

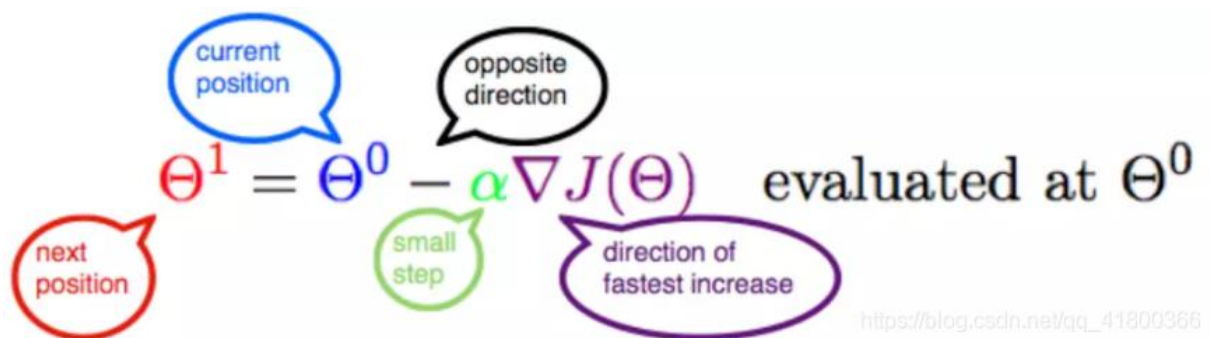
深度学习之数学基础

高数部分

梯度下降算法

1、概述

公式：



The diagram illustrates the gradient descent formula: $\Theta^1 = \Theta^0 - \alpha \nabla J(\Theta)$ evaluated at Θ^0 . Annotations include: 'current position' for Θ^0 , 'next position' for Θ^1 , 'small step' for α , 'opposite direction' for the minus sign, and 'direction of fastest increase' for $\nabla J(\Theta)$. A URL https://blog.csdn.net/qq_41800366 is visible at the bottom right.

BP 算法：BP 算法即是通过链式法则对复合函数求各个参数的偏导数，再通过梯度下降更新参数

核心代码：

```
xs_ = [i/100 for i in range(100)]
ys_ = [3*e+4+random.random()/10 for e in xs_]
w = random.random()
b = random.random()
lr = 0.01

for i in range(100000):
    for x,y in zip(xs_,ys_):
        h = w*x + b
        o = h - y
        loss = o ** 2

        dw = 2 * o * x
        db = 2 * o
```

$$w = w - lr * dw$$
$$b = b - lr * db$$

2、分类

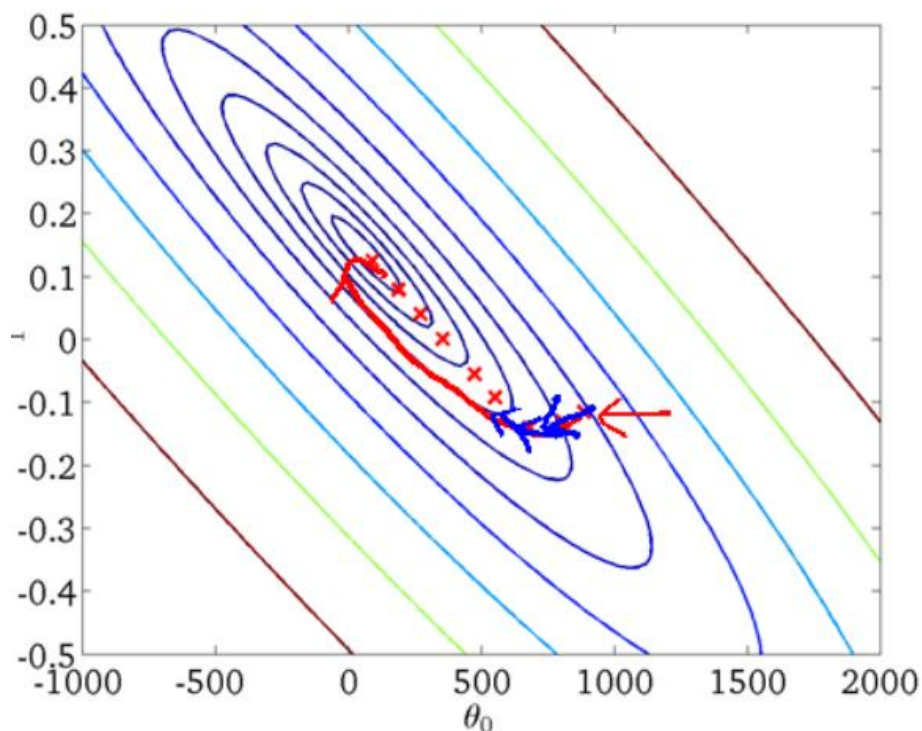
1) 批量梯度下降法 BGD

批梯度下降法(Batch Gradient Descent)针对的是整个数据集，通过对所有的样本的计算来求解梯度的方向。每迭代一步，都要用到训练集所有的数据，如果样本数目很大，那么可想而知这种方法的迭代速度！

优点：全局最优解；易于并行实现；

缺点：当样本数目很多时，训练过程会很慢。

从迭代的次数上来看，BGD 迭代的次数相对较少。其迭代的收敛曲线示意图可以表示如下：



2) 小批量梯度下降法 MBGD

在上述的批梯度的方式中每次迭代都要使用到所有的样本，对于数据量特别大的情况，如大规模的机器学习应用，每次迭代求解所有样本需要花费大量的计算成本。是否可以在每次的迭代过程中利用部分样本代替所有的样本呢？基于这样的思想，便

出现了 mini-batch 的概念。

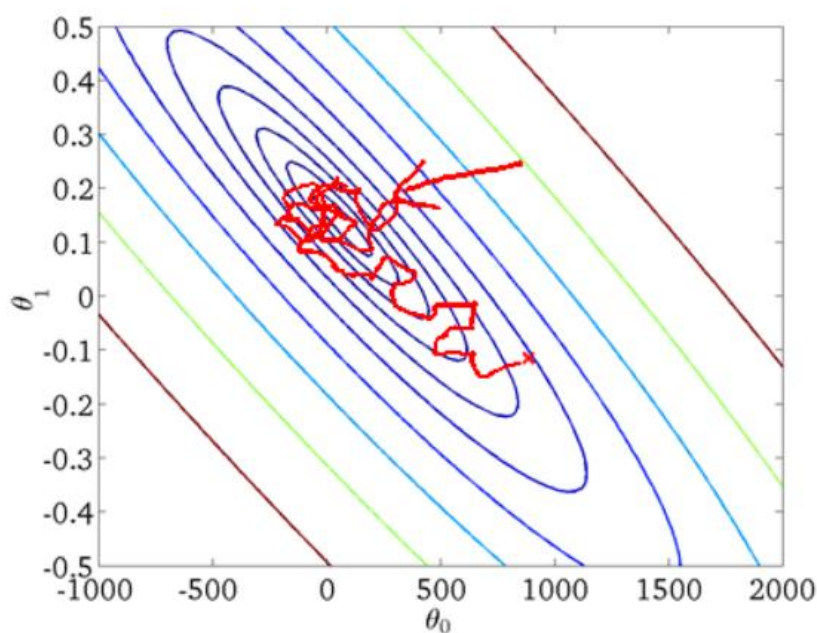
3) 随机梯度下降法 SGD

随机梯度下降算法(stochastic gradient descent)即每个 mini-batch 中只有一个训练样本。随机梯度下降是通过每个样本来迭代更新一次，如果样本量很大的情况（例如几十万），那么可能只用其中几万条或者几千条的样本，就已经将 θ 迭代到最优解了，对比上面的批量梯度下降，迭代一次需要用到十几万训练样本，一次迭代不可能最优，如果迭代 10 次的话就需要遍历训练样本 10 次。但是，SGD 伴随的一个问题是噪音较 BGD 要多，使得 SGD 并不是每次迭代都向着整体最优化方向。

优点：训练速度快；

缺点：准确度下降，并不是全局最优；不易于并行实现。

从迭代的次数上来看，SGD 迭代的次数较多，在解空间的搜索过程看起来很盲目。其迭代的收敛曲线示意图可以表示如下：



最优化问题

1、 梯度下降

2、 最小二乘法

概念：最小二乘法（又称最小平方法）是一种数学优化技术。它通过最小化误差的平方和寻找数据的最佳函数匹配。利用最小二乘法可以简便地求得未知的数据，并使得这些求得的数据与实际数据之间误差的平方和为最小。

公式推导：

y' 为预测值, y 为实际值, w 为参数

$$y' = w_0 + w_1x + w_2x \dots$$

M 为各点平方误差之和

$$M = \sum_{i=1}^n (y_i - y_i')^2 = \sum_{i=1}^n (y_i - w_0 - w_1x_{i1} - w_2x_{i2} \dots)^2$$

最小化误差 M , M 对各个 w 求偏导

$$\begin{cases} \frac{\partial M}{\partial w_0} = -2 \sum_{i=1}^n (y_i - w_0 - w_1x_{i1} - w_2x_{i2} \dots) = 0 \\ \frac{\partial M}{\partial w_j} = -2 \sum_{i=1}^n (y_i - w_0 - w_1x_{i1} - w_2x_{i2} \dots) x_{ij} = 0 \end{cases}$$

右边展开

$$\begin{cases} nw_0 + \sum x_{i1}w_1 + \sum x_{i2}w_2 + \dots = \sum y_i \\ \sum x_{i1}w_0 + \sum x_{i1}^2w_1 + \sum x_{i1}x_{i2}w_2 + \dots = \sum x_{i1}y_i \\ \dots \\ \sum x_{ip}w_0 + \sum x_{ip}x_{i1}w_1 + \sum x_{ip}x_{i2}w_2 + \dots = \sum x_{ip}y_i \end{cases}$$

写成矩阵

$$X'XW = X'Y$$

$$W = (X'X)^{-1}X'Y$$

<http://blog.csdn.net/liuqiao18434391822>

3、 牛顿法

牛顿法

1. 求 $f(x)=0$ 的根 (x 可以是向量)
2. 初始化离零点较近的 x_0 ，在 $x=x_0$ 处做曲线的切线

$$\frac{f(x)-f(x_0)}{x-x_0}=\nabla f(x_0) \quad (1)$$

3. 因为 $f(x)=0$ ，所以：

$$x=x_0-\frac{f(x_0)}{\nabla f(x_0)} \quad (2)$$

4. 迭代方法为：

$$x_{n+1}=x_n-\frac{f(x_n)}{\nabla f(x_n)} \quad (3)$$

牛顿法用于优化问题

一、算法

求凸优化的极小值时， $f'(x)=0$ ，代入上式得：

$$x_{n+1}=x_n-\frac{\nabla f(x_n)}{\nabla^2 f(x_n)} \quad (4)$$

$\nabla^2 f(x_n)$ 是 hessian 矩阵。

另一种理解方法也是一样求 $f'(x)=0$ ，只不过用的是泰勒公式：

$$f(x)=f(x_0)+\nabla f(x_0)(x-x_0)+\frac{\nabla^2 f(x_0)(x-x_0)^2}{2} \quad (5)$$

对上式求导

$$\nabla f(x)=\nabla f(x_0)+\nabla^2 f(x_0)(x-x_0) \quad (6)$$

并令其为 0，则：

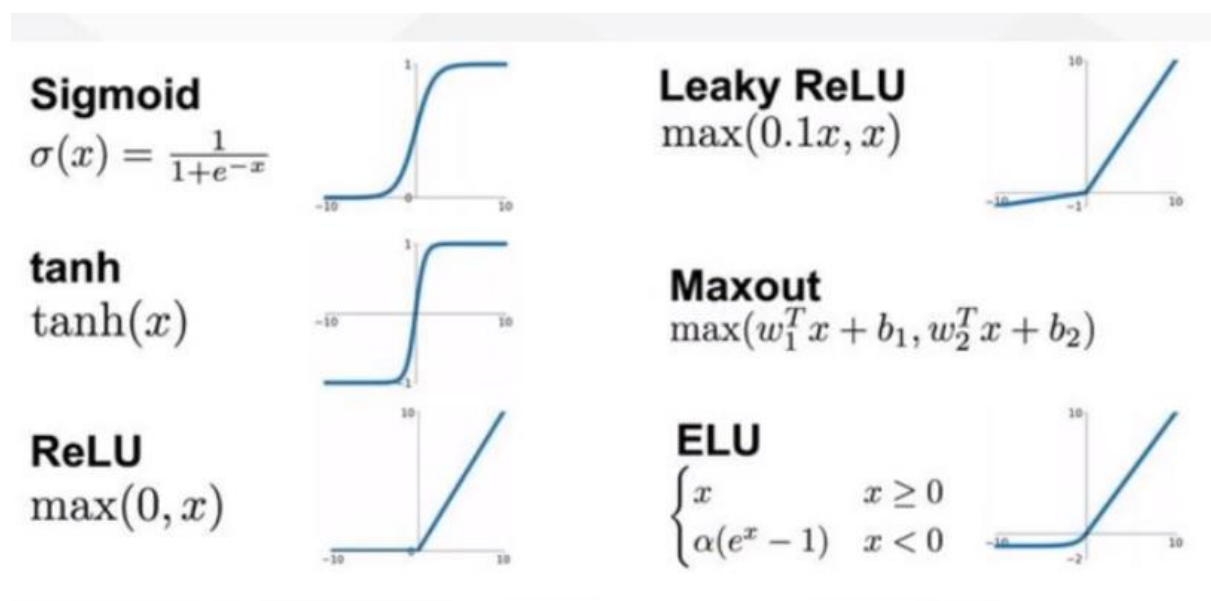
$$x=x_0-\frac{\nabla f(x_0)}{\nabla^2 f(x_0)} \quad (7)$$

二、优点

4、 有约束条件的最优化问题

拉格朗日乘子法

重要函数



线代部分

理解过拟合


用线代的方式理解过拟合即是方程数代表数据量，未知数代表参数量，第一个式子两方程线性相关即参数量多方程量少有无穷组解，在深度学习中便是过拟合；第二个式子方程量和参数量匹配，则为拟合状态；第三个式子，方程量多，参数量少，即数据量太多、模型参数太少，无解。

$$\begin{aligned} 2x + y &= 4 \\ 4x + 2y &= 8 \end{aligned}$$

$$\begin{aligned} 2x + y &= 4 \\ 4x + 3y &= 9 \end{aligned}$$

$$\begin{aligned} 2x + y &= 4 \\ 4x + 3y &= 9 \\ 5x + 2y &= 7 \end{aligned}$$

范数

$$\|x\|_p = \left(\sum_i |x_i|^p \right)^{\frac{1}{p}}$$


$p = \infty$ $p = 2$ $p = 1$ $0 < p < 1$ $p = 0$

上图表示了 p 从无穷到 0 变化时，三维空间中到原点的距离（范数）为 1 的点构成的图形的变化情况。以常见的 L-2 范数（ $p=2$ ）为例，此时的范数也即欧氏距离，空间中到原点的欧氏距离为 1 的点构成了一个球面。

几种特殊范数：

L-0 范数：用来统计向量中非零元素的个数。

L-1 范数：向量中所有元素的绝对值之和。可用于优化中去除没有取值的信息，又称稀疏规则算子。L1 范数又称曼哈顿距离、最小绝对误差。

L-2 范数：典型应用——欧式距离。可用于优化正则化项，避免过拟合。

L- ∞ 范数：计算向量中的最大值。

常见距离汇总

1、闵可夫斯基距离

闵可夫斯基距离 (Minkowski distance) 是衡量数值点之间距离的一种非常常见的方法，是有Lp范数引申而来，假设数值点 P 和 Q 坐标如下：

$$P = (x_1, x_2, \dots, x_n) \text{ and } Q = (y_1, y_2, \dots, y_n) \in \mathbb{R}^n$$

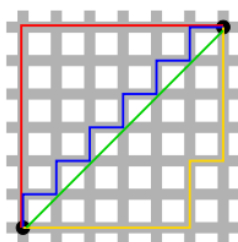
那么，闵可夫斯基距离定义为：

$$\left(\sum_{i=1}^n |x_i - y_i|^p \right)^{1/p}.$$

p=2是欧几里得距离 (Euclidean distance)

p=1是曼哈顿距离 (Manhattan distance)

假设在曼哈顿街区乘坐出租车从 P 点到 Q 点，白色表示高楼大厦，灰色表示街道：



绿色的斜线表示欧几里得距离，在现实中是不可能的。其他三条折线表示了曼哈顿距离，这三条折线的长度是相等的。

当 $p \rightarrow \infty$ 时，闵可夫斯基距离转化成切比雪夫距离 (Chebyshev distance)：

$$\lim_{p \rightarrow \infty} \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}} = \max_{i=1}^n |x_i - y_i|.$$

2、余弦相似度

做过二范数归一化后的欧氏距离等价于余弦相似度。

$$\text{CosSim}(x, y) = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2} \sqrt{\sum_i y_i^2}} = \frac{\langle x, y \rangle}{\|x\| \|y\|}$$

余弦相似度与向量的幅值无关，只与向量的方向相关，需要注意一点的是，余弦相似度受到向量的平移影响，上式如果将 x 平移到 $x+1$ ，余弦值就会改变。怎样才能实现平移不变性？这就是下面要说的皮尔逊相关系数 (Pearson correlation)

3、皮尔逊系数（相关系数）

$$Corr(x, y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \sqrt{\sum (y_i - \bar{y})^2}} = \frac{\langle x - \bar{x}, y - \bar{y} \rangle}{\|x - \bar{x}\| \|y - \bar{y}\|} = CosSim(x - \bar{x}, y - \bar{y}) \quad (1)$$

皮尔逊系数又称相关系数，皮尔逊相关系数具有平移不变性和尺度不变性，计算出了两个向量（维度）的相关性。不过，一般我们在谈论相关系数的时候，将 x 与 y 对应位置的两个数值看作一个样本点，皮尔逊系数用来表示这些样本点分布的相关性。

$$\rho_{xy} = \frac{Cov(X, Y)}{\sqrt{D(X)} \sqrt{D(Y)}} = \frac{E((X - EX)(Y - EY))}{\sqrt{D(X)} \sqrt{D(Y)}}$$

5、KL 散度和卡方

在统计学里面经常需要测量两分布之间的距离，进而判断出它们是否出自同一个 population，常见的方法有卡方检验（Chi-Square）和 KL 散度（KL-Divergence）

KL 散度介绍：

信息量：

$$h(x) = -\ln p(x)$$

熵：对假设一个发送者要将随机变量 X 产生的一长串随机值传送给接收者，接受者获得的平均信息量就是求它的数学期望

$$H[x] = -\sum_x p(x) \ln p(x)$$

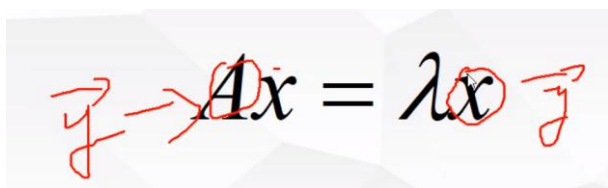
KL 散度：熵的大小与字符平均最短编码长度是一样的（shannon）。设有一个未知的分布 $p(x)$ ，而 $q(x)$ 是我们所获得的一个对 $p(x)$ 的近似，按照 $q(x)$ 对该随机变量的各个值进行编码，平均长度比按照真实分布的 $p(x)$ 进行编码要额外长一些，多出来的长度这就是 KL 散度（之所以不说距离，是因为不满足对称性和三角形法则）

$$\begin{aligned} KL(p||q) &= -\int p(\mathbf{x}) \ln q(\mathbf{x}) d\mathbf{x} - \left(-\int p(\mathbf{x}) \ln p(\mathbf{x}) d\mathbf{x} \right) \\ &= -\int p(\mathbf{x}) \ln \left\{ \frac{q(\mathbf{x})}{p(\mathbf{x})} \right\} d\mathbf{x}. \end{aligned}$$

特征方程

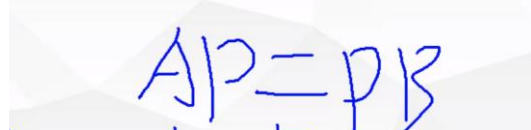
$$Ax = \lambda x$$

特征方程含义是对一个向量 X 作一系列变化等于对这个向量作一次变化 (λ 可为实数或复数), 广泛用于结果导向的过程检测。


$$\vec{x} \rightarrow Ax = \lambda \vec{x}$$

另一种理解, 输入向量 Y 进过变换矩阵 A 进行一系列变换然后在向量 X 上的投影等于输入向量 Y 直接在向量 X 上的投影。此理论在生成模型中有体现 (自编码, GAN)。

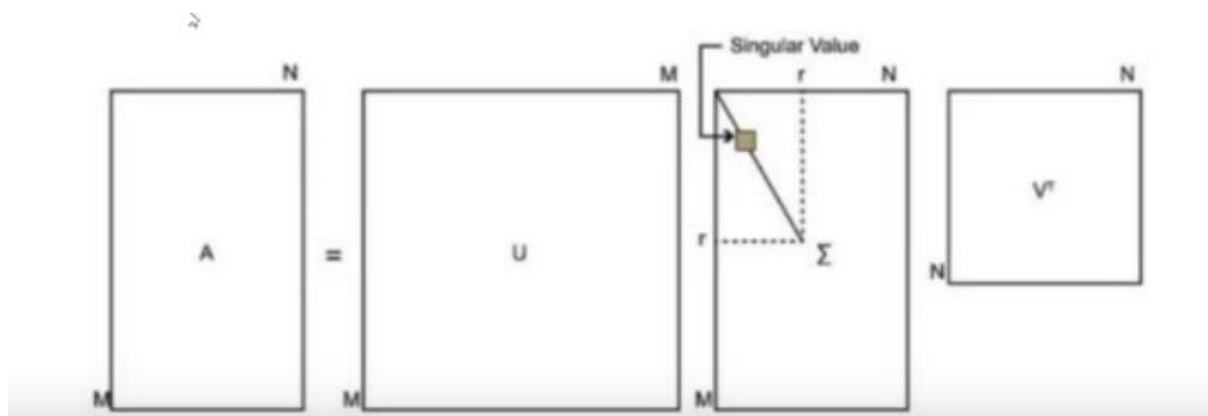
相似矩阵

$$P^{-1}AP = B$$

$$AP = PB$$

A 和 B 为相似矩阵, P 为特征矩阵, B 为对角阵

奇异值分解

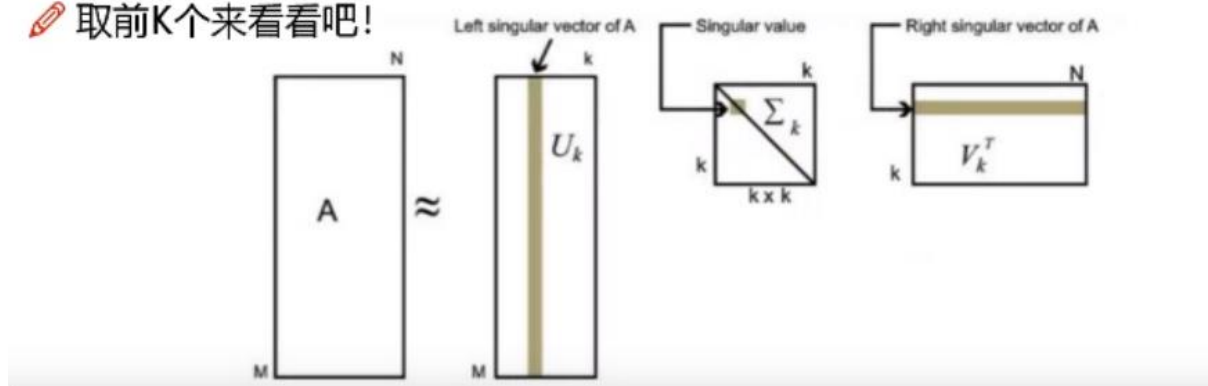
作用: 奇异值分解主要是为了让我们得到一个原始矩阵的近似, 若原始矩阵较大, 也可起到降维的作用



中间的矩阵为特征值矩阵也是我们最终所需的近似矩阵，存放的是数据的特征值

📌 照样按照特征值的大小来进行筛选，一般前10%的特征值（甚至更少）的和就占到了总体的99%了。

📌 取前K个来看看吧！



概率与统计

基本概念

- 随机事件：在随机试验中，可能出现也可能不出现，而在大量重复试验中具有某种规律性的事件叫做随机事件(简称事件)。
 - 样本点：随机试验中的每一个可能出现的试验结果称为这个试验的一个样本点。
 - 样本空间：全体样本点组成的集合称为这个试验的样本空间 Ω
 - 必然事件：样本空间 Ω 也是其自身的一个子集， Ω 也是一个“随机”事件，每次试验中必定有 Ω 中的一个样本点出现，必然发生。
 - 不可能事件：记作 Φ ，空集 Φ 也是样本空间的一个子集， Φ 也是一个特殊的“随机”事件，不包含任何样本点，不可能发生。
 - 空集：空集是指不含任何元素的集合。空集是任何集合的子集，是任何非空集合的真子集。空集不是无；它是内部没有元素的集合。
 - 全集：如果一个集合含有我们所研究问题中涉及的所有元素，那么就称这个集合为全集（通常也把给定的集合称为全集），通常记作 U 。
-
- 包含：事件A是事件B的子事件，事件A发生必然导致事件B发生，事件A的样本点都是事件B的样本点，记作 $A \subset B$ ，也叫做A包含于B，或B包含A
 - 相等：若 $A \subset B$ 且 $B \subset A$ ，那么 $A=B$ ，称A和B为相等事件，事件A与事件B含有相同的样本点。
 - 和事件：即事件A发生或事件B发生，事件A与事件B至少一个发生，由事件A与事件B所有样本点组成，记作 $A \cup B$ 或 $A+B$ ，也叫做A和B的并集
 - 积事件：即事件A和事件B同时发生，由事件A与事件B的公共样本点组成，记作 AB 或 $A \cap B$ ，也叫做A和B的交集
 - 差事件：即事件A发生且事件B不发生，是由属于事件A但不属于事件B的样本点组成，记作 $A - B$ ，也叫做A和B的差集
 - 互斥事件：事件A与事件B， $AB = \Phi$ ，事件A与事件B不能同时发生，事件A与事件B没有公共的样本点。
 - 事件A的对立事件：事件A不发生，事件A的对立事件是由不属于事件A的样本点组成，记作 \bar{a} ，若在一个S集合中， \bar{a} 也可以叫做A的补集

古典概型

如果每个情况都等可能的出现，此时某一事件的概率为：

$P(A) = \text{事件 } A \text{ 包含的基本事件数} / \text{全部可能的基本事件数}$

$P(A) = \text{事件 } A \text{ 所占区域大小} / \text{样本空间所占区域大小}$

其中一维为线段的长度，二维为面积，三维为体积称为古典概型，也称为等可能概型。

条件概率、联合概率、边缘概率

- 条件概率：事件A在另外一个事件B已经发生条件下的发生概率，记为 $P(A|B)$
- 联合概率：在多元的概率分布中多个随机变量分别满足各自条件的概率，记为 $P(AB)$
- 边缘概率：在多元的概率分布中单个随机变量概率，记为 $P(A)$

Y \ X	X				$p_Y(Y)$
	x_1	x_2	x_3	x_4	
y_1	$\frac{4}{32}$	$\frac{2}{32}$	$\frac{1}{32}$	$\frac{1}{32}$	$\frac{8}{32}$
y_2	$\frac{2}{32}$	$\frac{4}{32}$	$\frac{1}{32}$	$\frac{1}{32}$	$\frac{8}{32}$
y_3	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{2}{32}$	$\frac{8}{32}$
y_4	$\frac{8}{32}$	0	0	0	$\frac{8}{32}$
$p_X(X) \rightarrow$	$\frac{16}{32}$	$\frac{6}{32}$	$\frac{4}{32}$	$\frac{4}{32}$	$\frac{32}{32}$

Joint and marginal distributions of a pair of discrete, random variables X, Y having nonzero mutual information $I(X; Y)$. The values of the joint distribution are in the 4x4 square, and the values of the marginal distributions are along the right and bottom margins.

https://blog.csdn.net/qq_37177324/article/details/79410400

贝叶斯和朴素贝叶斯

1、贝叶斯

公式：后验概率 = (似然概率 x 先验概率) / 全概率

$$p(y|x) = \frac{p(x|y) \times p(y)}{p(x)}$$

先验概率： $p(y)$

后验概率： $p(y|x)$

2、朴素贝叶斯

公式：在贝叶斯假设基础上，假设样本 X 中所有事件相互独立

$$P(y_i|x_1, x_2, \dots, x_d) = \frac{P(y_i) \prod_{j=1}^d P(x_j|y_i)}{\prod_{j=1}^d P(x_j)}$$

常见分布

大数定理和中心极限定理

大数定理：在随机事件的大量重复出现中，往往呈现几乎必然的规律，这个规律就是大数定律。通俗地说，这个定理就是，在试验不变的条件下，重复试验多次，随机事件的频率近似于它的概率。偶然中包含着某种必然。

中心极限定理：中心极限定理，是指概率论中讨论随机变量序列部分和分布渐近于正态分布的一类定理。最早的中心极限定理在伯努利试验中，事件 A 出现的次数渐近于正态分布的问题。

学派划分

划分标准：若 w 是采样出来的离散的值则为频率派，若 w 是一个连续的分布，则为贝叶斯派。

$$y = P(x; w)$$

频率派：神经网络

贝叶斯派：图模型（GCN）是个拓扑结构，贝叶斯网络，条件随机场

马尔科夫和隐马尔科夫

1、马尔科夫

概述：一个一阶马尔科夫模型，包含一组状态集合，两组概率集合。

状态：一个系统真实状态，如天气系统（假设只有晴天、雨天、阴天三种天气，则这三种天气就组成一个状态集合）

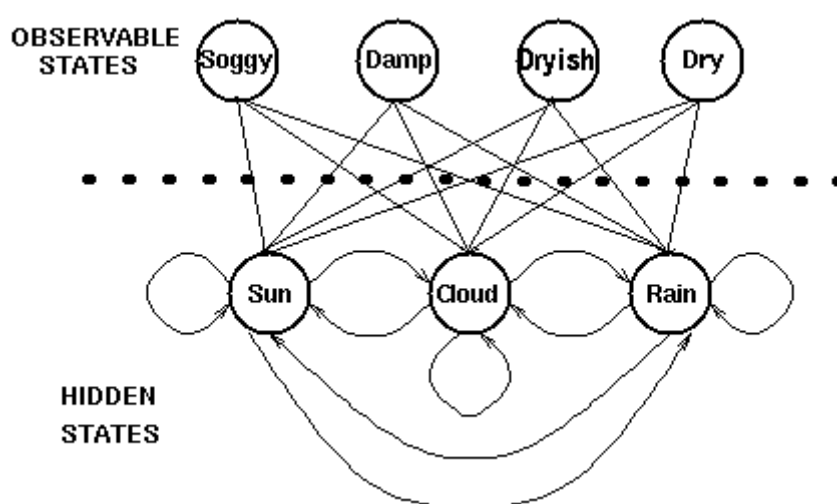
pi 向量：定义系统初始化时每一个状态的概率。

状态转移矩阵： M 个状态对应 M^2 个状态转移，即是 M 种状态都能够按一定概率转移到 M 个任意的状态中去，由此构成一个 M 阶状态转移矩阵。

任何一个可以用这种方式描述的系统都是一个马尔科夫过程。

马尔科夫过程的局限性：在某些情况下，我们希望找到的模式用马尔科夫过程描述还显得不充分。回顾一下天气那个例子，一个隐士也许不能够直接获取到天气的观察情况，但是他有一些水藻。民间传说告诉我们水藻的状态与天气状态有一定的概率关系——天气和水藻的状态是紧密相关的。我们希望为隐士设计一种算法，在不能够直接观察天气的情况下，通过水藻和马尔科夫假设来预测天气。需要着重指出的是，隐藏状态的数目与观察状态的数目可以是不同的。

2、 隐马尔可夫



模型介绍：

在图中，上面的状态是观察状态，下面的状态是隐藏状态。他们之间的连线，实际上应该从底下指向上面的观察状态：也就是说，这种连接表示，在给定的马尔科夫过程中，一个特定的隐藏状态生成特定的观察状态的概率。由此生成一个混淆矩阵，如下

		Seaweed			
		Dry	Dryish	Damp	Soggy
weather	Sun	0.60	0.20	0.15	0.05
	Cloud	0.25	0.25	0.25	0.25
	Rain	0.05	0.10	0.35	0.50

概述：

我们使用一个隐马尔科夫模型（HMM）对这些例子建模。这个模型包含两组状态集合

和三组概率集合：

* **隐藏状态**：一个系统的（真实）状态，可以由一个马尔科夫过程进行描述（例如，天气）。

* **观察状态**：在这个过程中‘可视’的状态（例如，海藻的湿度）。

* **pi 向量**：包含了（隐）模型在时间 $t=1$ 时一个特殊的隐藏状态的概率（初始概率）。

* **状态转移矩阵**：包含了一个隐藏状态到另一个隐藏状态的概率

* **混淆矩阵**：包含了给定隐马尔科夫模型的某一个特殊的隐藏状态，观察到的某个观察状态的概率。

因此一个隐马尔科夫模型是在一个标准的马尔科夫过程中引入一组观察状态，以及其与隐藏状态间的一些概率关系。

注意：在状态转移矩阵及混淆矩阵中的每一个概率都是与时间无关的——也就是说，当系统演化时这些矩阵并不随时间改变。实际上，这是马尔科夫模型关于真实世界最不现实的一个假设。

参考链接：[HMM 介绍-CSDN 博客](#)

蒙特卡罗方法

1、 概念

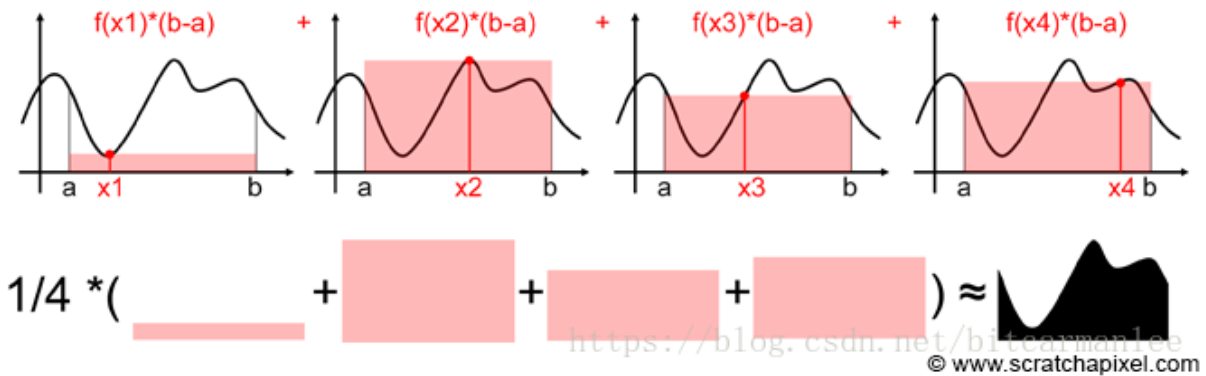
蒙特卡罗方法也称统计模拟方法，是 1940 年代中期由于科学技术的发展和电子计算机的发明，而提出的一种以概率统计理论为指导的数值计算方法。是指使用随机数（或更常见的伪随机数）来解决很多计算问题的方法。

2、 基本思想

通常蒙特卡罗方法可以粗略地分成两类：一类是所求解的问题本身具有内在的随机性，借助计算机的运算能力可以直接模拟这种随机的过程。另一种类型是所求解问题可以转化为某种随机分布的特征数，比如随机事件出现的概率，或者随机变量的期望值。通过随机抽样的方法，以随机事件出现的频率估计其概率，或者以抽样的数字特征估算随机变量的数字特征，并将其作为问题的解。这种方法多用于求解复杂的多维积分问题。

3、 应用

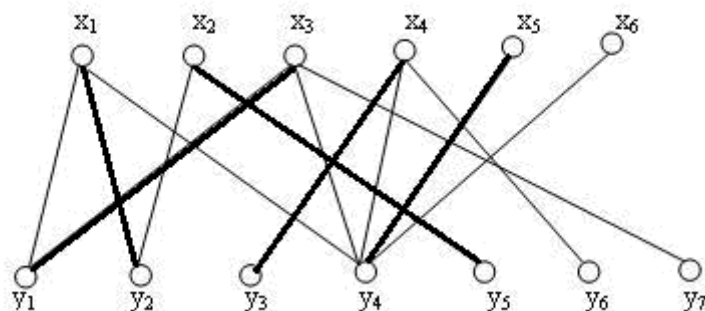
蒙特卡洛方法的一个重要应用就是求定积分。



积分中值定理其实是蒙特卡洛方法的一种特殊情况（正好取样到中点）

基本思想：求积分时，用一个长方形的面积来近似曲线下的面积，矩形的一条边就为 x 轴上的区间长度 $(b-a)$ ，而另一条边长计算方式为：在区间 (a, b) 中进行采样（可以是任意采样方法）一般选用均值分布来采样，采样 n 个点，计算每个点的 $f(n)$ 函数值，再对这些函数值求期望，便是另一条边的长度，由此便可计算矩形面积进而用来近似积分的值。

匈牙利算法



1. 匈牙利算法寻找最大匹配，就是通过不断寻找原有匹配 M 的增广路径，因为找到一条 M 匹配的增广路径，就意味着一个更大的匹配 M' ，其恰好比 M 多一条边。

2. 对于图来说，最大匹配不是唯一的，但是最大匹配的大小是唯一的。

3. 匈牙利算法其实就是一个后来者优先的递归算法，根据后来者优先的原则，寻找原匹配 M 的增广路径。

卡尔曼滤波

信息论