

第15章：使用推荐算法提升客户价值

《Python数据科学：技术详解与商业实践》

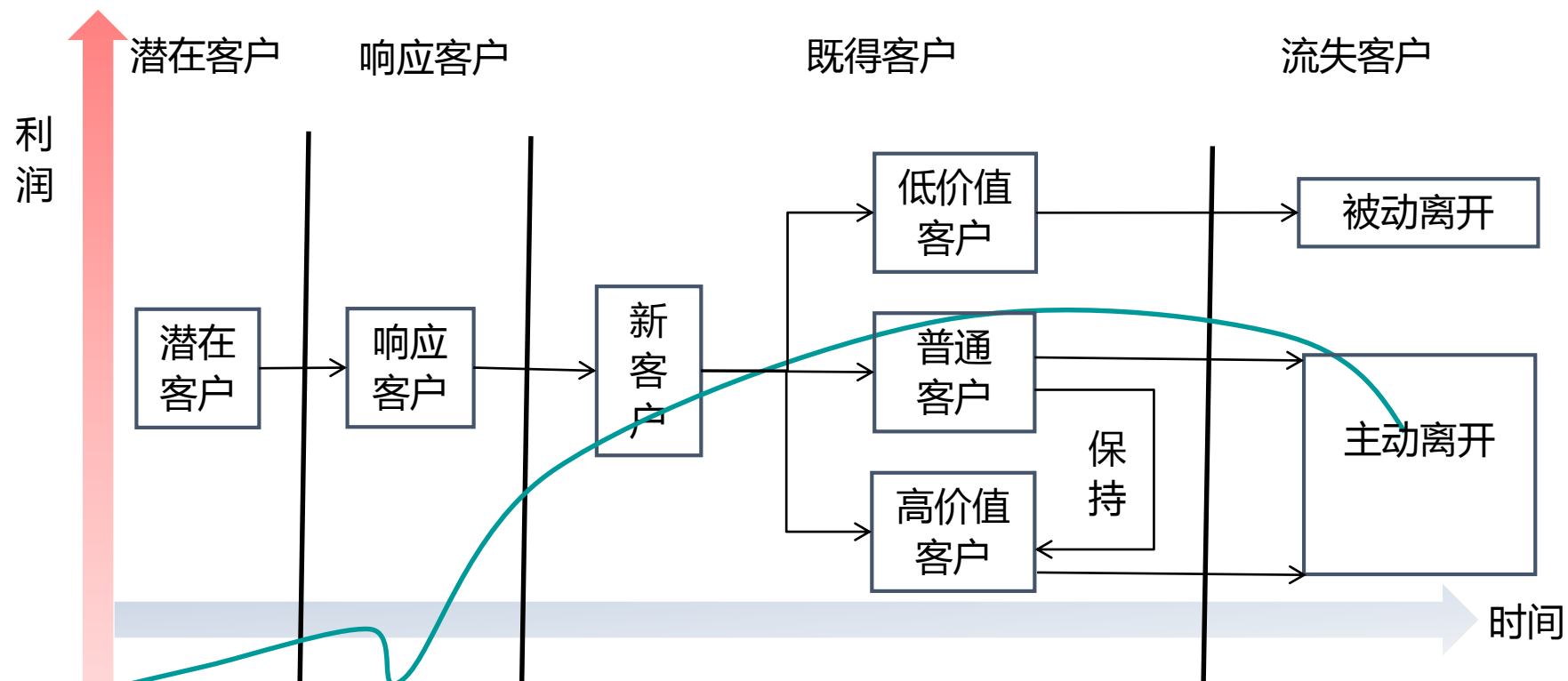
讲师：Ben

自我介绍

- 天善商业智能和大数据社区 讲师 – Ben
- 天善社区 ID - Ben_Chang
- <https://www.hellobi.com> – 学习过程中有任何相关的问题都可以提到技术社区数据挖掘版块。

- 智能推荐
- 购物篮分析与运用
- 相关性在推荐中的运用

课程由来：客户关系管理方面数据挖掘运用全景图



- 发掘潜在客户

- 客户获取
- 初始信用评分
- 客户价值预测

- 客户分群(市场细分)
- 交叉销售
- 产品精准营销
- 行为信用评分
- 欺诈侦测
- 客户保留
- 客户关系网

- 流失客户时间判断
- 流失客户类型判断

1 智能推荐

- 亚马逊把推荐成功地应用到购物网站，如买了X的人还买了Y，亚马逊有20%-30%的销售来自推荐。



深入解析SAS:数据处理、分析优化与商业应用 平装 – 2015年1月1日

夏坤庄 (作者), 徐唯 (作者), 潘红莲 (作者), 林建伟 (作者)

★★★★★ 4 条商品评论 | 天天低价·正品质优 | 分享

显示所有 2 格式和版本

Kindle电子书
¥ 64.68

平装
¥ 76.10

立即在免费软件上阅读

优惠信息: 优惠 再买好书减10元

配送至: 北京东城区 库存中仅剩 1 件 (更多商品正在运送途中)

送达日期: 明天(5月12日), 请在10小时37分钟内下单并选择“快递送货上门”。

(精确送达时间请于结账页面查询)

购买此商品的顾客也同时购买



CDA数据分析师系列丛书:
胸有成竹!数据分 ...
常国珍
★★★★★ 4
平装
¥ 36.30



SAS应用统计分析(第5版)
罗纳德·科迪(Ro ...
★★★★★ 45
平装
¥ 67.70



CDA数据分析师系列丛书:
从零进阶!数据分 ...
曹正凤
★★★★★ 6
平装
¥ 37.10



如虎添翼!数据处理的SAS
EG实现
徐筱刚
★★★★★ 4
平装
¥ 36.30



SAS编程与数据挖掘商业案
例
姚志勇
★★★★★ 51
平装
¥ 33.60



SAS统计分析与应用从入门
到精通(第2版) ...
汪海波
★★★★★ 15
平装
¥ 55.90



互联网金融时代:消费信贷评
分建模与应用
单良
★★★★★ 2
平装
¥ 41.30

- 从搜索到推荐是大势所趋，个性化成为潮流

互联网正从搜索时代进入发现时代。搜索是你明确知道需要什么东西；发现是说你并不十分清楚的知道一种东西存在，或者你并不清楚怎么样才能找到这种东西。

--- 美国财富杂志

推荐系统将成为未来十年里最重要的变革，社会化网站将由推荐系统所驱动。

--- John Riedl 明尼苏达大学教授

- 精准营销就是找到精确找到有直接购买欲望需求的客户。



- 孕妇对于零售商来说是个含金量很高的顾客群体。
- Target的市场营销人员求助于Target的顾客数据分析部要求建立一个模型，在孕妇第2个妊娠期就确认出来。
- Target 比一个女孩的父亲更早知道她怀孕的信息

Target



说几个发生在您身边的数据挖掘应用的例子！

推荐系统分类

- 基于应用领域分类：电子商务推荐，社交好友推荐，搜索引擎推荐，信息内容推荐
- 基于设计思想：基于协同过滤的推荐，基于内容的推荐，基于知识的推荐，混合推荐
- 基于使用何种数据：基于用户行为数据的推荐，基于用户标签的推荐，基于社交网络数据，基于上下文信息（时间上下文，地点上下文等）

推荐系统的基本思想

- 产品推荐一般是在海量的用户中发掘出一小部分和你品位比较类似的，这些用户成为邻居，然后根据他们喜欢的其他东西组织成一个排序的目录作为推荐。
- 核心问题：
 - 如何确定一个用户是不是和你有相似的品位？
 - 如何将邻居们的喜好组织成一个排序的目录？

推荐算法归纳

分类	算法	建模理念	适用场景
基于客户需求的推荐	分类模型（逻辑回归、神经网络）	根据客户的属性、行为数据对客户的需求进行建模，进而预测客户的（购买等）行为	只适用于有销售记录的产品，新老客户都适用，与协同过滤算法相比，建模过程复杂。
基于购物篮的推荐	关联规则	物品被同时购买的模式反映了顾客的需求模式	无需个性化定制的场景；有销售记录的产品，向老客户推荐；在套餐设计、产品摆放中运用。
基于物品相似性的推荐	基于Item的协同过滤	同类型的商品被同类的人购买	个性化定制的场景；一般适用于有购买行为的老客户，而且推荐的产品也不能是全新的产品；电商的商品推荐
基于客户相似性的推荐	基于User的协同过滤，各种相似度、KNN、	同类型的客户消费行为相似	一般适用于有购买行为的老客户，而且推荐的产品也不能是全新的产品；社交网站的好友推荐
基于内容的推荐	规则或SVD方法	客户不只关注一个产品，而是一个领域	可以是全新的产品，但是向老客户推荐。
市场细分	K-means	相似的客户具有相似的需求	全新的产品，全新的客户都是适用，但是模型效果只能事后检验

2 购物篮分析与运用

什么是购物篮

单个客户一次购买商品的总和称为一个购物篮

购物篮分析手段

- 重点在商品与商品之间的关联
 - 啤酒与尿布的关联

购物篮分析运用

- 超市货架布局：互补品与替代品
- 套餐设计

常用算法

- 不考虑购物顺序：关联规则
- 考虑购物顺序：序贯模型



2.1 关联规则

关联规则：评判规则的标准

- 支持度

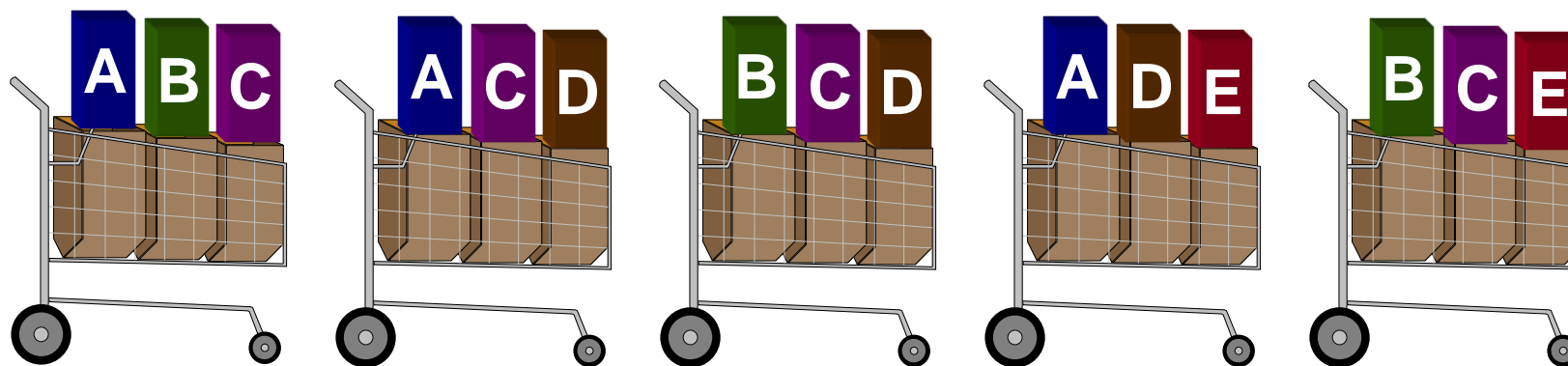
支持度 (Support) 指的是LHS 和 RHS 所包括的商品都同时出现的概率。即，包含规则。

$\text{支持度} = \frac{\text{LHS和RHS商品的交易次数}}{\text{总的交易次数}}$

- 置信度

可信度(Confidence)是指在所有的购买了左边商品的同时购买了右边商品的交易机率。即， $\text{可信度} = \frac{\text{包含规则两边商品的交易次数}}{\text{包含规则左边商品的交易次数}}$

关联规则



规则	支持度	置信度
$A \Rightarrow D$	2/5	2/3
$C \Rightarrow A$	2/5	2/4
$A \Rightarrow C$	2/5	2/3
$B \ \& \ C \Rightarrow D$	1/5	1/3

理解支持度与置信度

规则X的支持度=规则X的交易次数/交易的总数

理解：支持度表示规则X是否普遍。过小支持度与过大支持度？

规则X(A→B)的置信度=规则X的交易次数/规则X中商品B的交易次数

理解：置信度是一种条件概率，表示购买了A产品的客户再购买B产品的概率。低置信度与高置信度？

仅支置信度和支持度是否靠谱？

食堂卖饭，1000份打饭记录中，买米饭的有800人次，买牛肉的有600人次，两个共同买的有400人次，那么可以得出对于规则(牛肉-米饭)

$\text{Support} = P(\text{牛肉} \& \text{米饭}) = 400 / 1000 = 0.40$

$\text{Confidence} = P(\text{米饭} | \text{牛肉}) = 400 / 600 = 0.67$

置信度和支持度都很高，但是给买牛肉的人推荐米饭有意义吗？

无任何条件下用户购买米饭的概率：

$P(\text{米饭}) = 800 / 1000 = 0.8$

提升度 = 置信度 / 无条件概率 = $0.67 / 0.8$

- 提升度

提升度指的是两种可能性的比较，一种是在已知购买了左边商品情况下购买右边商品的可能性，另一种是任意情况下购买右边商品的可能性。两种可能性比较方式可以定义为两种可能性的概率之差值，或者两种可能性的概率之比值。

即规则的置信度/包含规则右边商品的交易次数占总交易量的比例。

$$\text{Lift} = P(\text{购买} = \text{右边商品} | \text{购买} = \text{左边商品}) / P(\text{购买} = \text{右边商品})$$

Apriori算法 (1)

几个概念

1. 项集—项目的集合
2. 频繁项集—符合最小支持度要求的项集
3. 最小支持度
4. 最小置信度

Apriori算法 (2)

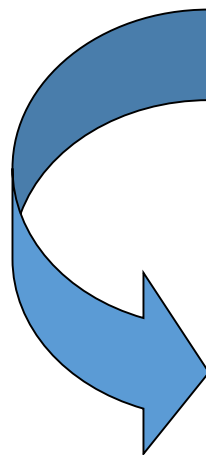
- Apriori算法利用频繁项集性质的先验知识 (prior knowledge)，通过逐层搜索的迭代方法，即将k-项集用于探索(k+1)-项集，来穷尽数据集中的所有频繁项集。
 - 先找到频繁1-项集集合 L_1 ，然后用 L_1 找到频繁2-项集集合 L_2 ，接着用 L_2 找 L_3 ，直到找不到频繁k-项集，找每个 L_k 需要一次数据库扫描。

Apriori算法步骤：举例

支持度阈值=20%

TID	ID列表
T10	A,B,E
T20	B,D
T30	B,C
T40	A,B,D
T50	A,C
T60	B,C
T70	A,C
T80	A,B,C,E
T90	A,B,C

1-项集C1



频繁1-项集
L1

算法第1次扫描：最小支持频度2

项集	支持频度
{A}	6
{B}	7
{C}	6
{D}	2
{E}	2

项集	支持频度
{A}	6
{B}	7
{C}	6
{D}	2
{E}	2

Apriori算法步骤：举例

频繁1-项集L1

项集	支持频度
{A}	6
{B}	7
{C}	6
{D}	2
{E}	2

算法第2次扫描：最小支持频度2

项集	支持频度
{A,B}	4
{A,C}	4
{A,D}	1
{A,E}	2
{B,C}	4
{B,D}	2
{B,E}	2
{C,D}	0
{C,E}	1
{D,E}	0

2-项集C2

项集	支持频度
{A,B}	4
{A,C}	4
{A,E}	2
{B,C}	4
{B,D}	2
{B,E}	2

频繁2-项集L2

Apriori算法步骤：举例

频繁2-项集

L2

项集	支持频度
{A,B}	4
{A,C}	4
{A,E}	2
{B,C}	4
{B,D}	2
{B,E}	2



连接项集	可用否
{A,B,C}	Y
{A,B,E}	Y
{A,C,E}	N
{B,C,D}	N
{B,C,E}	N
{B,D,E}	N



算法第3次扫描：最小支持
频度2 3-项集C3

项集	支持频度
{A,B,C}	2
{A,B,E}	2

频繁3-项集L3

项集	支持频度
{A,B,C}	2
{A,B,E}	2

候选C4={A,B,C,E},但是{B,C,E}不是频繁项集，因此，C4为空，算法无法发现新的频繁项集而终止

2.2 序贯模型

序贯模式挖掘(Sequence Analysis)

序贯模型算法分为四步：

1. 排序阶段。数据库D以客户号为主键，交易时间为次键进行排序。这个阶段将原来的事务数据库转换成由客户序列组成的数据库。
2. 频繁项集阶段。找出所有频繁项集组成的集合L。也同步得到所有频繁1-序列组成的集合。
3. 转换阶段。在找序列模式的过程中，要不断地进行检测一个给定的频繁集是否包含于一个客户序列中。
4. 序列阶段利用已知的频繁集的集合来找到所需的序列。类似于关联的Apriori算法。

序贯模式挖掘(Sequence Analysis)

- 1、在给出的数据库中，找出所有符合最小支持频数要求的频繁1-序列，这个例子中，频繁1序列为(30)、(40)、(70)、(40,70)和(90)

最小支持度=2

客户号	客户购物序列	频繁1序列
1	(30),(90)	(30)
2	(10,20),(30),(40,60,70)	(40)
3	(30,50,70)	(70)
4	(30)(40,70)(90)	(40, 70)
5	(90)	(90)

序贯模式挖掘(Sequence Analysis)

2、给一个可行的映射如下所示，其目的在于将频繁1序列当做一个整体，在后续计算中更加简便与高效。

频繁1序列	映射成
(30)	1
(40)	2
(70)	3
(40,70)	4
(90)	5

序贯模式挖掘(Sequence Analysis)

3、进行转换：为了使这个过程尽量快，用另一种形式来替换每一个客户序列。需注意：在转换完成的客户序列中，每条交易被其所包含的所有频繁1序列所取代。如果一条交易不包含任何频繁1序列，在转换完成的序列中它将不被保留。如果一个客户序列不包含任何的频繁1序列，在转换好的数据库中这个序列也将不复存在。

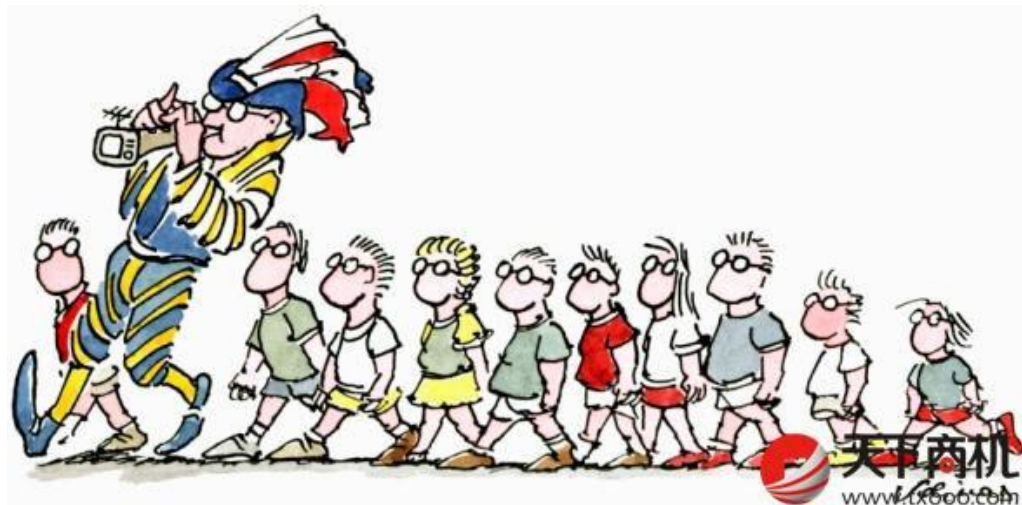
这个例子中，将映射好的频繁1序列的映射值 带回原来的数据，原数据会发生改变，如下所示。

客户号	客户购物序列	频繁项集	映射成
1	(30),(90)	{{(30)}{(90)}}	{1},{5}
2	(10,20),(30),(40,60,70)	{{(30)}{(40)(70)(40,70)}}	{1},{2,3,4}
3	(30,50,70)	{{(30)(70)}}	{1,3}
4	(30)(40,70)(90)	{{(30)}{(40)(70)(40,70)}{(90)}}	{1}{2,3,4}{5}
5	(90)	{{(90)}}	{5}

3 相关性在推荐中的运用

基于相似性的物品推荐

- 收集用户偏好
- 根据客户偏好计算客户的相似性
- 对匹配好的客户对，推荐产品
- 系统过滤的初级形态



风格与偏好
尝试者与跟随者
从众

收集用户偏好的方法

用户行为	类型	特征	作用
评分	显式	整数量化的偏好，可能的取值是 $[0, n]$ ； n 一般取值为 5 或者是 10	通过用户对物品的评分，可以精确的得到用户的偏好
投票	显式	布尔量化的偏好，取值是 0 或 1	通过用户对物品的投票，可以较精确的得到用户的偏好
转发	显式	布尔量化的偏好，取值是 0 或 1	通过用户对物品的投票，可以精确的得到用户的偏好。
			如果是站内，同时可以推理得到被转发人的偏好（不精确）
保存书签	显示	布尔量化的偏好，取值是 0 或 1	通过用户对物品的投票，可以精确的得到用户的偏好。
标记标签 (Tag)	显示	一些单词，需要对单词进行分析，得到偏好	通过分析用户的标签，可以得到用户对项目的理解，同时可以分析出用户的情感：喜欢还是讨厌
评论	显示	一段文字，需要进行文本分析，得到偏好	通过分析用户的评论，可以得到用户的情感：喜欢还是讨厌
点击流	隐式	一组用户的点击，用户对物品感兴趣，需要进行分析，得到偏好	用户的点击一定程度上反映了用户的注意力，所以它也可以从一定程度上反映用户的喜好。
页面停留时间	隐式	一组时间信息，噪音大，需要进行去噪，分析，得到偏好	用户的页面停留时间一定程度上反映了用户的注意力和喜好，但噪音偏大，不好利用。
购买	隐式	布尔量化的偏好，取值是 0 或 1	用户的购买是很明确的说明这个项目它感兴趣。

相似度的计算

相似度和距离是相反的概念：

- 闵可夫斯基距离Minkowski/欧式距离

$$\text{dist}(X, Y) = \left(\sum_{i=1}^n |x_i - y_i|^p \right)^{\frac{1}{p}}$$

- 杰卡德相似系数(Jaccard)

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

- 余弦相似度(cosine similarity)

$$\cos(\theta) = \frac{a^T b}{|a| \cdot |b|}$$

- Pearson相似系数

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

- 相对熵(K-L距离)

$$D(p \parallel q) = \sum_x p(x) \log \frac{p(x)}{q(x)} = E_{p(x)} \log \frac{p(x)}{q(x)}$$

- Hellinger距离

$$D_\alpha(p \parallel q) = \frac{2}{1 - \alpha^2} \left(1 - \int p(x)^{\frac{1+\alpha}{2}} q(x)^{\frac{1-\alpha}{2}} dx \right)$$

基于相似度的推荐

- 某影院收集了N个用户对于M个电影的观影记录。每个用户一行，第i行的记录形式为：

"<用户ID> \t <电影1>; <电影2>;"

- 已知奶茶的观影记录为：84, 14, 90, 91, 29, 21, 9, 44, 24, 89, 8, 42, 41, 40, 25, 37, 30, 16, 97, 52, 62, 56, 80, 83, 36, 26, 73, 64, 32, 27, 67, 65, 79, 87, 17。
- 找出与奶茶最匹配的前15名用户。
 - 思考：如何定义“最匹配”？

```
1347842 44
1347847 30;44
134790 28;30;97
1347912 44;30;78;28;58;68
1347915 8;30
1347916 24;8;58;44;83;32;28;26
1347925 3
1347943 8
1347949 28;58;35;30
1347974 30
1347982 30
1347991 83;77;57
1347999 30
1348027 30;8
1348066 71
1348088 30;97
1348110 83;30
1348118 28;47;30;16;58
1348135 30
1348148 47;48;44;58;33;30;76;18;28
1348153 52
134816 8;30;81
1348186 91
1348187 30
1348191 83;30;97
1348202 25
1348207 28;8
1348221 30;26
1348226 46;28;58
1348230 30
1348252 18
1348253 30;52
1348262 79;52;33
```

余弦相似度

[1001129, 35, 1.0, 35, [8, 9, 14, 16, 17, 21, 24, 25, 26, 27, 29, 30, 32, 36, 37, 40, 41, 42, 44
[305344, 35, 0.606976978666884, 95, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 1
[387418, 34, 0.5865557254410011, 96, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 1
[2118461, 30, 0.581675050747111, 76, [1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 13, 16, 17, 18, 19, 20, 1
[1664010, 31, 0.5617822916268811, 87, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 18, 1
[2439493, 32, 0.5608858475195935, 93, [1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 14, 15, 16, 17, 18, 1
[1114324, 22, 0.5606119105813882, 44, [5, 7, 8, 9, 12, 16, 17, 18, 21, 24, 26, 28, 30, 32, 33, 3
[1403217, 16, 0.5204800389058843, 27, [2, 3, 5, 8, 9, 10, 16, 17, 23, 24, 26, 27, 28, 29, 30, 3
[825819, 11, 0.5156879540323449, 13, [8, 16, 17, 24, 26, 30, 32, 41, 44, 58, 73, 77, 83]]
[1932594, 21, 0.5019960159204453, 50, [3, 5, 6, 7, 9, 10, 15, 16, 17, 18, 19, 20, 24, 25, 26, 2
[1314869, 19, 0.4897622991151437, 43, [3, 5, 6, 8, 9, 13, 16, 17, 18, 23, 24, 26, 27, 28, 30, 3
[525356, 15, 0.4791574237499549, 28, [1, 6, 8, 9, 16, 17, 25, 26, 30, 33, 36, 41, 43, 48, 49, 5
[2606799, 17, 0.4661472225410195, 38, [2, 3, 5, 8, 16, 17, 23, 25, 26, 27, 28, 29, 30, 32, 35, 3
[2056022, 19, 0.4635525346505533, 48, [1, 3, 8, 14, 17, 18, 19, 20, 21, 23, 24, 26, 27, 28, 3
[397001, 9, 0.458682472293863, 11, [8, 18, 24, 25, 29, 30, 44, 57, 79, 83, 97]]
[322009, 13, 0.4581897949526569, 23, [1, 8, 16, 17, 18, 19, 26, 28, 30, 32, 44, 45, 52, 55, 5
[2238060, 15, 0.4553825555391872, 31, [12, 16, 17, 18, 23, 24, 25, 26, 27, 28, 29, 30, 32, 3
[952063, 10, 0.45175395145262565, 14, [8, 16, 17, 24, 25, 26, 28, 30, 58, 76, 79, 83, 91, 92
[786312, 11, 0.4509560339299333, 17, [1, 3, 6, 18, 21, 24, 25, 30, 37, 44, 52, 56, 57, 62, 71
[1473980, 13, 0.4485426135725303, 24, [5, 8, 17, 18, 21, 24, 26, 28, 30, 33, 35, 36, 45, 55,

Jaccard相似度

[1001129, 35, 1.0, 35, [8, 9, 14, 16, 17, 21, 24, 25, 26, 27, 29, 30, 32, 36, 37, 40, 41, 42, 44,
[1114324, 22, 0.38596491228070173, 44, [5, 7, 8, 9, 12, 16, 17, 18, 21, 24, 26, 28, 30, 32, 3
[2118461, 30, 0.37037037037037035, 76, [1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 13, 16, 17, 18, 19, 20
[305344, 35, 0.3684210526315789, 95, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 1
[387418, 34, 0.35051546391752575, 96, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17,
[1403217, 16, 0.34782608695652173, 27, [2, 3, 5, 8, 9, 10, 16, 17, 23, 24, 26, 27, 28, 29, 30
[1664010, 31, 0.34065934065934067, 87, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 18,
[2439493, 32, 0.33333333333333333, 93, [1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 14, 15, 16, 17, 18,
[1932594, 21, 0.328125, 50, [3, 5, 6, 7, 9, 10, 15, 16, 17, 18, 19, 20, 24, 25, 26, 28, 30, 37,
[1314869, 19, 0.3220338983050847, 43, [3, 5, 6, 8, 9, 13, 16, 17, 18, 23, 24, 26, 27, 28, 30,
[525356, 15, 0.3125, 28, [1, 6, 8, 9, 16, 17, 25, 26, 30, 33, 36, 41, 43, 48, 49, 52, 58, 62, 71
[2606799, 17, 0.30357142857142855, 38, [2, 3, 5, 8, 16, 17, 23, 25, 26, 27, 28, 29, 30, 32, 3
[825819, 11, 0.2972972972972973, 13, [8, 16, 17, 24, 26, 30, 32, 41, 44, 58, 73, 77, 83]]
[2056022, 19, 0.296875, 48, [1, 3, 8, 14, 17, 18, 19, 20, 21, 23, 24, 26, 27, 28, 30, 32, 33, 3
[2238060, 15, 0.29411764705882354, 31, [12, 16, 17, 18, 23, 24, 25, 26, 27, 28, 29, 30, 32,
[322009, 13, 0.288888888888888886, 23, [1, 8, 16, 17, 18, 19, 26, 28, 30, 32, 44, 45, 52, 55,
[1663888, 16, 0.2857142857142857, 37, [3, 6, 8, 10, 12, 16, 18, 24, 29, 33, 37, 44, 46, 47, 4
[1473980, 13, 0.2826086956521739, 24, [5, 8, 17, 18, 21, 24, 26, 28, 30, 33, 35, 36, 45, 55,
[1784150, 13, 0.2826086956521739, 24, [2, 5, 8, 16, 17, 18, 26, 28, 30, 32, 37, 40, 48, 52, 5
[786312, 11, 0.2682926829268293, 17, [1, 3, 6, 18, 21, 24, 25, 30, 37, 44, 52, 56, 57, 62, 71

欧氏相似度

[1001129, 35, 35.0, 35, [8, 9, 14, 16, 17, 21, 24, 25, 26, 27, 29, 30, 32, 36, 37, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[305344, 35, 35.0, 95, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[387418, 34, 34.0, 96, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[2439493, 32, 32.0, 93, [1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1664010, 31, 31.0, 87, [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 15, 16, 18, 19, 20, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[2118461, 30, 30.0, 76, [1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 13, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1114324, 22, 22.0, 44, [5, 7, 8, 9, 12, 16, 17, 18, 21, 24, 26, 28, 30, 32, 33, 35, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1932594, 21, 21.0, 50, [3, 5, 6, 7, 9, 10, 15, 16, 17, 18, 19, 20, 24, 25, 26, 28, 30, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1314869, 19, 19.0, 43, [3, 5, 6, 8, 9, 13, 16, 17, 18, 23, 24, 26, 27, 28, 30, 32, 33, 35, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1461435, 19, 19.0, 56, [2, 3, 4, 5, 6, 7, 8, 10, 15, 17, 19, 23, 24, 27, 28, 29, 31, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[2056022, 19, 19.0, 48, [1, 3, 8, 14, 17, 18, 19, 20, 21, 23, 24, 26, 27, 28, 30, 32, 33, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[2606799, 17, 17.0, 38, [2, 3, 5, 8, 16, 17, 23, 25, 26, 27, 28, 29, 30, 32, 35, 36, 40, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1403217, 16, 16.0, 27, [2, 3, 5, 8, 9, 10, 16, 17, 23, 24, 26, 27, 28, 29, 30, 32, 33, 36, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1639792, 16, 16.0, 51, [3, 7, 10, 11, 12, 15, 16, 19, 23, 24, 29, 30, 31, 35, 36, 39, 41, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1663888, 16, 16.0, 37, [3, 6, 8, 10, 12, 16, 18, 24, 29, 33, 37, 44, 46, 47, 48, 50, 52, 54, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[2238060, 15, 15.0, 31, [12, 16, 17, 18, 23, 24, 25, 26, 27, 28, 29, 30, 32, 33, 35, 44, 45, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[525356, 15, 15.0, 28, [1, 6, 8, 9, 16, 17, 25, 26, 30, 33, 36, 41, 43, 48, 49, 52, 58, 62, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1473980, 13, 13.0, 24, [5, 8, 17, 18, 21, 24, 26, 28, 30, 33, 35, 36, 45, 55, 57, 58, 64, 67, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[1784150, 13, 13.0, 24, [2, 5, 8, 16, 17, 18, 26, 28, 30, 32, 37, 40, 48, 52, 58, 68, 74, 77, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]
[2237185, 13, 13.0, 27, [3, 6, 8, 9, 16, 18, 19, 24, 26, 29, 30, 37, 44, 46, 47, 49, 50, 55, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100], 100]

3.协同过滤

协同过滤

- 协同（collaborating）是群体行为，过滤（filtering）则是针对个人的行为。
- 协同过滤基于如下基本假设：如果一个人A在一个问题上和另一个人B持相同观点，那么对于另外一个问题，比起随机选择的一个路人甲，A更有可能同B持相同观点。

关联规则v.s.协同过滤

关联规则：

回答问题：某消费者购买商品A，那么他还可能买什么商品？

特征：直接的推荐，从整体的数据中挖掘潜在关联，与单个人的偏好无关，适用于item不多，并且非重度个性化的场景，如超市购物，汽车导购，交通规划等。

协同过滤：

回答问题：1) 与A客户相似的客户群是谁？将客户群中的物品推荐给A(A没有的物品)；2) 与A物品相似的物品群是什么？对某用户，若其已经购买或收藏A，则推荐该用户与A相似的物品群。

特征：间接推荐，即先找到相似的人(user based)或物(item based)，然后再根据品味相似的人的偏好进行推荐。适用于重个性化并且item非常多的场景，比如音乐，电影等。

关联规则v.s. 协同过滤

相同点：

协同过滤和关联规则，都使用了集体智慧，基于大众行为数据作出决策推荐

不同点：

协同过滤= 协同（集体智慧的部分）+过滤（针对具体某个人又做了一次个性化）

关联规则，基本上只做了集体智慧的部分，在关联规则应用场景，你和我看到的结果基本一致的。

Item-based CF 和 User-based CF

➤ Item-based CF

通过用户对不同item的评分来评测item之间的相似性，基于item之间的相似性做出推荐；

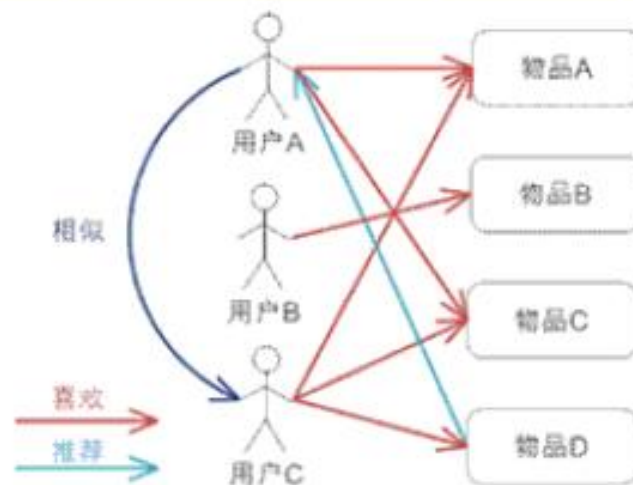
➤ User-based CF

通过不同用户对item的评分来评测用户之间的相似性，基于用户之间的相似性做出推荐。

推荐算法：基于用户的协同过滤算法（UserCF）

- 基于用户的协同过滤，通过不同用户对物品的评分来评测用户之间的相似性，基于用户之间的相似性做出推荐。
- 简单来讲就是：给用户推荐和他兴趣相似的其他用户喜欢的物品。

用户/物品	物品A	物品B	物品C	物品D
用户A	√		√	推荐
用户B		√		
用户C	√		√	√



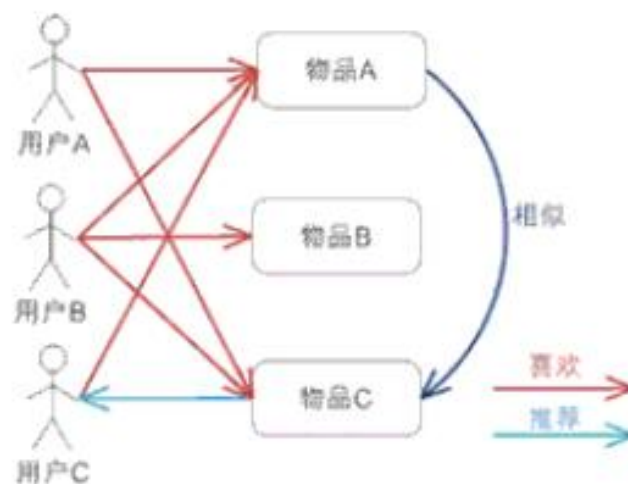
基于用户的协同过滤算法 (UserCF)

- 基于UserCF的基本思想相当简单，基于用户对物品的偏好找到相邻邻居用户，然后将邻居用户喜欢的推荐给当前用户。
- 计算上，就是将一个用户对所有物品的偏好作为一个向量来计算用户之间的相似度，找到K 邻居后，根据邻居的相似度权重以及他们对物品的偏好，预测当前用户没有偏好的未涉及物品，计算得到一个排序的物品列表作为推荐。
- 上图给出了一个例子，对于用户A，根据用户的历史偏好，这里只计算得到一个邻居-用户C，然后将用户C 喜欢的物品D 推荐给用户A。

基于物品的协同过滤算法 (ItemCF)

- 基于item的协同过滤，通过用户对不同item的评分来评测item之间的相似性，基于item之间的相似性做出推荐。
- 简单来讲就是：给用户推荐和他之前喜欢的物品相似的物品。

用户/物品	物品A	物品B	物品C
用户A	√		√
用户B	√	√	√
用户C	√		推荐



基于物品的协同过滤算法 (ItemCF)

- 基于ItemCF的原理和基于UserCF类似，只是在计算邻居时采用物品本身，而不是从用户的角度，即基于用户对物品的偏好找到相似的物品，然后根据用户的历史偏好，推荐相似的物品给他。
- 从计算的角度看，就是将所有用户对某个物品的偏好作为一个向量来计算物品之间的相似度，得到物品的相似物品后，根据用户历史的偏好预测当前用户还没有表示偏好的物品，计算得到一个排序的物品列表作为推荐。
- 上图给出了一个例子，对于物品A，根据所有用户的历史偏好，喜欢物品A的用户都喜欢物品C，得出物品A和物品C比较相似，而用户C喜欢物品A，那么可以推断出用户C可能也喜欢物品C。

User CF vs. Item CF

- 对于电子商务，用户数量一般大大超过商品数量，此时Item CF的计算复杂度较低
- 在非社交网络的网站中，内容内在的联系是很重要的推荐原则，它比基于相似用户的推荐原则更加有效。比如在购书网站上，当你看一本书的时候，推荐引擎会给你推荐相关的书籍，这个推荐的重要性超过了网站首页对该用户的综合推荐。可以看到，在这种情况下，Item CF 的推荐成为了引导用户浏览的重要手段。基于物品的协同过滤算法，是目前电子商务采用最广泛的推荐算法。
- 在社交网络站点中，User CF 是一个更不错的选择，User CF 加上社会网络信息，可以增加用户对推荐解释的信服程度。
- 推荐多样性和精度，各有千秋
- 用户对推荐算法的适应度

3.3 基于物品的协同过滤

需求分析：案例介绍

- 通过简短的描述，我们可以粗略地看出，这个网站提供个性化推荐电影服务。
- 核心点：
 - 网站提供所有电影信息，吸引用户浏览
 - 网站收集用户行为，包括浏览行为，评分行为，评论行为，从而推测出用户的爱好。
 - 网站帮助用户找到，用户还没有看过，并满足他兴趣的电影列表。
 - 网站通过海量数据的积累了，预测未来新片的市场影响和票房
- 电影推荐将成为这个网站的核心功能。

测试数据集

- 测试数据集:
- 每行3个字段，依次是用户ID、电影ID、用户对电影的评分(0-5分，每0.5分为一个评分点！)

```
1, 101, 5.0
1, 102, 3.0
1, 103, 2.5
2, 101, 2.0
2, 102, 2.5
2, 103, 5.0
2, 104, 2.0
3, 101, 2.0
3, 104, 4.0
3, 105, 4.5
3, 107, 5.0
4, 101, 5.0
4, 103, 3.0
4, 104, 4.5
4, 106, 4.0
5, 101, 4.0
5, 102, 3.0
5, 103, 2.0
5, 104, 4.0
5, 105, 3.5
5, 106, 4.0
```

- 1. 建立物品的同现矩阵
- 2. 建立用户对物品的评分矩阵
- 3. 矩阵计算推荐结果

步骤1：建立物品的同现矩阵

按用户分组，找到每个用户所选的物品，单独出现计数及两两一组计数。

	[101]	[102]	[103]	[104]	[105]	[106]	[107]
[101]	5	3	4	4	2	2	1
[102]	3	3	3	2	1	1	0
[103]	4	3	4	3	1	2	0
[104]	4	2	3	4	2	2	1
[105]	2	1	1	2	2	1	1
[106]	2	1	2	2	1	2	0
[107]	1	0	0	1	1	0	1

步骤2：建立用户对物品的评分矩阵

按用户分组，找到每个用户所选的物品及评分。

```
U3  
[101] 2.0  
[102] 0.0  
[103] 0.0  
[104] 4.0  
[105] 4.5  
[106] 0.0  
[107] 5.0
```


步骤3：矩阵计算推荐结果

同现矩阵*评分矩阵=推荐结果

协同（集体行动） × 过滤（针对个人偏好） = 个性化推荐

	101	102	103	104	105	106	107		U3		R
101	5	3	4	4	2	2	1		2.0		40.0
102	3	3	3	2	1	1	0		0.0		18.5
103	4	3	4	3	1	2	0	X	0.0	=	24.5
104	4	2	3	4	2	2	1		4.0		40.0
105	2	1	1	2	2	1	1		4.5		26.0
106	2	1	2	2	1	2	0		0.0		16.5
107	1	0	0	1	1	0	1		5.0		15.5

3.4 基于内容的推荐

基于内容的推荐

根据商品属性/分类进行推荐
根据用户标签进行推荐

amazon.com Hello. Sign in to get personalized recommendations. New customer? Start here

Your Amazon.com Today's Deals Gifts & Wish Lists Gift

Shop All Departments Books

Search Books

Advanced Search Browse Subjects Hot New Releases Bestsellers The New

More in Books

- [Bestsellers](#)
- Hot New Releases**
- [Most Gifted](#)
- [Most Wished For](#)
- [Movers & Shakers](#)

Narrow by Category

< [Any Category](#)

Books

- [4-for-3 Books](#)
- [Accessories](#)
- [Amazon Shorts](#)
- [Arts & Photography](#)
- [Bargain Books](#)
- [Biographies & Memoirs](#)
- [Books on CD](#)
- [Books on Cassette](#)
- [Business & Investing](#)
- [Calendars](#)
- [Children's Books](#)
- [Comics & Graphic Novels](#)
- [Computers & Internet](#)

Hot New Releases in Books

[Any Category](#) > Books

The bestselling new & future releases in Books. Updated

- 


The Noble Wilds
by The Supreme Master Ching Hai
★★★★★ (20 customer reviews)
Publication Date: February 1, 2008
Not in stock; order now and we'll ship as soon as it's available.
Price: \$20.00
- 

A New Earth: Awakening to Your Life's Purpose
by Eckhart Tolle (Author)
★★★★★ (739 customer reviews)
Release Date: January 30, 2008
In Stock
List Price: \$14.00
Price: \$7.70
You Save: \$6.30 (45%)
[121 used & new from \\$5.38](#)

iTunes Store this week

What's Hot **Music** More


New Music Releases

- 

1. **Shine a Light (Deluxe Edition)**
The Rolling Stones
2. **Troubadour**
George Strait
3. **Keep It Simple (Best of)**
Van Morrison
4. **Attack & Release**
The Black Keys
5. **Nude (Voice Stem)...**
Radiohead

[View All](#)

Top Songs

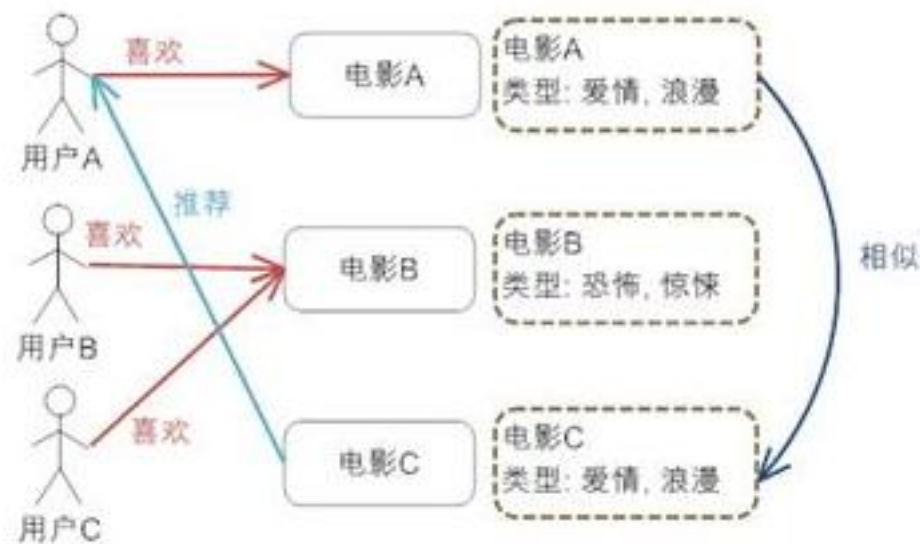
- 

1. **Love in This Club...**
Usher & Young Jeezy
2. **Hallelujah**
Jeff Buckley
3. **Bleeding Love**
Leona Lewis
4. **Love Song**
Sara Bareilles
5. **No Air**
Jordin Sparks & Chris Brown

[View All](#)

基于内容的推荐

已知道用户A喜欢的电影是A，
用户B和用户C喜欢的电影是
B。现在要推荐电影C。



基于内容的做法：要分析电影C的特征。目前的信息是A（爱情和浪漫），B（恐怖和惊悚）。用户A喜欢爱情和浪漫，因此将电影C推荐给用户A。适合于冷启动时期或给客户打来“新”东西，但是需要人为介入。

协同过滤的办法：完全根据用户ABC历史看电影的记录构建客户相似性或电影相似性矩阵。因此在新客户（消费记录很少）或新电影（从来没人看过）场景下无法使用。适合对半新人推荐半新的产品或老产品，不需要人为介入。

- 项亮，《推荐系统实践》，人民邮电出版社，2012。
- Ron Zacharski, www.guidetodatamining.com。

—— 秦路主讲 ——

七周成为数据分析师

七周为期，Get一条数据分析师职业黄金通道！



—— Python ——

数据分析与挖掘

集Python爬虫、数据采集、数据处理、数据分析与数据挖掘于一体，打造Python全栈工程师

主讲老师: 韦玮

VIP会员群+在线答疑+录播复习+1年反复观看

参团课程

案例为师,实战为王

开启Python机器学习之路

科学规划全套课程体系,从入门到进阶,从理论到技巧,嵌入丰富课程案例讲解,逐步推进

讲师: 唐宇迪 深度学习领域多年一线实践研究专家

独一无二的 数据仓库建模指南系列教程升级版

- 从企业视角进行数据规划以及数据仓库模型的搭建
- 高质量的数据库模型和技巧,以及丰富的例子
- 数据仓库架构理论和实践要领

资深讲师: BAO胖子 15年+BI从业经验
涉足电力、快消品、医药、信息服务行业的BI老兵

业务知识一站通

技术+业务,挣钱有门路!

—— 讲师: 陈文 ——



自己动手 丰衣足食

Python3网络爬虫实战案例

— 循序渐进,案例为王,诠释全面,思路制胜 —

讲师: 崔庆才 北航硕士,百万级热度爬文博主



讲师 丘祐玮

人人都爱数据科学家

Python数据科学精华实战课程

数据分析报告制作

秘籍升级版

讲师: 陈丹奕 知乎大神,前百度资深数据分析师

先机致胜 破冰AI

—— 深度学习模型/框架与实战 ——

讲师: 唐宇迪 同济大学硕士
深度学习领域多年一线实践研究专家



BI、商业智能
数据挖掘 大数据
数据分析师
R语言 Python
机器学习
深度学习
人工智能
Hive Hadoop
Tableau
BIEE ETL
数据科学家
PowerBI