# 论文查阅资料

# 一、推荐系统

## 1.1 推荐定义

### 1.1.1基于用户和基于物品

推荐就是通过对嗜好的模式进行预测，借以发现你尚未知晓，却合乎心意的新事物。找到可能喜欢的物品可以：

基于用户：观察与你志趣相投的人喜欢些什么

基于物品：观察其他人的明显偏好，你可以弄清楚哪些东西和你已然喜欢的物品相似。

实际上，它们是推荐引擎算法中应用最广的两大类：基于用户和基于物品的推荐程序。严格说来，上述场景均为协同过滤的范例-----仅仅通过了解用户与物品之间的关系进行推荐。

这些技术无须了解物品自身的属性。从某种意义上讲，这是一个优点。该推荐框架并不关心物品是否为书籍、主题公园、鲜花或其他人，因为根本不会导人它们的属性。

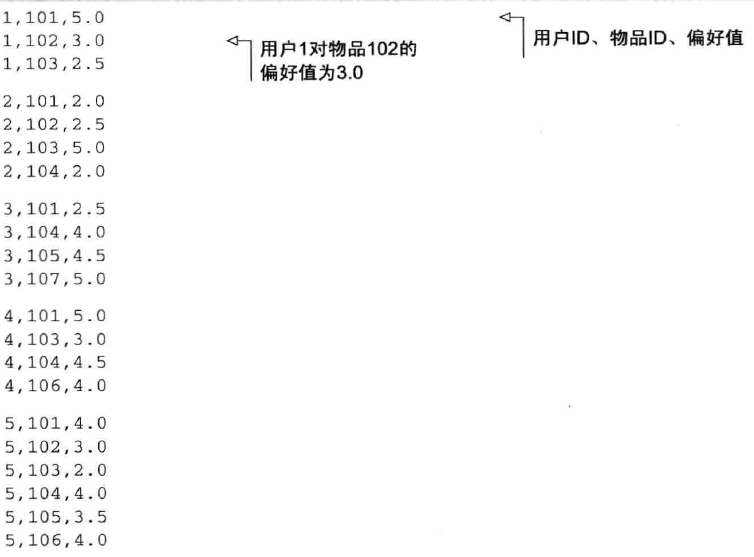
### 1.1.2基于内容

其他一些方法则立足于物品的属性，通常称为基于内容( content-based)的推荐技术。基于内容的推荐技术没有什么问题，相反，它们很有用。但是，它们必须与特定领域相结合，而难以规整为一个框架。

## 1.2 推荐引擎

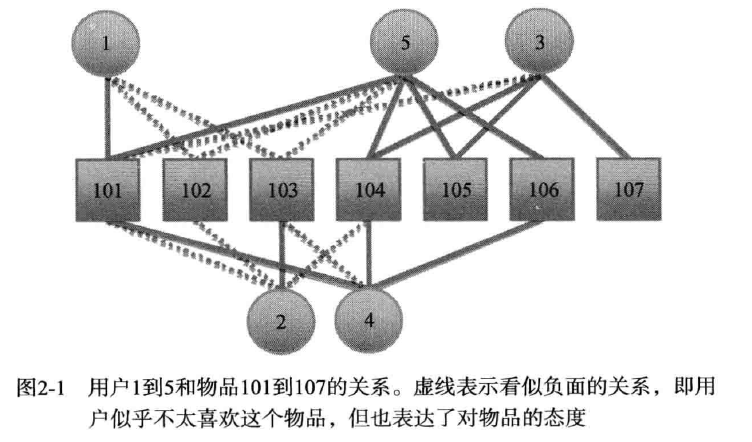
### 1.2.1 输入数据

推荐程序数据是以偏好的形式来表达的。因为最常见的推荐引擎总是把项目推荐给用户，所以谈论偏好最简便的方法是建立从用户到物品的关联。

一个偏好包含一个用户ID、一个物品ID，通常还有一个表达用户对物品的偏爱程度的数值。偏好值可任意设定，只需保证更大的值代表更强的正向偏好。

### 1.2.2推荐方式

根据上面的数据，得出用户与物品的关系如下：



根据上面的关系图，若向用户1进行推荐：可以为用户1推荐什么书呢？不是101、102和103，因为用户1显然已经知道这些书了，而推荐是用来发现新事物的。直观上看，既然用户4和5与用户1类似，那么把用户4或5喜欢的东西推荐给用户1是个好主意。

这样一来，可能的推荐结果就是书104、105和106。总体上看，104似乎最有可能，因为物品104对应的偏好值是4.5和4.0。

## 1.3评估推荐

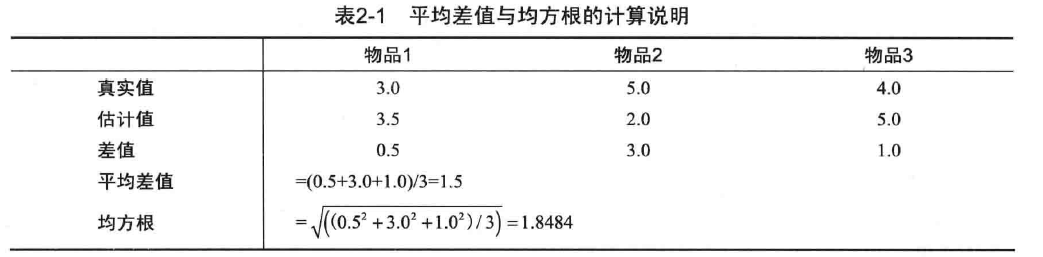
大多数推荐引擎仅会试图给出某些或其他所有物品的估计评分。由此，一种评估推荐程序推荐结果的方法是评估其估计偏好值的质量。即评估所估计的偏好在多大程度上与实际偏好相匹配。

### 1.3.1 训练数据和评分

在推荐引擎中，这可以通过提取一小段真实数据作为测试数据来仿真。这些用于测试的偏好不会作为训练数据导人到被评估的推荐引擎。相反，推荐程序需要为这些缺失的测试数据估计出偏好值，然后估计结果用于与真实值进行对照。

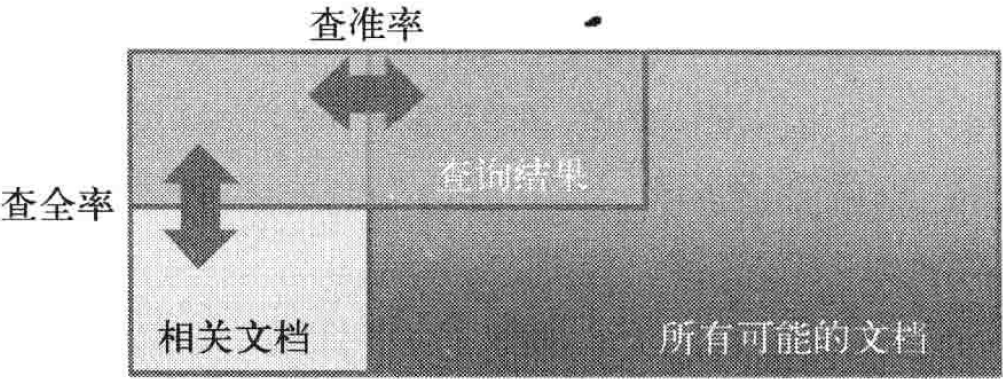
进而，我们可以非常简单地为推荐程序做一种评分。例如，可以计算出在估计和实际偏好之

间的平均差值。在这种评分中，值越低越好，因为值越低意味着估计值与实际偏好值的差别越小。



### 1.3.2 评估差准率和查全率

在一些对“相关”的定义中，**查准率**是指在top结果中相关结果的比例。“Precision at 10”（推荐10个结果时的查准率）是指这个比例来自对前1 0个top结果的判定。**查全率**是指所有相关结果包含在top结果中的比例。



# 二、进行基于用户推荐

## 2.1 基于用户的推荐

基于用户的推荐，它是在这个领域早期研究中阐述的方法，Mahout自然会有它的实现。“基于用户”这个说法有些不准确，因为所有推荐算法都建立在与用户和物品相关的数据上。基于用户的推荐算法的典型特征是，它建立在用户间有某种相似性的基础之上。事实上，这种算法在日常生活中很常见。

人群间的相似度

好友间的相似度

个人品味相似度

## 2.2 推荐算法

在实际过程中，对每个物品都检查实在是太慢了。通常会先计算出一个最相似用户的邻域，

然后仅考虑这些用户评价过的物品：

for每个其他用户w

计算用户u和用户w的相似度s

按相似度排序后，将位置靠前的用户作为邻域n

for (n中用户有偏好，u用户无偏好的)每个物品i

for (n中用户对i有偏好的) 每个其他用户v

计算用户u和用户v的相似度s

按权重s将v对i的偏好并入平均值

首先确定相似的用户，再考虑这些最相似用户对什么物品感兴趣。这些物品，就成为推荐的候选项。

**外层循环**简单地把每个已知物品，并且用户未对其表达过偏好的作为候选的推荐项。**内层循环**逐个查看对候选物品做过评价的其他用户，并记下他们对该物品的偏好值。最终，将这些值的**加权**平均作为目标用户对该物品偏好值的预测。每个偏好值的权重取决于该用户与目标用户之间的相似度。与目标用户越相似，他的偏好值所占权重越大。

## 三、相似性度量

基于用户的推荐程序的另一个重要部分，是UserSimilarity实现。基于用户的推荐程序非常

依赖这个组件。如果对用户之间的相似性缺乏可靠并有效的定义，这类推荐方法是没有意义的。这也适用于基于用户的推荐程序的“近亲”——基于物品的推荐程序，它同样依赖于相似性。

（1）基于用户时，用户之间的相似性：

**User1[**Item 1,Item 2,Item 3,Item 4,Item 8,Item 0,Item 1**]**

**User2[**Item 3,Item 6,Item 7,Item 4,Item 8,Item 0,Item 11**]**

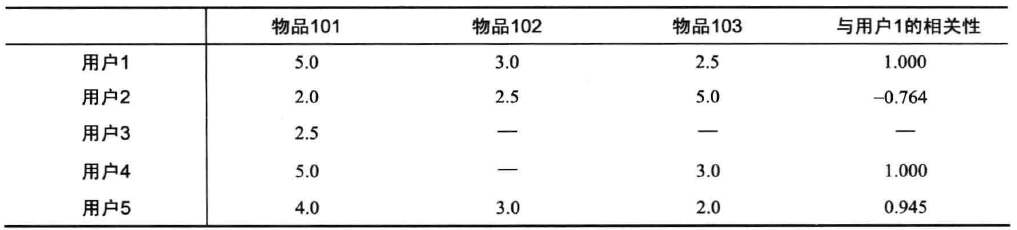
（2）基于物品时，物品之间的相似性

**Item1[**User 3,User 6,User 7,User 4,User 8,User 0,User 11**]**

**Item2[**User 13,User 5,User 8,User 14,User 9,User 0,User 11**]**

## 3.1 基于皮尔逊相关系数的相似度

皮尔逊相关系数是一个介于-1和1之间的数，它度量两个一一对应的数列之间的线性相关程度。也就是说，它表示两个数列中对应数字一起增大或一起减小的可能性。它度量数字一起按比例改变的倾向性，也就是说两个数列中的数字存在一个大致的线性关系。当该倾向性强时，相关值趋于1。当相关性很弱时，相关值趋于0。在负相关的情况下，—个序列的值高而另一个序列的值低，相关值趋于-1。

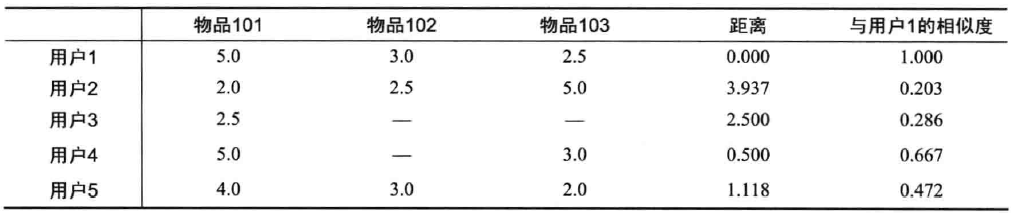


在统计学上，皮尔逊相关系数是两个序列协方差与二者方差乘积的比值。协方差计算的是两个序列变化趋势一致的绝对量。当两个序列相对于各自的均值点向同一方向移动得越远，协方差值就越大。除以方差则是为了对这一变化进行归一化。

这一统计学中广泛使用的概念，同样可以用于度量用户之间的相似性。它度量两个用户针对同一物品的偏好值变化趋势的一致性，都偏高或都偏低。

## 3.2 基于欧式距离定义相似度

**欧式距离**定义相似度，是基于用户之间的距离。你可以将用户想象成多维空间中的点，维数等于总的物品数，偏好值是坐标。这种相似性度量：计算两个用户点之间的欧氏距离d。这个值本身并不代表相似度，因为该值越大表示距离越远，也就是说两个用户越不相似。用户越相似，这个值应该越小。因此，实际应用中取1/(1+d)为相似度。



## 3.3 采用余弦相似度量

余弦相似性度量也将用户偏好值视为空间中的点，并基于此进行相似性度量。你需要将用户偏好值，视为n维空间中的点。现在，假设有两条从原点出发，分别到这两个点的射线。如果两个用户相似，则他们的打分也相似，也就是说他们的空间位置是很接近的。这样一来，至少这两条射线的方向也会差不多，两条射线之间的夹角会比较小。反之，如果两个用户不相似，则相应的两个点会相隔较远，从原点到这两点的射线很有可能指向不同的方向，形成的夹角会比较大。

与欧氏距离类似，这个夹角同样可以用来度量相似性。在这种情况下，夹角余弦代表相似度

值。如果你对三角函数不熟悉，那么记住这点就行了：余弦取值范围在-1到1之间，小的夹角余弦接近1，大的夹角（接近1800）余弦接近-1。这个性质很好，因为小的夹角映射到了较高的相似度值趋于1，而大的夹角则映射到-1附近。

## 3.4 斯皮尔曼相关系数基于相对排名定义相似度

## 3.5 忽略偏好值基于谷本系数计算相似度

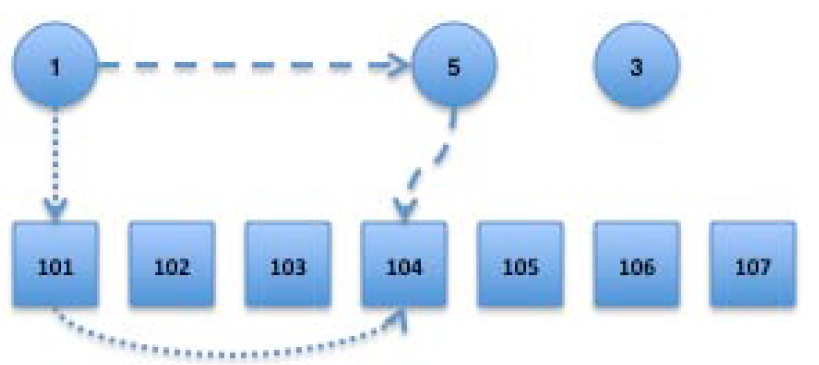
## 3.6 基于对数似然比更好的计算相似度

# 四、基于物品的推荐

## 4.1 基于物品推荐的概述

基于物品的推荐是以**物品**之间的相似度为基础的。基于用户的推荐，是根据相似的用户来推荐一个物品。而基于物品的推荐，根据物品的相似性进行推荐。

下图显示了基于用户和基于物品推荐之间的区别，他们通过不同的途径选择要推荐的物品：分别是通过相似的用户和相似的物品。



基于用户的推荐，寻找相似的用户，并了解他们喜欢什么；

基于物品的推荐了解用户的喜好，寻找相似的物品。

## 4.2 推荐算法

基于物品的推荐算法如下：

for(用户u尚未表达偏好的) 每个物品i

for(用户u表达偏好的) 每个物品j

计算i和j之间的相似度s

按权重为s将u对j的偏好并入平均值

return 值最高的物品（按加权平均排序）

第三行显示了它基于物品之间的相似度，而非象前面那样基于用户的相似性。两种算法比较相似，但也不完全是彼此相同的。他们有一些显著的区别：

**（1）运行时间**

基于物品的推荐程序运行时间随着物品个数的增长，基于用户推荐程序运行时间随用户数量增长。当物品数量比用户数量少很多的话，基于物品的推荐程序会带来显著的性能提升。

**（2）稳定性**

物品要比用户更稳定一些，随着时间的推移，搜集到的数据越来越多，对物品之间的相似性的估计值会趋于收敛。不会剧烈或频繁的发生变化。相对用户，随着时间的推移，用户会有一些新的认识，用户的的喜好也会随之改变。