

机器学习导论

习题四

191250026, 丁云翔, 191250026@smail.nju.edu.cn

2021 年 5 月 24 日

学术诚信

本课程非常重视学术诚信规范，助教老师和助教同学将不遗余力地维护作业中的学术诚信规范的建立。希望所有选课学生能够对此予以重视。¹

- (1) 允许同学之间的相互讨论，但是**署你名字的工作必须由你完成**，不允许直接照搬任何已有的材料，必须独立完成作业的书写过程；
- (2) 在完成作业过程中，对他人工作（出版物、互联网资料）中文本的直接照搬（包括原文的直接复制粘贴及语句的简单修改等）都将视为剽窃，剽窃者成绩将被取消。**对于完成作业中有关键作用的公开资料，应予以明显引用；**
- (3) 如果发现作业之间高度相似将被判定为互相抄袭行为，**抄袭和被抄袭双方的成绩都将被取消**。因此请主动防止自己的作业被他人抄袭。

作业提交注意事项

- (1) 请在L^AT_EX模板中**第一页填写个人的姓名、学号、邮箱信息**；
- (2) 本次作业需提交该pdf文件、问题1,2可直接运行的源码(nn.py, DF21.py, **不需要提交数据集**)，将以上三个文件压缩成zip文件后上传。zip文件格式为**学号.zip**，例如190000001.zip；pdf文件格式为**学号_姓名.pdf**，例如190000001_张三.pdf，**并通过教学立方提交**。
- (3) 未按照要求提交作业，或提交作业格式不正确，将会**被扣除部分作业分数**；
- (4) 本次作业提交截止时间为**5月25日23:55:00**。

¹参考尹一通老师高级算法课程中对学术诚信的说明。

1 [55pts] Neural Networks in Practice

在训练神经网络之前，我们需要确定的是整个网络的结构，在确定结构后便可以输入数据进行端到端的学习过程。考虑西瓜书第101-102页以及书中图5.7中描述的神经网络，即：输入是 d 维向量 $\mathbf{x} \in \mathbb{R}^d$ ，隐藏层由 q 个隐层单元组成，输出层为 l 个输出单元，其中隐层第 h 个神经元的阈值用 γ_h 表示，输出层第 j 个神经元的阈值用 θ_j 表示，输入层第 i 个神经元与隐层第 h 个神经元之间的连接权重为 v_{ih} ，隐层第 h 个神经元与输出层第 j 个神经元之间的连接权重为 w_{hj} ，记隐层第 h 个神经元接收到的输入为 $\alpha_h = \sum_{i=1}^d v_{ih} x_i$ ，输出为 $b_h = f(\alpha_h - \gamma_h)$ ，输出层第 j 个神经元接收到的输入为 $\beta_j = \sum_{h=1}^q w_{hj} b_h$ ，输出为 $\hat{y}_j = f(\beta_j - \theta_j)$ ， f 为对应的激活函数。

- (1) [30 pts (编程题)] 若隐层单元和输出层单元的激活函数都是Sigmoid函数，使用均方误差作为网络的损失函数（见西瓜书式(5.4)）。请打开`nn.py`程序并完成以下任务：完成Sigmoid函数及其梯度函数的编写，完成MSE损失函数和accuracy函数的编写，完成NeuralNetwork()类中向前传播predict函数，以及train函数的编写，其中包括向前传播、梯度计算、更新参数三个部分。对测试集完成尽量准确的分类预测（accuracy到0.5以上即可）。每轮梯度下降时，使用所有样本做累积BP算法，更新梯度时使用样本的平均梯度。请截图汇报测试集的MSE和Accuracy。

请注意，你需要基于numpy实现模型，除了示例代码中使用到的sklearn库函数，你将不允许使用其他sklearn函数。为方便并行计算，请使用numpy的矩阵/点积/均值/求和等函数而不是for循环进行运算，如果结果正确但使用了for循环，将会酌情扣分。

- (2) [20 pts] 神经网络学习分类问题时，模型输出层更加常用的设置是Softmax加交叉熵损失，即：假定隐层单元的激活函数是Sigmoid函数不变，对输出层，令 $z_j = \beta_j - \theta_j$ ，则输出为 $\hat{y}_j = \frac{e^{z_j}}{\sum_{i=1}^l e^{z_i}}$ ，损失函数 $E = -\sum_{j=1}^l y_j \log \hat{y}_j$ ， y_j 为该输出对应真实标记。请给出此时的梯度 $\frac{\partial E}{\partial w_{hj}}$ ， $\frac{\partial E}{\partial \theta_j}$ ， $\frac{\partial E}{\partial v_{ih}}$ 和 $\frac{\partial E}{\partial \gamma_h}$ 。（需给出计算步骤，可以像西瓜书一样定义新的符号 g_j 、 e_h ，但已有符号需使用题目中给定符号表示）

- (3) [5 pts] 相比依次直接求 v 、 γ 和 w 、 θ 的梯度，简述BP算法在计算上的优点？

Solution. 此处用于写解答(中英文均可)

(1)

完整代码见“nn.py”程序，完成需要完成的部分后，调整隐层节点数、学习率和轮数，得到一个性能较好的结果如下：测试集MSE为0.152，Accuracy为91.111%，如下图：

```

nn.py
100 # 训练网络 (可以对下面的n_hidden_layer_size, learning_rate和epochs进行修改, 以取得更好性能)
101 n_features = X.shape[1]
102 n_hidden_layer_size = 55
103 n_outputs = len(np.unique(y))
104 network = NeuralNetwork(d = n_features, q = n_hidden_layer_size, l = n_outputs)
105 network.train(X_train, y_train, learning_rate = 1, epochs = 5000)
106
107 # 预测结果
108 y_pred = network.predict(X_test)
109
110 NeuralNetwork > train() > for epoch in range(epochs)

```

Run: nn ×

Epoch 4990 loss: 0.060

Testing MSE: 0.152

Testing Accuracy: 91.111 %

Process finished with exit code 0

PyCharm 2021.1.1 available // Update... (yesterday 17:54)

(2)

① 若采用Softmax加交叉熵损失, 则 $E = -\sum_{j=1}^l y_j \log \hat{y}_j$, 设 β_j 为第 j 个输出层神经元的输入值, 对于 w_{hj} , 由链式法则, 同样有

$$\frac{\partial E}{\partial w_{hj}} = \frac{\partial E}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial w_{hj}} \quad (1)$$

$\beta_j = \sum_{h=1}^q w_{hj} b_h$, 所以有

$$\frac{\partial \beta_j}{\partial w_{hj}} = b_h \quad (2)$$

令 $z_j = \beta_j - \theta_j$, 则输出为 $\hat{y}_j = \frac{e^{z_j}}{\sum_{i=1}^l e^{z_i}}$, 损失函数为 $E = -\sum_{j=1}^l y_j \log \hat{y}_j$, 有

$$\begin{aligned}
 g_j &= \frac{\partial E}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial \beta_j} \\
 &= -\frac{y_j}{\hat{y}_j} \cdot \hat{y}_j (1 - \hat{y}_j) \\
 &= -y_j (1 - \hat{y}_j) \\
 &= y_j (\hat{y}_j - 1)
 \end{aligned} \quad (3)$$

联立(1)(2)(3)式, 可得BP算法中对 w_{hj} 的梯度为

$$\begin{aligned}
 \frac{\partial E}{\partial w_{hj}} &= g_j b_h \\
 &= y_j (\hat{y}_j - 1) b_h
 \end{aligned} \quad (4)$$

② 对于 θ_j , 由链式法则

$$\frac{\partial E}{\partial \theta_j} = \frac{\partial E}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial \theta_j} \quad (5)$$

由 z_j 、 \hat{y}_j 和 E 的定义可知

$$\begin{aligned}\frac{\partial E}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial \theta_j} &= -g_j \\ &= -y_j(\hat{y}_j - 1) \\ &= y_j(1 - \hat{y}_j)\end{aligned}\quad (6)$$

联立(5)(6)式，可得BP算法中对 θ_j 的梯度为

$$\frac{\partial E}{\partial \theta_j} = y_j(1 - \hat{y}_j) \quad (7)$$

③ 对于 v_{ih} ，设 α_h 为第 h 个隐层神经元的输入值，由链式法则

$$\frac{\partial E}{\partial v_{ih}} = \frac{\partial E}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \cdot \frac{\partial \alpha_h}{\partial v_{ih}} \quad (8)$$

$\alpha_h = \sum_{i=1}^d v_{ih}x_i$ ，所以有

$$\frac{\partial \alpha_h}{\partial v_{ih}} = x_i \quad (9)$$

设

$$\begin{aligned}e_h &= \frac{\partial E}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \\ &= \sum_{j=1}^l \frac{\partial E}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \\ &= \sum_{j=1}^l \frac{\partial E}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h}\end{aligned}\quad (10)$$

又 $b_h = \text{Sigmoid}(\alpha_h - \gamma_h)$ ，故

$$\frac{\partial b_h}{\partial \alpha_h} = b_h(1 - b_h) \quad (11)$$

$\beta_j = \sum_{h=1}^q w_{hj}b_h$ ，所以有

$$\frac{\partial \beta_j}{\partial b_h} = w_{hj} \quad (12)$$

联立(3)(10)(11)(12)式，可得

$$\begin{aligned}e_h &= \sum_{j=1}^l g_j w_{hj} b_h (1 - b_h) \\ &= b_h (1 - b_h) \sum_{j=1}^l g_j w_{hj}\end{aligned}\quad (13)$$

联立(8)(9)(10)(13)式，可得BP算法中对 v_{ih} 的梯度为

$$\begin{aligned}\frac{\partial E}{\partial v_{ih}} &= e_h x_i \\ &= b_h (1 - b_h) \left(\sum_{j=1}^l g_j w_{hj} \right) x_i\end{aligned}\quad (14)$$

④ 对于 γ_h ，由链式法则

$$\frac{\partial E}{\partial \gamma_h} = \frac{\partial E}{\partial b_h} \cdot \frac{\partial b_h}{\partial \gamma_h} \quad (15)$$

由 b_h 和 E 的定义可知

$$\begin{aligned} \frac{\partial E}{\partial b_h} \cdot \frac{\partial b_h}{\partial \gamma_h} &= -e_h \\ &= -b_h(1 - b_h) \sum_{j=1}^l g_j w_{hj} \\ &= b_h(b_h - 1) \sum_{j=1}^l g_j w_{hj} \end{aligned} \quad (16)$$

联立(15)(16)式，可得BP算法中对 γ_h 的梯度为

$$\frac{\partial E}{\partial \gamma_h} = b_h(b_h - 1) \sum_{j=1}^l g_j w_{hj} \quad (17)$$

(3)

BP算法在计算上更为简便，由于 E 和 v, γ, w, θ 不是简单的线性关系，其函数关系十分复杂，若直接对参数求梯度，计算会十分复杂；而BP算法利用链式法则求梯度，可以简化运算的复杂度，使计算更简便。而且在使用BP算法计算 v, γ 和 w, θ 的梯度时有相同的中间量，相比依次直接计算四个梯度可以避免重复计算，提高计算效率。

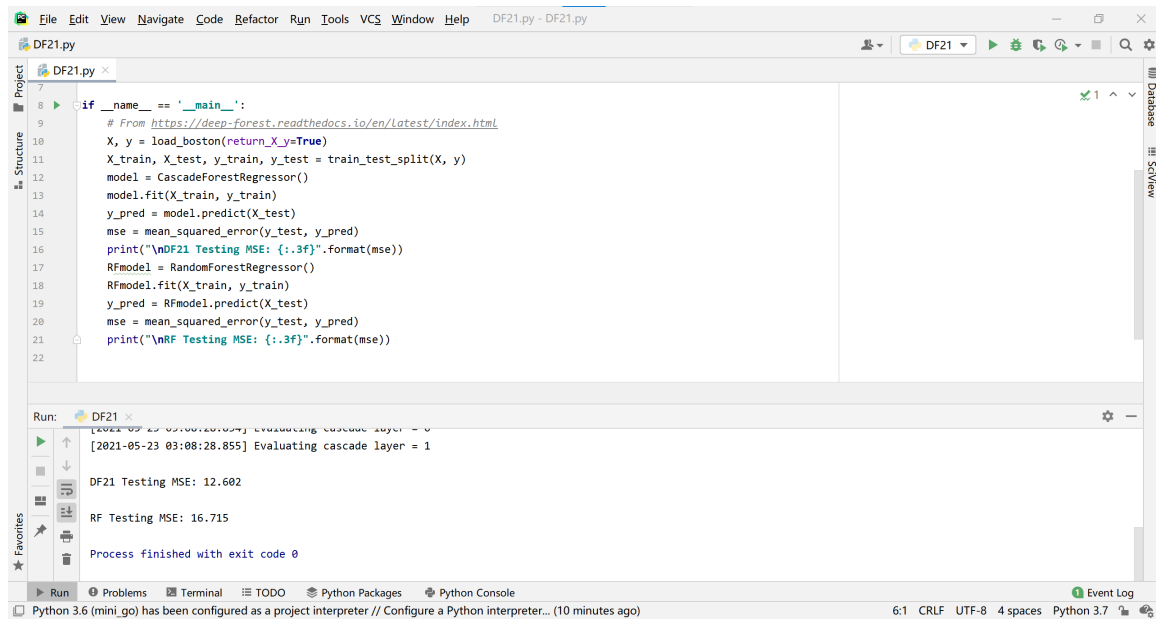
2 [10pts (编程题)] Deep Forest in Practice

深度森林是周志华老师等提出的新型深度学习模型。2021年2月1日，深度森林软件包DF21开源发布²，它拥有比其他基于决策树的集成学习方法更好的性能，更少的超参数，并且无需大量的调参，训练效率高。请安装DF21，并参考**DF21.py**，在波士顿房价预测数据上比较DF21和sklearn 随机森林的性能，汇报两个模型在测试集上的Mean Square Error (MSE)，以及将DF21的n_estimators 超参数在不同取值（大于等于3个取值）时的测试集MSE绘制成折线图。（由于模型和数据划分有随机性，可以直接用跑一次的结果，也可取跑多次的均值）

Solution. 此处用于写解答(中英文均可)

修改“DF21.py”以训练和测试随机森林，运行多次后。发现每次DF21的测试集MSE均小于随机森林的测试集MSE，在波士顿房价预测数据上DF21的性能更好，其中一次运行结果如下图：

²<https://deep-forest.readthedocs.io/en/latest/>

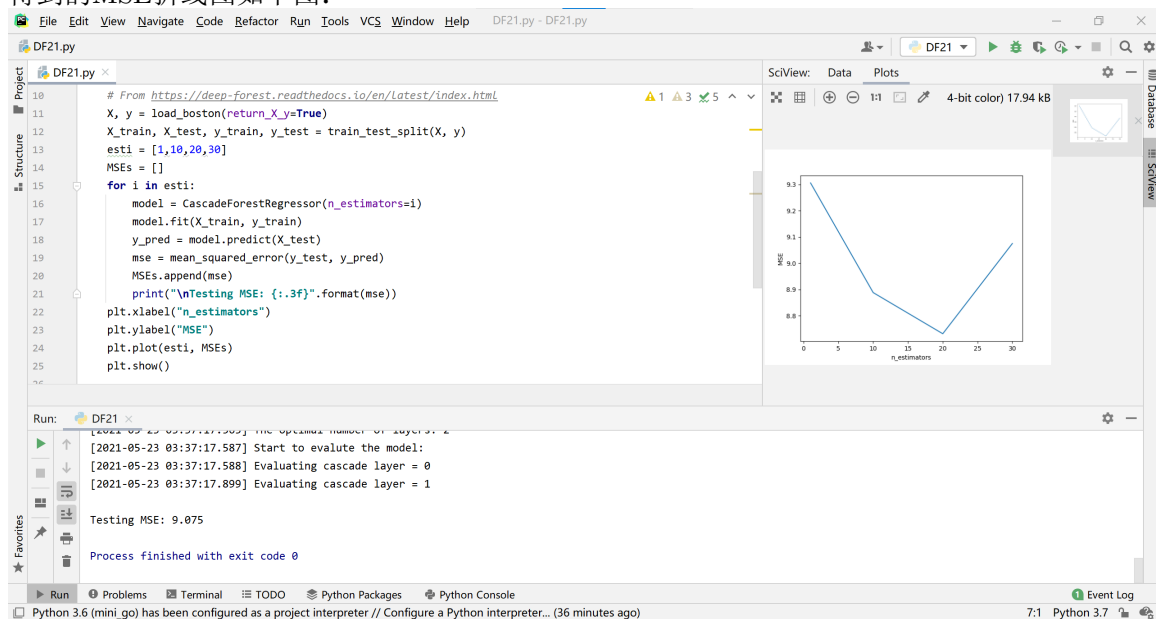


```

7  # From https://deep-forest.readthedocs.io/en/Latest/index.html
8  X, y = load_boston(return_X_y=True)
9  X_train, X_test, y_train, y_test = train_test_split(X, y)
10 model = CascadeForestRegressor()
11 model.fit(X_train, y_train)
12 y_pred = model.predict(X_test)
13 mse = mean_squared_error(y_test, y_pred)
14 print("\nDF21 Testing MSE: {:.3f}".format(mse))
15 RFmodel = RandomForestRegressor()
16 RFmodel.fit(X_train, y_train)
17 y_pred = RFmodel.predict(X_test)
18 mse = mean_squared_error(y_test, y_pred)
19 print("\nRF Testing MSE: {:.3f}".format(mse))
20
Run: DF21
[2021-05-23 03:08:20.854] Evaluating cascade layer = 0
[2021-05-23 03:08:28.855] Evaluating cascade layer = 1
DF21 Testing MSE: 12.602
RF Testing MSE: 16.715
Process finished with exit code 0

```

修改DF21的 `n_estimators` 超参数，分别选取 `n_estimators` 为1、10、20、30进行训练和测试，所得到的MSE折线图如下图：



3 [35 pts] Naive Bayes Classifier

通过对课本的学习，我们了解了采用“属性条件独立性假设”的朴素贝叶斯分类器。现在我们有如下表所示的一个数据集，其中 x_1 与 x_2 为特征，其取值集合分别为 $x_1 = \{-1, 0, 1\}$ ， $x_2 = \{B, M, S\}$ ， y 为类别标记，其取值集合为 $y = \{0, 1\}$ ：

- (1) [5pts] 通过查表直接给出的 $x = \{0, B\}$ 的类别；
- (2) [15pts] 使用所给训练数据，学习一个朴素贝叶斯分类器，并用学习得到的分类器确定 $x =$

表 1: 数据集

编号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_1	-1	-1	-1	-1	-1	0	0	0	0	0	0	1	1	1	1
x_2	B	M	M	B	B	B	M	M	S	S	S	M	M	S	S
y	0	0	1	1	0	0	0	1	1	1	1	1	1	1	0

$\{0, B\}$ 的标记，要求写出详细计算过程；

- (3) [15pts] 使用“拉普拉斯修正”，再学习一个朴素贝叶斯分类器，以及重新计算 $x = \{0, B\}$ 的标记，要求写出详细计算过程。

Solution. 此处用于写解答(中英文均可)

(1)

表中只有一个样本满足 $x = \{0, B\}$ ，其类别为 $y = 0$ ，故查表给出 $x = \{0, B\}$ 的类别为 $y = 0$ 。

(2)

首先估计类先验概率 $P(c)$ ，有

$$P(y = 0) = \frac{6}{15}$$

$$P(y = 1) = \frac{9}{15}$$

然后，为每个属性估计条件概率

$$P(x_1 = -1 \mid y = 0) = \frac{3}{6}$$

$$P(x_1 = -1 \mid y = 1) = \frac{2}{9}$$

$$P(x_1 = 0 \mid y = 0) = \frac{2}{6}$$

$$P(x_1 = 0 \mid y = 1) = \frac{4}{9}$$

$$P(x_1 = 1 \mid y = 0) = \frac{1}{6}$$

$$P(x_1 = 1 \mid y = 1) = \frac{3}{9}$$

$$P(x_2 = B \mid y = 0) = \frac{3}{6}$$

$$P(x_2 = B \mid y = 1) = \frac{1}{9}$$

$$P(x_2 = M \mid y = 0) = \frac{2}{6}$$

$$P(x_2 = M \mid y = 1) = \frac{4}{9}$$

$$P(x_2 = S \mid y = 0) = \frac{1}{6}$$

$$P(x_2 = S \mid y = 1) = \frac{4}{9}$$

由以上的概率值，学得的朴素贝叶斯分类器的表达式为

$$h_{nb}(x) = \arg \max_{c \in Y} P(c) \prod_{i=1}^d P(x_i | c) \quad (18)$$

对于 $x = \{0, B\}$ ，有

$$\begin{aligned} P(y=0) \times P(x_1=0 | y=0) \times P(x_2=B | y=0) &= \frac{6}{15} \times \frac{2}{6} \times \frac{3}{6} = \frac{1}{15} \\ P(y=1) \times P(x_1=0 | y=1) \times P(x_2=B | y=1) &= \frac{9}{15} \times \frac{4}{9} \times \frac{1}{9} = \frac{4}{135} \end{aligned}$$

$\frac{1}{15} > \frac{4}{135}$ ，由式(18)，朴素贝叶斯分类器将该样本判别为 $y=0$ 。

(3)

使用“拉普拉斯修正”，首先估计类先验概率 $\hat{P}(c)$ ，有

$$\begin{aligned} \hat{P}(y=0) &= \frac{6+1}{15+2} = \frac{7}{17} \\ \hat{P}(y=1) &= \frac{9+1}{15+2} = \frac{10}{17} \end{aligned}$$

然后，为每个属性估计条件概率

$$\begin{aligned} \hat{P}(x_1=-1 | y=0) &= \frac{3+1}{6+3} = \frac{4}{9} \\ \hat{P}(x_1=-1 | y=1) &= \frac{2+1}{9+3} = \frac{3}{12} \\ \hat{P}(x_1=0 | y=0) &= \frac{2+1}{6+3} = \frac{3}{9} \\ \hat{P}(x_1=0 | y=1) &= \frac{4+1}{9+3} = \frac{5}{12} \\ \hat{P}(x_1=1 | y=0) &= \frac{1+1}{6+3} = \frac{2}{9} \\ \hat{P}(x_1=1 | y=1) &= \frac{3+1}{9+3} = \frac{4}{12} \\ \hat{P}(x_2=B | y=0) &= \frac{3+1}{6+3} = \frac{4}{9} \\ \hat{P}(x_2=B | y=1) &= \frac{1+1}{9+3} = \frac{2}{12} \\ \hat{P}(x_2=M | y=0) &= \frac{2+1}{6+3} = \frac{3}{9} \\ \hat{P}(x_2=M | y=1) &= \frac{4+1}{9+3} = \frac{5}{12} \\ \hat{P}(x_2=S | y=0) &= \frac{1+1}{6+3} = \frac{2}{9} \\ \hat{P}(x_2=S | y=1) &= \frac{4+1}{9+3} = \frac{5}{12} \end{aligned}$$

由以上的概率值，学得的“拉普拉斯修正”后的朴素贝叶斯分类器的表达式为

$$\hat{h}_{nb}(x) = \arg \max_{c \in Y} \hat{P}(c) \prod_{i=1}^d \hat{P}(x_i | c) \quad (19)$$

对于 $x = \{0, B\}$, 有

$$\begin{aligned}\hat{P}(y=0) \times \hat{P}(x_1=0 \mid y=0) \times \hat{P}(x_2=B \mid y=0) &= \frac{7}{17} \times \frac{3}{9} \times \frac{4}{9} = \frac{28}{459} \\ \hat{P}(y=1) \times \hat{P}(x_1=0 \mid y=1) \times \hat{P}(x_2=B \mid y=1) &= \frac{10}{17} \times \frac{5}{12} \times \frac{2}{12} = \frac{25}{612}\end{aligned}$$

$\frac{28}{459} > \frac{25}{612}$, 由式(19), “拉普拉斯修正”后的朴素贝叶斯分类器将该样本判别为 $y=0$ 。

4 参考文献

- (1) 《机器学习》周志华著
- (2) NumPy Reference
- (3) DF21 Documentation