天津大学

机器学习实验报告

题目：机器学习实验---单层感知器

学　　院　智能与计算学部

专　　业　人工智能

年　　级　2021

姓　　名

学　　号

2023 年 3 月 14 日

机器学习实验-单层感知机

一、 实验目的：

1. 理解单层感知机算法原理，能实现单层感知机算法；
2. 掌握机器学习算法的代码构建流程；
3. 针对特定应用场景及数据，能构建单层感知机模型并进行预测。

二、 实验内容

1. 安装 matplotlib, numpy, pandas；
2. 熟悉 dry_bean_dataset 数据集，并能使用单层感知机算法对该数据集构建模型并应用。

三、 实验报告要求

1. 按实验内容撰写实验过程；
2. 报告中涉及到的代码，每一个模块需要有详细的注释；
3. 输出最终的测试结果。

四、 实验过程

1. 下载 dry_bean_dataset 数据集并读取，为了直观体现该模型的效果，我们仅选取 MajorAxisLength 和 MinorAxisLength 两个指标，数据选取前 3320 组数据，此时数据集仅含有 SEKER 和 BARBUNYA 两类

2. 单层感知机模型构建，并用随机梯度下降法进行训练
完整代码如下：

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

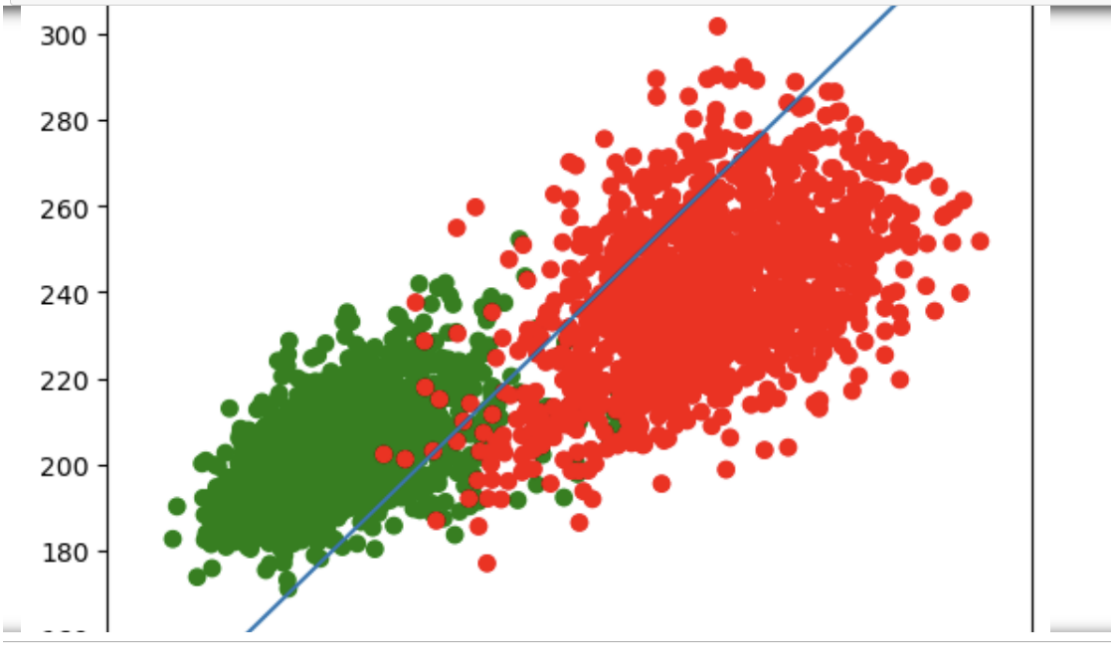
```
class perceptron:

    def __init__(self, x, y, alpha=0.001, circle=500, batchlength=20):
        self.x = x                  # 训练样本
        self.y = y                  # 训练样本中各组数据对应的类别
        self.alpha = alpha      # 学习率
        self.circle = circle    # 学习次数
        self.n = x.shape[0]     # 样本个数
        self.p = x.shape[1]     # 样本指标个数
        self.w = np.random.normal(size=(self.p, 1))
        self.b = np.random.normal(size=1)
        self.batchlength = batchlength     # 每次训练样本中使用的数据个数

    def batches(self):
        data = list(zip(self.x, self.y))
        np.random.shuffle(data)
        batches = [data[i:i+self.batchlength] for i in range(0, self.n,
self.batchlength)]
        return batches

    def sign(self, x):
        '''sign 激活函数'''
        if x > 0:
            return 1
        elif x < 0:
            return -1
        else:
            return 0

    def train(self):
        for i in range(self.circle):
            print('the {} circle'.format(i))
            for batch in self.batches():
                dw = db = 0
                num = 1
                for x, y in batch:
                    if y * (np.dot(self.w.T, x.T)+self.b) >= 0:
                        continue
                    else:
                        dw += -y * x.T
                        db += -y
                        num += 1
                if num != 0:
                    self.w -= self.alpha * dw / num
```

```
                    self.b -= self.alpha * db / num
                else:
                    continue
            color = []
            for c in self.y:
                if c == 1:
                    color.append('green')
                else:
                    color.append('red')
            x = np.arange(180, 470, 1)
            y = -self.w[0] * x/self.w[1]-self.b/self.w[1]   #分割线
            plt.plot(x, y)
            plt.scatter(np.array(self.x[:, 0]), np.array(self.x[:, 1]), color=color)
            plt.xlim([180,470])
            plt.ylim([160,320])
            plt.pause(0.1)
            plt.clf()

    def prediction(self, x):
        '''根据数据 x 判断该数据属于哪一类'''
        s = np.dot(self.w.T, x)+self.b
        output = self.sign(s)
        return output


if __name__ == '__main__':
    df = pd.DataFrame(pd.read_csv('Dry_Bean_Dataset.csv'))
    x = df.loc[0:3320, 'MajorAxisLength':'MinorAxisLength']
    y = df.loc[0:3320, 'Class']
    X = np.mat(x)
    Y = []
    for c in y:
        if c =='SEKER':
            Y.append(1)
        else:
            Y.append(-1)
    p = perceptron(x=X, y=Y)
    p.train()
```

五、 实验结果

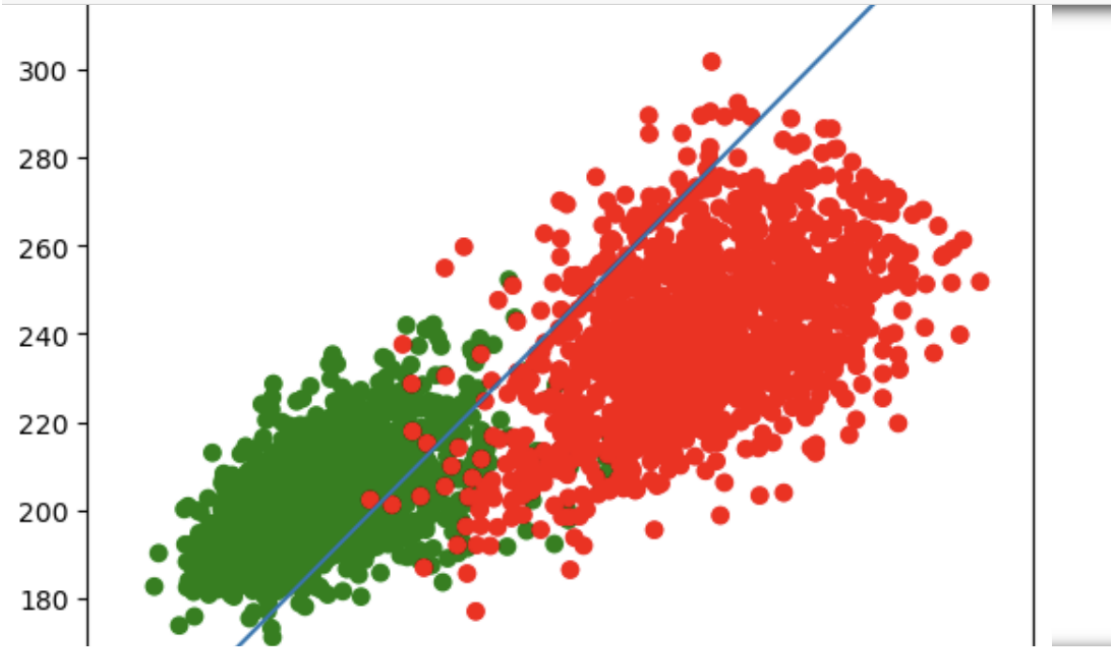
第六次

```
    p = perceptron(x=X, y=Y)
    p.train()
```



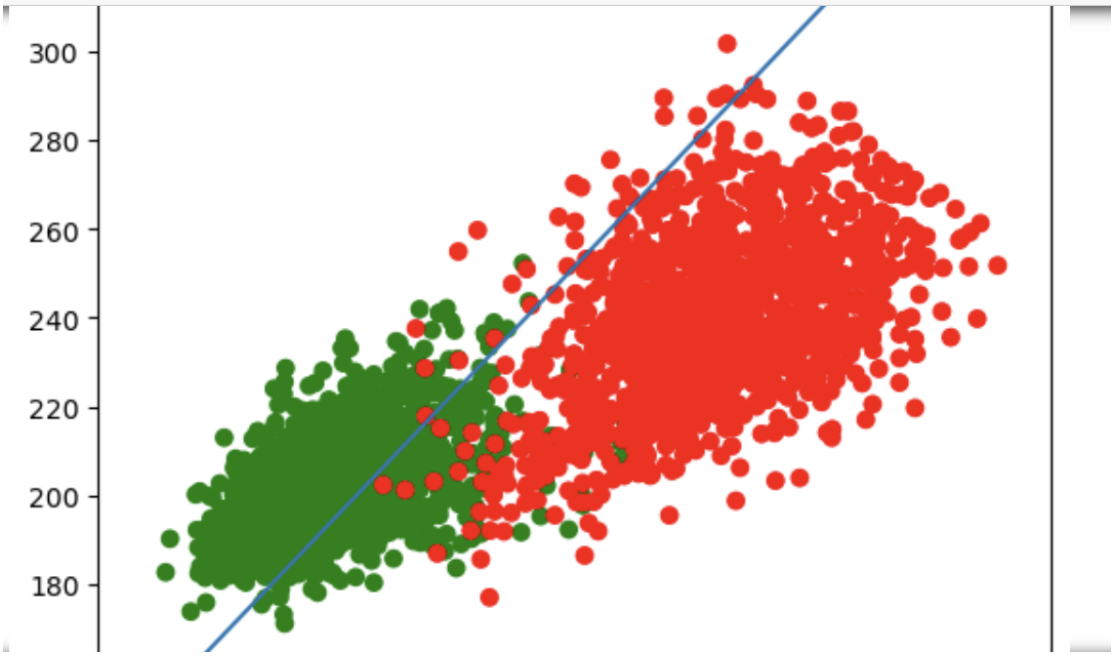
第 40 次

```
            Y.append(-1)
    p = perceptron(x=X, y=Y)
    p.train()
```

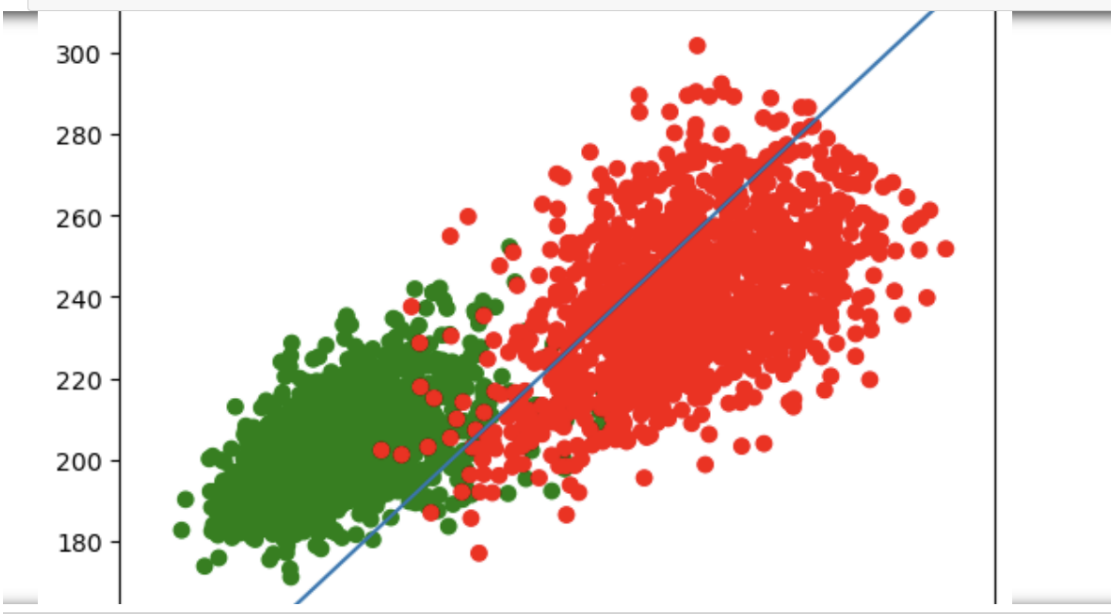


第 123 次

```
p = perceptron(x=X, y=Y)
p.train()
```



第 400 次



可见单层感知机的线性分类能力较差，甚至不如逻辑回归

六、 实验小结

本次实验是理解感知机算法的原理并利用单层感知机对 dry_bean_dataset 数据集进行分类。