Microsoft

Reactor

一起学人工智能系列
- 线性分类1

2021-09-15

Map


Stockholm
COMING SOON
London
Redmond
Toronto
New York
San Francisco
Tel Aviv
Shanghai
Abu Dhabi
Bangalore
Sao Paulo
Sydney

个人介绍

Kinfey Lo – (卢建晖)

Microsoft Cloud Advocate

前微软MVP、Xamarin MVP和微软RD，拥有超过10年的云原生、人工智能和移动应用经验，为教育、金融和医疗提供应用解决方案。 Microsoft Iginte，Teched 会议讲师，Microsoft AI 黑客马拉松教练，目前在微软，为技术人员和不同行业宣讲技术和相关应用场景。

爱编程(Python， C#，TypeScript，Swift，Rust，Go ）

专注于人工智能，云原生，跨平台移动开发

Github : https://github.com/kinfey

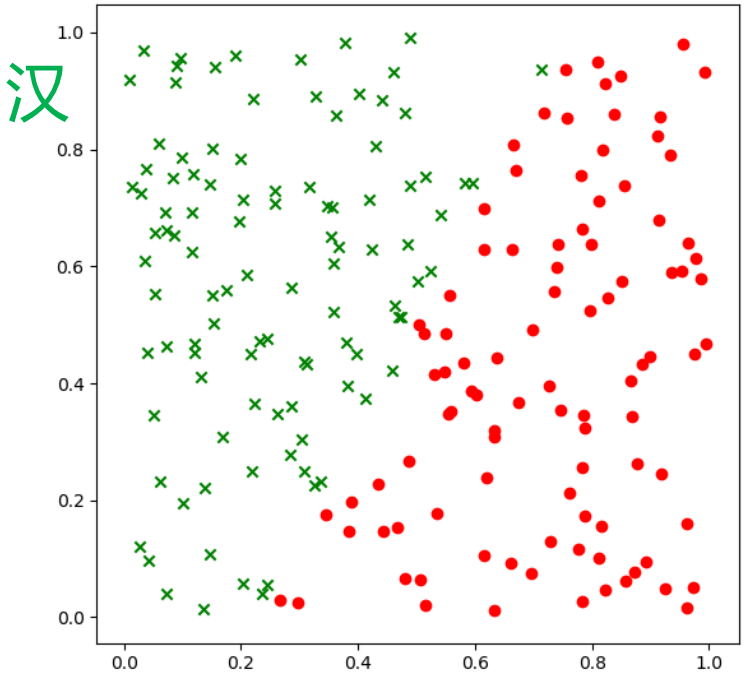
Email : kinfeylo@microsoft.com **Blog :** https://blog.csdn.net/kinfey

Twitter : @Ljh8304

引入

知识引入

分类问题在很多资料中都称之为逻辑回归，Logistic Regression，其原因是使用了线性回归中的线性模型，加上一个Logistic二分类函数，共同构造了一个分类器



样本序号	X_1:经度相对值	X_2:纬度相对值	Y:1=汉, 0=楚
1	0.325	0.888	1
2	0.656	0.629	0
3	0.151	0.101	1
4	0.785	0.024	0
...	...	...	...
200	0.631	0.001	0

归一化

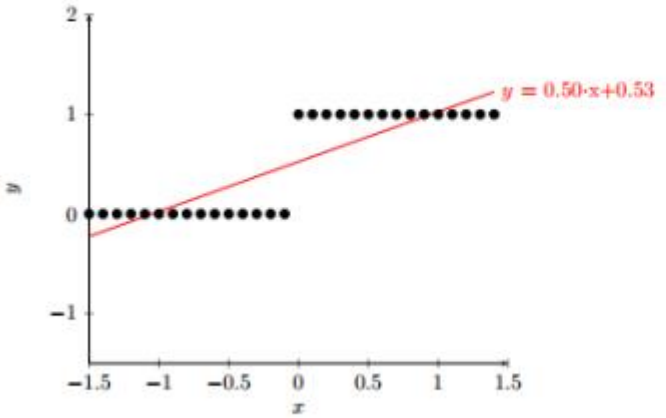
1.经纬度相对坐标值为 (0.58,0.92) 时，属于楚还是汉?

2.经纬度相对坐标值为 (0.62,0.55) 时，属于楚还是汉?

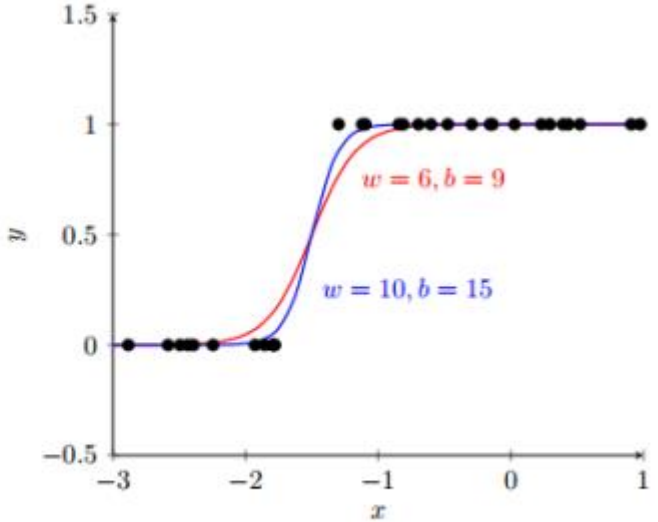
3.经纬度相对坐标值为 (0.39,0.29) 时，属于楚还是汉?

回归

- 回归
  - 线性回归
  - 逻辑回归



(a) 线性回归

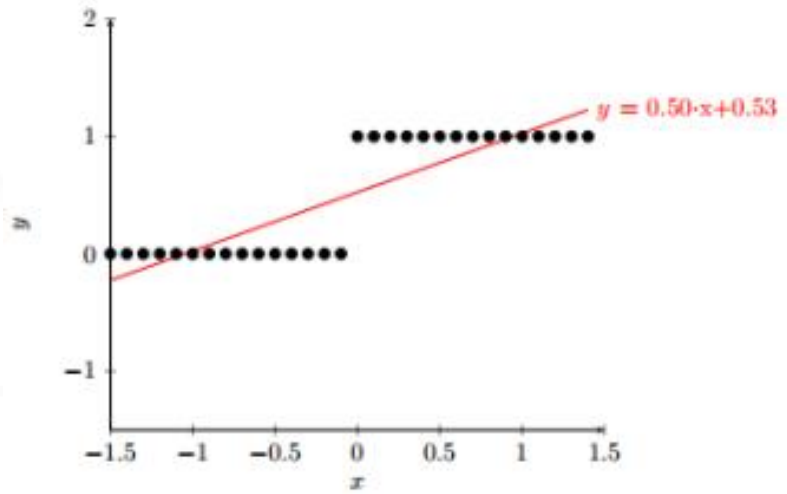


(b) Logistic 回归

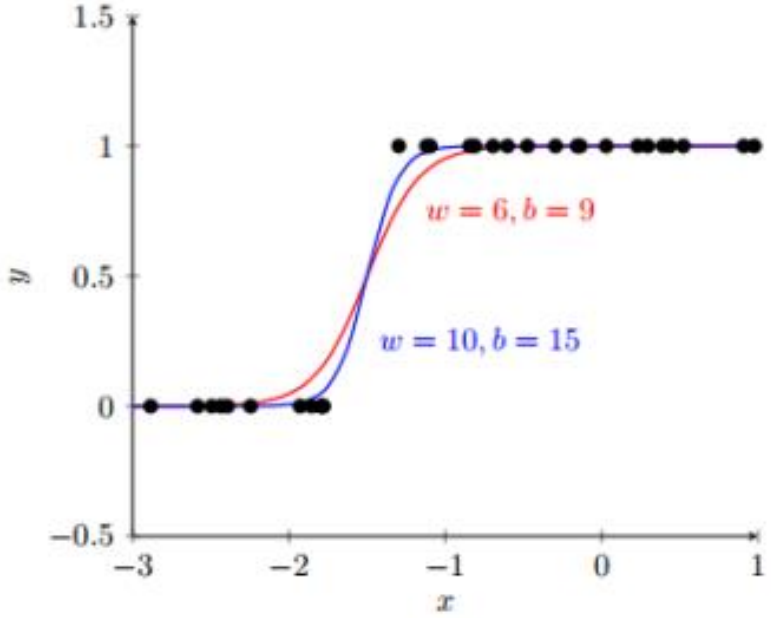
一.逻辑回归

逻辑回归 (Logistic Regression)

逻辑回归（Logistic Regression），回归给出的结果是事件成功或失败的概率。当因变量的类型属于二值（1/0，真/假，是/否）变量时，我们就应该使用逻辑回归



(a) 线性回归



(b) Logistic 回归

线性回归使用一条直线拟合样本数据

逻辑回归的目标是“拟合”0或1两个数值，而不是具体连续数值，所以称为广义线性模型

逻辑回归 (Logistic Regression)

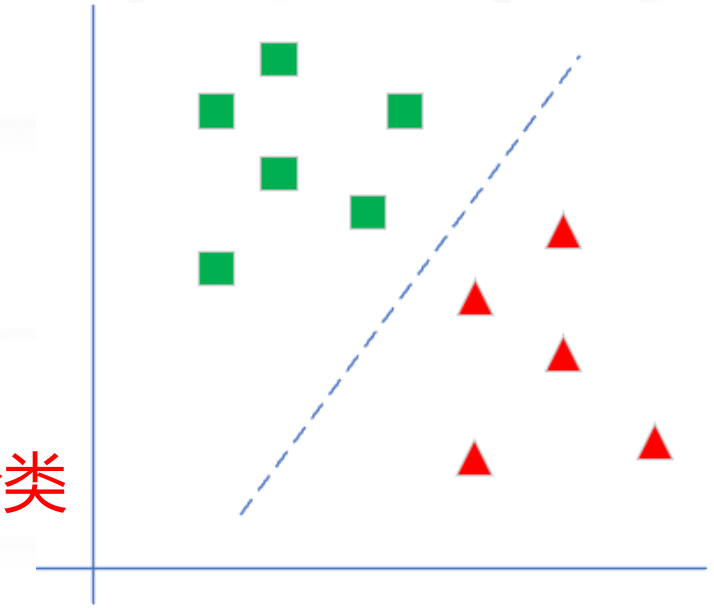
探讨引发疾病的危险因素，并根据危险因素预测疾病发生的概率等。

以胃癌病情分析为例，选择两组人群，一组是胃癌组，一组是非胃癌组，两组人群必定具有不同的体征与生活方式等。因此因变量就为是否胃癌，值为“是”或“否”；自变量就可以包括很多了，如年龄、性别、饮食习惯、幽门螺杆菌感染等。

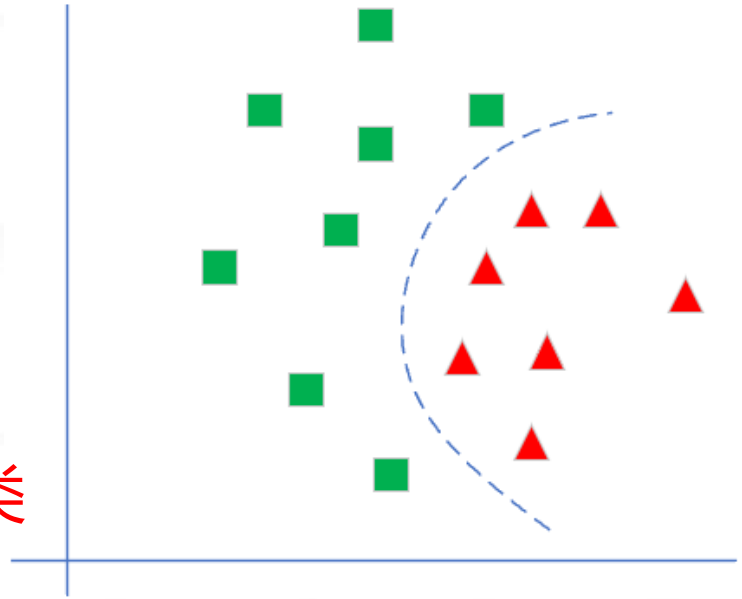
逻辑回归 (Logistic Regression)

自变量既可以是连续的，也可以是分类的。然后通过Logistic回归分析，可以得到自变量的权重，从而可以大致了解到底哪些因素是胃癌的危险因素。同时根据该权值可以根据危险因素预测一个人患癌症的可能性。

逻辑回归的另外一个名字叫做分类器，分为线性分类器和非线性分类器，本章中我们学习线性分类器。而无论是线性还是非线性分类器，又分为两种：二分类问题和多分类问题



线性二分类



非线性二分类

二分类函数

- 公式

$$Logistic(z) = \frac{1}{1+e^{-z}} \rightarrow a$$

- 导数

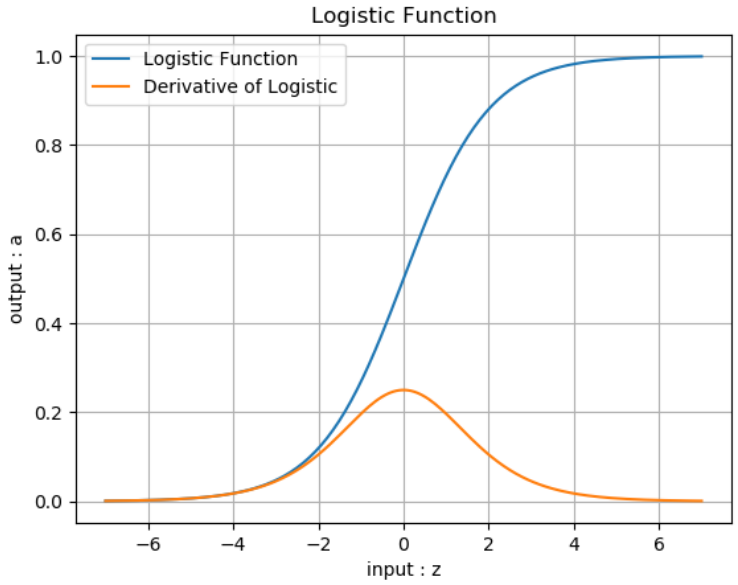
$$Logistic'(z) = a(1-a)$$

- 输入值域

$$(-\infty, \infty)$$

- 输出值域

$$(0, 1)$$



此函数实际上是一个概率计算，它把(−∞,∞)之间的任何数字都压缩到(0,1)之间，返回一个概率值，这个概率值接近1时，认为是正例，否则认为是负例。

训练时，一个样本x在经过神经网络的最后一层的矩阵运算结果作为输入z，经过Logistic计算后，输出一个(0,1)之间的预测值。我们假设这个样本的标签值为0属于负类，如果其预测值越接近0，就越接近标签值，那么误差越小，反向传播的力度就越小。

推理时，我们预先设定一个阈值比如0.5，则当推理结果大于0.5时，认为是正类；小于0.5时认为是负类；等于0.5时，根据情况自己定义。阈值也不一定就是0.5，也可以是0.65等等，阈值越大，准确率越高，召回率越低；阈值越小则相反，准确度越低，召回率越高。

•input=2时，output=0.88，而0.88>0.5，算作正例

•input=-1时，output=0.27，而0.27<0.5，算作负例

正向传播

正向传播

矩阵运算

$$z = x \cdot w + b \tag{1}$$

分类计算

$$a = Logistic(z) = \frac{1}{1+e^{-z}} \tag{2}$$

损失函数计算

二分类交叉熵损失函数：

$$loss(w, b) = -[y \ln a + (1-y) \ln(1-a)] \tag{3}$$

反向传播

反向传播

求损失函数对a的偏导

$$\frac{\partial loss}{\partial a} = -[\frac{y}{a} + \frac{-(1-y)}{1-a}] = \frac{a-y}{a(1-a)} \tag{4}$$

求a对z的偏导

$$\frac{\partial a}{\partial z} = a(1-a) \tag{5}$$

求损失函数loss对z的偏导

使用链式法则链接公式4和公式5：

$$\frac{\partial loss}{\partial z} = \frac{\partial loss}{\partial a}\frac{\partial a}{\partial z}$$

$$= \frac{a-y}{a(1-a)} \cdot a(1-a) = a - y \tag{6}$$

多样本

多样本情况

我们用三个样本做实例化推导：

$$
\begin{aligned}
Z &= \begin{pmatrix} z_1 & z_2 & z_3 \end{pmatrix} \\
A &= Logistic\begin{pmatrix} z_1 & z_2 & z_3 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 & a_3 \end{pmatrix}
\end{aligned}
$$

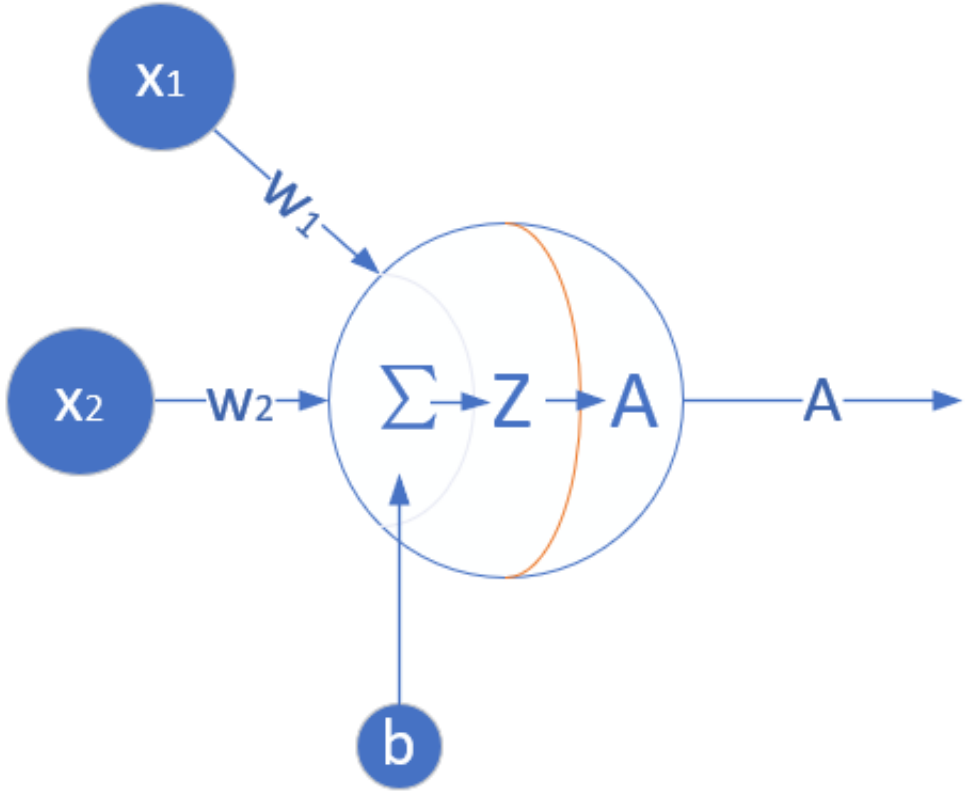
$$
\begin{aligned}
J(w,b) = &-\left[y_1 \ln a_1 + (1-y_1)\ln(1-a_1)\right] \\
&-\left[y_2 \ln a_2 + (1-y_2)\ln(1-a_2)\right] \\
&-\left[y_3 \ln a_3 + (1-y_3)\ln(1-a_3)\right]
\end{aligned}
$$

代入公式6结果：

$$
\begin{aligned}
\frac{\partial J(w,b)}{\partial z} &= \begin{pmatrix} \frac{\partial J(w,b)}{\partial z_1} & \frac{\partial J(w,b)}{\partial z_2} & \frac{\partial J(w,b)}{\partial z_3} \end{pmatrix} \\
&= \begin{pmatrix} a_1 - y_1 & a_2 - y_2 & a_3 - y_3 \end{pmatrix} \\
&= A - Y
\end{aligned}
$$

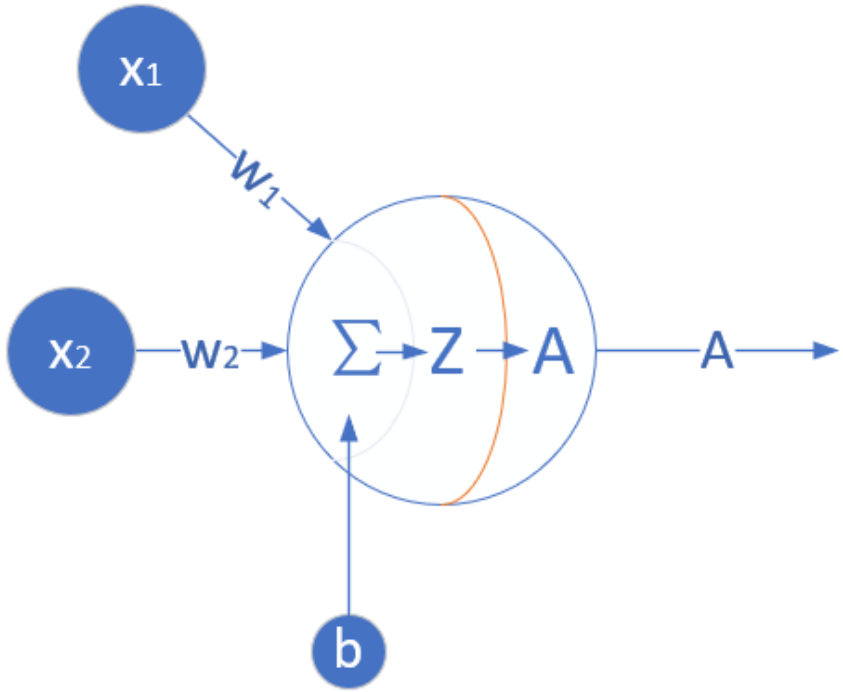
所以，用矩阵运算时可以简化为矩阵相减的形式： $A - Y$ 。

神经网络实现线性二分类



- 从视觉上判断是线性可分的，所以我们使用单层神经网络即可；
- 输入特征是经度和纬度，所以我们在输入层设置两个输入X1=经度，X2=维度；
- 最后输出的是一个二分类，分别是楚汉地盘，可以看成非0即1的二分类问题，所以我们只用一个输出单元就可以了。

神经网络实现线性二分类



输入层

输入经度 x_1 和纬度 x_2 两个特征：

$$X = \begin{pmatrix} x_1 & x_2 \end{pmatrix}$$

权重矩阵

输入是2个特征，输出一个数，则 W 的尺寸就是2x1：

$$w = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix}$$

B的尺寸是1x1，行数永远是1，列数永远和W一样。

$$B = \begin{pmatrix} b_1 \end{pmatrix}$$

输出层

$$\begin{aligned} z = x \cdot w + b &= \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} \\ &= x_1 \cdot w_1 + x_2 \cdot w_2 + b \end{aligned} \tag{1}$$

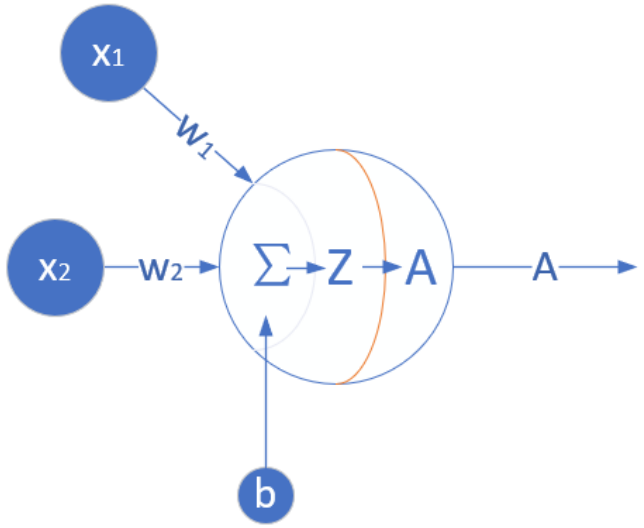
$$a = Logistic(z) \tag{2}$$

损失函数

二分类交叉熵损失函数：

$$loss(w,b) = -[ylna + (1-y)ln(1-a)] \tag{3}$$

神经网络实现线性二分类



反向传播

我们在上一节已经推导了loss对z的偏导数，结论为 $A-Y$ 。接下来，我们求loss对w的导数。本例中，w的形式是一个2行1列的向量，所以求w的偏导时，要对向量求导：

$$\begin{aligned}\frac{\partial \text{loss}}{\partial w} &= \begin{pmatrix} \partial \text{loss}/\partial w_1 \\ \partial \text{loss}/\partial w_2 \end{pmatrix} \\ &= \begin{pmatrix} \frac{\partial \text{loss}}{\partial z}\frac{\partial z}{\partial w_1} \\ \frac{\partial \text{loss}}{\partial z}\frac{\partial z}{\partial w_2} \end{pmatrix} = \begin{pmatrix} (a-y)x_1 \\ (a-y)x_2 \end{pmatrix} \\ &= (x_1 x_2)^T (a-y)\end{aligned} \tag{4}$$

上式中 x_1, x_2 是一个样本的两个特征值。如果是多样本的话，公式4将会变成其矩阵形式，以3个样本为例：

$$\begin{aligned}\frac{\partial J(w,b)}{\partial w} &= \begin{pmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ x_{31} & x_{32} \end{pmatrix}^T \begin{pmatrix} a_1 - y_1 \\ a_2 - y_2 \\ a_3 - y_3 \end{pmatrix} \\ &= X^T(A-Y)\end{aligned} \tag{5}$$

二.线性二分类

线性分类 vs 线性回归

	线性回归	线性分类
相同点	需要在样本群中找到一条直线	需要在样本群中找到一条直线
不同点	用直线来拟合所有样本，使得各个样本到这条直线的距离尽可能最短	用直线来分割所有样本，使得正例样本和负例样本尽可能分布在直线两侧

二分类数学原理-代数方式

代数方式：通过一个分类函数计算所有样本点在经过线性变换后的概率值，使得正例样本的概率大于0.5，而负例样本的概率小于0.5。

1. 正向计算

$$z = x_1 w_1 + x_2 w_2 + b \tag{1}$$

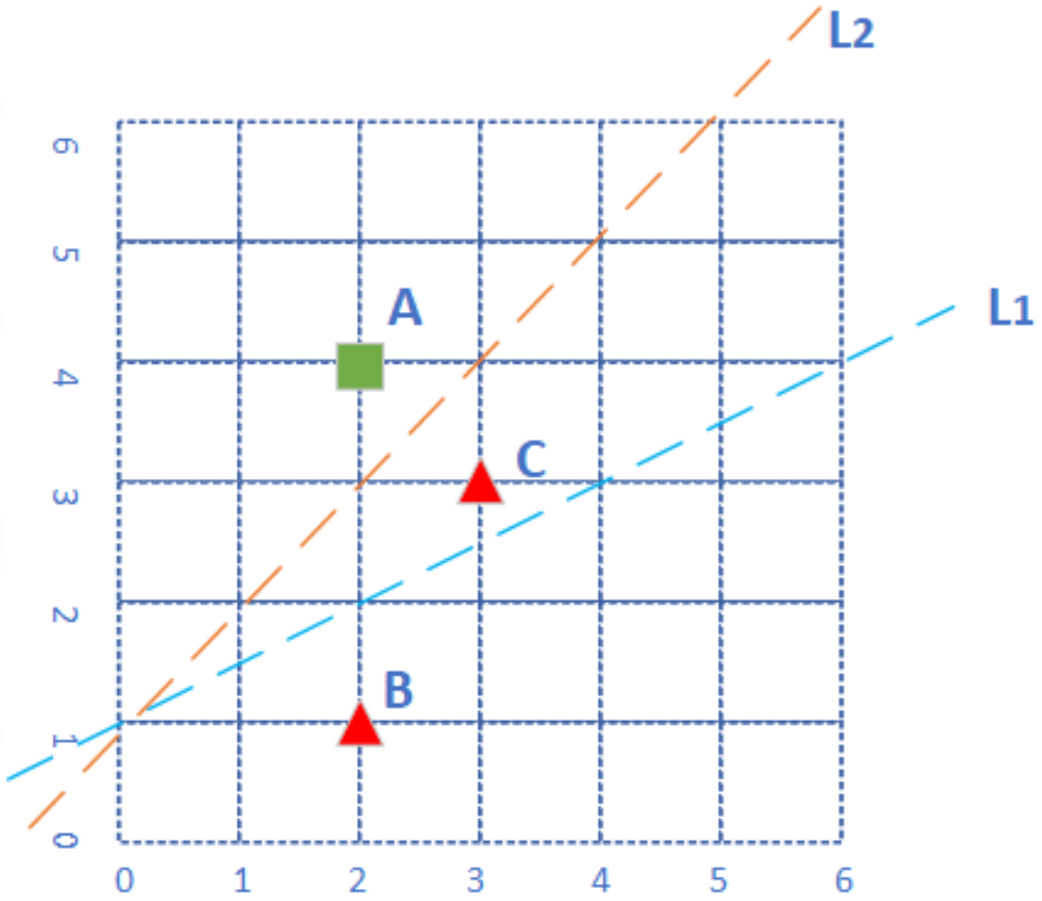
1. 分类计算

$$a = \frac{1}{1+e^{-z}} \tag{2}$$

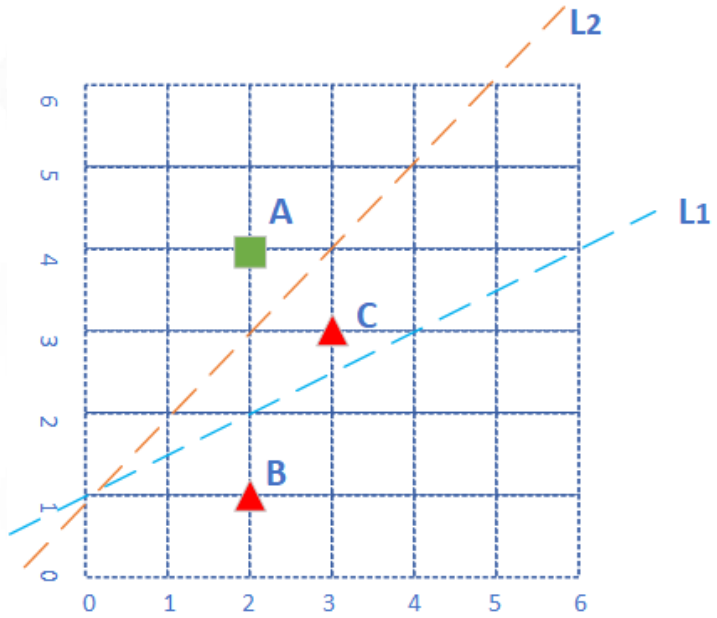
1. 损失函数计算

$$loss = -[y \ln(a) + (1-y) \ln(1-a)] \tag{3}$$

二分类数学原理-代数分析



二分类数学原理-代数分析



分类线为L1时

假设神经网络第一次使用 L_1 做为分类线，此时： $w_1 = -1, w_2 = 2, b = -2$ ，我们来计算一下三个点的情况。

A 点:

$$z_A = (-1) \times 2 + 2 \times 4 - 2 = 4 > 0 \quad \text{(正确)}$$

B 点:

$$z_B = (-1) \times 2 + 2 \times 1 - 2 = -2 < 0 \quad \text{(正确)}$$

C 点:

$$z_C = (-1) \times 3 + 2 \times 3 - 2 = 1 > 0 \quad \text{(错误)}$$

我们知道当 $z > 0$ 时， $Logistic(z) > 0.5$ 为正例，反之为负例，所以我们只需要看三个点的 z 值是否大于0或小于0就可以了，不用再计算 $Logistic$ 的函数值。

其中， A、B 点是处于正确的分类区，而 C 点处于错误的分类区。此时 C 点的损失函数值为（注意 C 的标签值 $y = 0$ ）：

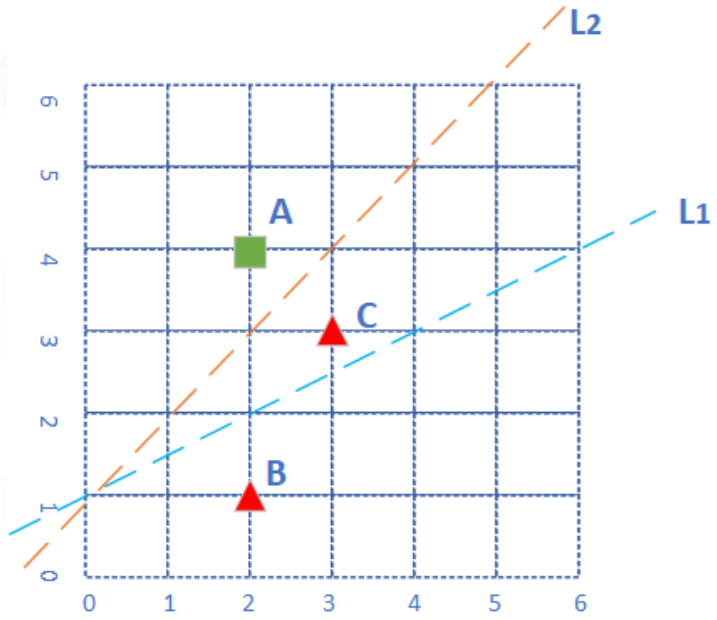
$$a_C = Sigmoid(z_C) = 0.731$$

$$loss_Z = -(0 \cdot \ln(0.731) + 1 \cdot \ln(1 - 0.731)) = 1.313$$

读者可能对1.313这个值没有什么概念，是大还是小呢？我们不妨计算一下分类正确的 A、B 点的坐标：

$$loss_A = 0.018, \quad loss_B = 0.112$$

二分类数学原理-代数分析



分类线为L1时

假设神经网络第一次使用 L_1 做为分类线，此时： $w_1 = -1, w_2 = 2, b = -2$ ，我们来计算一下三个点的情况。

A 点:

$$z_A = (-1) \times 2 + 2 \times 4 - 2 = 4 > 0 \quad \text{(正确)}$$

B 点:

$$z_B = (-1) \times 2 + 2 \times 1 - 2 = -2 < 0 \quad \text{(正确)}$$

C 点:

$$z_C = (-1) \times 3 + 2 \times 3 - 2 = 1 > 0 \quad \text{(错误)}$$

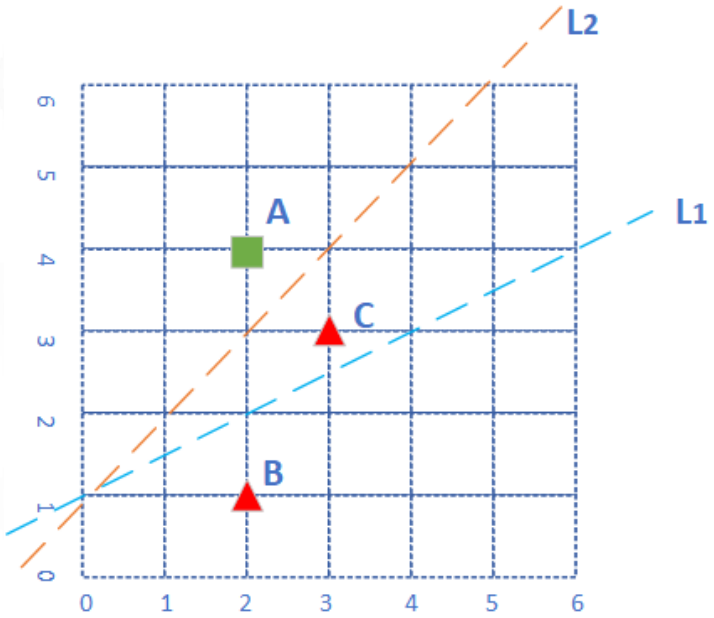
我们知道当 $z > 0$ 时， $Logistic(z) > 0.5$ 为正例，反之为负例，所以我们只需要看三个点的 z 值是否大于0或小于0就可以了，不用再计算 $Logistic$ 的函数值。

其中， A、B 点是处于正确的分类区，而 C 点处于错误的分类区。此时 C 点的损失函数值为（注意 C 的标签值 $y = 0$ ）：

$$a_C = Sigmoid(z_C) = 0.731$$

$$loss_Z = -(0 \cdot \ln(0.731) + 1 \cdot \ln(1 - 0.731)) = 1.313$$

二分类数学原理-代数分析



分类线为L2时

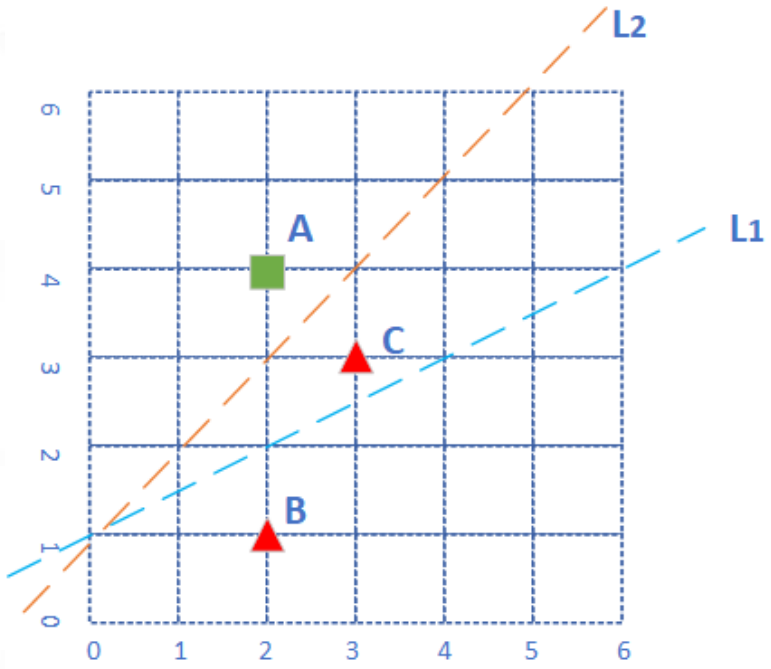
我们假设经过反向传播后，神经网络把直线的位置调整到 L_2，以 L_2 做为分类线，即 $w_1 = -1, w_2 = 1, b = -1$，则三个点的 z 值都会是符合其分类的：

$$z_A = (-1) \times 2 + 1 \times 4 - 1 = 1 > 0 \quad \text{(正确)}$$

$$z_B = (-1) \times 2 + 1 \times 1 - 1 = -2 < 0 \quad \text{(正确)}$$

$$z_C = (-1) \times 3 + 1 \times 3 - 1 = -1 < 0 \quad \text{(正确)}$$

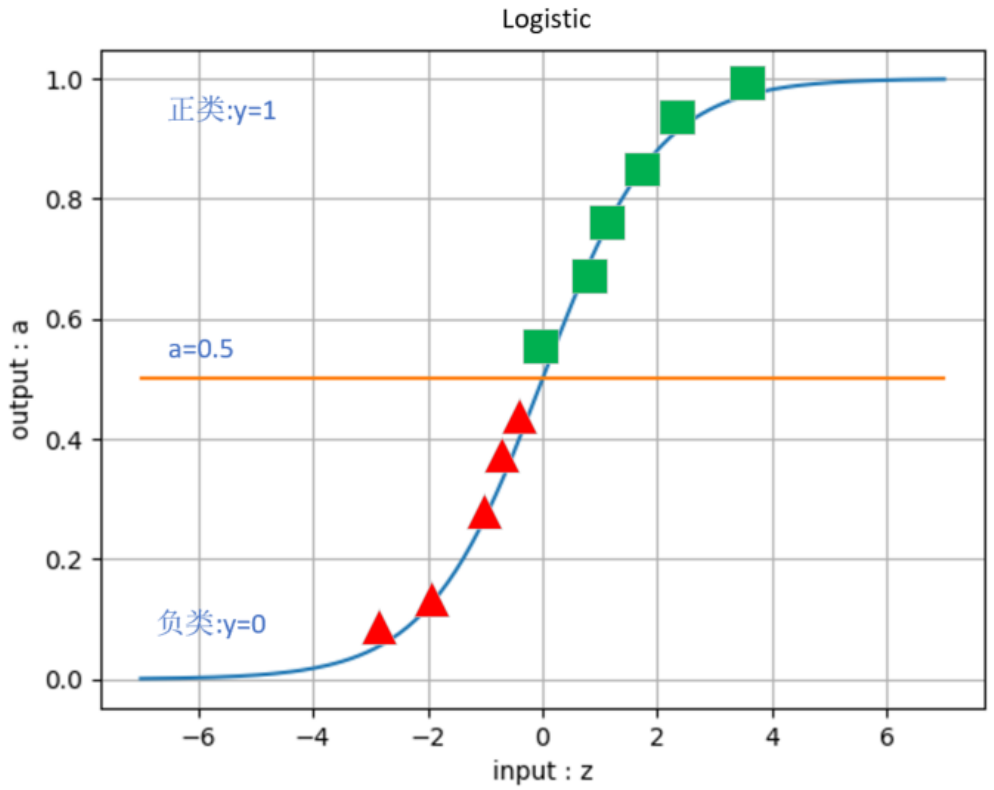
既然用z值是否大于0这个条件就可以判断出分类是否正确，那么二分类理论中为什么还要用Logistic函数做一次分类呢



只有z值的话，我们只能知道是大于0还是小于0，并不能有效地进行反向传播，也就是说我们无法告诉神经网络反向传播的误差的力度有多大。比如z=5和z=−1相比，难度意味着前者的力度是后者的5倍吗?

而有了Logistic分类计算后，得到的值是一个(0,1)之间的概率，比如：当z=5时，Logistic(5)=0.993；当z=−1时，Logistic(−1)=0.269。这两个数值的含义是这两个样本在分类区内的概率，前者概率为99.3%，偏向正例，后者概率为26.9%，偏向负例。然后再计算损失函数，就可以得到神经网络可以理解的反向传播误差，比如上面曾经计算过的lossA、lossB、lossC。

二分类数学原理-几何



$$a = Logistic(z) = \frac{1}{1+e^{-z}} > 0.5$$

做公式变形，两边取自然对数，可以得到：

$$z > 0$$

即： $z = x_1 \cdot w_1 + x_2 \cdot w_2 + b > 0$

对上式做一下变形，把 x_2 放在左侧，其他项放在右侧（假设 $w_2 > 0$，则不等号方向不变）：

$$x_2 > -\frac{w_1}{w_2}x_1 - \frac{b}{w_2} \tag{5}$$

简化一下两个系数，令 $w' = -w1/w2$，$b' = -b/w2$：

$$x_2 > w' \cdot x_1 + b' \tag{6}$$

公式6用几何方式解释，就是：有一条直线，方程为 $z = w' \cdot x_1 + b'$，所有的正例样本都处于这条直线的上方；同理可得所有的负例样本都处于这条直线的下方。

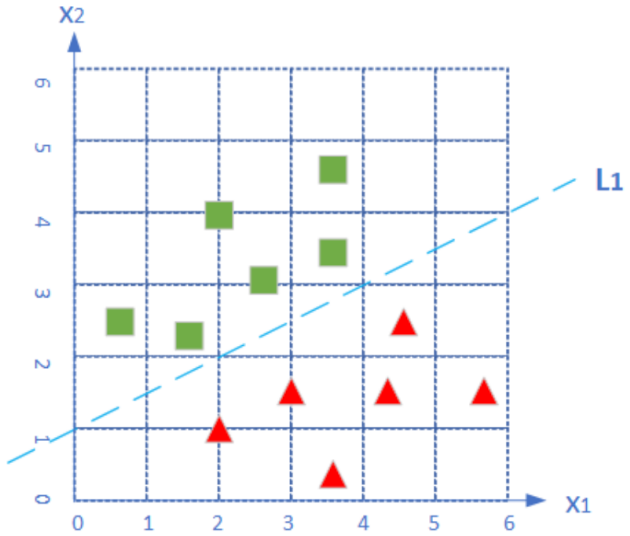
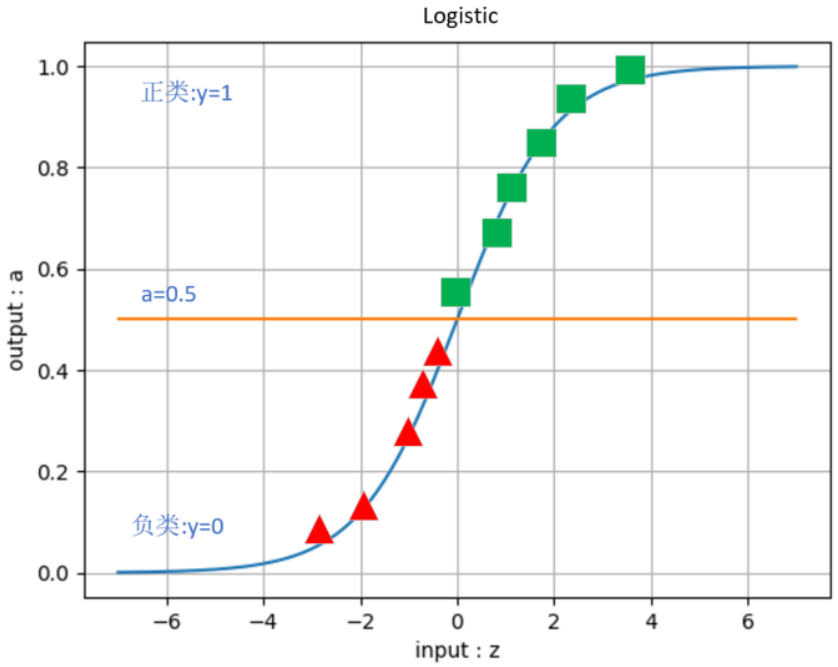


图6-7 用直线分开的两类样本

二分类数学原理-几何



假设绿色方块为正类：标签值y=1，红色三角形为负类：标签值y=0。从几何关系上理解，如果我们有一条直线，其公式为：$z=w'\cdot x1+b'$，如图中的虚线L1所示，则所有正类的样本的x2都大于z，而所有的负类样本的x2都小于z，那么这条直线就是我们需要的分割线。

这就说明神经网络的工作原理和我们在二维平面上的直观感觉是相同的，即神经网络的工作就是找到这么一条合适的直线，尽量让所有正例样本都处于直线上方时，负例样本处于直线的下方。其实这与线性回归中找到一条直线穿过所有样本点的过程有异曲同工之处。

我们还有一个额外的收获，即：

$$w'=-w1/w2(7)$$

$$b'=-b/w2(8)$$

我们可以使用神经网络计算出w1，w2，bw_1，w_2，bw1，w2，b三个值以后，换算成w'，b'w'，b'w'，b'，以便在二维平面上画出分割线，来直观地判断神经网络训练结果的正确性。

三.分类可视化

可视化的重要性

- 样本数据量比较少，一共只有200个样本，如果再分成两部分，会造成数据集覆盖不全面，存在很大的差异，对训练、验证、测试都没有帮助

- 由于本例的数据特征比较少，所以我们有更好的手段：可视化。在神经网络学习初期，可视化的训练过程与结果会对读者有巨大的帮助。

权重值的含义

```
W= [[-7.66469954]
 [ 3.15772116]]
B= [[2.19442993]]
A= [[0.65791301]
 [0.30556477]
 [0.53019727]]
```

在上一节中我们一起学习了线性二分类的原理，其中提到了如果我们能够根据训练结果，在图上画出一条直线来分割正例和负例两个区域，是不是就很直观了呢?

$$z = x_1 \cdot w_1 + x_2 \cdot w_2 + b \tag{1}$$

$$a = Logistic(z) \tag{2}$$

对公式2来说，当a大于0.5时，属于正例（属于汉），当a小于0.5时，属于负例（属于楚）。那么a=0.5时，就是楚汉边界啦!

$a = 0.5$,相当于$z = 0$

$$z = x_1 \cdot w_1 + x_2 \cdot w_2 + b = 0$$

把x2留在等式左侧，其它的挪到右侧去，就可以得到一条直线的方程了:

$$x_2 \cdot w_2 = -x_1 \cdot w_1 - b$$

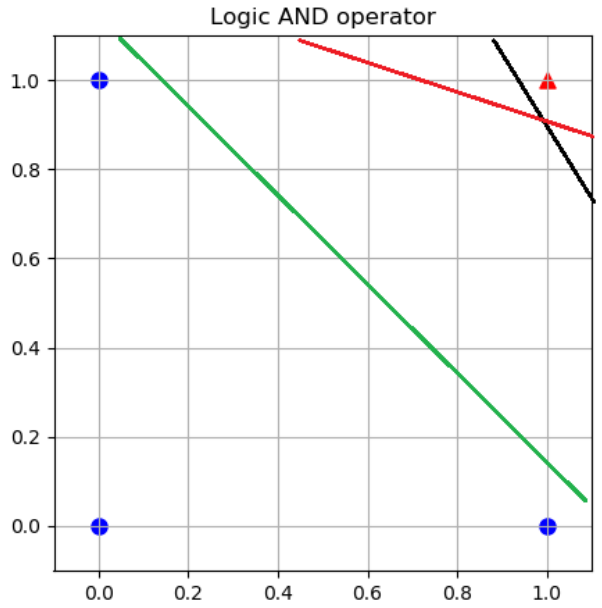
$$x_2 = -\frac{w_1}{w_2}x_1 - \frac{b}{w_2} \tag{3}$$

好了，这就是标准的直线方程 $y = ax + b$ 的形式了。这个公式等同于二分类原理中的公式7，8。

实现逻辑与或非门

单层神经网络，又叫做感知机，它可以轻松实现逻辑与、或、非门。由于逻辑与、或门，需要有两个变量输入，而逻辑非门只有一个变量输入。但是它们共同的特点是输入为0或1，可以看作是正负两个类别。
所以，在学习了二分类知识后，我们可以用分类的思想来实现下列5个逻辑门：

- 与门 AND
- 与非门 NAND
- 或门 OR
- 或非门 NOR
- 非门 NOT



应该在红色点和蓝色点之间划出一条分割线来，可以正好把正例和负例完全分开

实现逻辑非门

实现逻辑非门

很多阅读材料上会这样介绍：有公式 $y = wx + b$，令 $w = -1, b = 1$，则：

- 当 $x = 0$ 时，$y = -1 \times 0 + 1 = 1$
- 当 $x = 1$ 时，$y = -1 \times 1 + 1 = 0$

于是有如图6-13所示的神经元结构。

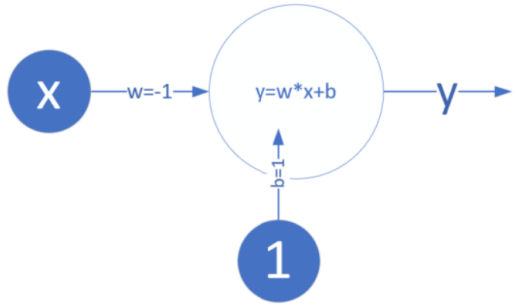


图6-13 不正确的逻辑非门的神经元实现

但是，这变成了一个拟合问题，而不是分类问题。比如，令 $x = 0.5$，带入公式中有：

$$y = wx + b = -1 \times 0.5 + 1 = 0.5$$

即，当 $x = 0.5$ 时，$y = 0.5$，且其结果 x 和 y 的值并没有丝毫“非”的意思。所以，应该定义如图6-14所示的神经元来解决问题，而其样本数据也很简单，如表6-6所示，一共只有两行数据。

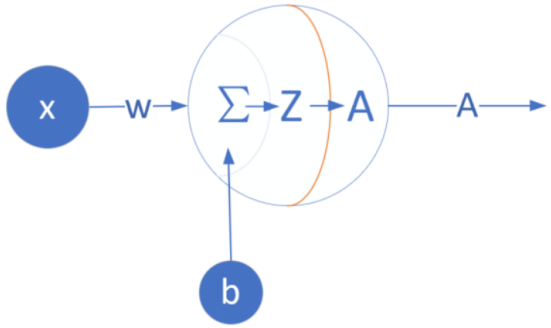
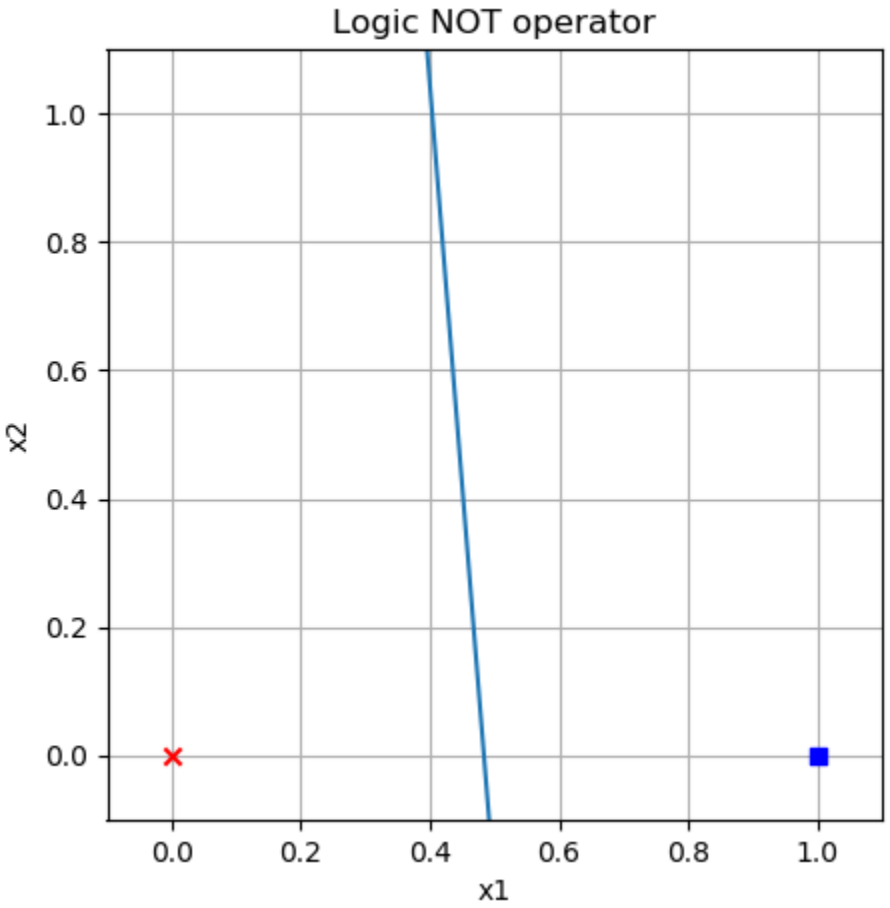


图6-14 正确的逻辑非门的神经元实现

表6-6 逻辑非问题的样本数据

样本序号	样本值x	标签值y
1	0	1
2	1	0



实现逻辑非门

实现逻辑非门

很多阅读材料上会这样介绍：有公式 $y = wx + b$，令 $w = -1, b = 1$，则：

- 当 $x = 0$ 时，$y = -1 \times 0 + 1 = 1$
- 当 $x = 1$ 时，$y = -1 \times 1 + 1 = 0$

于是有如图6-13所示的神经元结构。

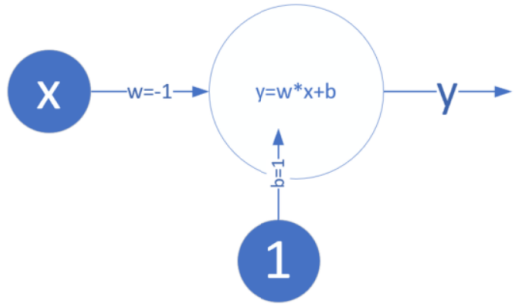


图6-13 不正确的逻辑非门的神经元实现

但是，这变成了一个拟合问题，而不是分类问题。比如，令 $x = 0.5$，带入公式中有：

$$y = wx + b = -1 \times 0.5 + 1 = 0.5$$

即，当 $x = 0.5$ 时，$y = 0.5$，且其结果 x 和 y 的值并没有丝毫“非”的意思。所以，应该定义如图6-14所示的神经元来解决问题，而其样本数据也很简单，如表6-6所示，一共只有两行数据。

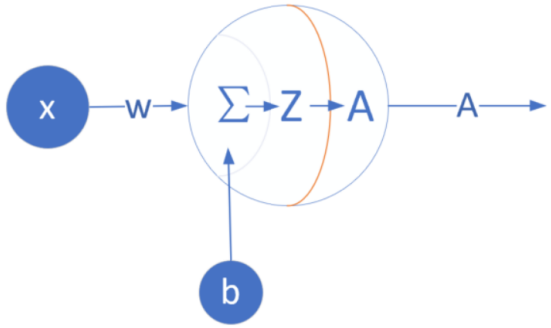
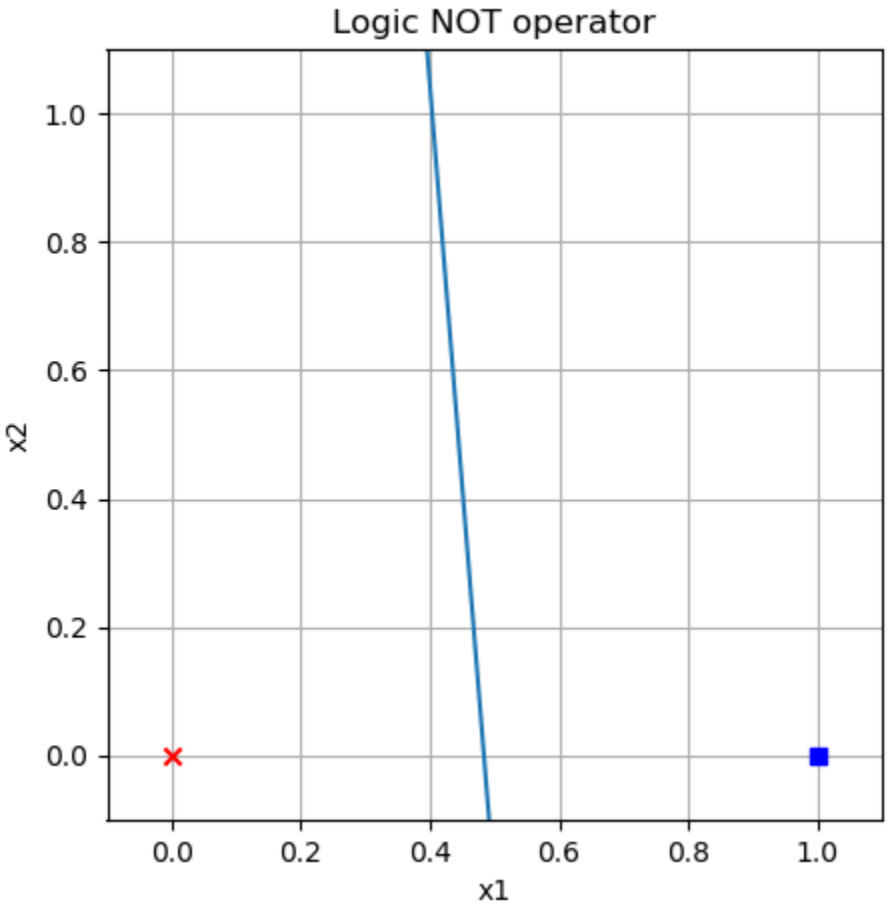


图6-14 正确的逻辑非门的神经元实现

表6-6 逻辑非问题的样本数据

样本序号	样本值x	标签值y
1	0	1
2	1	0



实现逻辑非门

实现逻辑与或门

神经元模型

依然使用之前的神经元模型，如图6-16。

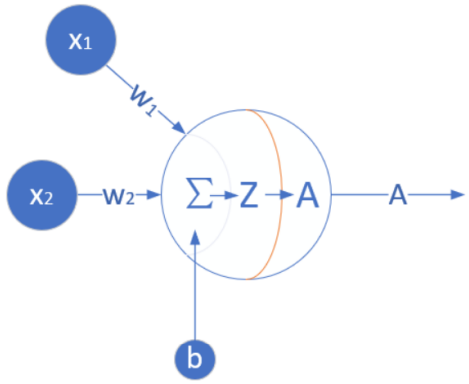


图6-16 逻辑与或门的神经元实现

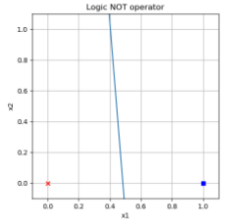
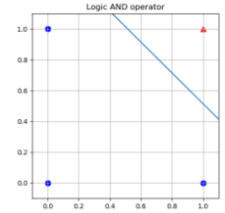
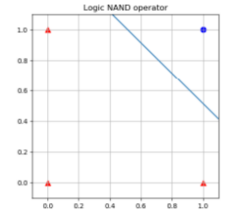
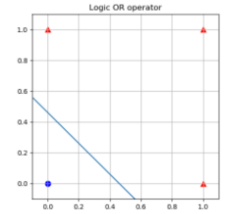
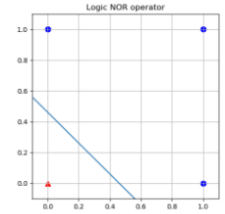
因为输入特征值只有两个，输出一个二分类，所以模型和前一节的一样。

训练样本

每个类型的逻辑门都只有4个训练样本，如表6-7所示。

表6-7 四种逻辑门的样本和标签数据

样本	x1	x2	逻辑与y	逻辑与非y	逻辑或y	逻辑或非y
1	0	0	0	1	0	1
2	0	1	0	1	1	0
3	1	0	0	1	1	0
4	1	1	1	0	1	0

逻辑门	分类结果	参数值
非		W=-12.468 B=6.031
与		W1=11.757 W2=11.757 B=-17.804
与非		W1=-11.763 W2=-11.763 B=17.812
或		W1=11.743 W2=11.743 B=-11.738
或非		W1=-11.738 W2=-11.738 B=5.409

四. 用双曲正切函数做二分类函数

提出问题

在二分类问题中，一般都使用对率函数（Logistic Function，常被称为Sigmoid Function）作为分类函数，并配合二分类交叉熵损失函数：

$$Logisitc(z_i) = \frac{1}{1+e^{-z_i}} \rightarrow a_i \tag{1}$$

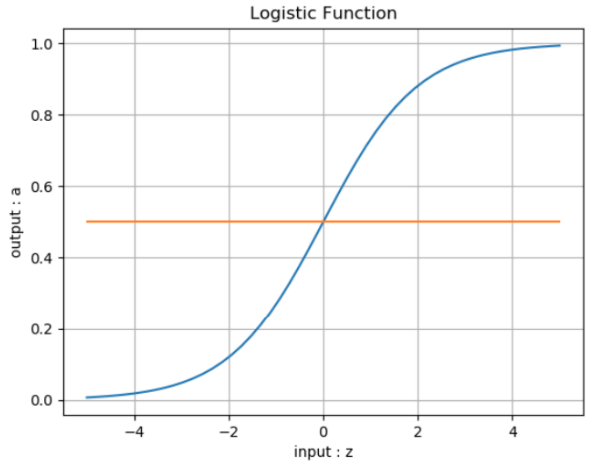
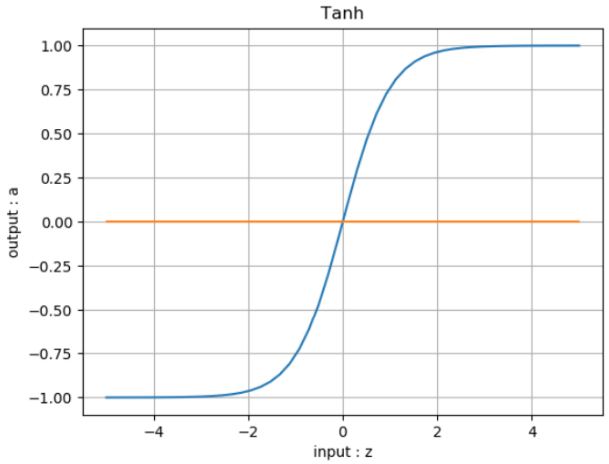
$$loss(w,b) = -[y_i \ln a_i + (1-y_i)\ln(1-a_i)] \tag{2}$$

还有一个与对率函数长得非常像的函数，即双曲正切函数（Tanh Function），公式如下：

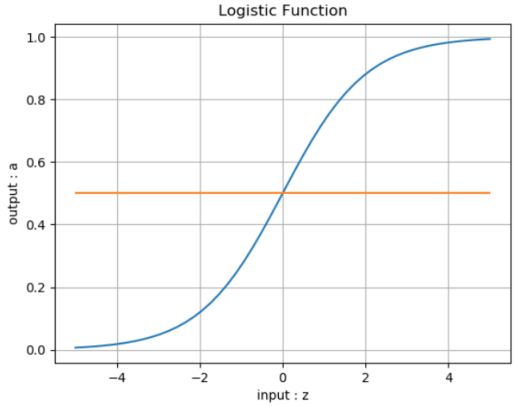
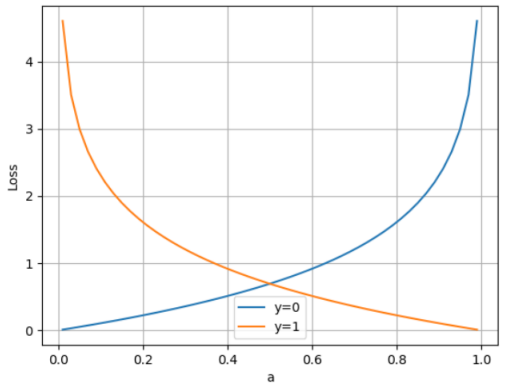
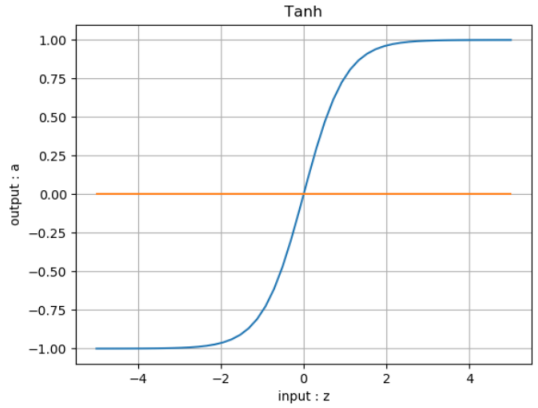
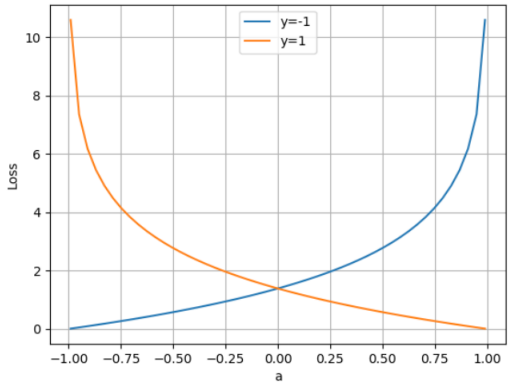
$$Tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} = \frac{2}{1+e^{-2z}} - 1 \tag{3}$$

提出问题：能不能用双曲正切函数作为分类函数呢?

分界线，其实只是人们理解神经网络做二分类的一种方式，对于神经网络来说，其实并没有分界线这个概念，它要做的是通过线性变换，尽量把正例向上方推，把负例向下方推

对率函数	双曲正切函数
	
正负类分界线：a=0.5	正负类分界线：a=0

对于对率函数来说，一般使用0.5作为正负类的分界线，那我们会自然地想到，对于双曲正切函数来说，可以使用0作为正负类的分界线。

分类函数	交叉熵函数
	
输出值域a在(0,1)之间，分界线为a=0.5，标签值为y=0/1	y=0为负例，y=1为正例，输入值域a在(0,1)之间，符合对率函数的输出值域
	
输出值域a在(-1,1)之间，分界线为a=0，标签值为y=-1/1	y=-1为负例，y=1为正例，输入值域a在(-1,1)之间，符合双曲正切函数的输出值域

五. 示例

Microsoft

Reactor

Thank You!