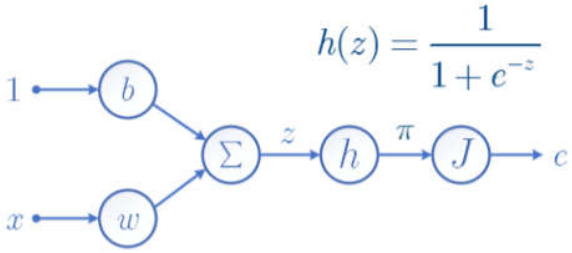
人工智能基础第三次编程

自 02 彭程 2020011075

一、推导用随机梯度下降法求解一元 Logistic 回归的过程



1.从训练数据中随机选取数据

2.从输入算输出：

$$z = wx + b$$
$$\pi = h(z) = \frac{c^z}{1+c^z} = \frac{1}{1+c^{-z}}$$
$$e = J(\pi) = -y\log\pi - (1-y)\log(1-\pi)$$

3.从输出算梯度：

$$\frac{\partial e}{\partial b} = \frac{de}{d\pi}\frac{d\pi}{dz}\frac{\partial z}{\partial b} = \frac{\pi - y}{\pi(1-\pi)}\pi(1-\pi) = \pi - y$$
$$\frac{\partial e}{\partial w} = \frac{de}{d\pi}\frac{d\pi}{dz}\frac{\partial z}{\partial w} = \frac{\pi - y}{\pi(1-\pi)}\pi(1-\pi)x = (\pi - y)x$$

其中：

$$\frac{dc}{d\pi} = \frac{d}{d\pi}J(\pi) = -\frac{y}{\pi} + \frac{1-y}{1-\pi} = \frac{\pi - y}{\pi(1-\pi)}$$
$$\frac{d\pi}{dz} = \frac{d}{dz}h(z) = \frac{c^{-z}}{(1+e^{-z})^2} = \frac{1}{1+e^{-z}}\frac{c^{-z}}{1+e^{-z}} = \pi(1-\pi)$$
$$\frac{\partial z}{\partial b} = 1 \qquad \frac{\partial z}{\partial w} = x$$

4.进行梯度更新：

$$w \leftarrow w - \alpha\frac{\partial e}{\partial w}$$
$$b \leftarrow b - \alpha\frac{\partial e}{\partial b}$$

5.如果没有达到要求，重复上述过程。

二、编程实现该随机梯度下降算法，根据花瓣长度特征，对数据集中的山鸢尾和维吉尼亚鸢尾使用 Logistic 回归进行二分类。

此处我们编程实现了上述的 logistic 回归，其中超参数的选取为：lr = 0.1, epoch = 50, batch_size = 32。经过 20-30 个 epoch，在训练集得到的正确率为 1。

```
Epoch50, Average Loss:0.2934882849022163, Train Acc:1.0
```

在测试集上同样取得正确率 1（图见下一部分）。

三、使用 Accuracy、 Sensitivity、 Specificity、 Recall、 Precision、 F1、 auROC 等指标评价你的分类方法。

测试的参考量的计算公式如下:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}$$

$$BER = \frac{1}{2}\left(\frac{FP}{FP + TN} + \frac{FN}{FN + TP}\right)$$

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(FP + TN)(TN + FN)(FN + FP)}}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{TN + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{Precision} = \frac{\text{TP}}{TP + FP}$$

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

此外的 auROC、auPRC 可以调用 sklearn 中的相关函数求取。

实验结果如下：

```
My model:
Accuracy: 1.0000
BER: 0.0000
MCC: 1.0000
Sensitivity: 1.0000
Specificity: 1.0000
Recall: 1.0000
Precision: 1.0000
F1: 1.0000
auPRC: 1.0000
auROC: 1.0000
```

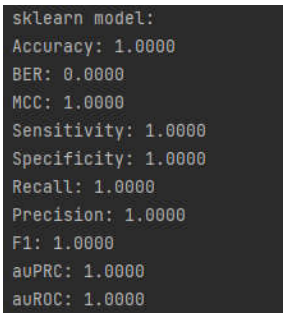
可见该分类任务比较简单，在测试集上能够达到全部正确。

四、使用 scikit-learn 中的 LogisticRegression 分类器求解该二分类问题，与自己实现的效果进行对比。

使用如下的函数：

```
# sklearn
clf = linear_model.LogisticRegression(random_state=0).fit(x_train.reshape(-1, 1), y_train)
y_test_pred = clf.predict(x_test.reshape(-1, 1))
print("sklearn model:")
model_evaluate(y_test, y_test_pred)
```

评价效果如下图：



由于此二分类比较简单，故 scikit-learn 中的 LogisticRegression 分类器和自己实现的模型没有差异。