逻辑回归第一步：函数设置

根据上一章可知我们需要确定样本的是它的几率

$$P_{w,b}(C_1|x),$$

$$\text{如果} P_{w,b}(C_1|x) \geqslant 0.5\text{，那么输出} C_1$$

$$\text{否则，输出} C_2$$

$$\text{假如我们使用的是高斯分布来确定几率的话，那么有}$$

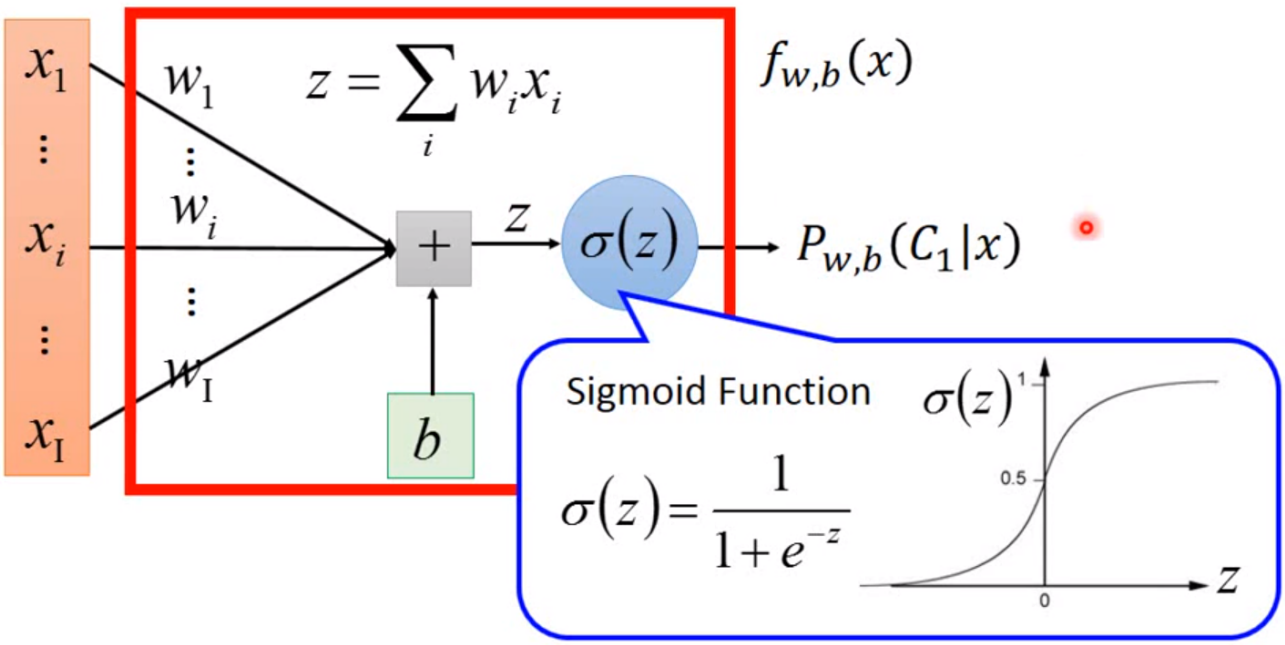
$$P_{w,b}(C_1|x) = \sigma(z)$$

$$\text{其中} \sigma(z) = \frac{1}{1+exp(-z)}$$

$$z = w*x+b = \sum w_i x_i + b \text{因此，我们的函数为}$$

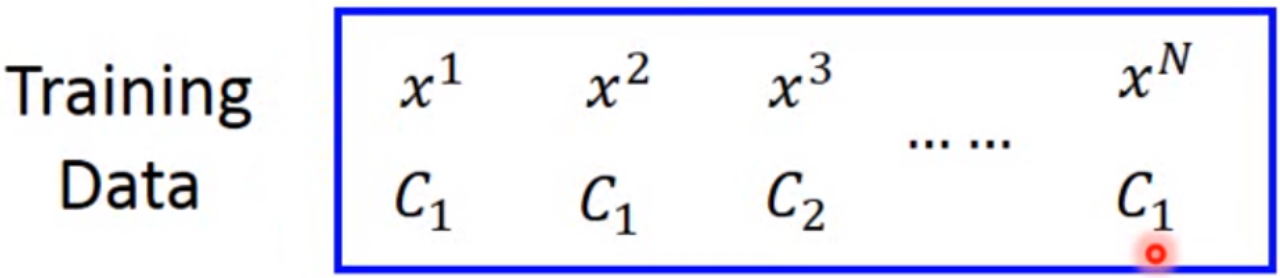
$$f_{w,b}(x) = P_{w,b}(C_1|x)$$

如果我们用图像来表示模型流程的话，如下所示



逻辑回归第二步：函数的优劣比较

在第一步已经确立模型之后，假设已经有了训练集



假设这组训练集的结果是由

$$f_{w,b} = P_{w,b}(C_1|x) \text{产生的，}$$

$$\text{那么给定一个} w,b \text{就决定了这个模型函数，而生成如上整组训练集的结果的概率为}$$

$$L(w,b) = f_{w,b}(x^1)f_{w,b}(x^2)(1-f_{w,b}(x^1))\dots f_{w,b}(x^N)$$

$$\text{而能够让} L(w,b) \text{最大化的} w,b \text{两个参数，我们取为} w^*,b^*$$

$$\text{则有} w^*,b^* = argmax_{w,b}L(w,b)$$

已知

$$L(w,b) = f_{w,b}(x^1)f_{w,b}(x^2)(1-f_{w,b}(x^1))\dots f_{w,b}(x^N)$$

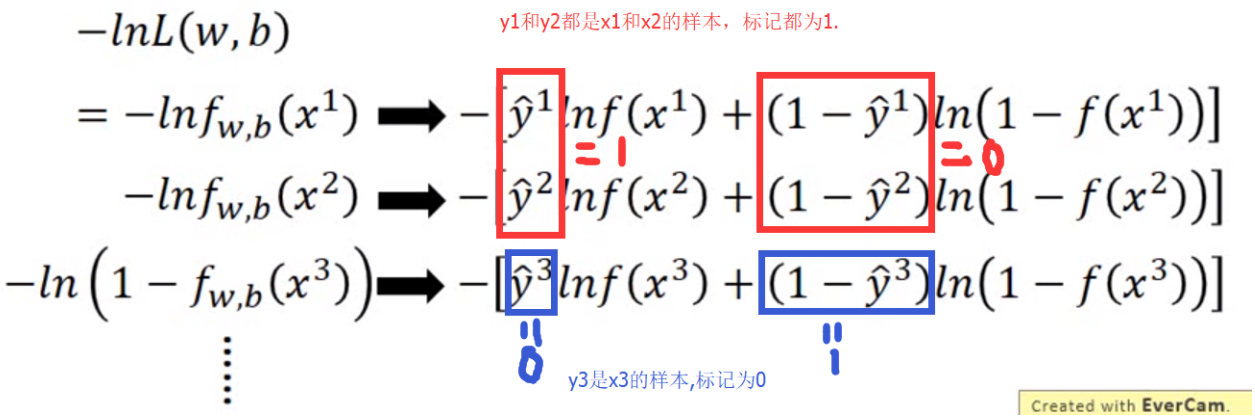
$$w^*,b^* = argmax_{w,b}L(w,b)$$

我们可以通过转换，将公式转化为

$$w^*,b^* = argmin_{w,b} -lnL(w,b)$$

因为函数取对数并不改变函数的性质。而转换后的函数更容易计算。

而假设C1类型标为1，C2类型标记为0.通过计算可以知道函数变化如下：



通过简化之后则有

$$-lnL(w,b) = lnf_{w,b}(x^1)lnf_{w,b}(x^2)ln(1-f_{w,b}(x^1))\dots$$
$$= \sum_n -[y^n lnf_{w,b}(x^n) + (1-y^n)ln(1-f_{w,b}(x^1))]$$

逻辑回归第三步：寻找最优函数

已知损失函数为：

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

那么可以使用梯度下降来进行求解，即对w,b进行偏微分梯度下降更迭。前面已知

$$f_{w,b}(x) = P_{w,b}(C_1|x) = \sigma(z) = \frac{1}{1+exp(-z)}$$
$$\text{其中 } z = w*x+b = \sum_i w_i x_i + b$$

因此我们对w，b求偏微分，可以等价视为

函数左边的偏微分：

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

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

函数右边的偏微分：

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

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

经过整理之后，我们得到的对w进行梯度下降的偏微分，结果如下：

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

$$\frac{\left(1-f_{w,b}(x^n)\right)x_i^n}{}\qquad\qquad \frac{f_{w,b}(x^n)x_i^n}{}$$

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

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

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

Larger difference, larger update

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

逻辑回归于线性回归的区别

	Logistic Regression	***Linear Regression***
Step 1:	$f_{w,b}(x) = \sigma\left(\sum_i w_i x_i + b\right)$ Output: between 0 and 1	$f_{w,b}(x) = \sum_i w_i x_i + b$ Output: any value
Step 2:	Training data: $(x^n, \hat{y}^n)$ $\hat{y}^n$: 1 for class 1, 0 for class 2 $L(f) = \sum_n C(f(x^n), \hat{y}^n)$	Training data: $(x^n, \hat{y}^n)$ $\hat{y}^n$: a real number $L(f) = \frac{1}{2}\sum_n (f(x^n) - \hat{y}^n)^2$
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: $w_i \leftarrow w_i - \eta \sum_n -\left(\hat{y}^n - f_{w,b}(x^n)\right) x_i^n$	

交叉熵VS平方差

交叉熵

1.信息量

交叉熵（cross entropy）是常用的一个概念，一般用来求目标与预测值之间的差距。 交叉熵是信息论中的一个概念，要想了解交叉熵的本质，需要先从最基本的概念讲起。

首先是信息量。假设我们听到了两件事，分别如下：

事件A：巴西队进入了2018世界杯决赛圈。

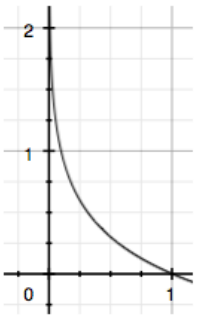
事件B：中国队进入了2018世界杯决赛圈。

仅凭直觉来说，显而易见事件B的信息量比事件A的信息量要大。究其原因，是因为事件A发生的概率很大，事件B发生的概率很小。所以当越不可能的事件发生了，我们获取到的信息量就越大。越可能发生的事件发生了，我们获取到的信息量就越小。

假设X是一个离散型随机变量，其取值集合为χ,概率分布函数$p(x) = Pr(X = x), x \in \chi$,则定义事件$X = x_0$的信息量为：

$$I(x_0) = -log(p(x_0))$$

由于是概率所以$p(x_0)$的取值范围是$[0, 1]$,绘制为图形如下：



可见该函数符合我们对信息量的直觉

2.熵

考虑另一个问题，对于某个事件，有n种可能性，每一种可能性都有一个概率$p(x_i)$
这样就可以计算出某一种可能性的信息量。举一个例子，假设你拿出了你的电脑，按下开关，会有三种可能性，下表列出了每一种可能的概率及其对应的信息量

序号	事件	概率p	信息量I
A	电脑正常开机	0.7	-log(p(A))=0.36
B	电脑无法开机	0.2	-log(p(B))=1.61
C	电脑爆炸了	0.1	-log(p(C))=2.30

> 注：文中的对数均为自然对数

我们现在有了信息量的定义，而熵用来表示所有信息量的期望，即：

$$H(X) = -\sum_{i=1}^{n} p(x_i)log(p(x_i))$$

其中n代表所有的n种可能性，所以上面的问题结果就是

$$\begin{aligned} H(X) &= -[p(A)log(p(A)) + p(B)log(p(B)) + p(C))log(p(C))] \\ &= 0.7 \times 0.36 + 0.2 \times 1.61 + 0.1 \times 2.30 \\ &= 0.804 \end{aligned}$$

3.相对熵

相对熵又称KL散度,如果我们对于同一个随机变量 x 有两个单独的概率分布 P(x) 和 Q(x)，我们可以使用 KL 散度（Kullback-Leibler (KL) divergence）来衡量这两个分布的差异 。

在机器学习中，P往往用来表示样本的真实分布，比如[1,0,0]表示当前样本属于第一类。Q用来表示模型所预测的分布，比如[0.7,0.2,0.1] 直观的理解就是如果用P来描述样本，那么就非常完美。而用Q来描述样本，虽然可以大致描述，但是不是那么的完美，信息量不足，需要额外的一些“信息增量”才能达到和P一样完美的描述。如果我们的Q通过反复训练，也能完美的描述样本，那么就不再需要额外的“信息增量”，Q等价于P。

KL散度的计算公式：

$$D_{KL}(p||q) = \sum_{i=1}^{n} p(x_i) log(\frac{p(x_i)}{q(x_i)}) \tag{3.1}$$

n为事件的所有可能性。
D_{KL}的值越小，表示q分布和p分布越接近

现在有两个分布，真实分布p和非真实分布q，我们的样本来自真实分布p。

按照真实分布p来编码样本所需的编码长度的期望为 $\sum_i p(i) * log\frac{1}{p(i)}$，这就是上面说的**信息熵H(p)**

按照不真实分布q来编码样本所需的编码长度的期望为 $\sum_i p(i) * log\frac{1}{q(i)}$，这就是所谓的**交叉熵H(p,q)**

这里引申出**KL散度D(p||q)** = H(p,q) - H(p) = $\sum_i p(i) * log\frac{p(i)}{q(i)}$，也叫做**相对熵**，它表示两个分布的差异，差异越大，相对熵越大。

机器学习中，我们用非真实分布q去预测真实分布p，因为真实分布p是固定的，D(p||q) = H(p,q) - H(p) 中 H(p) 固定，也就是说交叉熵H(p,q)越大，相对熵D(p||q)越大，两个分布的差异越大。

所以交叉熵用来做损失函数就是这个道理，它衡量了真实分布和预测分布的差异性。

4.为什么要用交叉熵做loss函数?

见逻辑回归使用平方差。

交叉熵可在神经网络(机器学习)中作为损失函数，p表示真实标记的分布，q则为训练后的模型的预测标记分布，交叉熵损失函数可以衡量p与q的相似性。交叉熵作为损失函数还有一个好处是使用sigmoid函数在梯度下降时能避免均方误差损失函数学习速率降低的问题，因为学习速率可以被输出的误差所控制。

逻辑回归使用平方差

假设我们使用平方差来计算损失函数并且进行梯度下降，将会出如下的情况

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:

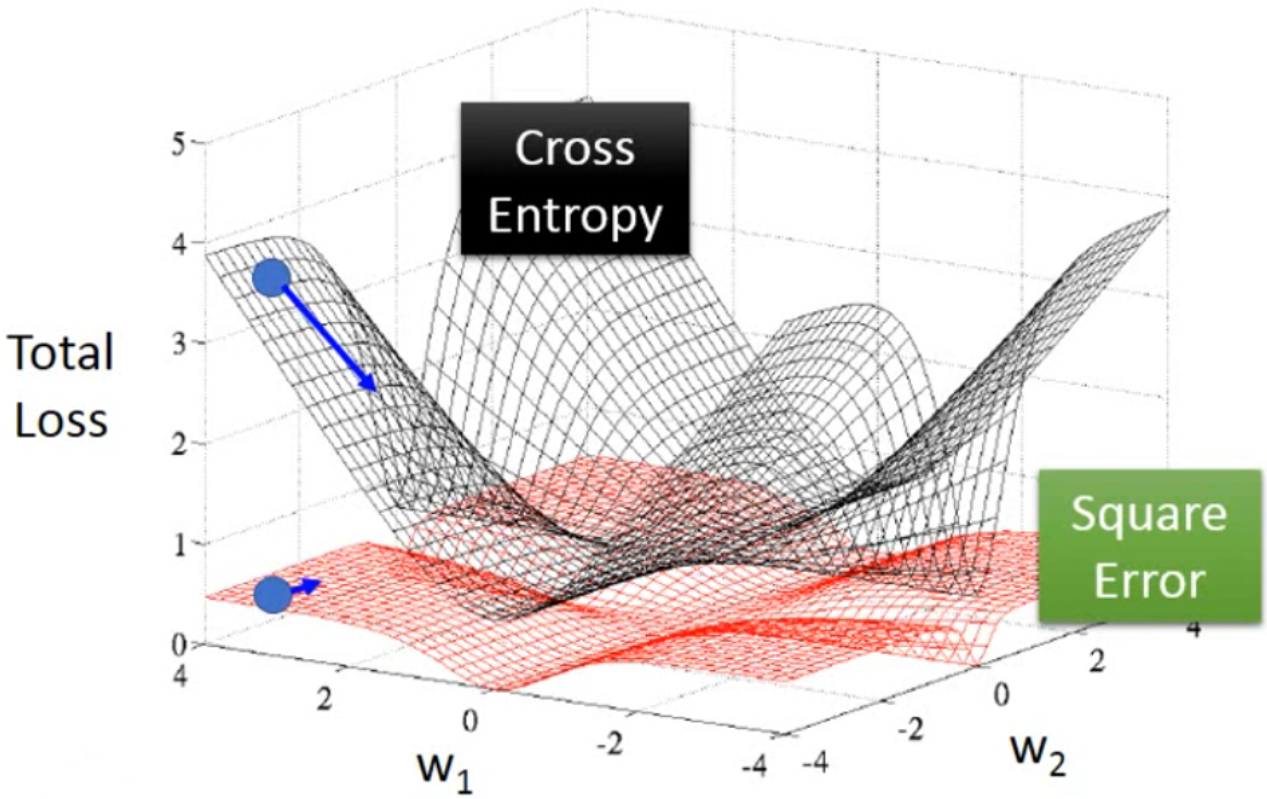
$$\frac{\partial\, (f_{w,b}(x)-\hat{y})^2}{\partial w_i} = 2\left(f_{w,b}(x) - \hat{y}\right) \frac{\partial f_{w,b}(x)}{\partial z} \frac{\partial z}{\partial w_i}$$

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

$\hat{y}^n = 0$ 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) $\Rightarrow \partial L/\partial w_i = 0$

即当使用平方差的时候，对于某一个样本的标签，无论预测结果是否准确，偏微分都将为0，对于距离很远的样本，这样的估计是不准确的。



如果使用交叉熵，距离越远的点，将会下降的更快，而距离越近的点，下降速度减缓。而如果使用的是平方差，距离远的点下降速度将会非常缓慢。

生成模型(Generative)和判别模型(Discriminative)

生成模型

生成模型估计的是**联合概率分布**（joint probability distribution），p(y, x)=p(y|x)*p(x)，由数据学习联合概率密度分布P(X,Y)，然后求出条件概率分布P(Y|X)作为预测的模型，即生成模型：P(Y|X)= P(X,Y)/ P(X)。基本思想是首先建立样本的联合概率概率密度模型P(X,Y)，然后再得到后验概率P(Y|X)，再利用它进行分类。生成方法关心的是给定输入x产生输出y的生成关系。

【生成模型Generative Model】——intra-class probabilistic description

又叫产生式模型。估计的是联合概率分布（joint probability distribution），p(class, context)=p(class|context)*p(context)。

用于随机生成的观察值建模，特别是在给定某些隐藏参数情况下。在机器学习中，或用于直接对数据建模（用概率密度函数对观察到的draw建模），或作为生成条件概率密度函数的中间步骤。通过使用贝叶斯rule可以从生成模型中得到条件分布。

如果观察到的数据是完全由生成模型所生成的，那么就可以fitting生成模型的参数，从而仅可能的增加数据相似度。但数据很少能由生成模型完全得到，所以比较准确的方式是直接对条件密度函数建模，即使用分类或回归分析。

与描述模型的不同是，描述模型中所有变量都是直接测量得到。

- **主要特点：**
  一般主要是对后验概率建模，从统计的角度表示数据的分布情况，能够反映同类数据本身的相似度。
  只关注自己的inclass本身（即点左下角区域内的概率），不关心到底 decision boundary在哪。
- **优点：**
  实际上带的信息要比判别模型丰富，
  研究单类问题比判别模型灵活性强
  模型可以通过增量学习得到
  能用于数据不完整（missing data）情况
  modular construction of composed solutions to complex problems
  prior knowledge can be easily taken into account
  robust to partial occlusion and viewpoint changes
  can tolerate significant intra-class variation of object appearance
- **缺点：**
  tend to produce a significant number of false positives. This is particularly true for object classes which share a high visual similarity such as horses and cows
  学习和计算过程比较复杂

判别模型

判别模型估计的是**条件概率分布**(conditional distribution)， p(y|x)，是给定观测变量x和目标变量y的条件模型。由数据直接学习决策函数y=f(X)或者条件概率分布P(y|x)作为预测的模型。判别方法关心的是对于给定的输入X，应该预测什么样的输出Y。

例如：比如说要确定一只羊是山羊还是绵羊，**用判别模型**的方法是先从历史数据中学习到模型，然后通过提取这只羊的特征x来预测出这只羊f(X)是山羊的概率，是绵羊的概率。**用生成模型**的方法是我们可以根据山羊的特征首先学习出一个山羊模型，然后根据绵羊的特征学习出一个绵羊模型。然后从这只羊中提取特征，放到山羊模型P(w1|X)中看概率是多少，再放到绵羊模型P(w2|X)中看概率是多少，如果P(w1|X)>P(w2|X)，那么我们就认为X是属于w1类，即该羊属于山羊。

再例如：比如说你的任务是识别一个语音属于哪种语言。例如对面一个人走过来，和你说了一句话，你需要识别出她说的到底是汉语、英语还是法语等。那么你可以有两种方法达到这个目的：**用生成模型**的方法是学习每一种语言，你花了大量精力把汉语、英语和法语等都学会了，我指的学会是你知道什么样的语音对应什么样的语言。然后再有人过来对你说话，你就可以知道他的语言对应什么语言；**用判别模型**的方法是不去学习每一种语言，你只学习这些语言模型之间的差别，然后再分类。意思是指我学会了汉语和英语等语言的发音是有差别的，我学会这种差别就好了。

【判别模型Discriminative Model】——inter-class probabilistic description

又可以称为条件模型，或条件概率模型。估计的是条件概率分布(conditional distribution)，p(class|context)。

利用正负例和分类标签，focus在判别模型的边缘分布。目标函数直接对应于分类准确率。

- 主要特点:

寻找不同类别之间的最优分类面，反映的是异类数据之间的差异。

- 优点:

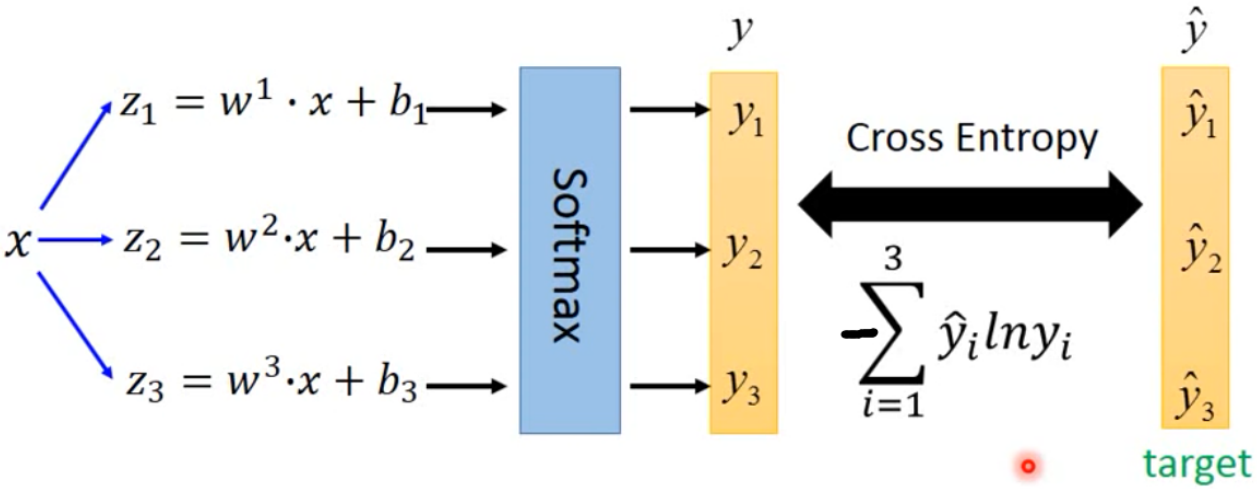
分类边界更灵活，比使用纯概率方法或生产模型得到的更高级。
能清晰的分辨出多类或某一类与其他类之间的差异特征
在聚类、viewpoint changes, partial occlusion and scale variations中的效果较好
适用于较多类别的识别
判别模型的性能比生成模型要简单，比较容易学习

- 缺点:

不能反映训练数据本身的特性。能力有限，可以告诉你的是1还是2，但没有办法把整个场景描述出来。
Lack elegance of generative: Priors, 结构, 不确定性
Alternative notions of penalty functions, regularization, 核函数
黑盒操作：变量间的关系不清楚，不可视

	判别式模型（discriminative model）	产生式模型（generative model）
特点	寻找不同类别之间的最优分类面，反映的是异类数据之间的差异	对后验概率建模，从统计的角度表示数据的分布情况，能够反映同类数据本身的相似度
区别(假定输入x, 类别标签y)	估计的是条件概率分布(conditional distribution)：P(y|x)	估计的是联合概率分布（joint probability distribution: P(x, y),
联系	由产生式模型可以得到判别式模型，但由判别式模型得不到产生式模型。	
常见模型	– logistic regression – SVMs – traditional neural networks – Nearest neighbor	–Gaussians, Naive Bayes –Mixtures of Gaussians, Mixtures of experts, HMMs –Sigmoidal belief networks, Bayesian networks – Markov random fields

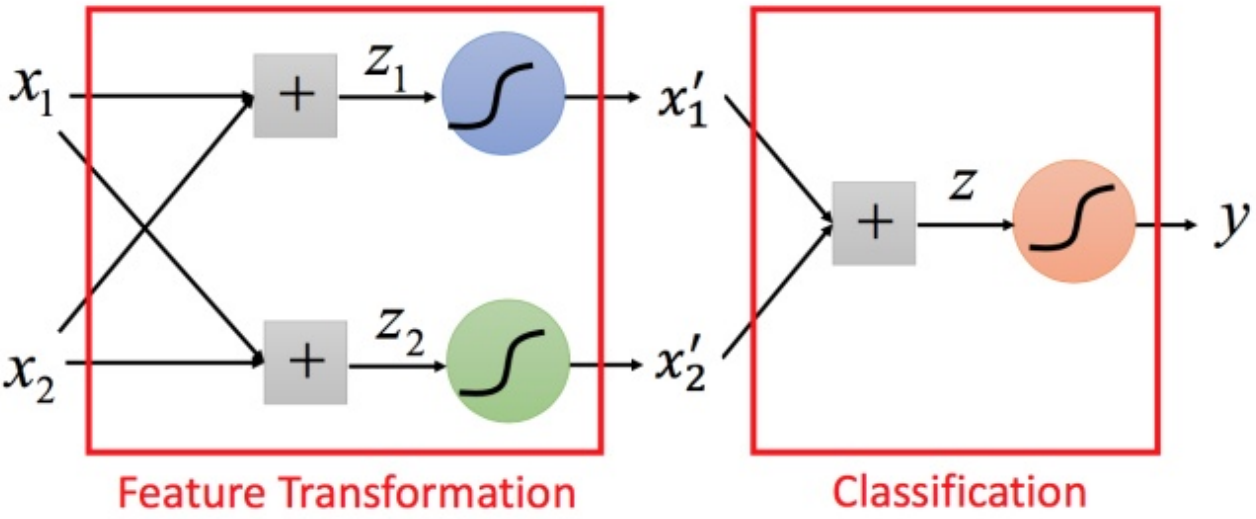
多分类模型（3类为例）



对于多分类模型，按照上述的流程，对于一个测试样本x进行z1,z2,z3三个模型函数的计算，并且通过softmax方法，将计算结果转化成y1,y2,y3，并与测试样本中原有的标签进行交叉熵计算。最后根据交叉熵所计算出的结果，来判断x的分类。

级联逻辑回归模型

当存在样本没办法通过一根直线直接将样本进行逻辑分类，那么这个时候可以通过级联逻辑回归模型，将样本转换之后再进行分类。



一个逻辑回归的输入可以来源于其他逻辑回归的输出，这个逻辑回归的输出也可以是其他逻辑回归的输入。把每个逻辑回归称为一个 **Neuron（神经元）**，把这些神经元连接起来的网络，就叫做 **Neural Network（神经网络）**。