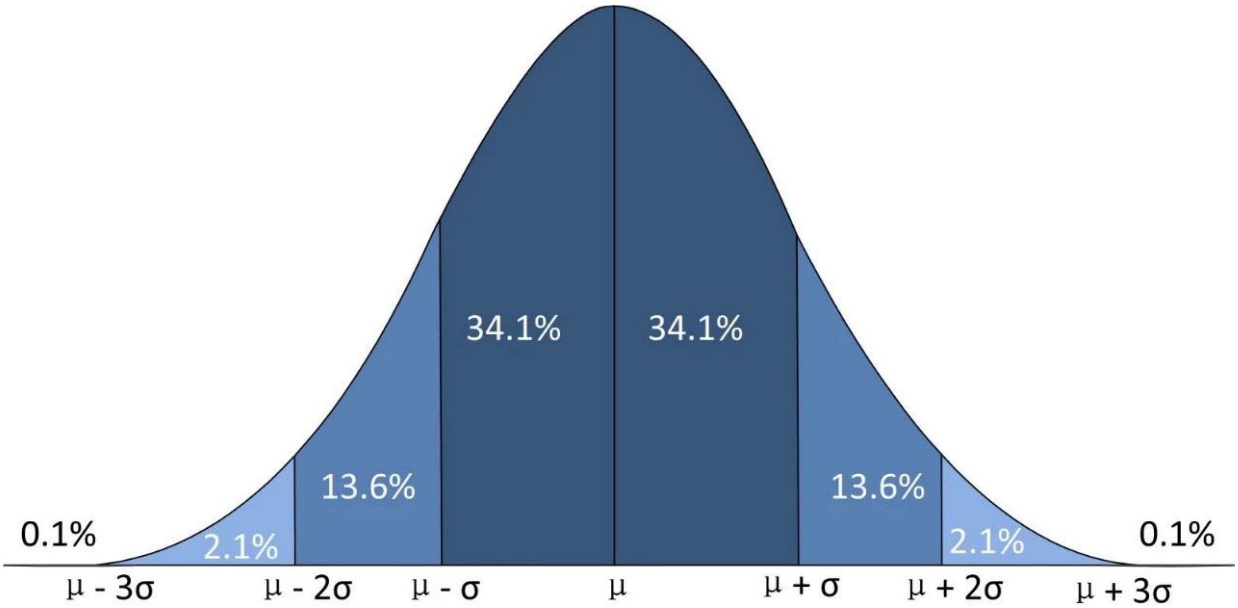
1、概率论与机器学习

机器学习其实是集合了统计学、概率论、计算机科学、数学算法等方面的交叉研究，即便你对机器学习的应用炉火纯青，但对这些技术没有一个**全面的数学理解**，极有可能出现应用失误。因此与其说为什么概率论与数理统计在机器学习中为什么这么重要，不如说为什么数学在机器学习中为什么这么重要！

概率论研究的是事物的不确定性，它是统计学、信息论的前置课程。概率论的难度系数属中等，毕竟你在高中就学习过如何计算一个随机变量的期望、方差。从机器学习的视角来看，概率论是必须要了解的，但不需要达到精通的程度。你只需要灵活运用它，把机器学习世界的不确定性变量算清楚就足够了。因此，当你掌握了概率论，你就揭开机器学习世界神秘的一层面纱。

对于有监督机器学习，其属性特征数据对应 X，它的目标值标签对应 y，如果我们把它当作是随机变量的话，那我们就可以用概率论的观点对它进行建模。假设它服从某种**概率分布**，比如说人的身高大体是服从正太分布的，像**姚明**一样非常高的非常少，像**郭敬明**一样矮也是非常少的，比如中国的男性平均身高 1.75 左右，画出来就是我们学概率论和数理统计时候的一个正太分布：



我们要是对数据进行分类的话，根据他的身高、体重等等，那我们就可以对他的身高进行建模来计算它服从某种分布，然后计算他的概率，这就是我们要学习概率论的原因。

2、随机事件

什么是**随机事件**呢？就是可能发生，也可能不发生的事件。比如你抛硬币，它正面朝上或者反面朝上，这就是一个随机事件；生孩子，生男生女这也是一个随机事件。

如果一定发生的话，这种称为**必然事件**，比如说太阳明天会升起，这肯定是必然事件；不可能发生的事件，我们称之为不可能事件，比如水往高处流，这就是不可能事件。

我们一般把随机事件用大写字母 A 或 B 这样来表示，每一个随机事件它关联有一个发生的概率，记作 P(A)，像抛硬币它正面朝上的概率是 0.5，反面朝上的概率也是 0.5，$0 \le P(A) \le 1$。 如果概率等于 1 那就是必然事件，如果等于 0 那就是不可能事件。

以前学概率论的时候，老师交了我们各种计算概率的方法，比如抽各种颜色的球等等这样的问题，一般都是用排列组合来算的。

举例说明：40个球，分4种颜色,比例为：1、5、9、25。一次抽四个，抽中不同颜色各一个的概率是多少？

事件随机抽球四次的排列数为（分母）：$A_{40}^{4}=\frac{40!}{(40-4)!}=40\times 39\times 38\times 37$

各取出一种颜色的排列组合数为（分子）：$A_{4}^{4}\times C_{1}^{1}\times C_{5}^{1}\times C_{9}^{1}\times C_{25}^{1}=(4\times 3\times 2)\times(5\times 9\times 25)$

那么，一次抽四个，抽中不同颜色各一个的概率为：

$$P=\frac{A_{4}^{4}\times C_{1}^{1}\times C_{5}^{1}\times C_{9}^{1}\times C_{25}^{1}}{A_{40}^{4}}\approx 0.01231$$

3、条件概率

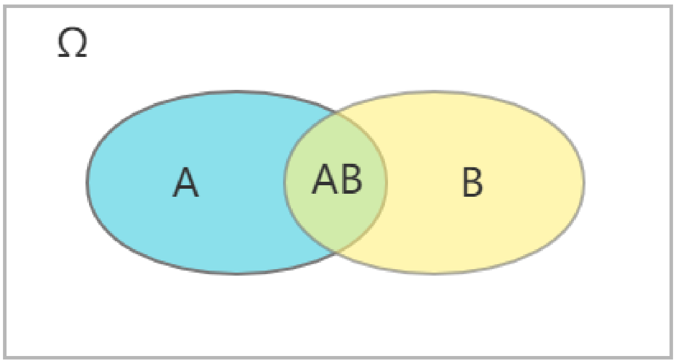
3.1、条件概率公式

条件概率是针对于两个或更多个有相关关系、因果关系的随机事件而言的。对于两个随机事件 A 和 B 而言，在 A 发生的情况下 B 发生的概率，那记住 $P(B|A)$，即 AB 同时发生的概率除以 A 发生的概率：

$$P(B|A)=\frac{P(AB)}{P(A)}$$

同理，在 B 发生的情况下 A 发生的概率，那记住 $P(A|B)$：1

$$P(A|B)=\frac{P(AB)}{P(B)}$$



Ω 表示样本空间，即表示全部事件。

举例说明，已知家庭中有两个孩子，其中一个是女孩，问这时另一个孩子也是女孩的概率是多少？

$$\Omega=\{(\text{男孩},\text{男孩}),(\text{男孩},\text{女孩}),(\text{女孩},\text{男孩}),(\text{女孩},\text{女孩})\}$$

$$\mathrm{A}=\{\text{已知一个是女孩}\}=\{(\text{男孩},\text{女孩}),(\text{女孩},\text{男孩}),(\text{女孩},\text{女孩})\}$$

$$\mathrm{AB}=\{\text{另一个也是女孩}\}=\{(\text{女孩},\text{女孩})\}$$

那么，其中一个是女孩，这时另一个孩子也是女孩的概率是：

$$\mathrm{P(B|A)}=\frac{\mathrm{P(AB)}}{\mathrm{P(A)}}=\frac{1}{3}$$

3.2、贝叶斯公式

根据条件概率公式，可以推导出贝叶斯公式：

$$P(B)P(A|B)=P(AB)=P(A)P(B|A)$$

$$P(A|B)=\frac{P(A)P(B|A)}{P(B)}$$

贝叶斯公式得到的结果是后验概率，后验概率是指依据得到"结果"信息所计算出的最有可能是那种事件发生，如贝叶斯公式中的，是"执果寻因"问题中的"因"。

举例说明：餐桌上有一块肉和一瓶醋，你如果吃了一块肉，很酸，那你觉得肉里加了醋的概率有多大？你说：80%可能性加了醋。OK，你已经进行了一次后验概率的猜测，没错，就这么简单！这就是"执果寻因"。

贝叶斯公式在整个机器学习和深度学习中是非常有用的，因为很多时候我们要用一种叫做最大化后验概率（Maximum a posteriori estimation，简称MAP）的思想。

4、随机事件独立性

说白了就是两个事情是不相关的， B 在 A 发生的条件下发生的概率是等于 B 本身发生的概率

$$P(B|A)=P(B)$$

$$P(AB)=P(A)P(B)$$

我们可以把它推广到 n 个事件相互独立的情况上面去，就是等于各自发生概率的乘积：

$$P(a_1,a_2,\cdots,a_n)=\prod_{i=1}^{n}P(a_i)$$

生活实例：

比如说生孩子，生男生女的概率都是$\frac{1}{2}$，这个事件就是相互独立事件。

第一胎生女孩定义为事件 A 概率为 $P(A)=\frac{1}{2}$，第二胎生男孩定义为事件 B，则 $P(B|A)=P(B)=\frac{1}{2}$。

第一胎生了女孩第二胎生了男孩的概率为：

$$P(AB)=P(A)P(B)=\frac{1}{2}\times\frac{1}{2}=\frac{1}{4}$$

5、随机变量

整个概率论的核心。变量是什么呢?

我们中学的时候就学习过变量了，它是取值可以变化的量，比如可以取 0 到 1 区间上所有的实数，或者取从 1 到 100 之间的整数。

5.1、离散随机变量

随机变量是什么呢?

就是变量取值都有一个概率。第一种情况是离散型的随机变量，比如前面说的抛硬币正面朝上还是反面朝上，这是两个事件，我们可以把这两个事件编号，得到：

$$\mathbf{x}=\begin{cases}0, & \text{正面}\\ 1, & \text{反面}\end{cases}$$

这就是随机变量，x 取值有两种情况，0 或 1，取每种值的概率都是 0.5，离散型的随机变量它的取值只可能是有限可能个，像掷色子，就有 6 中可能（1、2、3、4、5、6），取每种值的概率都是 $\frac{1}{6}$；或者无穷可列个，比如，1 到 $+\infty$，虽然是无穷个，但是一定可以用整数编号表示。

5.2、连续随机变量

连续型的随机变量，理解起来抽象一些，它的取值是无限不可列个，比如 0 到 1 之间的所有的实数，首先它肯定是无限个，而它比无限可列个更高级，它不可列，比如其中的 0.001 到 0.002 之间还是有无限个，不管怎么细分，a 和 b 之间还是有无限个，这就是连续型的随机变量，比如说抛石子在 0 到 1 的矩形范围内，它可能落在区域内任何一个位置，那么石子落在的位置 x,y 就是连续型随机变量，说白了就是它坐标取 0 到 1 之间任何一个值都是有可能的。对于离散型随机变量，写成如下：

$$P(x=x_i)=p_i$$

$$p_i\geq 0$$

$$\sum p_i=1$$

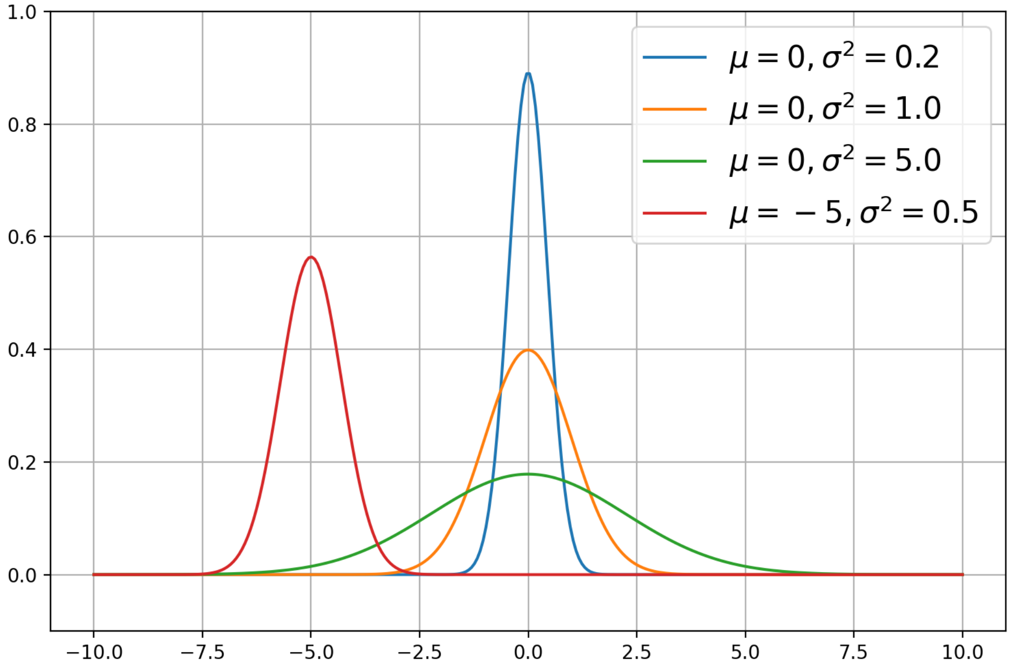
对于连续型随机变量我们是这么定义的，利用它的概率密度函数来定义

$$f(x)\geq 0$$

$$\int_{-\infty}^{+\infty}f(x)dx=1$$

通过概率密度函数可以计算事件发生的概率 P(y)，注意对于连续随机变量，往往计算区间的概率：

$$P(y) = P(x \le y) = \int_{-\infty}^{y} f(x)dx$$



需要注意的是，连续性随机变量它取某个具体值的概率是 0 的，$p(x = x_i) = 0$ 但是它落在某个区间范围的概率是有值的，因为算的是面积，就好像前面提到的石子落到区域内的面积：

$$\int_{x_1}^{x_2} f(x)dx = F(x_2) - F(x_1)$$

5.3、概率密度函数概率计算

```
'''
横轴区间（μ-σ,μ+σ）内的面积为68.268949%
横轴区间（μ-2σ,μ+2σ）内的面积为95.449974%
横轴区间（μ-3σ,μ+3σ）内的面积为99.730020%
'''
import numpy as np
import matplotlib.pyplot as plt
from scipy import integrate
def f(x,sigma,u):
    return 1/(np.sqrt(2*np.pi) * sigma) * np.exp(-(x - u)**2/(2 * sigma**2))
x = np.linspace(-10,10,300)
y = f(x,1,0)
plt.plot(x,y)
print('横轴区间（μ-σ,μ+σ）内的面积（概率）为：',np.round(integrate.quad(f,-1,1,args=(1,0))[0],4))
print('横轴区间（μ-2σ,μ+2σ）内的面积（概率）为：',np.round(integrate.quad(f,-2,2,args=(1,0))[0],4))
print('横轴区间（μ-3σ,μ+3σ）内的面积（概率）为：',np.round(integrate.quad(f,-3,3,args=(1,0))[0],4))
```

6、数学期望与方差

6.1、期望

这个在学概率论的时候同学们都是学过的，这是核心概念之一。什么是数学期望，从均值开始看起：

$$X = (x_1, x_2, \cdots, x_n)$$

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

这里的 $\frac{1}{n}$ 可以看作是每个样本 x_i 的权重，或者叫概率，如果把它替换称概率 p_i，就得到了我们的数学期望。

举个例子，比如说买彩票有 0.1 的概率中 500 万，0.3 的概率中 200 万，0.6 的概率中 50 万，那可能的收益不能用 (500+200+50)/3 = 250 来平均一下，肯定要考虑各自的概率值：

$$E=500\times0.1+200\times0.3+50\times0.6=140$$

说白了，对于离散型的随机变量而言，数学期望就是概率意义的**平均值。**

对于连续型的随机变量，把它推广一下变成定积分，求一个广义积分就是数学期望

$$E(X)=\sum x_ip(x_i)$$

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

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import integrate
def f(x,sigma,u):
    return 1/(np.sqrt(2*np.pi) * sigma) * np.exp(-(x - u)**2/(2 * sigma**2))
x = np.linspace(-10,10,300)
y = f(x,2,2.5)
plt.plot(x,y)
# 计算数学期望函数
def E(x,sigma,u):
    return x * f(x,sigma,u)
print('不定积分计算正太分布数学期望：',np.round(integrate.quad(E,-100,100,args=(2,2.5))[0],1))
```

6.2、方差

方差反应的数据的波动程度的，就是它和均值，我们的数学期望偏离程度的平均。这里每个数据减去期望的平方，不平方的话正负抵消掉了，然后再乘以 P 概率值

$$D(X)=\frac{1}{n}\sum_{i=1}^{n}(x_i-E(X))^2=\sum_{i=1}^{n}(x_i-E(X))^2p(x_i)$$

$$D(X)=\int_{-\infty}^{+\infty}(x_i-E(X))^2f(x)dx$$

离散性随机变量：

```
import numpy as np
X = np.random.randint(-10,10,size = 200)
print('Numpy库提供的函数计算方差：%0.2f'%(np.var(X)))
# X.mean为期望
var = ((X - X.mean())**2).sum()/200
print('根据公式计算的方差为：%0.2f'%(var))
```

连续型随机变量：

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy import integrate
def f(x,sigma,u):
    return 1/(np.sqrt(2*np.pi) * sigma) * np.exp(-(x - u)**2/(2 * sigma**2))
x = np.linspace(-10,10,300)
y = f(x,2,2.5)
plt.plot(x,y)
# 计算数学期望函数
def D(x,sigma,u):
    return (x - 2.5)**2 * f(x,sigma,u)
print('不定积分计算正太分布方差是：',np.round(integrate.quad(D,-100,100,args=(2,2.5))[0],1))
```

6.3、重要公式

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

$$D(X) = \frac{1}{n}\sum_{i=1}^{n}(x_i - E(X))^2 = \sum_{i=1}^{n}(x_i - E(X))^2 p(x_i)$$

上式中的 $(x - E(x))^2$ 化简如下。利用数学期望的线性性质：

$$E(a + bX) = a + bE(x)$$

$$E(x) = \mu$$

$$D(x) = \sum_{i=1}^{n}(x_i - E(X))^2 p(x_i) = E((X - E(X))^2) = E((X - \mu)^2)$$

$$E((X - \mu)^2) = E(X^2 - 2\mu X + \mu^2)$$

$$= E(X^2) - 2\mu E(X) + \mu^2$$

$$= E(X^2) - \mu^2$$

$$= E(X^2) - (E(X))^2$$

$$D(X) = Var(X) = E(X^2) - (E(X))^2$$

这是求方差时非常常用的一个公式！

7、随机向量

线性代数中，我们把标量 x 推广到向量，就是它有多个分量。

同样我们把单个随机变量可以推广到随机向量，就是它有多个分量，这样就有了随机向量的概念了，这是很自然的延申。

离散型的随机向量向量 X 取某一个具体的值为向量 X_i，然后取每一个向量值的概率都大于等于 0，所有的概率加起来要等于 1，符合这两个约束条件就可以了。

$$p(X = X_i) \geq 0$$

$$\sum_{i=1}^{n} p(X_i) = 1$$

连续型的随机向量，它是用 0 和概率密度函数来描述的，n 重积分等于 1，相当于体积等于 1。

$$f(x) \geq 0$$

$$\int\int\int f(x)dx = 1$$

下面是二维的随机向量:

$$f(x_1, x_2) \geq 0$$

$$\int_{-\infty}^{+\infty} f(x_1, x_2)dx_1dx_2 = 1$$

8、随机变量独立性

两个随机变量如果相互独立的话，它们的联合概率密度函数等于它们的分别的概率密度函数乘积

推广到多个随机变量相互独立

$$f(x_1, x_2, \cdots, x_n) = f(x_1)f(x_2)\cdots f(x_n) = \prod_{i=1}^{n} f(x_i)$$

这和随机事件的形式上是统一的，$f(x)$ 换成符号 $p(x)$ 就可以了。

9、协方差

协方差是对于方差的推广，对于两个随机变量，它们的协方差是反应它们两个之间的线性相关程度的，把 x_2 换成 x_1 那就是方差了，展开之后就是 x_1 和 x_2 的期望减去它们期望的乘积。

方差公式:

$$E((X - E(X))^2) = E((X - \mu)^2)$$

$$D(X) = Var(X) = E(X^2) - (E(X))^2$$

协方差公式:

$$cov(x_1, x_2) = E((x_1 - E(x_1))(x_2 - E(x_2)))$$

$$cov(x_1, x_2) = E(x_1x_2) - E(x_1)E(x_2)$$

对于 n 维的向量 X，它的协方差就构成了一个协方差矩阵，第一行第一个是 x_1 和 x_1 的协方差（即 x_1 自身方差），第一行第二个是 x_1 和 x_2 的协方差，第一行第 n 个是 x_1 和 x_n 的协方差。

$$\begin{bmatrix} x_1x_1, & x_1x_2, & \cdots, & x_1x_n \\ x_2x_1, & x_2x_2, & \cdots, & x_2x_n \\ \vdots, & \vdots, & \ddots, & \vdots \\ x_nx_1, & x_nx_2, & \cdots, & x_nx_n \end{bmatrix}$$

显然这是一个对称阵，这在我们机器学习里面会经常使用的！

```
import numpy as np
X = np.random.randint(1,20,size = (5,5))
display(np.cov(X,rowvar=False,bias = True))
print('第一行第一个协方差：%0.2f'%(np.mean(X[:,0]**2) - (np.mean(X[:,0]))**2))
print('第一行第二个协方差：%0.2f'%(np.mean(X[:,0] * X[:,1])- X[:,0].mean()*X[:,1].mean()))
```

```
print('第一行最后一个协方差：%0.2f'%(np.mean(X[:,0]*X[:,-1]) - (np.mean(X[:,0]))*(np.mean(X[:,-1]))))
'''
array([[ 32.24,   4.76, -19.2 ,  -7.6 ,  -8.4 ],
       [  4.76,  44.24,  19.6 ,  19.4 ,  -6.6 ],
       [-19.2 ,  19.6 ,  31.2 ,  17.6 ,   9.  ],
       [ -7.6 ,  19.4 ,  17.6 ,  24.4 ,   4.6 ],
       [ -8.4 ,  -6.6 ,   9.  ,   4.6 ,   9.2 ]])
第一行第一个协方差：32.24
第一行第二个协方差：4.76
第一行最后一个协方差：-8.40
'''
```

10、机器学习中常见分布

正太分布

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$ 概率密度函数

均匀分布

$$f(x) = \begin{cases} \frac{1}{b-a} & a \le x \le b \\ 0 & x < a, x > b \end{cases}$$

二项分布

$$P(x=1) = p$$

$$P(x=0) = 1-p$$

自然界和生活中很多事件都服从正太分布的，或者近似的服从正太分布的，比如人的身高、体重和智商，大部分人是平均值，小部分人比较胖，或比较瘦；比较高，或比较矮；比较愚钝，或比较聪明。还有考试成绩啊，人的收入等这些都近似服从正太分布。

二项分布拿我们抛硬币的例子来说，比如 x = 0 是背面朝下的概率，x = 1 是正面朝上的概率，那么它取值只有 0 或 1 两种情况。当然不用 0 或 1，你用 -1 和 +1 也是可以的。都是二项分布，取每个值都有一个概率值。在我们机器学习中，主要用的就是这几种概率分布

11、最大似然估计

最大似然是估计（求解）一个概率密度函数中参数问题的。比如有个向量 X，θ 是它的参数，比如正太分布中的 μ 和 σ 这都是需要估计的参数。

$$p(X;\theta)$$

那我们怎么估计这组参数呢？肯定是根据一组样本来学习，假设我们有 n 个样本它们是独立同分布的，也就是说它们服从同样一个概率分布，并且它们之间相互独立的，抽样出来的

$$x_i \ ; \ i = 1, 2, \cdots, n$$

那么所有变量发生的概率就可以写成它们乘积的形式，因为它们之间是相互独立的嘛，这时 L 是似然函数，这里 x 是已经取了具体样本的值了，θ 是我们要估计的参数

$$L(\theta) = \prod_{i=1}^{n} p(x_i;\theta)$$

既然这组样本是已经抽样抽出来的，是已经发生的，我们肯定要把它发生的概率最大化，也就是说要最大化这样一个似然函数

$$max \prod_{i=1}^{n} p(x_i;\theta)$$

求解一个函数的极值，就是要求解它的导数，也就是梯度等于 0

$$\nabla L(\theta)=0$$

而这样的乘积形式求导是不容易的（多个累乘，就更加麻烦！），之前讲过导数求导公式

$$(fg)'=f'g+fg'$$

如果更多项展开是非常麻烦的，所以我们可以对函数取对数，因为对数函数是单调增函数，所以求原函数的极值，也等于求它的对数形式的极值，所以我们两边取对数的话，就把连乘的形式转化为了连加的形式，因为连加的形式的导数，就是等于导数的连加

$$\mathrm{ln(ab)=lna+lnb}$$

$$lnL(\theta)=ln\prod_{i=1}^{n}p(x_i;\theta)=\sum_{i=1}^{n}lnp(x_i;\theta)$$

所以我们要解决的问题就是求这个函数的极大值，这个可以对 θ 求导让它等于 0 得到，带 log 的是对数似然函数，这就是最大似然函数最基本的思想

$$max\sum_{i=1}^{n}lnp(x_i;\theta)$$

如果数据符合正太分布，那么通过最大似然可以推导出线性回归的损失函数 MSE （最小二乘法）：

$$J(\theta)=\frac{1}{2}\sum_{i=1}^{n}(h_\theta(x^{(i)})-y^{(i)})^2$$

大家可以参看，老师之前讲解的内容《**多元线性回归**》线性回归算法推导!