人工智能中的数学

一、函数

一次函数：

一个自变量的情形：

$$y = ax + b$$

多个自变量的情形：

$$y = ax_1 + bx_2 + c(a、b、c是常数)$$

神经网络中的一次函数：

神经网络中，神经单元的加权输入可以表示为一次函数关系。例如，神经单元有三个来自下层的输入，其加权输入 z 的式子如下所示：

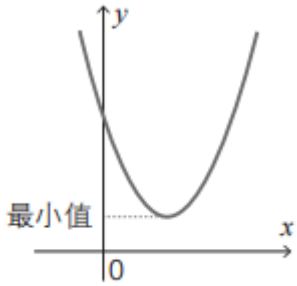
$$z = w_1x_1 + w_2x_2 + w_3x_3 + b$$

- 如果把作为参数的权重 w_1, w_2, w_3 与偏置 b 看作常数，那么加权输入 z 和 x_1, x_2, x_3 是一次函数关系。
- 另外，在神经单元的输入 x_1, x_2, x_3 作为数据值确定了的情况下，加权输入 z 和权重 w_1, w_2, w_3 以及偏置 b 是一次函数关系。
- 用误差反向传播法推导计算式时，这些一次函数关系使得计算可以简单地进行。

二次函数：

机器学习和深度学习中的代价函数使用了二次函数。

二次函数由下式表示： $y = ax^2 + bx + c$

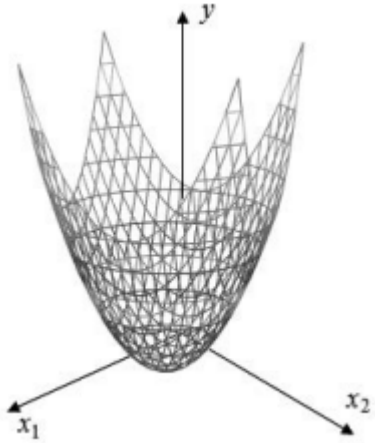


这个图像中重要的一点是，a 为正数时图像向下凸，从而存在最小值。

下凸的函数有最小值，这个性质是机器学习中最小二乘法（损失函数的平方误差总和）的基础

多个自变量的二次函数：

$y = ax_1^2 + bx_1x_2 + cx_2^2 + px_1 + qx_2 + r$

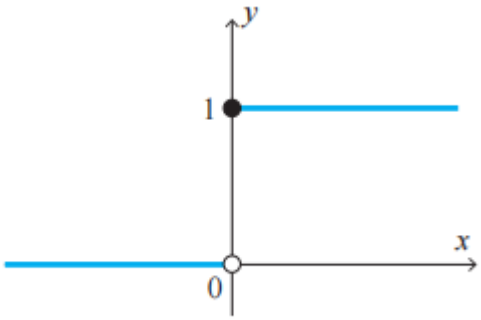


神经网络中要处理的二次函数

单位阶跃函数：

神经网络的原型模型是用单位阶跃函数作为激活函数

$f(x) = \begin{cases} 0, & (\mathrm{x} < 0) \\ 1, & (\mathrm{x} \geq 0) \end{cases}$ 此函数在原点不可导，所以不能作为主要的激活函数



单位阶跃函数的图像。在应用数学的世界里，这个函数活跃于线性响应理论之中。

指数函数与Sigmoid函数：

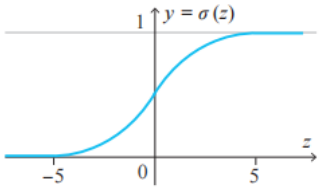
指数函数： $y = a^x, (a > 0, \mathrm{a} \neq 1)$

重要的底：自然数，欧拉数，纳皮尔数 $e = 2.71828\cdots$

Sigmoid函数：

$\sigma(x) = \frac{1}{1+e^{-x}}$　$\sigma(x)$是神经网络中非常具有代表性的激活函数之一

这个函数的图像如右图所示。可以看出，这个函数是光滑的，也就是处处可导。函数的取值在 0 和 1 之间，因此函数值可以用概率来解释。



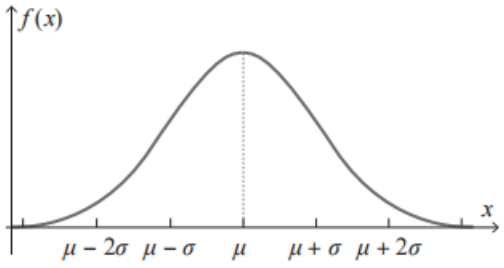
Sigmoid 函数的图像。

正态分布的概率密度函数

在用计算机实际确定神经网络时，必须设定权重和偏置的初始值。设定初始值时，正态分布（normal distribution）是一个有用的工具。使用服从这个分布的随机数，容易取得好的结果。

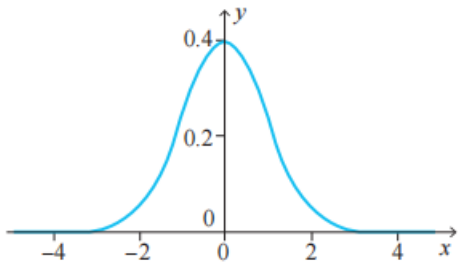
正态分布是服从以下概率密度函数 *f*(*x*) 的概率分布

$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu)^2}{2\sigma^2}}$　其中σ是标准差，μ 是期望值（平均值）



期望值为 μ，标准差为 σ 的正态分布。另外，这个 σ 与 Sigmoid 函数名 σ 的含义不同。

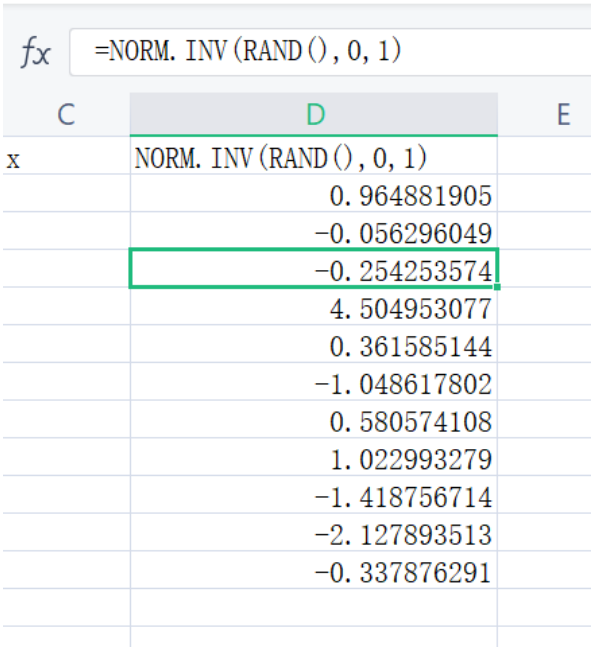
如下图所示，这个正态分布称为**标准正态分布**。



$\mu = 0$，$\sigma = 1$ 的正态分布概率密度函数的图像。

在 Excel 中，可以像下面这样产生正态分布随机数。

$= NORM.INV(RAND(), \mu, \sigma)$　　（μ、σ是期望值和标准差）



方差或者说标准差越大，离散程度越大。

如下，可点击跳转阅读相关概念：

关于概率密度函数

关于正态分布

关于期望值和均值

关于方差和标准差

二、数列

理解了数列的递推公式，就很容易理解：求解预估值的正向传播（链式求值）、误差的反向传播（链式求导）

注意：计算机更擅长用递推公式进行运算，联想下程序设计中的递归

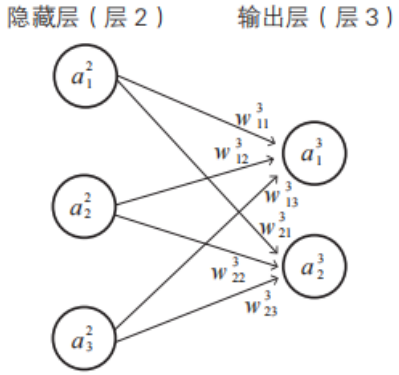
数列的通项公式和递推公式

- 等差数列通项公式：$a_n = a_1 + (n-1)d$
- 等比数列通项公式：$a_n = a_1 q^{n-1}$
- 等差数列递推公式：$a_{n+1} = a_n + d$
- 等比数列递推公式：$a_{n+1} = a_n q$
- 神经网络中，每一个神经单元可以理解为数列中的某一项

联立的递推公式

$$\begin{cases} a_{n+1} = a_n + 2b_n + 2 \\ b_{n+1} = 2a_n + 3b_n + 3 \end{cases}$$

如上，两个数列的联立递推公式，尝试理解如下神经网络中每个神经单元的值的计算



注：其中，a_n^m, b_n^m ：第m层第n个神经单元的输出值和偏置（激活神经单元的阈值）；

注：其中，w_{ij}^k： 第 k 层 第 i 个 神经单元，接收上层 第 j 个结点数据的权重;

$$\begin{cases} a_1^3 = \alpha(w_{11}^3 a_1^2 + w_{12}^3 a_2^2 + w_{13}^3 a_3^2 + b_1^3) \\ a_2^3 = \alpha(w_{21}^3 a_1^2 + w_{22}^3 a_2^2 + w_{23}^3 a_3^2 + b_2^3) \end{cases}$$

根据这个递推公式，第 3 层的输出 a_1^3和a_2^3由第 2 层的输出 a_1^2、 a_2^2 、a_3^2决定。

也就是说，第 2 层的输出与第 3 层的输出由联立递推公式联系起来。

三、∑符号的理解和使用

∑是一个需要下功夫来熟悉的符号。如果不理解∑，在阅读神经网络相关的文献时就比较麻烦。

这是因为将加权输入用∑符号来表示会简洁得多。

常见的希腊字母：

\sigma	Σ大写 σ小写
\alpha	A大写 α小写
\beta	B大写 β小写
\gamma	Γ大写 γ小写
\delta	Δ大写 δ小写
\epsilon	E大写 ϵ小写
\zeta	Z大写 ζ小写
\eta	H大写 η小写
\theta	Θ大写 θ小写
\lambda	Λ大写 λ小写

∑符号含义：表示数列的总和

$$\sum_{i=1}^{n} a_i = a_1 + a_2 + \cdots + a_n$$

∑符号性质：和的∑为∑的和，常数倍的∑为∑的常数倍

$$\sum_{i=1}^{n}(a_i + b_i) = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_i$$

$$\sum_{i=1}^{n} ca_i = c\sum_{i=1}^{n} a_i$$

在机器学习和深度学习的文献中，把加权输入用∑符号不是非常简洁。

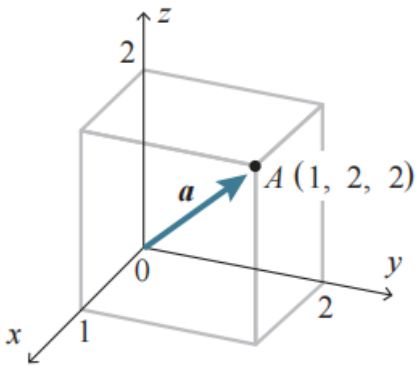
四、向量(Vector)

定义：

- 有向线段的属性：起点 *A* 的位置、指向 *B* 的方向，以及*AB* 的长度，也就是大小
- 把方向与大小抽象出来，这样的量叫作**向量**
- 向量是具有方向与大小的量，用箭头表示。
- 有向线段 *AB* 所代表的向量用 $\overrightarrow{AB}$ 表示，也可以用带箭头的单个字母$\vec{a}$表示。

向量的坐标表示法

- 把向量的箭头放在坐标系中，就可以用坐标的形式表示向量。
- 把箭头的起点放在原点，用箭头终点的坐标表示向量。
- 用坐标表示的向量 $\vec{a}$，$\vec{a} = (a_1, a_2)$
- 三维空间同理：$\vec{a} = (a_1, a_2, a_3)$ ，$\vec{a} = (1, 2, 2)$，如图



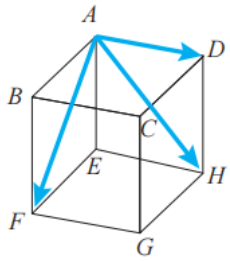
向量的大小：｜a｜

表示向量的箭头的**长度**称为这个向量的大小。向量 $\vec{a}$ 的大小用｜**a**｜表示。

向量的内积：$\vec{a}\cdot\vec{b}=|\vec{a}|\left|\vec{b}\right|\cos(\theta)$

在边长为 3 的立方体 $ABCD-EFGH$ 中，有

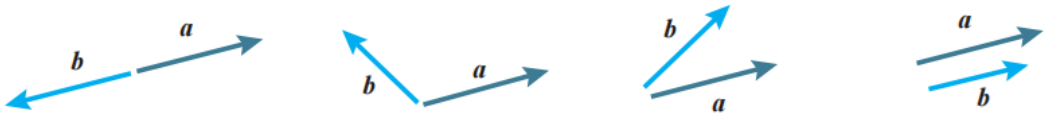
$$\overrightarrow{AD}\cdot\overrightarrow{AD}=|\overrightarrow{AD}||\overrightarrow{AD}|\cos 0^\circ=3\cdot 3\cdot 1=9$$
$$\overrightarrow{AD}\cdot\overrightarrow{AF}=|\overrightarrow{AD}||\overrightarrow{AF}|\cos 90^\circ=3\cdot 3\sqrt{2}\cdot 0=0$$
$$\overrightarrow{AF}\cdot\overrightarrow{AH}=|\overrightarrow{AF}||\overrightarrow{AH}|\cos 60^\circ=3\sqrt{2}\cdot 3\sqrt{2}\cdot\frac{1}{2}=9$$



柯西 - 施瓦茨不等式：$-|a||b|\le a\cdot b\le|a||b|$

根据柯西 - 施瓦茨不等式，可以得出以下事实。

- 当两个向量方向相反时，内积取得最小值。**梯度下降法的基本原理**
- 当两个向量不平行时，内积取平行时的中间值。**越大越相似，考察卷积神经网络**
- 当两个向量方向相同时，内积取得最大值



通过内积可以知道两个向量的相对的相似度。

内积的坐标表示

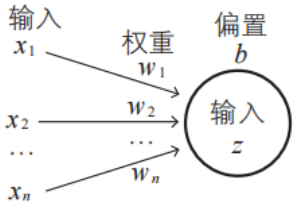
当$\vec{a}=(a1,a2)$，$\vec{b}=(b1,b2)$时，

$$\vec{a}\cdot\vec{b}=a_1b_1+a_2b_2$$

同理，三维空间情况下：当$\vec{a}=(a1,a2,a3)$，$\vec{b}=(b1,b2,b3)$时，

$$\vec{a}\cdot\vec{b}=a_1b_1+a_2b_2+a_3b_3$$

向量的一般化

- 向量的方便之处在于，二维以及三维空间中的性质可以照搬到任意维空间中；
- 神经网络虽然要处理数万维的空间，但是二维以及三维空间的向量性质可以直接利用。基于此，向量被充分应用在梯度下降法中；
- 二维以及三维空间中的向量公式推广到任意的 *n* 维空间：
  - 向量的坐标表示法：$\vec{a}=(a_1,a_2,\cdots,a_n)$
  - 内积的坐标表示：$\vec{a}\cdot\vec{b}=a_1b_1+a_2b_2+\cdots+a_nb_n$
- 在神经网络中使用内积形式：
  - 

  - 如图有两个向量：$\vec{x}=(x_1,x_2,\cdots,x_n)$ 和 $\vec{w}=(w_1,w_2,\cdots,w_n)$
  - 加权输入可表示为向量内积形式：$z=\vec{x}\cdot\vec{w}+b$

五、矩阵(Matrix)

什么是矩阵

- 矩阵就是数的阵列，如下：横排为行，竖排为列，构成一个3行3列的矩阵
- 行数和列数相同，称为**方阵**；
- (1,2,3)、(4,5,6)、(7,8,9)，称为**行向量**；(1,4,7)、(2,5,8)、(3,6,9)称为**列向量**；
- 行向量、列向量，也统称**向量**

$$\begin{Bmatrix}1&2&3\\4&5&6\\7&8&9\end{Bmatrix}$$

- 扩展为一般形式：m行n列的矩阵

$$\begin{Bmatrix}a_{11}&a_{12}&\cdots&a_{1n}\\a_{21}&a_{22}&\cdots&a_{2n}\\\vdots&\vdots&\ddots&\vdots\\a_{m1}&a_{m2}&\cdots&a_{mn}\end{Bmatrix}\quad(\mathrm{m,n的矩阵})$$

  位于第 i 行第 j 列的值（称为元素），用a_{ij}表示

- 很有名的矩阵（**单位矩阵**）：对角线的元素(a_{ii})为1，其他元素为0的**方阵**，用 E 或 I 表示

$$E=\begin{Bmatrix}1&0&0\\0&1&0\\0&0&1\end{Bmatrix}\quad(\text{注：E 为德语中表示 1 的单词 Ein 的首字母})$$

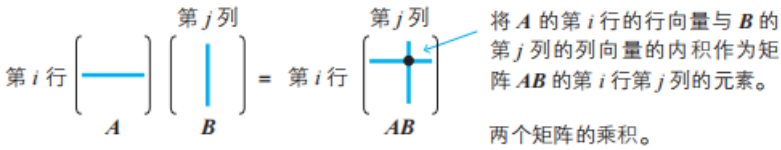
矩阵相等

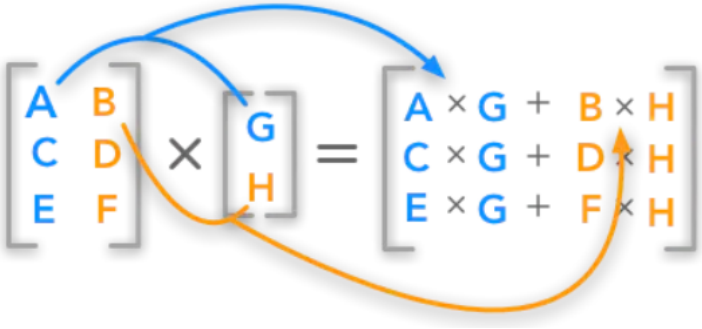
两个矩阵 **A**、**B** 相等的含义是它们对应的元素相等，记为 **A** = **B**

矩阵的和、差、常数倍

两个矩阵 **A**、**B** 的和 **A** + **B**、差 **A** - **B** 定义为相同位置的元素的和、差所产生的矩阵。此外，矩阵的常数倍定义为各个元素的常数倍所产生的矩阵。

矩阵的乘积

- 矩阵的乘积在神经网络的应用中特别重要。对于两个矩阵 **A**、**B**，将**A** 的第 *i* 行看作行向量，**B** 的第 *j* 列看作列向量，将它们的**内积**作为第 *i*行第 *j* 列元素，由此而产生的矩阵就是矩阵 **A**、**B** 的乘积 $A \cdot B$。
- 

- 矩阵的乘法不满足交换律。也就是说，除了例外情况，以下关系式成立：$A \cdot B \neq B \cdot A$
- 不是任何两个矩阵都能够相乘，**只有乘数矩阵A的列数和被乘矩阵B的行数相同的时候，两个矩阵才能相乘。**
- 维度为（m x n）的矩阵 A 和维度为（n x p）的矩阵 B 相乘，最终得到维度为（m x p）的矩阵 C。
- 

Hadamard 乘积

对于相同形状的矩阵 **A**、**B**，将相同位置的元素相乘，由此产生的矩阵称为矩阵 **A**、**B** 的 **Hadamard** 乘积，用 **A**⊙**B** 表示。

转置矩阵

- 将矩阵 **A** 的第 *i* 行第 *j* 列的元素与第 *j* 行第 *i* 列的元素交换，由此产生的矩阵称为矩阵 **A** 的转置矩阵（Transposed Matrix），用 t^A 或 A^T 等表示。
- 可以理解为把行向量转换成列向量（反之亦然，把列向量转换成行向量）

逆矩阵：

方阵 A 的逆矩阵记作A^{-1}，其满足：$A^{-1}A=E$

- **逆矩阵**在深度学习中的应用：
  - 牛顿法优化神经网络。
  - 在深度学习中，经常需要求逆矩阵，但是由于求逆矩阵的计算开销巨大，因此通常会将矩阵转换成其他特殊矩阵的形式以避免或简化矩阵求逆。

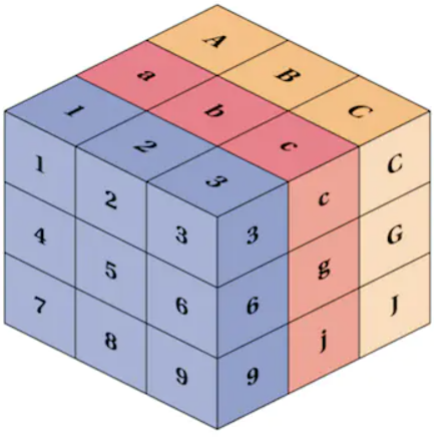
正交矩阵：

设 n 阶方阵 $A=(a_{ij})$，满足：$AA^T=A^TA=E$，则称A为正交矩阵，即$A^{-1}=A^T$。

- 正交矩阵的行向量之间与列向量之间都是两两正交（向量点积为0）的单位向量。
  - n 阶正交矩阵可以看作n维空间中任意相互垂直（正交）坐标基。
  - 向量乘以一个正交矩阵：可以看作是对向量只进行旋转，而没有伸缩和空间映射作用。
- 正交矩阵的应用：
  - RNN (循环神经网络) 中防止梯度消失和维度爆炸的方法：正交初始化。
  - 对于正交矩阵，可以将求逆矩阵的过程转化为求矩阵转置，大大减小了计算量。

标量（Scalar)和张量（Tensor）理解

- 在机器学习和深度学习中，核心的数据结构就是标量、向量、矩阵和张量；
- 其中标量是单个数字，或者说，若 x∈ℝ 表示 x 是一个标量。
- 其中张量是更泛化的实体，是对标量，向量和矩阵的更高层封装，可以理解为，张量的特例是矩阵，矩阵的特例是向量，向量的特例是标量；
- 如下图：多个矩阵，或者说6个矩阵组成的一个张量



TENSOR

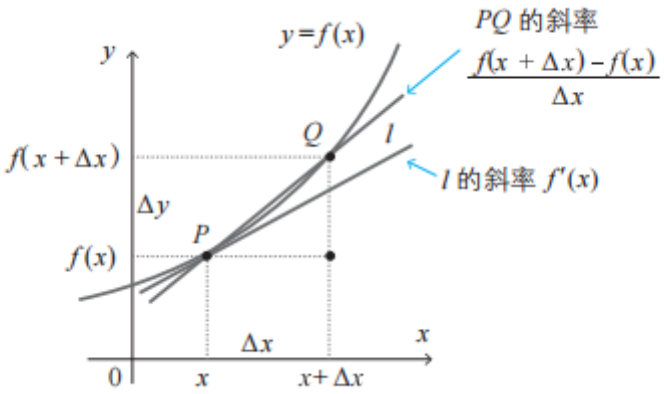
谷歌提供的人工智能学习系统（TensorFlow)的命名中使用了这个属于，据说来源于物理学中的"tension"(张力)，好比一个物体不同的截面上产生的应力方向和大小都不相同，张量是应力在数学上的抽象。

六、导数（derivatives）

导数的定义

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

导函数的含义如下图所示。作出函数 $f(x)$ 的图像，$f'(x)$ 表示图像切线的斜率。因此，具有光滑图像的函数是可导的。



导函数的含义。$f'(x)$ 表示图像切线的斜率。实际上，如果 Q 无限接近 P（也就是$\Delta x \to 0$），那么直线 PQ 无限接近切线 l。

神经网络中用到的函数的导数公式

我们很少使用上面的定义式来求导函数，而是使用导数公式。

下面我们就来看一下在神经网络的计算中使用的函数的导数公式（x 为变量、c 为常数）。

$$(c)' = 0 \qquad (e)' = 0 \qquad (\text{常数的导数为} 0)$$
$$(x)' = 1$$
$$(x^2)' = 2x$$
$$(e^x)' = e^x$$
$$(e^{-x})' = -e^{-x}$$

导数符号

函数 $y = f(x)$ 的导函数用 $f'(x)$ 表示，但也存在不同的表示方法，比如可以用分数形式来表示：
$f'(x) = \frac{dy}{dx}$

这个表示方法是十分方便的，这是因为复杂的函数可以像分数一样计算导数

导数的性质

线性性：和的导数为导数的和，常数倍的导数为导数的常数倍。

$$[f(x) + g(x)]' = f'(x) + g'(x)$$
$$[cf(x)]' = cf'(x)$$

导数的线性性是误差反向传播法背后的主角（损失函数loss的反向传播）

分数函数的导数和 Sigmoid 函数的导数

当函数是分数形式时，求导时可以使用下面的分数函数的求导公式：

$$[\frac{1}{f(x)}]' = -\frac{f(x)}{[f(x)]^2}$$

Sigmoid 函数 σ(x) 是神经网络中最有名的激活函数之一：$\sigma(x) = \frac{1}{1+e^{-x}}$

将 $1+e^{-x}$ 代入上式：

$$\sigma'(x) = -\frac{(1+e^{-x})'}{(1+e^{-x})^2} = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1+e^{-x}-1}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2} = \sigma(x) - \sigma(x)^2$$

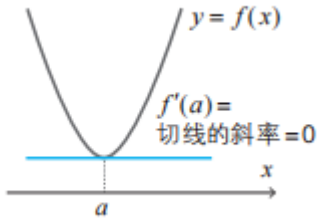
提取$\sigma(x)$**，得到：** $\sigma'(x) = \sigma(x)(1-\sigma(x))$

梯度下降法中，需要对这个Sigmoid函数求导。求导时使用上式会十分方便。

划重点：Sigmoid函数求导公式：$\sigma'(x) = \sigma(x)(1-\sigma(x))$

最小值的条件

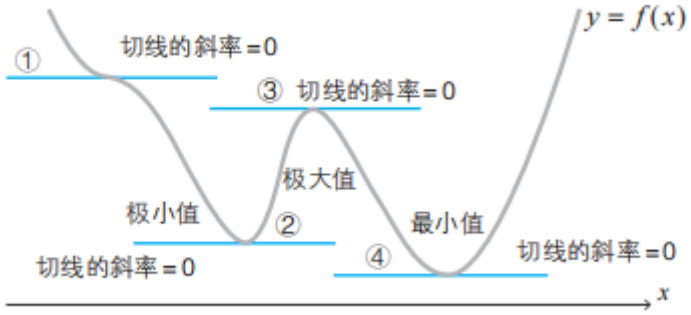
由于导函数 *f*′(*x*) 表示切线斜率，我们可以得到以下原理：**当函数*f*(*x*)在*x* = *a*处取得最小值时，*f*′(*a*) = 0。**



当 *f*(*x*) 在 *x* = *a* 处取最小值时，该函数在该点的切线的斜率（即导函数的值）为 0。

或者换句话说：***f*′(*a*) = 0 是函数 *f* (*x*) 在*x* = *a* 处取得最小值的必要条件**

但是，看看如下例子：



虽然 *f*′(*a*) = 0（切线斜率为 0，即切线与 *x* 轴平行），但在①、②、③的情况下函数不取最小值。

此函数，虽然是连续光滑的，但有不止一处增减性的变化，所以，**f'(a) = 0 是函数 *f* (*x*) 在*x* = *a* 处取得最小值的必要条件，仅仅是必要条件**

总结：需要连续光滑的**可导函数**，还要求是**凸函数**

七、神经网络的偏导数基础

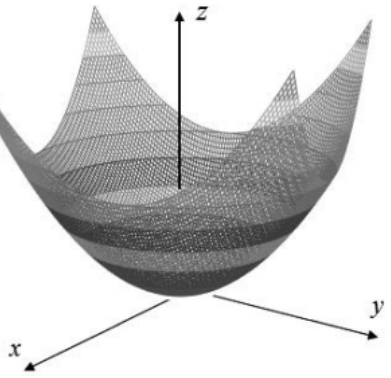
神经网络的计算往往会涉及成千上万个变量，这是因为构成神经网络的神经单元的权重和偏置都被作为变量处理。现在讨论下神经网络计算中所需的**多变量函数**的导数

多变量函数

有两个以上的自变量的函数称为多变量函数。如：

$z = x^2 + y^2$ （x和y都是自变量）

也可写成：$f(x, y) = x^2 + y^2$



$z = x^2 + y^2$ 的图像。

再如：函数$f(x_1, x_2, \cdots, x_n)$：是有 *n* 个自变量$x_1, x_2, \cdots, x_n$的函数

偏导数（partial derivative)

求导的方法也同样适用于多变量函数的情况。但是，由于有多个变量，所以必须指明对哪一个变量进行求导。在这个意义上，关于某个特定变量的导数就称为偏导数

对$f(x, y) = x^2 + y^2$ 这个多自变量函数，求偏导：

只关注变量 x，且 将 y 看作常数来求导：$\frac{\partial f(x,y)}{\partial x}$

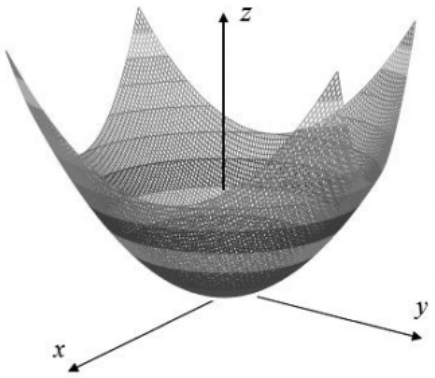
只关注变量 y， 且将 x 看作常数来求导：$\frac{\partial f(x,y)}{\partial y}$

当 $z = f(w, x, b) = wx + b$ 时，分别求偏导，得到：$\frac{\partial z}{\partial x} = w$　　$\frac{\partial z}{\partial w} = x$　　$\frac{\partial z}{\partial b} = 1$

问：当$z = w_1 x_1 + w_2 x_2 + b$ 时，求关于$x1, w2, b$的偏导数

多变量函数的最小值条件

- 光滑的单变量函数 *y* = *f* (*x*) 在点 *x* 处取得最小值的**必要条件**是：导函数在该点取值 0；
- 对于多变量函数，同样适用
- 函数$z = f(x, y)$取得最小值的必要条件是：$\frac{\partial f(x,y)}{\partial x} = 0$　　$\frac{\partial f(x,y)}{\partial y} = 0$
- 而且可以扩展到n个变量的多变量函数$f(x_1, x_2, \cdots, x_n)$
- 举例： 求函数$z = f(x, y) = x^2 + y^2$取得最小值时 *x*、*y* 的值
  - 首先求关于x和关于y的偏导数：$\frac{\partial z}{\partial x} = 2x$　　$\frac{\partial z}{\partial y} = 2y$
  - 函数取得最小值的必要条件是 *x* = 0，*y* = 0。此时函数值 *z* 为 0。由于$x^2 + y^2 \geq 0$，所以我们知道这个函数值 0 就是最小值。
  - 思考下：葡萄酒杯的底部~~



$z = x^2 + y^2$ 的图像。

八、复合函数链式求导法则（误差反向传播法的基础）

复杂（复合）函数求导的链式法则。这个法则对于理神经网络中误差反向传播法很有必要

复合函数

已知函数 $y = f(u)$，当 *u* 表示为 $u = g(x)$ 时，*y* 作为 *x* 的函数可以表示为形如 $y = f(g(x))$ 的嵌套结构（*u* 和 *x* 表示多变量）。这时，嵌套结构的函数 *f*(*g*(*x*)) 称为 *f*(*u*) 和 *g*(*x*) 的**复合函数。**

神经网络中，多个输入$x_1, x_2, \cdots, x_n$，将 *σ*(*x*) 作为激活函数，求神经单元的输出 *y* 的过程如下：

$$y = \alpha(w_1x_1 + w_2x_2 + \cdots + w_nx_n + b)$$

$w_1, w_2, \cdots, w_n$为各个输入的权重

b为神经单元的偏置

y即为：$f(x_1, x_2, \cdots, x_n)$多变量一次函数和$\alpha(x)$激活函数的复合函数

单变量函数的链式法则

单变量函数 $y = f(u)$，当 *u* 表示为单变量函数 $u = g(x)$时，复合函数 *f*(*g*(*x*)) 的导函数：

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dx}$$

同理：当 y为 u 的函数，u为 v 的函数，v为 x 的单变量函数时：

$$\frac{dy}{dx} = \frac{dy}{du}\frac{du}{dv}\frac{dv}{dx}$$

举例：

有Sigmoid函数和线性函数（w,b是常数，所以u是**单变量函数**）：

$$y = \frac{1}{1 + e^{-u}} \quad , \quad u = wx + b$$
$$y = f(g(x)) = \sigma(wx + b)$$

求: $\frac{dy}{dx}$ ？

有：Sigmoid函数的导数公式：$\sigma'(x) = \sigma(x)(1 - \sigma(x))$

$$\sigma'(u) = \sigma(u)(1 - \sigma(u))$$
$$\frac{dy}{du} = y(1 - y)$$
$$\frac{du}{dx} = w$$

得到：

$$\frac{dy}{dx}=\frac{dy}{du}\frac{du}{dx}=y(1-y)w=\frac{w}{1+e^{-(wx+b)}}[1-\frac{1}{1+e^{-(wx+b)}}]$$

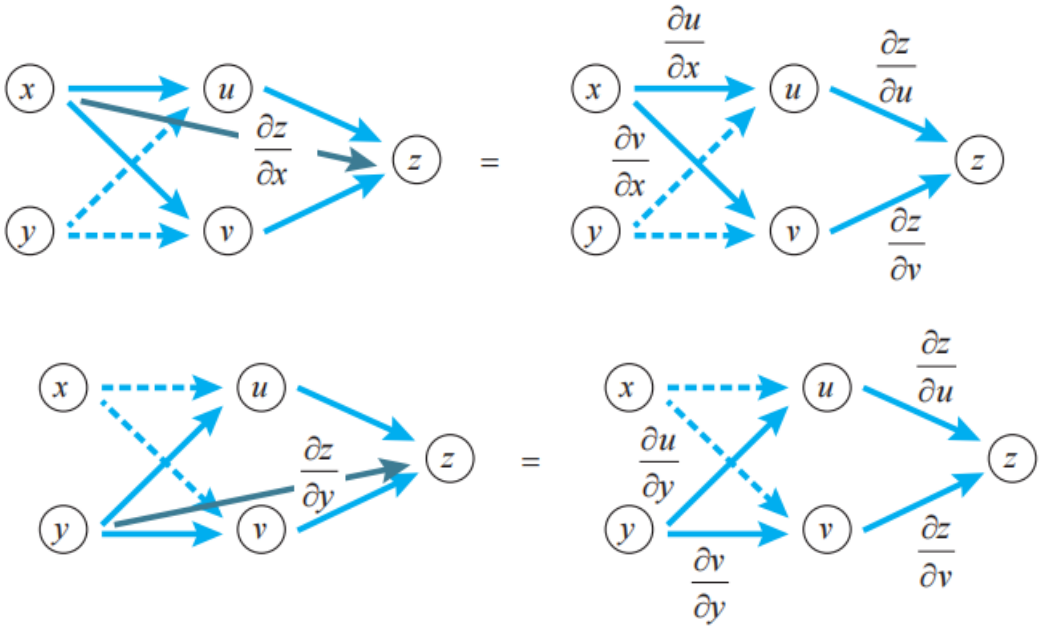
多变量函数的链式法则

在多变量函数的情况下，链式法则的思想也同样适用。只要像处理分数一样对导数的式子进行变形即可。然而事情并没有这么简单，因为必须对相关的全部变量应用链式法则。

变量 z 为 u、v 的函数 $z=f(u,v)$，如果 u、v 分别为 x、y 的函数$u=g(x,y),v=h(x,y)$，则 z 为 x、y的函数$z=f(g(x,y),h(x,y))$，此时下式（多变量函数的链式法则）成立。

$$\begin{aligned}\frac{\partial z}{\partial x}&=\frac{\partial z}{\partial u}\frac{\partial u}{\partial x}+\frac{\partial z}{\partial v}\frac{\partial v}{\partial x}\\\frac{\partial z}{\partial y}&=\frac{\partial z}{\partial u}\frac{\partial u}{\partial y}+\frac{\partial z}{\partial v}\frac{\partial v}{\partial y}\end{aligned}\qquad(\text{求偏导})$$

即：变量 z 为 u、v 的函数，u、v 分别为 x、y 的函数，z 关于 x 求导时，先对 u、v 求导，然后与 z 的相应导数相乘，最后将乘积加起来。如图：



三个以上的多变量函数的情况下也同样成立！

九、多变量函数的近似公式（梯度下降的基础）

梯度下降法是确定神经网络的一种代表性的方法。在应用梯度下降法时，需要用到多变量函数的近似公式。

单变量函数的近似公式

已知导数定义：

$$f'(x)=\lim_{\Delta x\to 0}\frac{f(x+\Delta x)-f(x)}{\Delta x}$$

将Δx这个"无限趋于0"或者说"无限小"的量，替换成"微小"的量，得到：

$$f'(x)\approx\frac{f(x+\Delta x)-f(x)}{\Delta x}$$

整理得：

$$f(x+\Delta x) \approx f(x) + f'(x)\Delta x$$

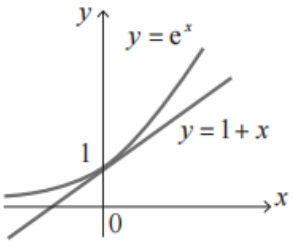
这个就是，**单变量函数的近似公式**，这里Δx是"微小得数"

举例：$y = f(x) = e^x$ ，求其 $x = 0$ 附近 得近似公式

$$(e^x)' = e^x$$
$$e^{x+\Delta x} \approx e^x + e^x \Delta x$$
$$x \to 0$$
$$e^x \approx 1 + \Delta x \approx 1 + x$$

得到：$y = e^x$ 和其在$x \to 0$时得近似函数 $y = 1 + x$

如图：



在 $x = 0$ 附近，$y = e^x$ 与 $y = 1 + x$ 的图像重叠。

多变量函数的近似公式

将单变量函数的近似公式扩展到两个变量的函数。如果x、y 作微小的变化，那么函数 $z = f(x, y)$ 的值将会怎样变化呢？答案就是**多变量函数近似公式**。Δx、Δy 为微小的数。

$$f(x + \Delta x,\ y + \Delta y) \approx f(x, y) + \frac{\partial f(x, y)}{\partial x}\Delta x + \frac{\partial f(x, y)}{\partial y}\Delta y$$

化简一下：定义$\Delta z = f(x + \Delta x,\ y + \Delta y) - f(x, y)$ 得到：

$$\Delta z \approx \frac{\partial z}{\partial x}\Delta x + \frac{\partial z}{\partial y}\Delta y$$

通过这样的简化方式，就很容易将近似公式进行推广到哪个变量$x_1, x_2, \cdots, x_n$。

例如，变量 z为四个变量 w, x, y, b的函数时，近似公式如下所示

$$\Delta z \approx \frac{\partial z}{\partial w}\Delta w + \frac{\partial z}{\partial x}\Delta x + \frac{\partial z}{\partial y}\Delta y + \frac{\partial z}{\partial b}\Delta b$$

近似公式的向量表示

如上，四个变量的函数的近似公式 ，可以表示为如下两个向量的内积：（复习一下向量内积的坐标表示）

$$\text{梯度向量：}\ \nabla\alpha = \left(\frac{\partial z}{\partial w}, \frac{\partial z}{\partial x}, \frac{\partial z}{\partial y}, \frac{\partial z}{\partial b}\right)$$
$$\text{位移向量：}\ \Delta\beta = (\Delta w, \Delta x, \Delta y, \Delta b)$$

对于一般的 *n* 变量函数，近似公式也可以像这样表示为内积的形式

十、梯度下降法的含义与公式

应用数学最重要的任务之一就是寻找函数取最小值的点。研究一下著名的寻找最小值的点的方法——梯度下降法

主要通过两个变量的函数来展开。在神经网络的计算中，往往需要处理成千上万个变量，但其数学原理和两个变量的情形是相同的

梯度下降法的思路

1. 已知函数 *z* = *f*(*x*, *y*)，怎样求使函数取得最小值的 *x*、*y* 呢？最有名的方法就是利用"使函数 *z* = *f* (*x*, *y*) 取得最小值的 *x*、*y* 满足以下关系"：

$$\frac{\partial f(x,y)}{\partial x}=0 \qquad \frac{\partial f(x,y)}{\partial y}=0$$

2. 然而，在实际问题中，联立上面的方程式通常不容易求解
3. 梯度下降法是一种具有代表性的替代方法。该方法不直接求解上面的方程，而是通过慢慢地移动图像上的点进行摸索，从而找出函数的最小值。
4.

向量内积的回顾

两个固定大小的非零向量 $\vec{a}$、$\vec{b}$。当 $\vec{b}$ 的方向与 $\vec{a}$ 相反时，内积 **a** · **b** 取最小值

换句话说，当向量 $\vec{b}$ 满足 $\vec{b}=-k\vec{a}$ 时，可以使得内积 **a** · **b** 取最小值

二变量函数的梯度下降法的基本式

$$(\Delta x,\Delta y)=-\eta(\frac{\partial f(x,y)}{\partial x},\frac{\partial f(x,y)}{\partial y})$$

利用关系式 (5)，如果从点(*x*, *y*)向点(*x* + Δ*x*, *y* + Δ*y*)移动，就可以从图像上点 (*x*, *y*) 的位置最快速地下坡。

定义来了，向量$(\frac{\partial f(x,y)}{\partial x},\frac{\partial f(x,y)}{\partial y})$称为函数$f(x,y)$在点$(x,y)$处的**梯度（gradient)**

举例：设 Δx、Δy 为微小的数。在函数 $z=x^2+y^2$ 中，当 x 从 1 变到 $1+\Delta x$、y 从 2 变到 $2+\Delta y$ 时，求使这个函数减小得最快的向量 $(\Delta x,\Delta y)$。

解：

有：$(\Delta x,\Delta y)=-\eta(\frac{\partial f(x,y)}{\partial x},\frac{\partial f(x,y)}{\partial y})$

有：$z=f(x,y)=x^2+y^2$

得：$\frac{\partial f(x,y)}{\partial x}=2x \qquad ,\frac{\partial f(x,y)}{\partial y}=2y$

有：$(x,y)=(1,2)$，从（1，2）出发

使得函数$f(x,y)$减小最快的向量是：$-\eta(2,4)$

梯度下降法及其用法

二变量函数的梯度下降法：

步骤1：利用$(\Delta x,\Delta y)=-\eta(\frac{\partial f(x,y)}{\partial x},\frac{\partial f(x,y)}{\partial y})$找到函数$f(x,y)$减小最快的方向$(\Delta x,\Delta y)$

步骤2：从点 (x,y) 向点 $(x+\Delta x,y+\Delta y)$ 移动，到达新的点(x,y)

步骤3：重复步骤1和步骤2，反复运算，得到$f(x,y)$的最小值

将梯度下降法推广到三个变量以上的情况

梯度下降法基本式：

$$(\Delta x_1, \Delta x_2, \cdots, \Delta x_n) = -\eta(\frac{\partial nf(x_1, x_2, \cdots, x_n)}{\partial x_1}, \frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_2}, \cdots, \frac{\partial f(x_1, x_2, \cdots, x_n)}{\partial x_n})$$

注意，左边的向量也称为位移向量，可以简写成Δx，右边的向量可以简写成：哈密顿算子 ∇f

得到梯度下降法基本式的**简洁写法**：

$$\Delta x = -\eta \nabla f$$

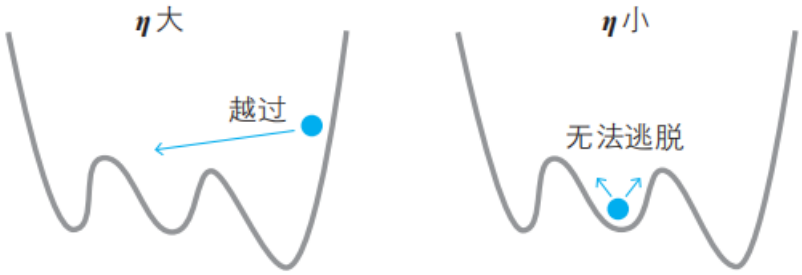
再注意：

η 的含义以及梯度下降法的要点

η 可以看作人移动时的“步长”，根据 η 的值，可以确定下一步移动到哪个点。

如果步长较大，那么可能会到达最小值点，也可能会直接跨过了最小值点（左图）。

而如果步长较小，则可能会滞留在极小值点（右图）

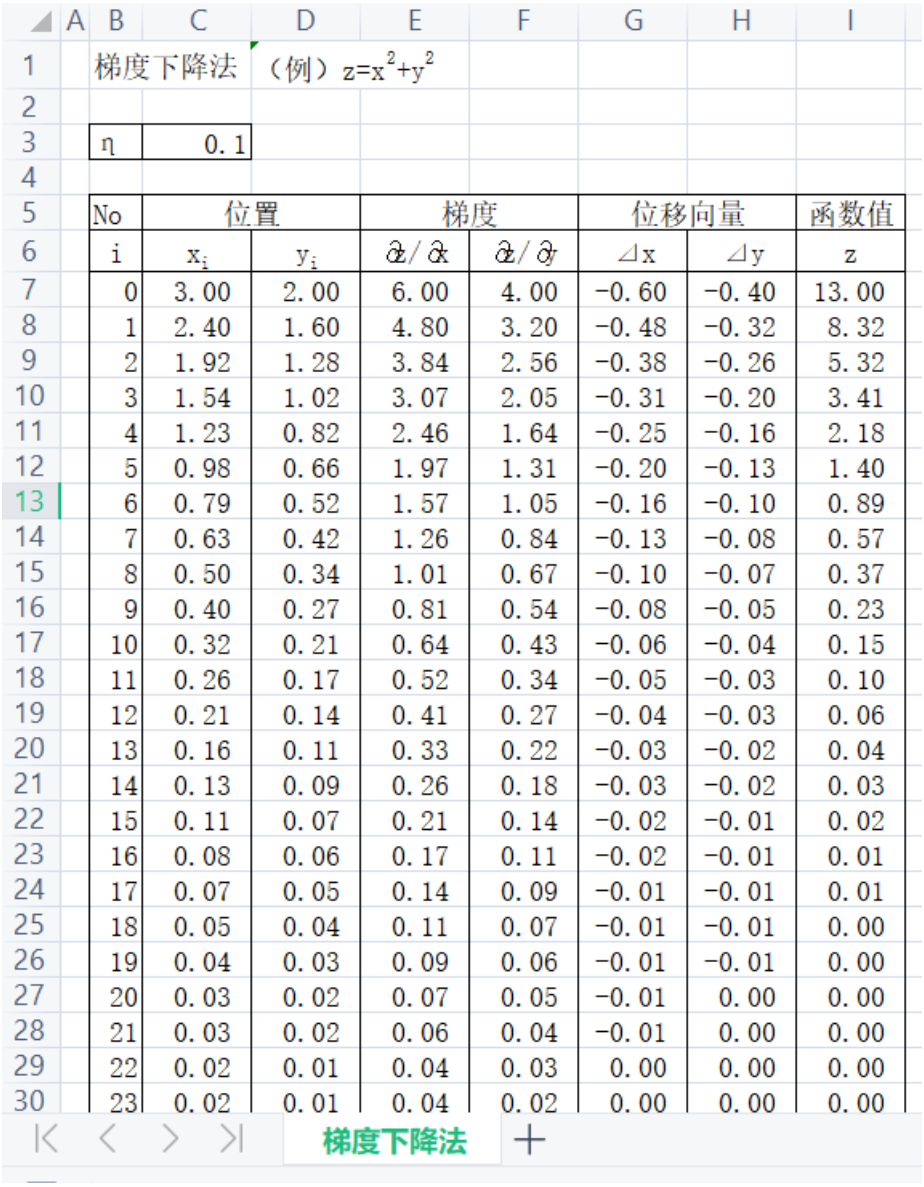


在神经网络的世界中，η 也称为学习率。遗憾的是，它的确定方法没有明确的标准，只能通过反复试验来寻找恰当的值。

说明1：梯度下降法也可以用于单变量函数，只要梯度向量解释为一维向量（$n = 1$）的情况就可以了。也就是说，将偏导数替换为导数，将得到的下式作为梯度下降法的基本式。$\Delta x = -\eta f'(x)$　（η 为正的微小常数）

说明2：将 η 看作步长，实际上这并不严谨，正确的说法是，位移向量的大小才是步长。不过，虽然人的步长大体上是固定的，但梯度下降法的“步长”是不均匀的。因为梯度在不同的位置大小不同。因此，在应用数学的数值计算中，有时会对梯度下降法的基本式进行其他变形~~。

十一、用Excel体验梯度下降法



	A	B	C	D	E	F	G	H	I
1		梯度下降法		（例）$z=x^2+y^2$					
2									
3		η	0.1						
4									
5		No	位置		梯度		位移向量		函数值
6		i	x_i	y_i	$\partial z/\partial x$	$\partial z/\partial y$	Δx	Δy	z
7		0	3.00	2.00	6.00	4.00	-0.60	-0.40	13.00
8		1	2.40	1.60	4.80	3.20	-0.48	-0.32	8.32
9		2	1.92	1.28	3.84	2.56	-0.38	-0.26	5.32
10		3	1.54	1.02	3.07	2.05	-0.31	-0.20	3.41
11		4	1.23	0.82	2.46	1.64	-0.25	-0.16	2.18
12		5	0.98	0.66	1.97	1.31	-0.20	-0.13	1.40
13		6	0.79	0.52	1.57	1.05	-0.16	-0.10	0.89
14		7	0.63	0.42	1.26	0.84	-0.13	-0.08	0.57
15		8	0.50	0.34	1.01	0.67	-0.10	-0.07	0.37
16		9	0.40	0.27	0.81	0.54	-0.08	-0.05	0.23
17		10	0.32	0.21	0.64	0.43	-0.06	-0.04	0.15
18		11	0.26	0.17	0.52	0.34	-0.05	-0.03	0.10
19		12	0.21	0.14	0.41	0.27	-0.04	-0.03	0.06
20		13	0.16	0.11	0.33	0.22	-0.03	-0.02	0.04
21		14	0.13	0.09	0.26	0.18	-0.03	-0.02	0.03
22		15	0.11	0.07	0.21	0.14	-0.02	-0.01	0.02
23		16	0.08	0.06	0.17	0.11	-0.02	-0.01	0.01
24		17	0.07	0.05	0.14	0.09	-0.01	-0.01	0.01
25		18	0.05	0.04	0.11	0.07	-0.01	-0.01	0.00
26		19	0.04	0.03	0.09	0.06	-0.01	-0.01	0.00
27		20	0.03	0.02	0.07	0.05	-0.01	0.00	0.00
28		21	0.03	0.02	0.06	0.04	-0.01	0.00	0.00
29		22	0.02	0.01	0.04	0.03	0.00	0.00	0.00
30		23	0.02	0.01	0.04	0.02	0.00	0.00	0.00

梯度下降法

十二、最优化问题和回归分析

从数学上来说，确定神经网络的参数是一个最优化问题，具体就是对神经网络的参数（即权重和偏置）进行拟合，使得神经网络的输出与实际数据相吻合。

为了理解最优化问题，最浅显的例子就是回归分析。

什么是回归分析

由多个变量组成的数据中，着眼于其中一个特定的变量，用其余的变量来解释这个特定的变量，这样的方法称为回归分析。回归分析的种类有

很多。

最简单的**一元线性回归分析**，以两个变量组成的数据作为考察对象，用一条直线近似地表示右图所示的散点图上的点列，通过该直线的方程来考察两个变量之间的关系。该直线称为**回归直线**

有训练数据如下：

身高（x）	体重（y）
174.1	61.5
...	...
177.8	66.1

回归方程，x是自变量，y是因变量，模型（model）： $y = wx + b$

训练数据：自变量x_i，因变量 y_i

预测值：$\hat{y} = wx + b$

预测值和实际测试数据的误差：$y_i - \hat{y_i} = y_i - (wx_i + b)$

平方误差：$C_i = \frac{1}{2}(y_i - (wx_i + b))^2$

平方误差和（损失函数，代价函数）：$loss(x, y) = \sum_{i=1}^{n} \frac{1}{2}(y_i - (wx_i + b))^2$

在这个简单的线性回归中，我们的目标是：找到合适的w, b，尽可能让损失函数$loass(x, y)$取最小值0，即：

$$\frac{\partial loss(x, y)}{\partial w} = 0 \qquad \frac{\partial loss(x, y)}{\partial b} = 0$$

根据复合函数链式偏导法则，得到：

$$\frac{\partial loss(x, y)}{\partial w} = \sum_{i=1}^{n} x_i(y_i - (wx_i + b)) = 0$$
$$\frac{\partial loss(x, y)}{\partial b} = \sum_{i=1}^{n} -(y_i - (wx_i + b)) = 0$$

联立求解w, b

总结：

综上，机器学习过程中

步骤1：确定了模型（线性回归直线模型中只有两个参数w, b），即，根据身高和体重的训练数据去找到预测体重的回归直线方程

步骤2：通过大量训练数据求累加和，使其等于0（或趋近于0），联立求解，得到模型中的参数w, b，完成机器学习

注意1：当模型中参数增多，要提供更多的训练数据，数据规模

注意2：训练数据的采集和使用前，要预先进行数据标准化和数据清洗

注意3：损失函数有很多种形式，本例中损失函数是平方误差总和（最小二乘法），最小值为0，本例中代价函数关于w和b两个参数求偏导，如果参数量巨大会怎样？

十三、概率初步

概率论是研究不确定的学科。

概率论是现有许多人工智能算法的基础。现阶段的很多人工智能算法都是数据驱动的，且目的大多为了做预测或是作出更好的决策。如：

- 机器翻译中，如何检测你输入的语言种类。一种简单的方法就是把你输入的词或句子进行分解，计算各语言模型的概率，然后概率最高的是最后确定的语言模型。
- 用神经网络进行图像分类，网络的输出是衡量分类结果可信程度的概率值，即分类的置信度，我们选择置信度最高的作为图像分类结果。
- 混合高斯模型、隐马尔科夫模型等传统语音处理模型都是以概率论为基础的

随机试验

满足以下三个特点的试验称为随机试验：

- 可以在相同的条件下重复进行。
- 每次试验的可能结果不止一个，并且能事先明确试验的所有可能结果。
- 进行一次试验之前不能确定哪一个结果会出现。

样本点、样本空间、随机事件

- 样本点：一次随机试验所有可能结果的集合是**样本空间**，而随机试验中的每个可能结果称为**样本点。**
- 随机事件：随机试验的某个或某些样本点组成的集合, 常用大写字母表示。
- 举例：
  - 随机试验E_1 ：扔一次骰子，观察可能出现的点数情况。
  - 扔一次骰子点数出现情况样本空间为：S = {1，2，3，4，5，6}。
  - 扔一次骰子样本点（可能的结果）为：e_i =1，2，3，4，5，6。
  - 随机事件A_1：“扔1次骰子出现的点数为5”，即A_1=$\{x \mid x=5\}$。
  - 随机事件A_2：“扔2次骰子出现的点数均为5”，即A_2=$\{x_i \mid x_1=5, x_2=5\}$。

随机变量

随机变量：本质是一个函数，是从样本空间的子集到实数的映射，将事件转换成一个数值，也就是，一次随机事件的数量化表示，是一个实值单值函数。

一些随机试验的结果可能不是数，因此很难进行描述和研究，比如S={正面，反面}。因此将随机试验的每一个结果与实数对应起来，从而引入了随机变量的概念。随机变量用大写字母表示，其取值用小写字母表示。

举例1：随机试验E_2：抛两枚骰子，观察可能出现的点数的和

$$X=X(e)=X(i,\ j)=i+j,\quad (\mathrm{i},\ \mathrm{j}=1,\ 2,\ \cdots,\ 6)$$

这里X 即为**随机变量**

按照随机变量的可能取值，可分为：

离散随机变量：随机变量的全部可能取到的值是有限个或可列无限多个。如：某年某地的出生人数。

连续随机变量：随机变量的全部可能取到的值有无限个，或数值无法一一列举。如：奶牛每天挤出奶的量，可能是一个区间中的任意值

随机向量

在实际应用中，经常需要对所考虑的问题用多个变量来描述。我们把多个随机变量放在一起组成向量，称为多维随机变量或者随机向量。

定义：如果$X_1(\omega)$，$X_2(\omega)$，$\ldots$，$X_n(\omega)$ 是定义在同一个样本空间$\Omega=\{\omega\}$上的 n 个随机变量，则称：$X（\omega）=（X_1(\omega)，X_2(\omega)，\ldots，X_n(\omega)）$，为 n 维(元)随机变量或**随机向量。**

如：我们通过人脸判断人的年龄，可能需要结合多个特征（随机变量），如脸形、脸部纹理、面部斑点、皮肤松弛度、发际线等，将这些特征结合映射为一个实数，即年龄。

频率派视角下的概率

频数：多次随机试验后，随机变量值中代表某特征的数（某个随机事件）的出现次数。

频率：多次随机试验后，某个随机事件的频数与随机试验总数之比。

概率：做大量重复试验时，随着试验次数的增加，某一随机事件出现的频率，总在一个定值的附近稳定地摆动，便将此定值称为该事件的**概率。**

好玩的经典的布丰投针实验

贝叶斯派视角下的概率

频率派与贝叶斯派的论战：

频率派：客观世界内在地存在随机性，这是自然规律

贝叶斯派：不，本身并没有客观的随机性，只是观察者没有上帝视角

频率派：我们要找出正确的概率模型参数，它是客观存在且唯一正确的

贝叶斯派：既然我们都没有上帝视角了，如何知道哪一个是正确的?

. . .

对于贝叶斯派而言，他们认为所谓的「随机」，只是由于人没有办法掌握全局信息。

比如，对于有内幕消息的人来说，股市的波动是确定的，对于普通散户而言，股市的波动是随机不确定的

所以，贝叶斯派将「概率」视为对某种结果出现的信任程度，而非频率

总结：频率派的缺陷：

- 试验必须可以进行很多次（易于操作，成本低，周期短）
- 缺乏一定的严谨性（如何确定"稳定"与否）
- 无法对不可多次重复的事件进行预判（毒理学检测, etc）
- 内在地觉得事件本身是随机的

古典概型

概率学习之路的启蒙者
定义：若一个随机试验包含的基本事件的数量是有限的，且各个基本事件发生的可能性均相等，则此种概率模型称为**古典概型**
特征：

1. 包含的基本事件数量有限
2. 所有基本事件发生的概率均相等
3. 任意两个基本事件之间是互斥的

场景：
- 掷骰子，抛硬币，彩票抽奖，婴儿的出生日期
- 选择题不会全靠蒙，so many……

计算公式： $P(A)=\frac{m}{n}$ m：A事件包含的基本事件个数，n： 基本事件总数

比如掷骰子：A 事件：偶数点数向上， m：是A事件出现的个数

突破极限：

- 若某一随机试验的基本事件数量不断增加不断增加……
- 古典概型：偶要hold不住了…怎么办……
- 几何概型：莫慌，俺来了
- 几何概型粉墨登场……

几何概型

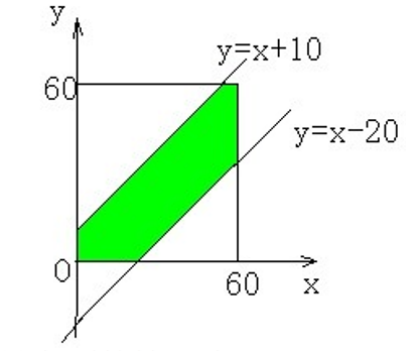
定义：若每个事件发生的概率只与构成该事件区域的度量（长度，面积，体积或度数）成比例，则称这样的概率模型为**几何概型。**

实质：将古典概型从有限到无限的扩展

特征：

- 包含的基本事件数量无限
- 所有基本事件发生的概率均相等
- 任意两个基本事件之间是互斥的
- 场景：
    - 对于不规则形状面积的测量
    - 相遇问题

      男生和女生相约晚上七点至八点见面，若女生先到，则会等男生十分钟，过时不候；若男生先到，则会等女生二十分钟，过时不候。求他们见上面的概率为多少。



联合概率

定义：两个或多个事件同时发生的概率

符号表示：$P(XY)$或$P(X,Y)$

$$P(XY) = P(X,Y) = P(Y|X) \cdot P(X) = P(X|Y) \cdot P(Y)$$

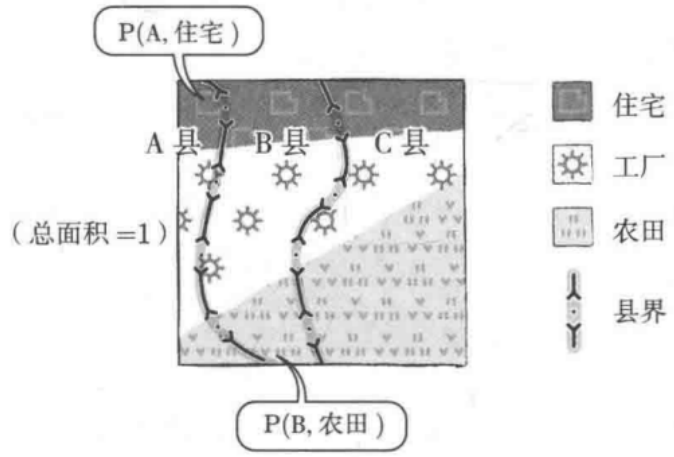
若事件相互独立，则

$$P(XY) = P(X,Y) = P(X) \cdot P(Y)$$

解读：

- X，Y 两事件同时发生的概率 ，等于，X发生的概率 乘以 X事件已发生前提下Y发生的概率
- 若X，Y两事件相互独立，即，事件Y发不发生，对事件X发生的概率没有影响，反之亦然，公式可以改写

理解：



举例：班级的同学达到多少人，至少有两个同学的生日一样的概率大于0.5?
解：考虑某个同学甲，第1个同学和其生日不同的概率为：$\frac{364}{365}$，第2个同学和前两个人生日不同的概率为：$\frac{363}{365}$，以此类推，第n个同学和前面所有人生日不同的概率为：$\frac{365-n}{365}$，则所有 n个同学生日彼此不同的概率依据联合概率的规则，为：

$P = \frac{364}{365} \times \frac{363}{365} \times \cdots \times \frac{365-n}{365} \leq 0.5$

经计算： $n \geq 23$ 即可!

条件概率、贝叶斯公式

条件概率的定义：事件Y已发生的前提下，事件X发生的概率。也叫 **后验概率**

符号表示：$P(X \mid Y)$

$$P(X \mid Y) = \frac{P(XY)}{P(Y)}$$

理解：

♢J	♢Q	♢K	♡J
♡Q	♡K	♡1	♡2
♠K	♠1	♠2	♡3
♠3	♠4	♠5	♠6

	$Y=$数字牌	$Y=$人头牌
$X=$红色	3/16	6/16
$X=$黑色	6/16	1/16

有：

$$P(Y=\text{数字牌}|X=\text{红色})=\frac{1}{3}$$
$$P(Y=\text{人头牌}|X=\text{红色})=\frac{2}{3}$$
$$P(Y=\text{数字牌},\ X=\text{红色})=\frac{3}{16}$$
$$P(X=\text{红色})=\frac{9}{16}$$

思辨：条件概率描绘的是B发生的前提下，A发生的概率，那A和B不都是发生了吗？那和联合概率P(AB)不是一样吗？P(AB)和P(A|B)有什么不同？

- 两者的“外部环境”不一样；在条件概率的情形下，我们把事件 B 的发生当做一件已经确定的事，B先已发生了，然后再看 A **在此情形下**发生的概率；
- 而在联合概率的情形下，没有哪个事件先发生的说法，没有先决条件，既关注一个事件发生的概率，也关注其它事件发生的概率。

贝叶斯公式：

由条件概率和联合概率公式，“超牛”的贝叶斯公式如下，很easy嘛：

$$P(Y|X)=\frac{P(X|Y)P(Y)}{P(X)}$$

按照贝叶斯的语言，解读下：

X: 样本（可被观测到的）	**Y: 产生样本的内在机制参数**
P(Y|X)：后验概率(posterior)	P(X)：证据(evidence)
P(X|Y)：似然估计(likeli$h$ood)	P(Y)：先验概率(prior)

贝叶斯公式应用：中文分词、统计机器翻译、深度贝叶斯网络等

尝试理解分词：

如何对这个句子进行分词（词串）才最靠谱？ 杭州 | 西湖、杭 | 州西湖、杭州西 | 湖、.....

令 X为字串（句子），Y 为词串（一种特定的分词假设：杭，杭州，杭州西，州西湖.....）。

我们就是需要寻找使得 P(Y|X) 最大的 Y，使用贝叶斯公式：$P(Y|X)=\frac{P(X|Y)P(Y)}{P(X)}$

- 若已知
  - P(Y)：对于每种分词假设都不变;
  - P(X)：这种分词方式（词串）的可能性
  - P(X|Y)：这个词串生成我们的句子的可能性，我们就可以成功分词。

频率派只关注 X（观测到的样本）
贝叶斯派认为除了 X 之外，还应对样本产生的机制参数预估一个先验概率，甚至可以从主观的角度预估，比如去猜测一本书的价格，人会在心里预估出一个价格范围，例如不会超过一千美金，贝叶斯派做分析必须用到参数先验概率，哪怕主观意识去预估先验概率（$P(Y)$），是在抽样之前对参数的预估，而在获得样本之后，人们对于参数的判断会根据样本发生变化，便得到了参数的后验分布（$P(Y \mid X)$）通过贝叶斯公式可以求得后验分布

离散型随机变量的概率分布和概率分布函数

有离散型随机变量X的所有可能取值为$x_i(i=1,2,3,\cdots,n)$

$$P(X=x_i)=P_i, \qquad (i=1,2,\cdot,n)$$

称为随机变量X的**概率分布**、**分布律**、或**概率质量函数**或**概率函数**，至今未统一这些术语。

分布律，又叫概率质量函数（Probability Mass Function，PMF）

常用表格形式来表示，如

X	x_1	x_2	x_3	$\cdots$	x_n
P_i	P_1	P_2	P_3	$\cdots$	P_n

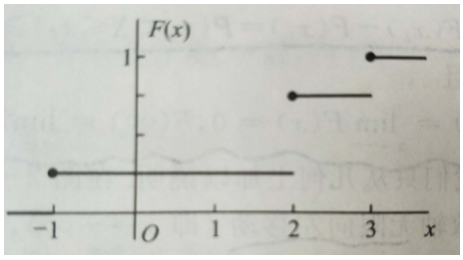
由概率定义：$P-i\geq 0$ 且 $\sum_{i=1}^{n}P_i=1$

概率分布函数：$F(x)=P\{X\leq x_i\}$ ，$F(x)$ 表示 随机变量X小于某个数的概率之和，累加的思想，是一个递增的函数。与分布律（概率分布、概率函数）不同！不同！

举例：随机变量X的分布函数如下，求X的分布律和$P\{-1<X\leq 3\}$

$$F(x)=\begin{cases}0 & x<-1\\ 0.4 & -1\leq x<1\\ 0.8 & 1\leq x<3\\ 1 & 3\leq x\end{cases}$$

解：用函数图像理解



分布律：

X	**-1**	**1**	**3**
P	0.4	0.4	0.2

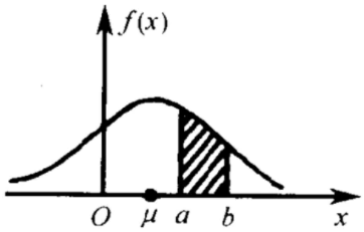
$P\{-1<X\leq 3\}=P\{x=1\}+P\{x=3\}=0.4+02=0.6$

连续型随机变量的概率密度、概率分布函数：

实际生活中，我们通常不太关心取到某一点的概率，而是取到某一区间的概率。所以我们需要研究分布函数。**分布函数**，又叫累计分布函数（Cumulative Distribution Function，CDF）

还是看图吧，**概率密度函数**$f(x)$，如下图所示：

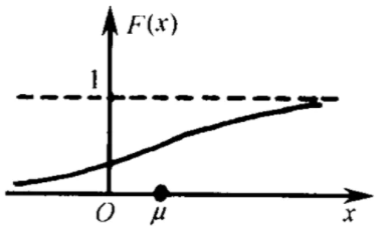
这里的μ 是数学期望，(a,b) 是区间，区间与函数曲线围起来的面积，就是P，是**概率**



函数 $f(x)$ 称为X 的概率密度函数（Probability Density Function，PDF），简称**概率密度**

概率密度函数 $f(x)$ 是 概率分布函数$F(x)$ 的导函数!

再看图，**概率分布函数**$F(x)$，如下图所示：

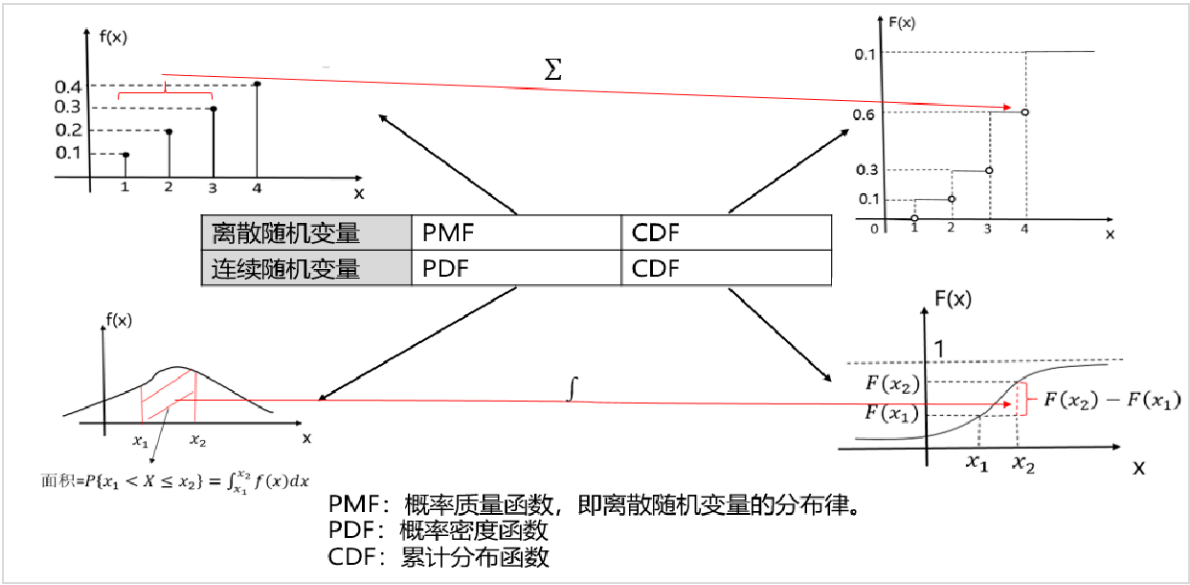


这里的μ 是数学期望，$F(x)$ 表示从$(-\infty,x)$ 区间里**概率的累加和**

概率分布函数$F(x)$ 是 概率密度函数 $f(x)$ 的 积分：

$$F(x)=\int_{-\infty}^{x}f(t)dt$$

总结:



数学期望

数学期望相当于统计数学中的**均值**，用$E(X)$ 或 μ 表示，分两种情况：

离散型：

若，随机变量$X=\{x_1,x_2,\cdots,x_n\}$，$P(x_i)$ 为x_i对应的概率，这里也可以理解为x_i出现频率，则:

$$E(X)=\sum_{i=1}^{n}x_iP(x_i)$$

连续型：

若，连续性随机变量X的概率密度函数为$f(x)$,则：

$$E(X)=\int_{-\infty}^{\infty}xf(x)dx$$

经典例：甲乙两人赌博，对于每一局而言，两人获胜的几率都是相等的，比赛为五局三胜制，最终的赢家可以获得100美元的奖励。在第四局比赛开始前，由于特殊情况需终止比赛，而前三局甲赢了两局，乙赢了一局，现在应如何公平地分配这100美元?

性质：

$$\begin{gathered}E(C)=C,\qquad C为常数\\E(CX)=CE(X),\qquad C为常数\\E(X+Y)=E(X)+E(Y)\\E(XY)=E(X)E(Y),\qquad 当X,Y相互独立时\\E(XY)\qquad 需按照定义去计算，当X,Y不独立时\end{gathered}$$

方差和标准差

方差（Variance）：度量随机变量和其数学期望之间的偏离程度，有时也用D 、σ^2 表示

$$Var(X)=D(X)=\sigma^2=E((X-E(X))^2)=\sum_{i=1}^{N}\frac{(X-E(X))^2}{N}$$

$(X-E(X))^2$ 是随机变量与数学期望（均值）的离散程度，对这个离散程度再求数学期望（均值）得到**方差**

标准差（均方差）：方差的平方根，用 σ 表示，度量效果与方差一致，但方差不同，标准差的量纲与随机变量一致！

$$\sigma=\sqrt{E((X-E(X))^2)}=\sqrt{\sum_{i=1}^{N}\frac{(X-E(X))^2}{N}}$$

协方差：

$$Cov(X,Y)=E(XY)-E(X)E(Y)$$

用于衡量两个变量的总体误差
协方差为正：两个变量的变化趋势相同，当一个大于其期望值时，另一个也大于，小于的情况亦然
协方差为负：两个变量的变化趋势相反，当一个大于其期望值时，另一个就小于，反之亦然
协方差为0：两个变量无线性相关性；两个变量独立，则协方差一定为0，但反之并不一定成立

离散分布(离散型随机变量)：伯努利分布

设随机变量X只可能取0与1两个值，它的分布律（概率质量函数，概率函数）是：

$$P\{X=k\}=p^k(1-p)^{1-k}, \qquad k=0,\ 1(0<p<1)$$

则称：X 服从以 p 为参数的伯努利分布（0-1分布、两点分布、a-b分布）

表格化：

X	**0**	**1**
P_k	$1-p$	p

当前，数学期望：$E(X)=p$ ， 方差： $D(X)=p(1-p)$

- 伯努利分布主要用于二分类问题，可以用伯努利朴素贝叶斯进行文本分类或垃圾邮件分类。伯努利模型中每个特征的取值为1和0，即某个单词在文档中是否出现过，或是否为垃圾邮件。
- 为防止模型过拟合，常会用dropout方法随机丢弃神经单元，每个神经单元都被建模为伯努利随机变量，被抛弃的概率为p，成功输出的比例为1−p。

特殊的离散分布：二项式分布

二项分布是重复 n 次伯努利试验满足的分布

若用X表示 n 重伯努利试验中事件A发生的次数，则 n 次试验中事件A发生 k 次的概率为：

$$P(X=k)=C_n^k p^k(1-p)^{n-k}, \qquad k=0,\ 1,\ 2,\ \cdots n$$

此时称：X 服从参数为 $n,\ p$ 的二项分布，记为：X ~ B(n，p) 。

其中：期望$E(X)=np$，方差：$D(X)=np(1-p)$。

- 二项分布在 NLP(自然语言处理) 中使用得非常广泛，例如估计文本中含有“的”字的句子所占百分比，或者确定一个动词在语言中常被用于及物动词还是非及物动词。
- 如在Dropout方法中，对于某一层的n个神经元在每个训练步骤中可以被看作是n 个伯努利实验的集合，即被丢弃的神经元总数服从参数为n，p的二项分布。

特殊的离散分布 - 泊松分布

若随机变量所有可能的取值为0，1，2，…，而取每个值的概率(分布律)为：

$$P\{X=k\}=\frac{\lambda^k e^{-\lambda}}{k!}, \qquad k=0,\ 1,\ 2,\ \cdots$$

则称：X服从参数为 λ 的泊松分布，记为：X~P(λ) 。

其中，$E(X)=\lambda,\ D(X)=\lambda$。参数 λ 是单位 时间或单位面积内随机事件的平均发生率。

泊松分布是二项分布当n很大p很小时的近似计算。

- 泊松分布用于描述单位时间内，随机事件发生的次数。如一段时间内某一客服电话受到的服务请求的次数、汽车站台的候客人数、机器出现的故障数、自然灾害发生的次数、DNA序列的变异数等。
- 图像处理中，图像会因为观点显示仪器测量造成的不确定性而出现服从泊松分布的泊松噪声，我们经常会给图像加泊松噪声用于图像的数据增强。

特殊的连续分布-正态分布（上帝的分布）

若连续型随机变量X的概率密度函数 $f(x)$ 为：

$$f(x)=\frac{1}{\sqrt{2\pi}\sigma}e^{\frac{(x-\mu)^2}{2\sigma^2}}$$

其中，方差 σ^2 ，期望 μ ，标准差 σ $(\sigma>0)$ 为常数，则称X服从参数为（μ，σ）的**正态分布**或**高斯分布**，记为：X-N(μ，σ^2)。

当μ=0，σ=1时，称随机变量X服从标准正态分布，记为X~N(0，1)。如下：

$$f(x)=\frac{1}{\sqrt{2\pi}}e^{\frac{x^2}{2}}$$

人群的身高、学生的成绩、社会的财富，在自然现象和社会现象中，大量随机变量都服从或近似服从正态分布。正态分布是机器学习中最常用的分布，如：

- 图像处理中，我们可以给图像添加高斯噪声用于图像增强等任务。也可以用高斯滤波器去除噪声并平滑图像。还可以用混合高斯模型进行图像的前景目标检测。
- 在传统语音识别模型GMM-HMM（高斯混合模型-隐马尔科夫）中，高斯混合模型就是由多个高斯分布混合起来的模型。
-

原图	加入泊松噪声（λ =15）	加入高斯噪声（μ=0，σ=10）
		

中心极限定理

为什么正态分布会如此多见？因为背后有个 中心极限定理

其基本内容是：大量相互独立的随机变量的均值经标准化后收敛于**正态分布**

例如：两个骰子朝上点数之和，近似于正态分布，作图