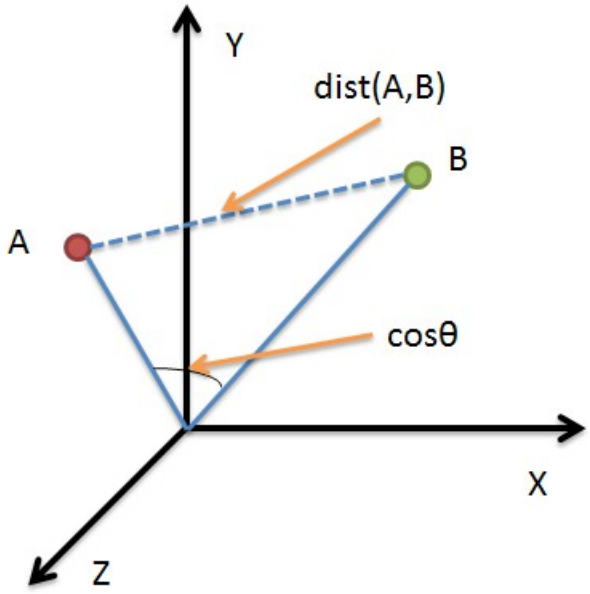
欧氏距离与余弦相似度

欧氏距离是最常见的距离度量，而余弦相似度则是最常见的相似度度量，很多的距离度量和相似度度量都是基于这两者的变形和衍生，所以下面重点比较下两者在衡量个体差异时实现方式和应用环境上的区别。

借助三维坐标系来看下欧氏距离和余弦相似度的区别：



从图上可以看出距离度量衡量的是空间各点间的绝对距离，跟各个点所在的位置坐标（即个体特征维度的数值）直接相关；而余弦相似度衡量的是空间向量的夹角，更加的是体现在方向上的差异，而不是位置。如果保持 A 点的位置不变，B 点朝原方向远离坐标轴原点，那么这个时候余弦相似度 cosθ 是保持不变的，因为夹角不变，而 A、B 两点的距离显然在发生改变，这就是欧氏距离和余弦相似度的不同之处。

根据欧氏距离和余弦相似度各自的计算方式和衡量特征，分别适用于不同的数据分析模型：欧氏距离能够体现个体数值特征的绝对差异，所以更多的用于需要从维度的数值大小中体现差异的分析，如使用用户行为指标分析用户价值的相似度或差异；而余弦相似度更多的是从方向上区分差异，而对绝对的数值不敏感，更多的用于使用用户对内容评分来区分用户兴趣的相似度和差异，同时修正了用户间可能存在的度量标准不统一的问题（因为余弦相似度对绝对数值不敏感）。

上面都是对距离度量和相似度度量的一些整理和汇总，在现实的使用中选择合适的距离度量或相似度度量可以完成很多的数据分析和数据挖掘的建模，后续会有相关的介绍。

欧氏距离和余弦相似度

时间 2013-07-15 23:39:59 CSDN 博客

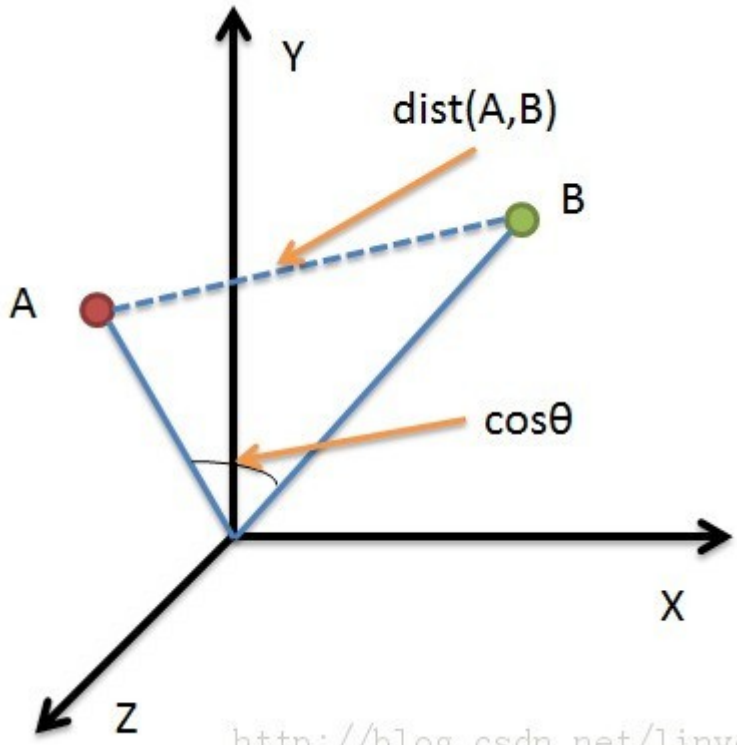
原文 http://blog.csdn.net/linvo/article/details/9333019

主题 向量

两者相同的地方，就是在机器学习中都可以用来计算相似度，但是两者的含义有很大差别，以我的理解就是：

前者是看成坐标系中两个 点 ，来计算两点之间的 距离 ；
后者是看成坐标系中两个 向量 ，来计算两向量之间的 夹角 。
前者因为是 点 ，所以一般指 位置 上的差别，即 距离 ；
后者因为是 向量 ，所以一般指 方向 上的差别，即所成 夹角 。

如下图所示：



数据项 A 和 B 在坐标图中当做点时，两者相似度为距离 dist(A,B)，可通过欧氏距离（也叫欧几里得距离）公式计算：

$$dist(X,Y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

当做向量时，两者相似度为 cosθ，可通过余弦公式计算：

$$\cos\theta = \frac{\sum_{i=1}^{n}(A_i \times B_i)}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} = \frac{A^T \cdot B}{\|A\| \times \|B\|}$$

假设||A||、||B||表示向量 A、B 的 2 范数，例如向量[1,2,3]的 2 范数为：

$\sqrt{(1^2+2^2+3^2)} = \sqrt{14}$

numpy 中提供了范数的计算工具： **linalg.norm()**

所以计算 cosθ 起来非常方便（假定 A、B 均为列向量）：

```
num = float(A.T * B) #若为行向量则 A * B.Tdenom = linalg.norm(A) *
linalg.norm(B)cos = num / denom #余弦值sim = 0.5 + 0.5 * cos #归一化
```

因为有了 linalg.norm()，欧氏距离公式实现起来更为方便：

```
dist = linalg.norm(A - B)sim = 1.0 / (1.0 + dist) #归一化
```

关于归一化：

因为余弦值的范围是 [-1,+1]，相似度计算时一般需要把值归一化到 [0,1]，一般通过如下方式：

sim = 0.5 + 0.5 * *cosθ*

若在欧氏距离公式中，取值范围会很大，一般通过如下方式归一化：

sim = 1 / (1 + *dist* (*X,Y*))

说完了原理，简单扯下实际意义，举个栗子吧：

例如某 T 恤从100 块降到了 50 块（A(100,50)），某西装从1000 块降到了 500 块（B(1000,500)）

那么T 恤和西装都是降价了 50%，两者的价格变动趋势一致，余弦相似度为最大值，即两者有很高的 **变化趋势相似度**

但是从商品价格本身的角度来说，两者相差了好几百块的差距，欧氏距离较大，即两者有较低的 **价格相似度**

-- EOF --