## Review

我们可以得出z的简易表达式$z = w \cdot x + b$，可得出

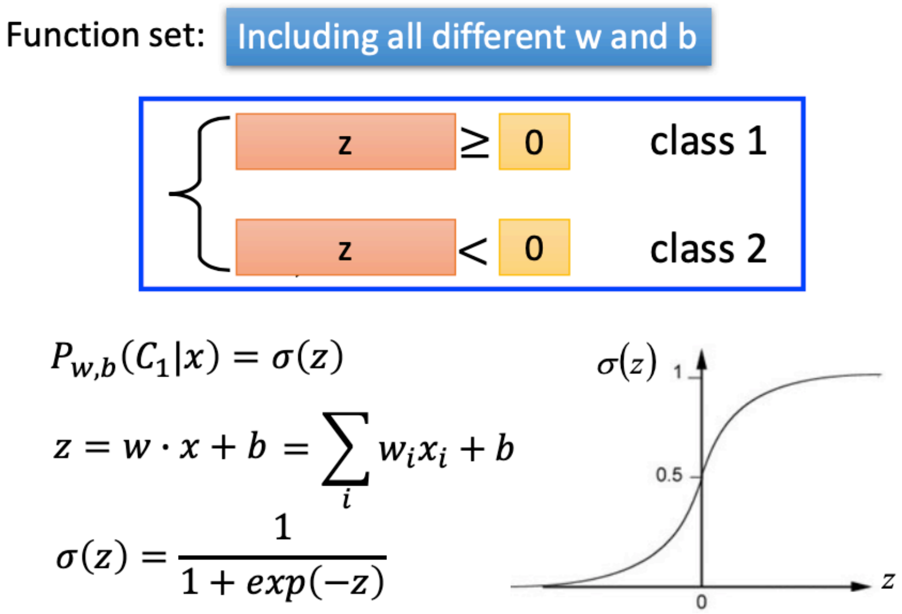
$$P(C_1|x) = \sigma(z) = \sigma(w \cdot x + b)$$

当得出$N_1, N_2, \mu^1, \mu^2, \sum$时，就可以计算出w和b的值。

## Three Steps

### Step1: Function Set

把所有的w和b都要包括进来，这里使用的function set就是sigmoid函数，




### Step2: Goodness of a Function

对于给定的一组w和b，得出似然函数L(w,b)的表达式，对于一个二分类问题，类别C1的概率为$f_{w,b}(x^i)$，$i = 1, 2, 4, \ldots N$，而类别C2的概率则为$1 - f_{w,b}(x^3)$。找出相对应的$w^*, b^*$，使得L取得最大值。

| Training Data | $x^1$ | $x^2$ | $x^3$ | …… | $x^N$ |
|---|---|---|---|---|---|
| | $C_1$ | $C_1$ | $C_2$ | | $C_1$ |

Assume the data is generated based on $f_{w,b}(x) = P_{w,b}(C_1|x)$

Given a set of w and b, what is its probability of generating the data?

$$L(w,b) = f_{w,b}(x^1)f_{w,b}(x^2)\left(1-f_{w,b}(x^3)\right)\cdots f_{w,b}(x^N)$$

The most likely w* and b* is the one with the largest $L(w,b)$.

$$w^*, b^* = arg\max_{w,b} L(w,b)$$

对于训练数据集，我们设C1的$\hat{y}=1$，C2的$\hat{y}=0$，服从Bernoulli distribution。在函数前面加-号就可以使原来的最大化函数，转化为对目标的最小化。

$$\boxed{w^*, b^* = arg\max_{w,b} L(w,b)} = \boxed{w^*, b^* = arg\min_{w,b} -lnL(w,b)}$$

$$
\begin{aligned}
&-lnL(w,b)\\
&= -lnf_{w,b}(x^1) \Rightarrow -\left[\boxed{1}\, lnf(x^1) \cancel{+ \boxed{0}\, ln\left(1-f(x^1)\right)}\right]\\
&\quad -lnf_{w,b}(x^2) \Rightarrow -\left[\boxed{1}\, lnf(x^2) \cancel{+ \boxed{0}\, ln\left(1-f(x^2)\right)}\right]\\
&-ln\left(1-f_{w,b}(x^3)\right) \Rightarrow -\left[\cancel{\boxed{0}\, lnf(x^3) +} \boxed{1}\, ln\left(1-f(x^3)\right)\right]\\
&\quad \vdots
\end{aligned}
$$

这时原来的似然函数L转化为了一个新形式，把原来的乘法变成了ln项相加，可以方便后边对w的求导

$$L(w,b) = f_{w,b}(x^1)f_{w,b}(x^2)\left(1-f_{w,b}(x^3)\right)\cdots f_{w,b}(x^N)$$

$$-lnL(w,b) = lnf_{w,b}(x^1) + lnf_{w,b}(x^2) + ln\left(1-f_{w,b}(x^3)\right)\cdots$$

$\hat{y}^n$: 1 for class 1, 0 for class 2

$$= \sum_n -\underbrace{\left[\hat{y}^n lnf_{w,b}(x^n) + (1-\hat{y}^n)ln\left(1-f_{w,b}(x^n)\right)\right]}_{\text{Cross entropy between two Bernoulli distribution}}$$

**Distribution p:**

$p(x=1) = \hat{y}^n$

$p(x=0) = 1-\hat{y}^n$

cross entropy

**Distribution q:**

$q(x=1) = f(x^n)$

$q(x=0) = 1-f(x^n)$

$$H(p,q) = -\sum_x p(x)ln\left(q(x)\right)$$

现在我们的目标就转化为了找出$w^*, b^* = argmin - lnL(w,b)$，交叉熵的形式为

$$-lnL(w,b) = \sum_n -[\hat{y}^n lnf_{w,b}(x^n) + (1-\hat{y}^n)ln(1-f_{w,b}(x^n))$$

**Step3: Find the best function**

为了找出那组使得$-lnL(w,b)$最小化的参数$w^*, b^*$，这里我们使用了Gradient Descent方法

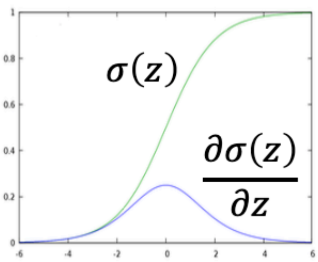
$$f_{w,b}(x) = \sigma(x) = \frac{1}{1+e^{-z}}, \quad z = w \cdot x + b = \sum_i w_i x_i + b$$

对wi求导，

$$\frac{-lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \overset{\left(1-f_{w,b}(x^n)\right)x_i^n}{\boxed{\frac{lnf_{w,b}(x^n)}{\partial w_i}}} + (1-\hat{y}^n)\frac{ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}\right]$$

$$\frac{\partial lnf_{w,b}(x)}{\partial w_i} = \frac{\partial lnf_{w,b}(x)}{\partial z}\frac{\partial z}{\partial w_i} \qquad \frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial ln\sigma(z)}{\partial z} = \frac{1}{\sigma(z)}\frac{\partial \sigma(z)}{\partial z} = \frac{1}{\cancel{\sigma(z)}}\cancel{\sigma(z)}\left(1-\sigma(z)\right)$$




$$\frac{-lnL(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \overset{\left(1-f_{w,b}(x^n)\right)x_i^n}{\boxed{\frac{lnf_{w,b}(x^n)}{\partial w_i}}} + (1-\hat{y}^n)\overset{-f_{w,b}(x^n)x_i^n}{\boxed{\frac{ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}}}\right]$$

$$\frac{\partial ln\left(1-f_{w,b}(x)\right)}{\partial w_i} = \frac{\partial ln\left(1-f_{w,b}(x)\right)}{\partial z}\frac{\partial z}{\partial w_i} \qquad \frac{\partial z}{\partial w_i} = x_i$$

$$\frac{\partial ln\left(1-\sigma(z)\right)}{\partial z} = -\frac{1}{1-\sigma(z)}\frac{\partial \sigma(z)}{\partial z} = -\frac{1}{\cancel{1-\sigma(z)}}\sigma(z)\cancel{\left(1-\sigma(z)\right)}$$

分别得出$\frac{\partial lnf_{w,b}(x)}{\partial w_i}$，$\frac{\partial ln(1-f_{w,b}(x))}{\partial w_i}$，代入原式子，化简可得

$$\frac{-\ln L(w,b)}{\partial w_i} = \sum_n -\left[\hat{y}^n \overset{\left(1-f_{w,b}(x^n)\right)x_i^n}{\boxed{\frac{\ln f_{w,b}(x^n)}{\partial w_i}}} + (1-\hat{y}^n) \overset{-f_{w,b}(x^n)x_i^n}{\boxed{\frac{\ln\left(1-f_{w,b}(x^n)\right)}{\partial w_i}}}\right]$$

$$= \sum_n -\left[\hat{y}^n\left(1-f_{w,b}(x^n)\right)x_i^n - (1-\hat{y}^n) f_{w,b}(x^n) x_i^n\right]$$

$$= \sum_n -\left[\hat{y}^n - \cancel{\hat{y}^n f_{w,b}(x^n)} - f_{w,b}(x^n) + \cancel{\hat{y}^n f_{w,b}(x^n)}\right] x_i^n$$

$$= \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$$

Larger difference, larger update

$$w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$$

得出梯度$\frac{\partial(-lnL(w,b))}{\partial w_i} = \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$，代入每次的梯度更新公式，

$$w_i \leftarrow w_i - \eta \frac{\partial(-lnL(w,b))}{\partial w_i} = w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$$

**Logistic Regression + Square error是否可行**

按照之前的步骤，先得出$f_{w,b}(x), L(f)$的表达式，第三步再求导，可以发现一个问题，代入训练数据集的$\hat{y}$后，梯度总是为0，模型最后无法训练，所以这样的结合是不可行的。

Step 1: $f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$

Step 2: Training data: $(x^n, \hat{y}^n)$, $\hat{y}^n$: 1 for class 1, 0 for class 2

$$L(f) = \frac{1}{2}\sum_n \left(f_{w,b}(x^n) - \hat{y}^n\right)^2$$

Step 3:

$$\frac{\partial\left(f_{w,b}(x) - \hat{y}\right)^2}{\partial w_i} = 2\left(f_{w,b}(x) - \hat{y}\right) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$

$$= 2\left(f_{w,b}(x) - \hat{y}\right) f_{w,b}(x)\left(1 - f_{w,b}(x)\right) x_i$$

$\hat{y}^n = 1$ If $f_{w,b}(x^n) = 1$ (close to target) $\Rightarrow$ $\partial L / \partial w_i = 0$

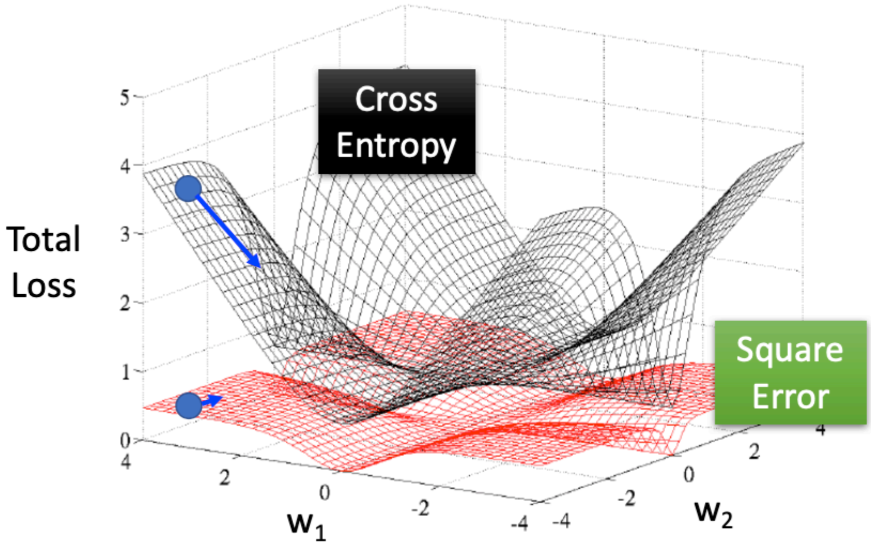
If $f_{w,b}(x^n) = 0$ (far from target) $\Rightarrow$ $\partial L / \partial w_i = 0$

$\hat{y}^n = 0$ If $f_{w,b}(x^n) = 1$ (far from target) $\Rightarrow$ $\partial L / \partial w_i = 0$

If $f_{w,b}(x^n) = 0$ (close to target) $\Rightarrow$ $\partial L / \partial w_i = 0$

**Cross Entropy v.s. Square Error**

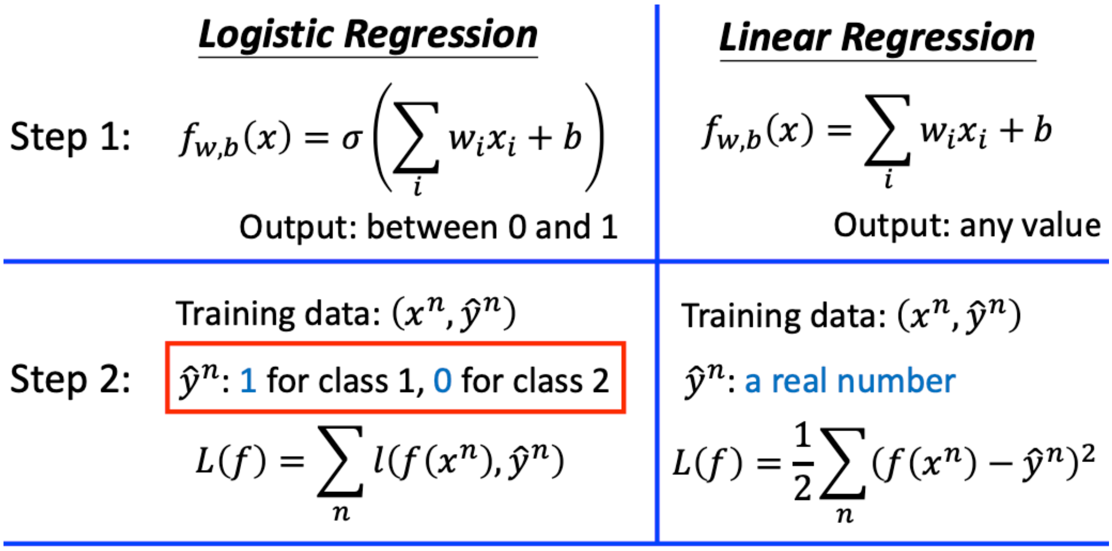
下图我们将Cross entropy和square error进行了对比，黑色网格线表示cross entropy，红色表示square error

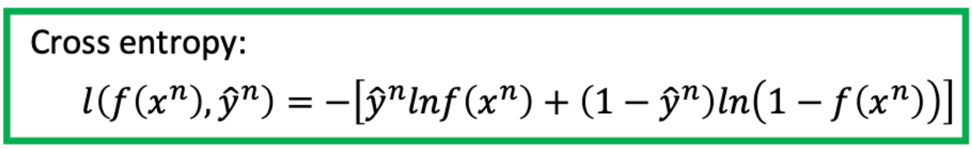



对于cross entropy，loss变化较大，曲线比较sharp，相应的微分也较大，每次跨越的步长也较长

对于square error，loss曲线变化比较平缓，微分值很小，每次跨越的步长也小，当gradient接近于0的时候，参数就很有可能不再更新，训练也会停下来。就算将gradient设置为很小的值，使训练不那么容易停下来，但由于每次跨越的步长很小很小，也会出现训练非常缓慢的问题

## Logistic vs Linear Regression



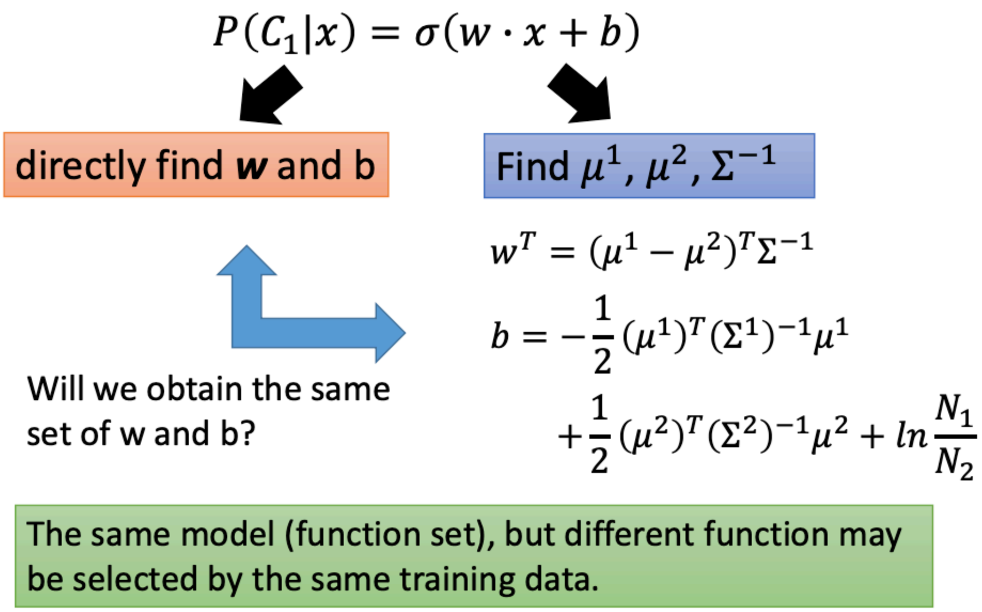
Step 3:

Logistic regression: $w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$

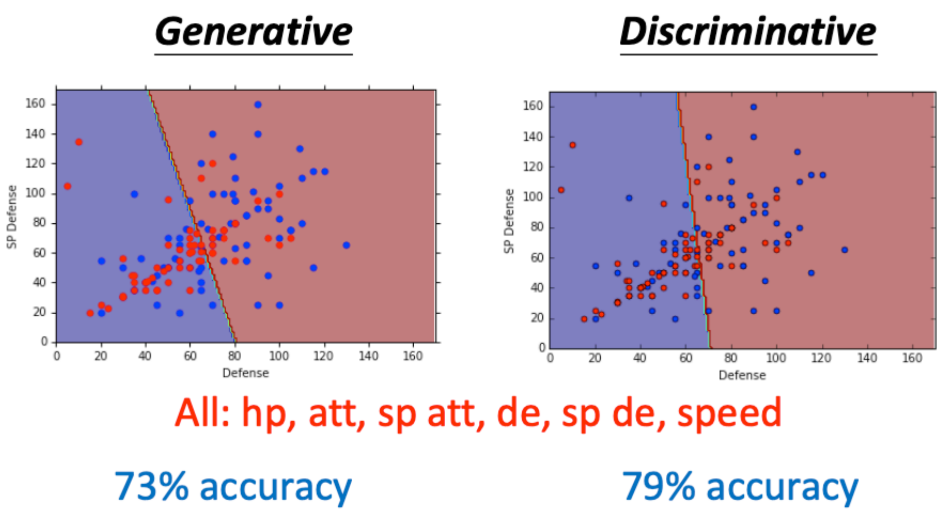
Linear regression: $w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$

### Discriminative v.s. Generative

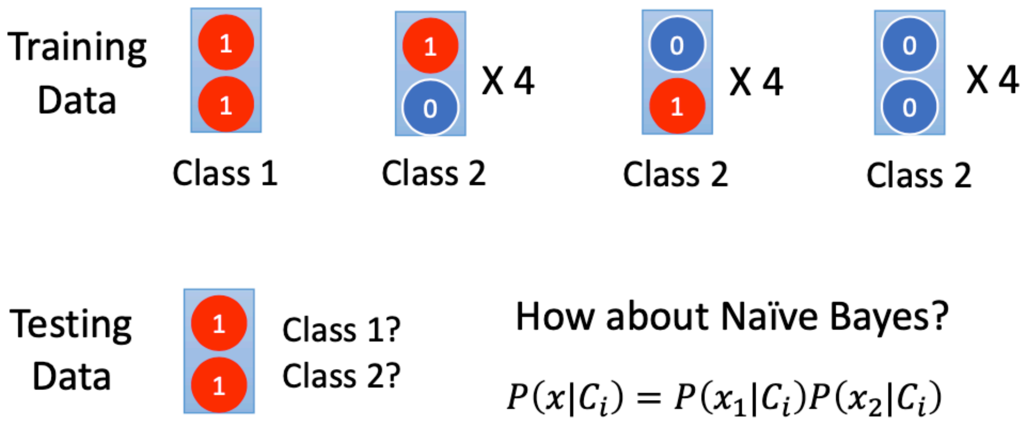
logistic regression我们称之为Discriminative方法；而我们将gaussian来描述posterior probability，称之为Generative方法。虽然都使用了相同的函数表达式，但需要找到的参数却是不同的。




logistic regression**没有实质性的假设**，要求直接找出对应的w和b。但generative model**做出了假设**，假设输入的数据是服从Gaussian分布的，需要先找出$\mu^1, \mu^2, \sum^{-1}$，再根据这些值得出相对应的w和b。

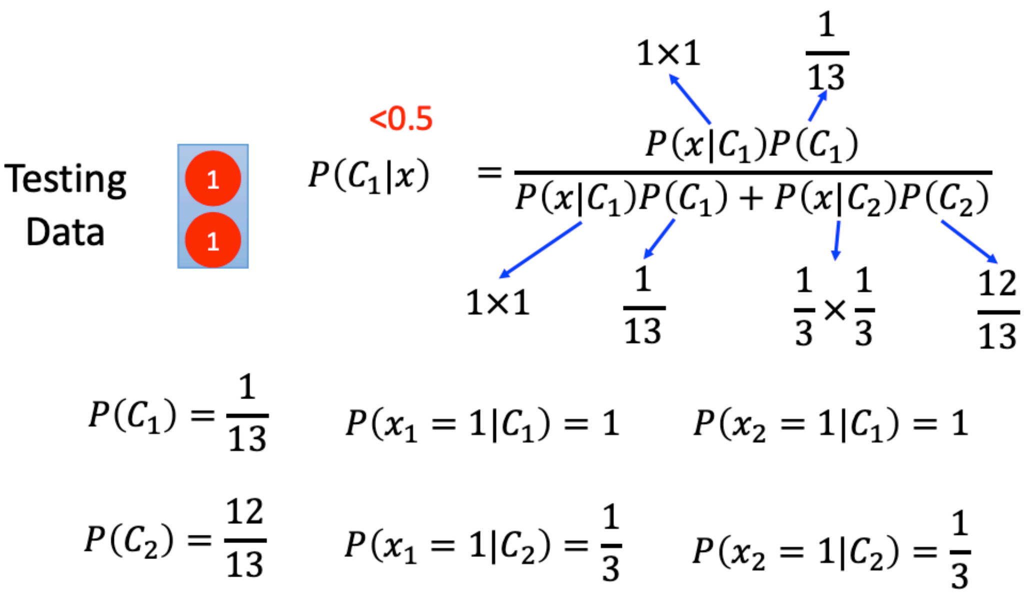



**Example**




对于包含13个example 的训练数据，对于图中所示的测试数据，我们可以明显看出测试example属于Class1，那么通过Naive Bayes（朴素贝叶斯）计算的结果也是这样吗？下面我们将开始验证，

$$P(x|C1) = P(x_1 = 1|C_1) \times P(x_2 = 1|C_1) = 1 \times 1$$
$$P(x|C2) = P(x_1 = 1|C_2) \times P(x_2 = 1|C_2) = \frac{1}{3} \times \frac{1}{3}$$




根据这个计算结果可知，属于Class1的概率是小于0.5的，因此可以看出根据朴素贝叶斯算法算出，测试的example是属于Class2，和我们的直觉是相反的。这是由于训练数据集中属于Class1的数量太少了，比例只有1/13。在实际生活中的模型训练中，我们也必须要避免数据集的差异对实验结果造成的影响，数据集中每个类别所占的比例应该是差别不大的。