数学 统计学 机器学习

关注者 1,028 被浏览 181,815

如何理解皮尔逊相关系数（Pearson Correlation Coefficient）？

做计算似度的时候经常会用皮尔逊相关系数，那么应该如何理解该系数？其数学含义、本质是什么？

微调
机器学习、数据挖掘、人工智能 等 4 个话题的优秀回答者

提供一个机器学习方向的解释。**先上结论：在数据标准化（ $\mu=0,\sigma=1$ ）后，Pearson相关性系数、Cosine相似度、欧式距离的平方可认为是等价的。**换句话说，如果你的数据符合正态分布或者经过了标准化处理，那么这三种度量方法输出等价，不必纠结使用哪一种。**对于标准化后的数据求欧氏距离平方并经过简单的线性变化，其实就是Pearson系数 [1]，详见证明2。**

我个人觉得比较容易理解的步骤是：我们一般用欧式距离（向量间的距离）来衡量向量的相似度，但欧式距离无法考虑不同变量间取值的差异。举个例子，变量a取值范围是0至1，而变量b的取值范围是0至10000，计算欧式距离时变量b上微小的差异就会决定运算结果。**而Pearson相关性系数可以看出是升级版的欧氏距离平方，因为它提供了对于变量取值范围不同的处理步骤。因此对不同变量间的取值范围没有要求（unit free），最后得到的相关性所衡量的是趋势，而不同变量量纲上差别在计算过程中去掉了，等价于z-score标准化。**

而未经升级的欧式距离以及cosine相似度，对变量的取值范围是敏感的，在使用前需要进行适当的处理。我个人的经验是，**在低维度可以优先使用标准化后的欧式距离或者其他距离度量，在高维度时Pearson相关系数更加适合**。不过说到底，这几个衡量标准差别不大，很多时候的输出结果是非常相似的。

回答的结构如下：1. 定义一些基础概念和公式 2. 证明这三种测量方法间的等价性 3. 通过实验结果验证等价性（实验代码需要Python 3，工具库numpy，scipy和sklearn）。

假设我们有两个向量 $X=[X_1,\dots X_n]$ 和 $Y=[Y_1,\dots Y_n]$ ，长度均为 n 。

欧氏距离（Euclidean Distance）是常见的相似性度量方法，可求两个向量间的距离，取值范围为0至正无穷。显然，如果两个向量间的距离较小，那么向量也肯定更为相似。此处需要注意的一点是，欧氏距离计算默认对于每一个维度给予相同的权重，因此如果不同维度的取值范围差别很大，那么结果很容易被某个维度所决定。解决方法除了对数据进行处理以外，还可以使用加权欧氏距离，不同维度使用不同的权重。本文中我们使用的是欧氏距离的平方。

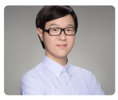
- 公式1： $d(X,Y)=\sum_{i=1}^{n}(X_n-Y_n)^2$

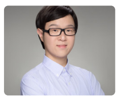
Pearson相关性系数（Pearson Correlation）是衡量向量相似度的一种方法。输出范围为-1到+1, 0代表无相关性，负值为负相关，正值为正相关。

- 公式2： $\rho(X,Y)=\frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X\sigma_Y}=\frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sqrt{\sum_{i=1}^{n}(X_i-\mu_X)^2}\sqrt{\sum_{i=1}^{n}(Y_i-\mu_Y)^2}}$

Cosine相似度也是一种相似性度量，输出范围和Pearson相关性系数一致，含义也相似。

- 公式3： $c(X,Y)=\frac{X\cdot Y}{|X|\,|Y|}=\frac{\sum_{i=1}^{n}X_iY_i}{\sqrt{\sum_{i=1}^{n}X_i^2}\sqrt{\sum_{i=1}^{n}Y_i^2}}$





标准化（Standardization）是一种常见的数据缩放手段，标准化后的数据均值为0，标准差为1。

- **公式4**：$z(X) = \frac{X_i - \mu_X}{\sigma_X}$

平方和（Summed Square）与样本方差（Sample Variance）之间的关系：

- **公式5**：$\sigma_X = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \mu_X)^2}{n-1}}$
- **公式6**：由公式5可得 $(n-1)\sigma_X^2 = \sum_{i=1}^{n}(X_i - \mu_X)^2$

证明1： Pearson相关性系数与Cosine Similarity在数据被标准化后等价

观察公式2和公式3，易发现如果将公式3中的X和Y代入公式4，可得

$c(X,Y) = \frac{\sum_{i=1}^{n} X_i Y_i}{\sqrt{\sum_{i=1}^{n} X_i^2}\sqrt{\sum_{i=1}^{n} Y_i^2}} = \frac{z(X)\cdot z(Y)}{|z(X)|\,|z(Y)|}$ 因为此时 $\mu = 0, \sigma = 1$ ，所以经过化简后会发现公式2和3等价。为了节省空间，过程略去，可参考其他答主的回答。

证明2：Pearson相关性系数和欧式距离方在标准化数据下等价

为了简化公式，此处的 X, Y 我们默认已经经过了**标准化处理**，因此均值为0，标准差为1。在这种情况下我们可以利用了公式5和6化简 $\sum_{i=1}^{n} X_i^2$ 和 $\sum_{i=1}^{n} Y_i^2$，得到下式：

$\sum_{i=1}^{n} X_i^2 = \sum_{i=1}^{n}(X_i - 0)^2 = \sum_{i=1}^{n}(X_i - \mu_X)^2 = (n-1)\sigma_X^2 = n-1$ ，当 n 取值很大时 $n-1 \rightarrow n$ ，所以我们可得到 $\sum_{i=1}^{t} X_i^2 = \sum_{i=1}^{t} Y_i^2 = n$ ，这个结论马上会用到。

让我们开始展开欧氏距离方（第二步到第三步使用了我们上边的推导）：

$$
\begin{aligned}
d(X,Y) &= \sum_{i=1}^{n}(X_n - Y_n)^2 \\
&= \sum_{i=1}^{n} {X_i}^2 - 2\sum_{i=1}^{n} X_i Y_i + \sum_{i=1}^{n} {Y_i}^2 \\
&= 2n - 2\sum_{i=1}^{n} X_i Y_i \\
&= 2n(1 - \sum_{i=1}^{n} X_i Y_i) \\
&= 2n(1 - \frac{\sum_{i=1}^{n}(X_i - 0)(Y_i - 0)}{1 \cdot 1}) \\
&= 2n(1 - \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y} \\
&= 2n(1 - \rho(X,Y))
\end{aligned}
$$

于是我们得到了结论 $d(X,Y) = 2n(1-\rho(X,Y))$ ，此处的n是向量的长度，是常数，因此我们依然可以认为是等价的。

划重点：欧氏距离的平方 = 2 * 常数n （也就是向量的长度）* （1-Pearson相关系数）

证明3：Cosine相似度和欧氏距离方等价

通过证明1和2，易得证明3，略去。

实验证明：

我随机生成了三个向量（长度为100），并分别计算两两之间的Pearson相关性系数，Cosine相似度和欧式距离方：

- 原始数据，没有任何处理
- 经过了标准化（公式4）后的结果

结果如下图，可见标准化后三者等价。**此处需要注意因为Pearson可能是负数，因此我用1-Pearson**，之后结果就会是非负数并处于区间 $[0, 2]$ ，这样就可以和欧氏距离这个非负进行对比。

```
原始数据，没有标准化

             x1&x2  x2&x3  x1&x3
pearson:     0.9196 1.0139 1.0571
cos:         0.264  0.3301 0.3024
euclidean sq 9.695  10.485 11.29

标准化后的数据：均值=0，标准差=1

             x1&x2  x2&x3  x1&x3
pearson:     0.9196 1.0139 1.0571
cos:         0.9196 1.0139 1.0571
euclidean sq 0.9196 1.0139 1.0571
```

```
import numpy as np
from scipy.stats import pearsonr
from scipy.spatial.distance import euclidean
from scipy.spatial.distance import cosine
from sklearn.preprocessing import StandardScaler

# 设定向量长度，均为100
n = 100
x1 = np.random.random_integers(0, 10, (n,1))
x2 = np.random.random_integers(0, 10, (n,1))
x3 = np.random.random_integers(0, 10, (n,1))

p12 = 1 - pearsonr(x1, x2)[0][0]
p13 = 1 - pearsonr(x1, x3)[0][0]
p23 = 1 - pearsonr(x2, x3)[0][0]

d12 = (euclidean(x1, x2)**2) / (2*n)
d13 = (euclidean(x1, x3)**2) / (2*n)
d23 = (euclidean(x2, x3)**2) / (2*n)

c12 = cosine(x1, x2)
c13 = cosine(x1, x3)
c23 = cosine(x2, x3)

print('\n原始数据，没有标准化\n')
print('             x1&x2  x2&x3  x1&x3')
print('pearson:     ', np.round(p12, decimals=4), np.round(p13, decimals=4),
      np.round(p23, decimals=4))
print('cos:         ', np.round(c12, decimals=4), np.round(c13, decimals=4),
      np.round(c23, decimals=4))
print('euclidean sq', np.round(d12, decimals=4), np.round(d13, decimals=4),
      np.round(d23, decimals=4))

# 标准化后的数据
x1_n = StandardScaler().fit_transform(x1)
x2_n = StandardScaler().fit_transform(x2)
x3_n = StandardScaler().fit_transform(x3)

p12_n = 1 - pearsonr(x1_n, x2_n)[0][0]
p13_n = 1 - pearsonr(x1_n, x3_n)[0][0]
p23_n = 1 - pearsonr(x2_n, x3_n)[0][0]

d12_n = (euclidean(x1_n, x2_n)**2) / (2*n)
d13_n = (euclidean(x1_n, x3_n)**2) / (2*n)
d23_n = (euclidean(x2_n, x3_n)**2) / (2*n)

c12_n = cosine(x1_n, x2_n)
c13_n = cosine(x1_n, x3_n)
c23_n = cosine(x2_n, x3_n)
```

```
print('\n标准化后的数据: 均值=0, 标准差=1\n')
print('             x1&x2  x2&x3  x1&x3')
print('pearson:     ', np.round(p12_n, decimals=4), np.round(p13_n, decimals=4),
      np.round(p23_n, decimals=4))
print('cos:         ', np.round(c12_n, decimals=4), np.round(c13_n, decimals=4),
      np.round(c23_n, decimals=4))
print('euclidean sq', np.round(d12_n, decimals=4), np.round(d13_n, decimals=4),
      np.round(d23_n, decimals=4))
```

[1] Berthold, M.R. and Höppner, F., 2016. On clustering time series using euclidean distance and pearson correlation. *arXiv preprint arXiv:1601.02213*.

编辑于 2018-04-10

陈小龙
新闻码农

332 人赞同了该回答

楼上这些的回答都太复杂了!!!

先说结论: 皮尔逊相关系数是余弦相似度在维度值缺失情况下的一种改进, 皮尔逊相关系数是余弦相似度在维度值缺失情况下的一种改进, 皮尔逊相关系数是余弦相似度在维度值缺失情况下的一种改进.

楼主如果高中正常毕业, 参加过高考, 那么肯定会这么一个公式

cos<a, b> = a • b / |a|•|b|

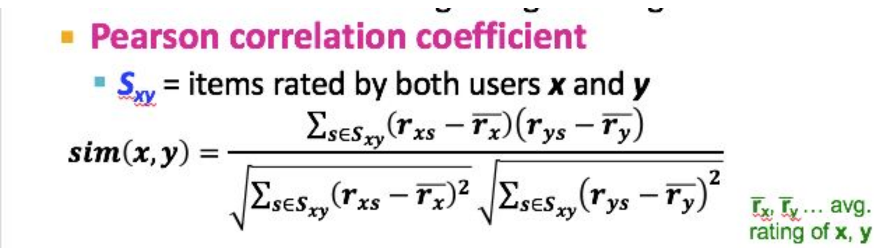
假设a = (3, 1, 0), b = (2, -1, 2)

分子是a, b两个向量的内积, (3, 1, 0) • (2, -1, 2) = 3•2 + 1•(-1) + 0•2 = 5

分母是两个向量模(模指的是向量的长度)的乘积.

总之这个cos的计算不要太简单...高考一向这是送分题...

然后问题来了, 皮尔逊系数和这个cos啥关系...(不好意思借用了我们学校老师的课件...)



皮尔森相关系数计算公式

其实皮尔逊系数就是cos计算之前两个向量都先进行中心化(centered)...就这么简单...

中心化的意思是说, 对每个向量, 我先计算所有元素的平均值avg, 然后向量中每个维度的值都减去这个avg, 得到的这个向量叫做被中心化的向量. 机器学习, 数据挖掘要计算向量余弦相似度的时候, 由于向量经常在某个维度上有数据的缺失, 预处理阶段都要对所有维度的数值进行中心化处理.

我们观察皮尔逊系数的公式:

分子部分: 每个向量的每个数字要先减掉向量各

分母部分: 两个根号式子就是在做取模运算, 里面的所有的 r 也要减掉平均值, 其实也就是在做中心化.

note: 我其实是今天上推荐系统课, 讲相似性计算的时候才发现原来余弦计算和皮尔逊相关系数计算就是一个东西两个名字啊......气死我了...高中的时候我还是靠背公式解题的...逃....

================2017-11-15更新: 对余弦相似度和皮尔森相关系数的进一步认识================

余弦距离(余弦相似度), 计算的是两个向量在空间中的夹角大小, 值域为[-1, 1]: 1代表夹角为0°, 完全重叠/完全相似; -1代表夹角为180°, 完全相反方向/毫不相似.

余弦相似度的问题是: 其计算严格要求"两个向量必须所有维度上都有数值", 比如:

v1 = (1, 2, 4),

v2=(3, -1, null),

那么这两个向量由于v2中第三个维度有null, 无法进行计算.

然而, 实际我们做数据挖掘的过程中, 向量在某个维度的值常常是缺失的, 比如v2=(3, -1, null), v2数据采集或者保存中缺少一个维度的信息, 只有两个维度. 那么, 我们一个很朴素的想法就是, 我们在这个地方填充一个值, 不就满足了"两个向量必须所有维度上都有数值"的严格要求了吗? 填充值的时候, 我们一般这个向量已有数据的平均值, 所以v2填充后变成v2=(3, -1, 1), 接下来我们就可以计算cos<v1, v2>了.

而皮尔逊相关系数的思路是, 我把这些null的维度都填上0, 然后让所有其他维度减去这个向量各维度的平均值, 这样的操作叫作**中心化**. 中心化之后所有维度的平均值就是0了, 也满足进行余弦计算的要求. 然后再进行我们的余弦计算得到结果. 这样先中心化再余弦计得到的相关系数叫作皮尔逊相关系数.

所以, 从本质上, 皮尔逊相关系数是余弦相似度在维度值缺失情况下的一种改进.

另外, 以movielens数据集计算两个用户之间相似度的协同过滤场景来说, 余弦相似度和皮尔逊相关系数所表现的都是在两个用户都有打分记录的那些特征维度下, 他们超过自身打分平均值的幅度是否接近. 如果各维度下的超出幅度都类似, 那么就是比较相似的.

编辑于 2018-03-02

▲ 赞同 332 ▼ 23 条评论 分享 收藏 感谢 收起

TimXP

216 人赞同了该回答

要理解Pearson相关系数，首先要理解协方差（Covariance），协方差是一个反映两个随机变量相关程度的指标，如果一个变量跟随着另一个变量同时变大或者变小，那么这两个变量的协方差就是正值，反之相反，公式如下：

Pearson相关系数公式如下：

由公式可知，Pearson相关系数是用协方差除以
随机变量的相关程度（协方差大于0的时候表示

▲ 赞同 313 ▼ 28 条评论 分享 收藏 感谢 收起

协方差值的大小并不能很好地度量两个随机变量的关联程度，例如，现在二维空间中分布着一些数据，我们想知道数据点坐标X轴和Y轴的相关程度，如果X与Y的相关程度较小但是数据分布的比较离散，这样会导致求出的协方差值较大，用这个值来度量相关程度是不合理的，如下图：

为了更好的度量两个随机变量的相关程度，引入了Pearson相关系数，其在协方差的基础上除以了两个随机变量的标准差，容易得出，pearson是一个介于-1和1之间的值，当两个变量的线性关系增强时，相关系数趋于1或-1；当一个变量增大，另一个变量也增大时，表明它们之间是正相关的，相关系数大于0；如果一个变量增大，另一个变量却减小，表明它们之间是负相关的，相关系数小于0；如果相关系数等于0，表明它们之间不存在线性相关关系。《数据挖掘导论》给出了一个很好的图来说明：

编辑于 2016-08-20

▲ 赞同 216 ▼ 8 条评论 分享 收藏 感谢 收起

知乎用户

5 人赞同了该回答

几何上可以理解为夹角的余弦值

编辑于 2017-05-31

▲ 赞同 5 ▼ 添加评论 分享 收藏 感谢 收起

知乎用户

4 人赞同了该回答

$L^2(\Omega,\mathcal{F},P)$ 是Hilbert空间，如果定义 $<X,Y>=E(XY)$ 的话， $E(XY)$ 满足内积的性质，就可以用夹角啊正交啊那套东西来考虑。

▲ 赞同 313 ▼ 28 条评论 分享 收藏 感谢 收起

这个相关系数 $\rho = \frac{<X,Y>}{||X||||Y||}$ 其实就是两者中心化保证零均值之后的夹角的cos。

编辑于 2017-05-31

▲ 赞同 4 ▼ 8 条评论 分享 收藏 感谢 收起

▲ 赞同 313 ▼ 28 条评论 分享 收藏 感谢 收起