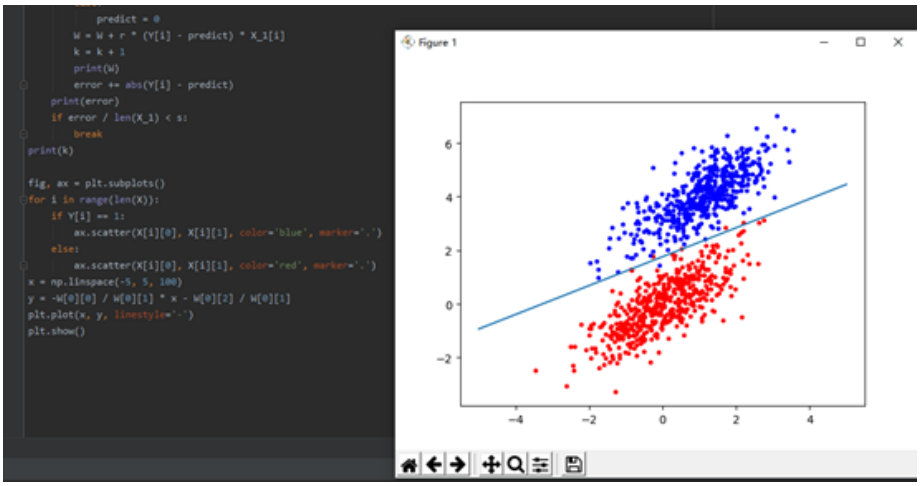
4、

重复第三步的过程，每次重复后计算总体错误率，与错误率阈值s对比，如果小于s，那么精度达标，跳出循环；如果不达标，那就继续重复。

5、

将得到的错误率达标的权值组取出，通过算式，在第一步的基础上做出分割线，将算法的结果可视化、具象化。至此，实验顺利完成。

四、实验结果



五、小组分工

组员	分工
高泽文	编写代码
李志康	编写报告
崔庆元	参与代码和报告编写
李锦洋	参与代码和报告编写

六、实验总结

通过本实验，我们深入了解了单层感知机的工作原理，对如何用python实现简单机器学习过程有了更加具体直观的感受，对于感知机的基本算法也有了更加清晰的认识。