**SVD原理与示例**

**1.SVD简介**

奇异值分解(Singular Value Decomposition, SVD)是在机器学习中应用广泛的算法之一。SVD算法主要应用于降维、推荐、图像压缩、自然语言处理等。

**2.特征值和特征向量**

假设A为n\*n的矩阵，即，x是n\*1的向量，为一个实数值；如果有，那么x称为A的对应特征值的特征向量。

**3.方阵的特征值分解**

假设A为n\*n的矩阵(方阵)，A的n个特征值为，A的n个特征向量为，那么A可以用如下的特征分解式表示。

其中，

一般我们把W的各个特征向量标准化，即满足，或者。这时候这些特征向量为一组标准正交基，即，也就是，这时候W也称为酉矩阵。

于是。

特征分解的几何意义，当矩阵是一个高维矩阵，它表示高维空间的一个线性变换。这个变换有很多的变换方向，通过特征值分解得到的前N个特征向量，对应这个矩阵的最主要的N个变换方向。利用前N个变换方向，可以近似表示原来的矩阵(变换)，或者说提取了最重要的N个特征。

**4.奇异值分解**

上述特征值分解一般适用于方阵。如果A为m\*n的矩阵，是否可以进行类似的分解呢？可以的，这就是奇异值分解。A的奇异值分解，定义为这样的分解式。

U是一个m\*m的矩阵，是一个m\*n的矩阵，除了对角线上的元素外，其余元素为0，对角线上的元素称为奇异值。U和V都是酉矩阵。

下面介绍如何计算U和V以及。

首先计算，这是一个n\*n的矩阵，对其进行特征值分解，有(请参考方阵的特征值分解)。的特征向量构成了右奇异向量。

为什么的特征向量构成了右奇异向量呢？由于，于是，，可以记。

其次，我们求，这是一个m\*m的矩阵，对其进行特征值分解，有(请参考方阵的特征值分解)。的特征向量构成了左奇异向量。

这时候，我们得到了U矩阵和V矩阵；接下来，求奇异值。

求奇异值的第一种方法如下。

所以。

求奇异值的第二种方法如下。

根据，我们得到和的特征值为奇异值的平方，即奇异值为特征值的开根方，。

**5．SVD实例**

假设有矩阵A如下。

首先计算和。

对进行特征分解，有

对进行特征分解，有

用上文介绍的第一种方法，求解奇异值。

有，有。

*，*有。

最终的奇异值分解式如下。

**6.奇异值分解特性**

在特征值分解中，一般把特征值按照从大到小、从左上角向右下角进行排列。在奇异值分解中，奇异值的排列方式类似。

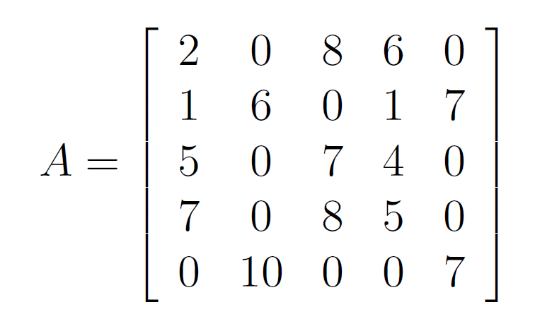
在很多情况下，前10%甚至1%的奇异值的和占到全部奇异值的和的99%以上。这时候，可以保留k个最大的奇异值及其对应的左右奇异向量，对矩阵进行近似处理，具体如下。

由于K <<< m/n，一个大型矩阵可以用三个小型矩阵来表示。正是由于这个性质，奇异值分解可以用于降维、数据压缩和去噪、推荐、自然语言处理(潜在语义分析LSI)等。

**7.文档-词项矩阵与SVD**

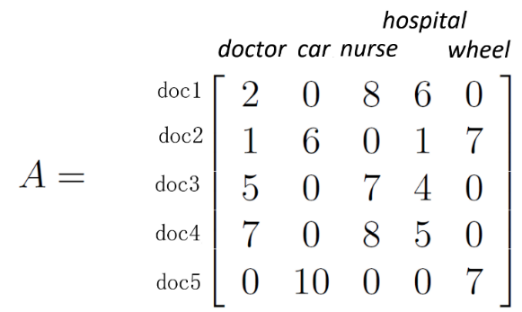
7.1 文档和词项的对应矩阵

现在有下面的文档-词项矩阵。

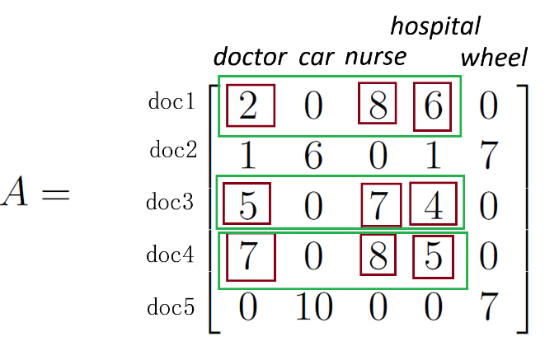


这个矩阵的每行，对应一个文档，分别是doc1,doc2,doc3,doc4,doc5等文档。

这个矩阵的每列，对应一个词项，分别是doctor，car，nurse，hospital，wheel。



我们把文档和词项对应一下，重新观察，我们看到，Doc1,doc3,doc4好像在doctor，nurse，hospital的权重更大一点。Doc2,doc5好像在car，wheel词项上的权重更大一点。



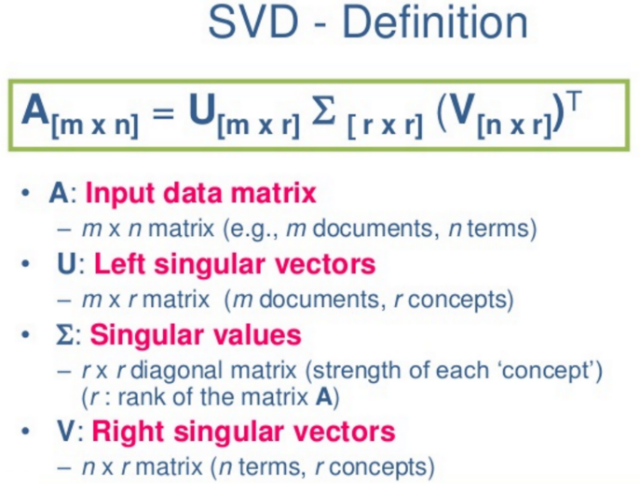
我们可以把Doc1,doc3,doc4归入“医院”主题，把Doc2,doc5归入“汽车”主题。

在实际应用中，文档数量很多，词项的数量很多，通过肉眼不容易观察这种文档-主题关系。

SVD的功能，一个是把文档集映射到一个概念空间，另外一个是对文档集的表示进行降维。我们先了解降维。

7.2 SVD与降维

奇异值分解的特性如下。



* 实数矩阵A ，总是可以进行奇异值分解的；
* 分解后的U 、S 、V都是唯一的；
* U、V都是列正交的(每一列都是相互垂直的单位向量)；
* S是一个对角矩阵，对角线上的元素是奇异值，且均为正数，并且从左上往下按照大小排列。

A是一个文档-词项矩阵(对其进行转置就是词项-文档矩阵，读者可以灵活掌握)。

对矩阵A进行奇异值分解，即A = U S VT，那么有

这个矩阵每一行代表一篇文档，每一列代表一个概念，绝对值越大表示相关性越高。比如文档1和概念1、3的相关性较高。

这个矩阵可以理解为5个概念的强度，揭示的是将数据投影到低维度空间中时，在各个坐标方向上数据的方差。

VT的每一行代表一个概念，每一列代表一个词项，每个元素代表的是词项与概念之间的相关关系。

现在，我们对数据进行降维，取K=3。U保留前3列为，S保留3行3列为，VT保留前3行为。的每一行代表一篇文章，每一列代表降维至3维后每篇文章在各个维度方向的投影。也可以类似理解。

=

对降维以后的矩阵进行重建，即计算，得到，结果和A有差别。但是差别是可以忍受的，因为我们获得了好处，也就是可以用3维向量表示每个词项和每个文档了。

在实际应用中，我们没有必要重建，而是在低维空间上，用低维向量尝试计算词项之间、文档之间的相似度。

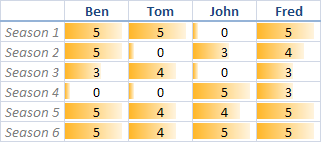
文档对应到的各行，通过计算的各个行向量的相似度，就可以计算文档之间的相似度。类似的，词项对应到的各个列，通过计算的各个列向量的相似度，就可以计算词项之间的相似度。

8. SVD与推荐

基于SVD的推荐策略，本质上是使用SVD方法做特征降维，然后再计算相似度。

下面给出一个实例。

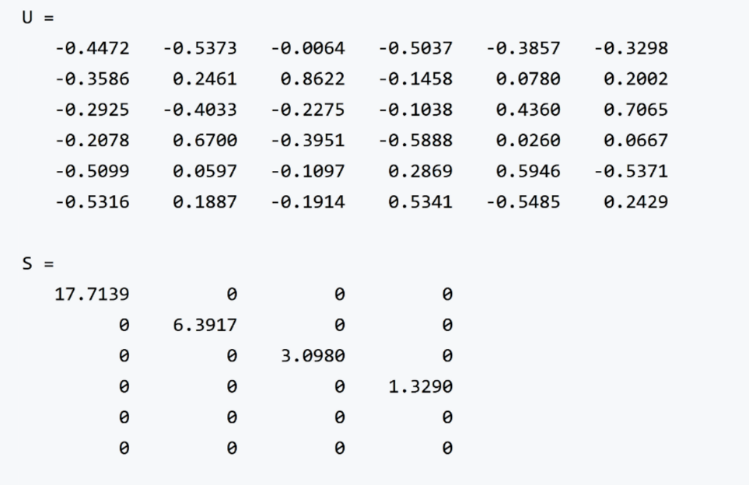
假设我们有一个矩阵，该矩阵每一列代表一个user，每一行代表一个item。



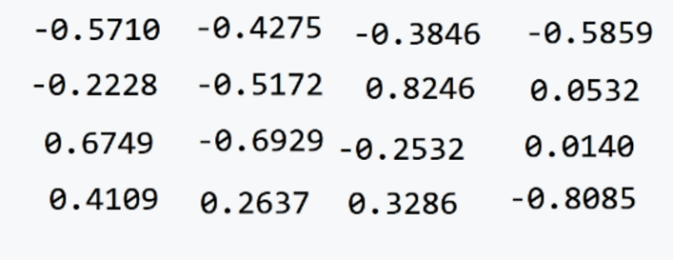
如上图，ben、tom….代表user，season1、season2…代表item。

矩阵值代表评分(0代表未评分)：如 ben对season1评分为5，tom对season1 评分为5，tom对season2未评分。

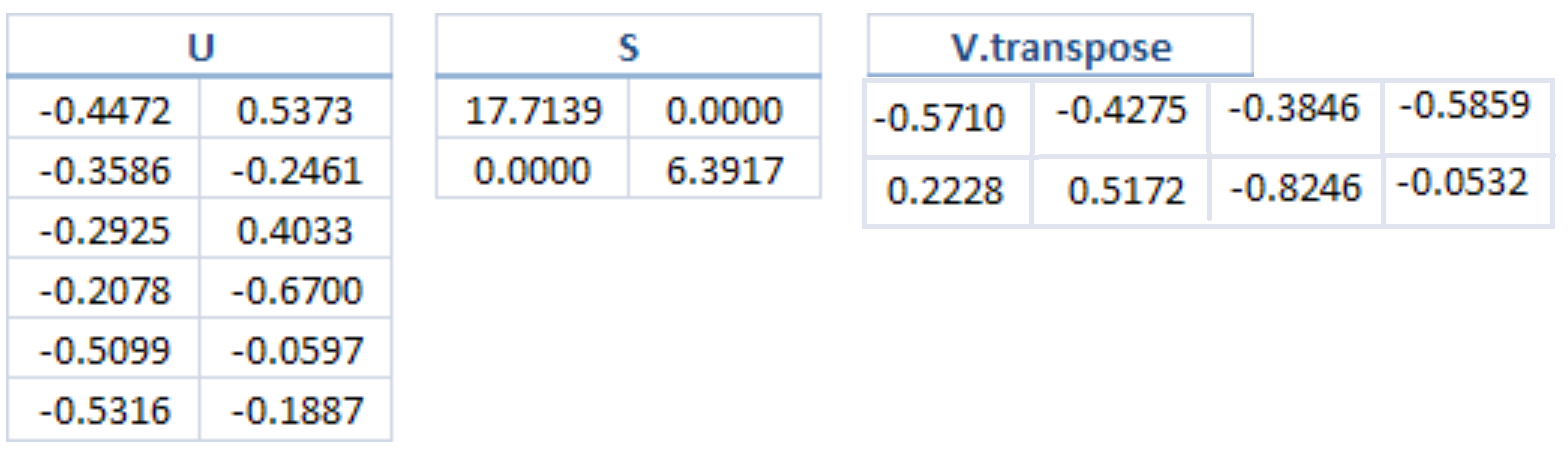
现在对该矩阵进行SVD奇异值分解。



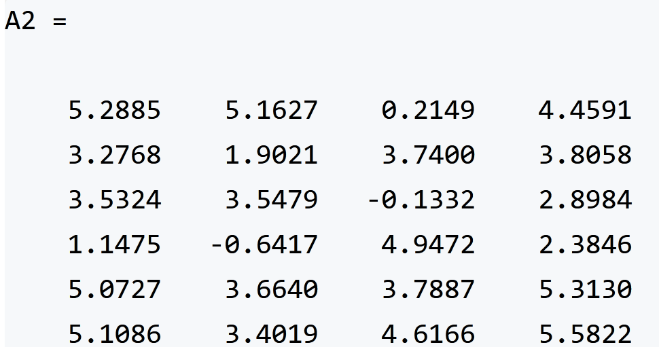
VT



取S对角线上前k个元素。

当k=2时候即将S(6\*4)降维成S(2\*2)，同时U(6\*6),VT(4\*4)相应地变为 U(6\*2),VT (2×4)。

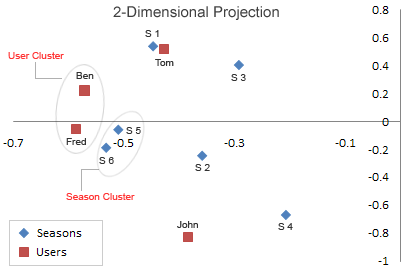
我们用降维后的U，S，VT来相乘得到A2。



我们可以看到，A2和A很接近，所以降维可以看成是一种数据的有损压缩。

U的每一行表示一个二维向量, 将u的第一列当成x值，第二列当成y值。

将VT的每一列表示一个2维向量，将VT的第一行当成x值，第二行当成y值。

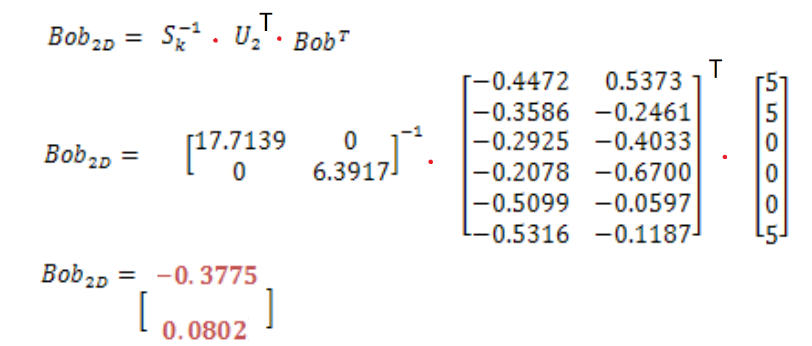


从图中可以看出, Season5，Season6特别靠近，Ben和Fred也特别靠近。

8.1准备推荐

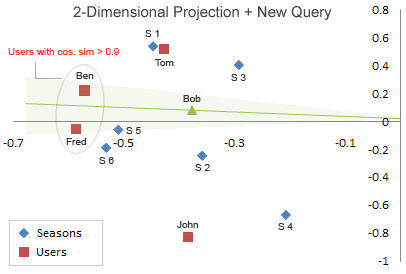
现在有个名字叫Bob的新用户，并且已知这个用户对season n的评分向量为：[5 5 0 0 0 5] (行向量)。

我们的任务是要对他做出个性化的推荐。我们的思路首先是，利用新用户的评分向量，找出该用户的相似用户。



|  |
| --- |
| 这个公式的证明如下。  A = USVT。  现在有新的user，为列向量。  根据A= USVT，有A=US(v1 v2 v3 v4)，备注v1,v2,v3,v4为VT的各个列向量，对应各个user。  降维表示即A’= U2S2(v1’ v2’ v3’ v4’)。  现有新用户的表示为，假设他的降维表示为v’，那么有= U2S2v’，于是有v’= U2TS2-1。S2为对角矩阵，U2为正交矩阵。 |

对特征进行降维转换，得到一个Bob的二维向量，就知道Bob的坐标。



从图中找出和Bob最相似的用户。

注意，最相似并不是距离最近的用户，这里的相似用余弦相似度计算，即夹角与Bob最小的用户坐标，可以计算出最相似的用户是ben。找出最相似的用户，即ben。

观察ben的评分向量为：[5 5 **3** 0 **5** 5]。

对比Bob的评分向量：[5 5 0 0 0 5]。

然后找出ben评分过而Bob未评分的item并排序，即[season 5：5，season 3：3]，即推荐给Bob的item依次为 season5 和 season3。

8.2 扩展思考

相似用户可以多个，每个由相似度作为权重来共同影响推荐的item的评分。

9.参考文献

[1] https://blog.csdn.net/syani/article/details/52297093, 2018.

[2] http://poloclub.gatech.edu/cse6242/2014spring/lectures/CSE6242-20140403-TextLsiVisApp.pdf, 2018.