

协同过滤

✓ 物以类聚，人以群分

✎ 基于用户的协同过滤算法 (user-based)

✎ 基于物品的协同过滤算法 (item-based)

✎ 矩阵分解 (如果得到用户对所有商品的评分)

✎ 矩阵分解中的隐式与显式情况解决思路

协同过滤

✓ 特征长什么样子

| 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。 |
| 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 | 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。 |

协同过滤

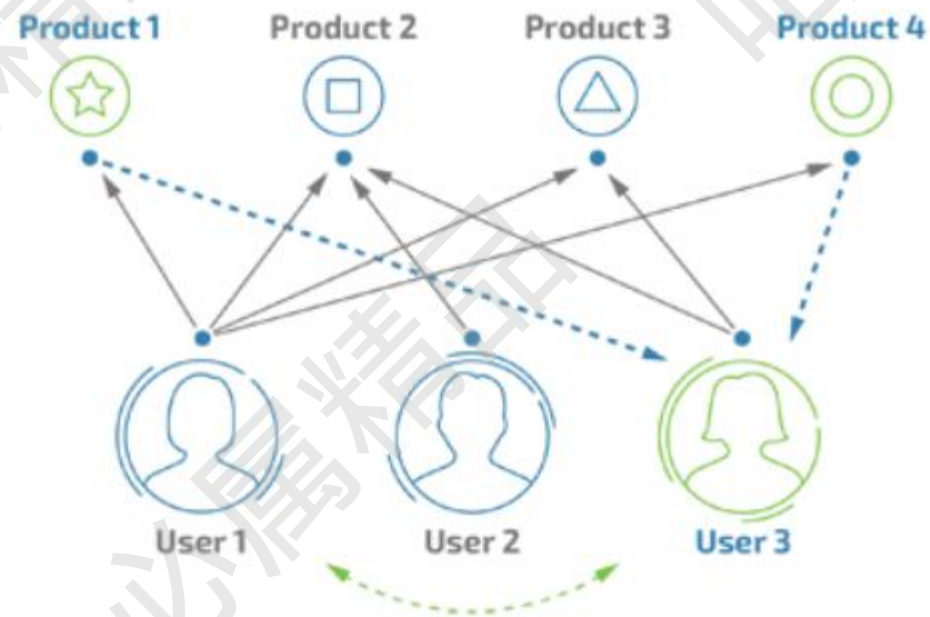
✓ 基于用户的协同过滤

✎ 首先找到相似用户（相似度计算）

✎ 属性特征，行为特征等都可以当做计算输入

✎ User1喜欢1,2,3,4； User3当前喜欢2,3

✎ 如果这俩用户计算后相似度较高，就可以把1,4推给User3



协同过滤

✓ 基于用户的协同过滤

✎ 存在的问题：数据稀疏，计算复杂度，人是善变的，冷启动问题

✎ 稀疏：通常商品非常多，用户购买的只是其中极小一部分

✎ 计算：计算相似度矩阵是个大活，用户和商品都比较多的时候就难了

✎ 冷启动：新用户来了怎么办？

协同过滤

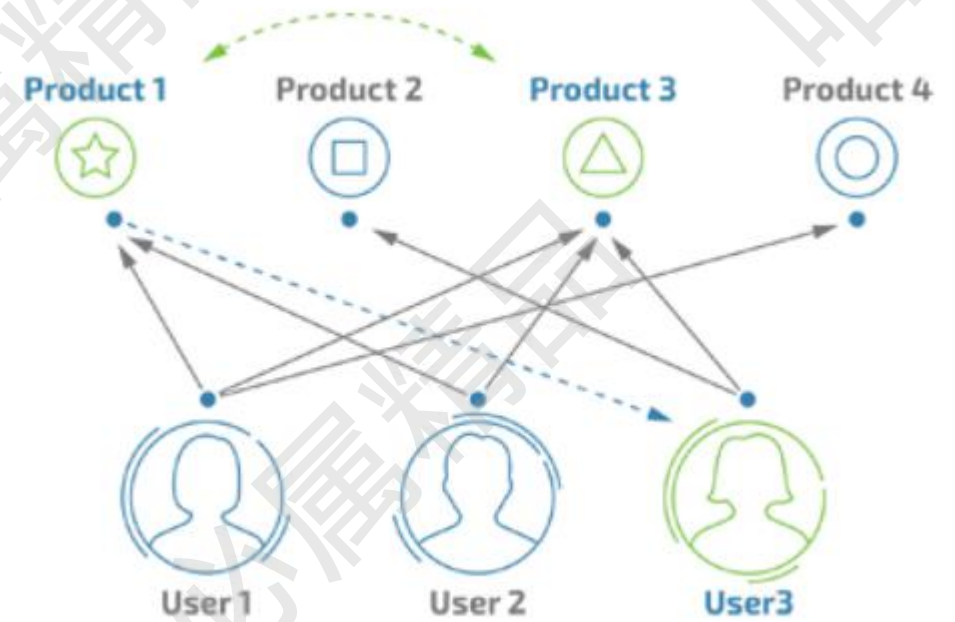
✓ 基于物品的协同过滤

✎ 还是要先得到用户与商品的交互数据

✎ 此时发现商品1和3经常在一起出现

✎ 那这两商品之间肯定有鬼。。。 (相关度)

✎ User3目前只买了商品2和3，此时可以推给他商品1



协同过滤

✓ 基于物品的协同过滤



- ✎ 更流行，现阶段各大网站基本都是用户较多，商品（种类）比用户少的多
- ✎ 商品属性通常比较固定，特征获取容易，而且基本不会改变
- ✎ 即便上架了新商品，它自身也有各种标签，不会像用户一样是张白纸
- ✎ 应用场景更适合当下各种网站，APP（实时的除外，例如新闻）

协同过滤

✓ 小例子

✎ 首先计算商品之间的相似度 (pearson)，邻居设置为2，预测 $r_{51}=?$

| | | users | | | | | | | | | | | |
|--------|---|-------|---|---|---|---|---|---|---|---|----|----|----|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| movies | 1 | 1 | | 3 | | 5 | | | 5 | | 4 | | |
| | 2 | | | 5 | 4 | | | 4 | | | 2 | 1 | 3 |
| | 3 | 2 | 4 | | 1 | 2 | | 3 | | 4 | 3 | 5 | |
| | 4 | | 2 | 4 | | 5 | | | 4 | | | 2 | |
| | 5 | | | 4 | 3 | 4 | 2 | | | | | 2 | 5 |
| | 6 | 1 | | 3 | | 3 | | | 2 | | | 4 | |

 - unknown rating  - rating between 1 to 5

| | | users | | | | | | | | | | | | sim(1,m) |
|--------|---|-------|---|---|---|---|---|---|---|---|----|----|----|-------------|
| | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
| movies | 1 | 1 | | 3 | | ? | 5 | | | 5 | | 4 | | 1.00 |
| | 2 | | | 5 | 4 | | | 4 | | | 2 | 1 | 3 | -0.18 |
| | 3 | 2 | 4 | | 1 | 2 | | 3 | | 4 | 3 | 5 | | <u>0.41</u> |
| | 4 | | 2 | 4 | | 5 | | | 4 | | | 2 | | -0.10 |
| | 5 | | | 4 | 3 | 4 | 2 | | | | | 2 | 5 | -0.31 |
| | 6 | 1 | | 3 | | 3 | | | 2 | | | 4 | | <u>0.59</u> |

$$r_{51} = (0.41 \cdot 2 + 0.59 \cdot 3) / (0.41 + 0.59) = 2.6$$

矩阵分解

✓ 为什么需要矩阵分解

✎ 用户：1个亿，商品100W，这得是多大的一个矩阵，要命了

✎ 能不能间接点来求呢？最终目标就是把每个用户对各个商品的喜好预测出来

✎ 跟找中介租房子差不多，通过中介来重新组合矩阵

✎ 矩阵分解已经成为推荐系统中用的最多的方法

$$R = \begin{pmatrix} 1 & ? & 2 & ? & ? \\ ? & ? & ? & ? & 4 \\ 2 & ? & 4 & 5 & ? \\ ? & ? & 3 & ? & ? \\ ? & 1 & ? & 3 & ? \\ 5 & ? & ? & ? & 2 \end{pmatrix}$$

矩阵分解

✓ 矩阵分解实例

✎ 用户-歌曲之间的行为数据

✎ 1代表听过该歌曲，0表示没有

✎ 可以想象成一个非常稀疏的矩阵

✎ 目标：预测空白值到底等于多少

| | 成都 | 董小姐 | 安河桥 | 洗白白 | 抓泥鳅 | 小白兔 | 西海情歌 | 青藏高原 | 呼伦贝尔 |
|------|----|-----|-----|-----|-----|-----|------|------|------|
| 用户1 | 1 | 1 | | | | | | | |
| 用户2 | 1 | | 1 | | | | | | |
| 用户3 | | 1 | 1 | | | | | | |
| 用户4 | 1 | 1 | | | | | | | |
| 用户5 | 1 | | 1 | | | | | | |
| 用户6 | | 1 | 1 | | | | | | |
| 用户7 | 1 | | | | | | | | |
| 用户8 | | | | 1 | 1 | | | | |
| 用户9 | | | | 1 | | 1 | | | |
| 用户10 | | | | | 1 | 1 | | | |
| 用户11 | | | | 1 | 1 | | | | |
| 用户12 | | | | 1 | | 1 | | | |
| 用户13 | | | | | 1 | 1 | | | |
| 用户14 | | | | 1 | | | | | |
| 用户15 | | | | | | | 1 | 1 | |
| 用户16 | | | | | | | 1 | | 1 |
| 用户17 | | | | | | | | 1 | 1 |
| 用户18 | | | | | | | 1 | 1 | |
| 用户19 | | | | | | | 1 | | 1 |
| 用户20 | | | | | | | | 1 | 1 |
| 用户21 | | | | | | | 1 | | |
| 用户22 | 1 | | | 1 | | | | | |
| 用户23 | 1 | | | | | | 1 | | |
| 用户24 | | | | 1 | | | 1 | | |

矩阵分解

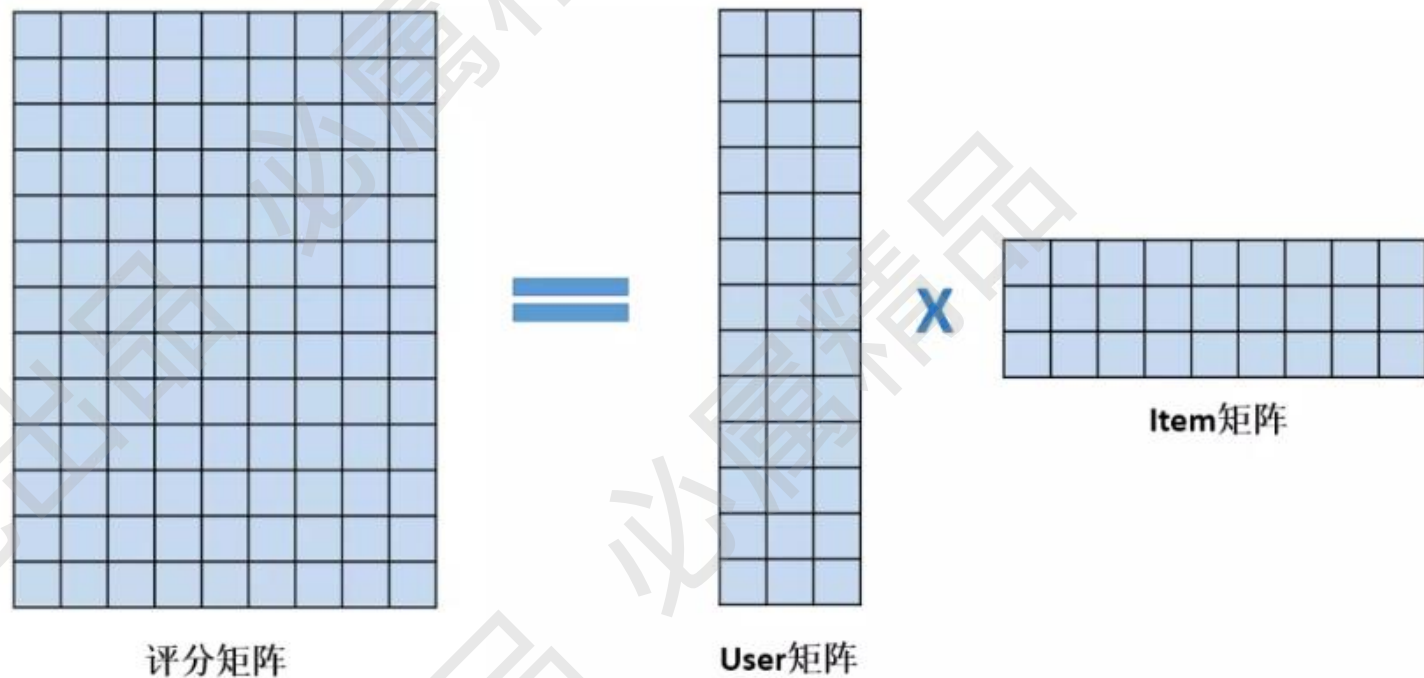
✓ 矩阵分解

✎ user-item矩阵分解

✎ 得到user,item两个矩阵

✎ 原矩阵: $m \times n$ (用户, 音乐)

✎ user($m \times k$), item($k \times n$)



矩阵分解

✓ 矩阵分解

✎ 这俩矩阵可有实际值

✎ K等于多少合适呢?

✎ 其中的数值代表什么?

✎ 如何计算得到?

| | 民谣 | 儿歌 | 草原风 |
|------|-------|-------|-------|
| 用户1 | 0.93 | -0.08 | 0.10 |
| 用户2 | 0.93 | -0.08 | 0.10 |
| 用户3 | 0.90 | -0.21 | 0.07 |
| 用户4 | 0.93 | -0.08 | 0.10 |
| 用户5 | 0.93 | -0.08 | 0.10 |
| 用户6 | 0.90 | -0.21 | 0.07 |
| 用户7 | 0.62 | 0.03 | 0.08 |
| 用户8 | 0.11 | 0.81 | -0.51 |
| 用户9 | 0.11 | 0.81 | -0.51 |
| 用户10 | 0.02 | 0.74 | -0.58 |
| 用户11 | 0.11 | 0.81 | -0.51 |
| 用户12 | 0.11 | 0.81 | -0.51 |
| 用户13 | 0.02 | 0.74 | -0.58 |
| 用户14 | 0.13 | 0.56 | -0.28 |
| 用户15 | -0.07 | 0.49 | 0.77 |
| 用户16 | -0.07 | 0.49 | 0.77 |
| 用户17 | -0.17 | 0.40 | 0.78 |
| 用户18 | -0.07 | 0.49 | 0.77 |
| 用户19 | -0.07 | 0.49 | 0.77 |
| 用户20 | -0.17 | 0.40 | 0.78 |
| 用户21 | 0.02 | 0.38 | 0.49 |
| 用户22 | 0.71 | 0.55 | -0.19 |
| 用户23 | 0.61 | 0.38 | 0.53 |
| 用户24 | 0.15 | 0.88 | 0.19 |

User矩阵

X

| | 成都 | 董小姐 | 安河桥 | 洗白白 | 抓泥鳅 | 小白兔 | 西海情歌 | 青藏高原 | 呼伦贝尔 |
|-----|------|-------|-------|-------|-------|-------|------|-------|-------|
| 民谣 | 0.99 | 0.74 | 0.74 | 0.23 | 0.03 | 0.03 | 0.08 | -0.11 | -0.11 |
| 儿歌 | 0.09 | -0.14 | -0.14 | 0.87 | 0.58 | 0.58 | 0.63 | 0.36 | 0.36 |
| 草原风 | 0.16 | 0.07 | 0.07 | -0.41 | -0.46 | -0.46 | 0.82 | 0.68 | 0.68 |

Item矩阵

矩阵分解

✓ 隐向量

✎ 其实就是特征的高维表达，只不过很难去理解

✎ 例如用户的隐向量可以想象成是这个样子：

| | | | | |
|-----------|------------------|-----------------|------------------|------|
| Alice | = 10% Action fan | +10% Comedy fan | +50% Romance fan | +... |
| Bob | = 50% Action fan | +30% Comedy fan | +10% Romance fan | +... |
| Titanic | = 20% Action | +00% Comedy | +70% Romance | +... |
| Toy Story | = 30% Action | +60% Comedy | +00% Romance | +... |

矩阵分解

✓ 隐向量

✎ 用户与商品向量可以当做其特征表示

✎ 这可不是随机值，可以观察下数值特点

✎ 不同颜色表示特征鲜明的地方，也就是喜好

| | 成都 | 董小姐 | 安河桥 | 洗白白 | 抓泥鳅 | 小白兔 | 西海情歌 | 青藏高原 | 呼伦贝尔 |
|-----|------|-------|-------|-------|-------|-------|------|-------|-------|
| 民谣 | 0.99 | 0.74 | 0.74 | 0.23 | 0.03 | 0.03 | 0.08 | -0.11 | -0.11 |
| 儿歌 | 0.09 | -0.14 | -0.14 | 0.87 | 0.58 | 0.58 | 0.63 | 0.36 | 0.36 |
| 草原风 | 0.16 | 0.07 | 0.07 | -0.41 | -0.46 | -0.46 | 0.82 | 0.68 | 0.68 |

Item矩阵

| | 民谣 | 儿歌 | 草原风 |
|------|-------|-------|-------|
| 用户1 | 0.93 | -0.08 | 0.10 |
| 用户2 | 0.93 | -0.08 | 0.10 |
| 用户3 | 0.90 | -0.21 | 0.07 |
| 用户4 | 0.93 | -0.08 | 0.10 |
| 用户5 | 0.93 | -0.08 | 0.10 |
| 用户6 | 0.90 | -0.21 | 0.07 |
| 用户7 | 0.62 | 0.03 | 0.08 |
| 用户8 | 0.11 | 0.81 | -0.51 |
| 用户9 | 0.11 | 0.81 | -0.51 |
| 用户10 | 0.02 | 0.74 | -0.58 |
| 用户11 | 0.11 | 0.81 | -0.51 |
| 用户12 | 0.11 | 0.81 | -0.51 |
| 用户13 | 0.02 | 0.74 | -0.58 |
| 用户14 | 0.13 | 0.56 | -0.28 |
| 用户15 | -0.07 | 0.49 | 0.77 |
| 用户16 | -0.07 | 0.49 | 0.77 |
| 用户17 | -0.17 | 0.40 | 0.78 |
| 用户18 | -0.07 | 0.49 | 0.77 |
| 用户19 | -0.07 | 0.49 | 0.77 |
| 用户20 | -0.17 | 0.40 | 0.78 |
| 用户21 | 0.02 | 0.38 | 0.49 |
| 用户22 | 0.71 | 0.55 | -0.19 |
| 用户23 | 0.61 | 0.38 | 0.53 |
| 用户24 | 0.15 | 0.88 | 0.19 |

User矩阵

矩阵分解

✓ 隐向量

✎ 隐向量真的可以理解吗？通常只是比喻而已，一般难以理解

✎ 例如一个50维的向量，鬼知道它具体表什么含义

✎ 没关系，咱们理解不了无所谓，计算机能更好的理解就可以了

| | | | | | | |
|----|--------|--------|--------|---------|-------|-------|
| 我 | 0.3351 | 0.5545 | 0.6798 | 0.4287 | 0.479 | 0.668 |
| 今天 | 0.357 | 0.68 | 0.066 | 0.4735 | 0.329 | 0.617 |
| 真 | 0.0522 | 0.4642 | 0.714 | 0.04378 | 0.696 | 0.795 |
| 帅气 | 0.2855 | 0.8701 | 0.4188 | 0.10523 | 0.039 | 0.94 |

矩阵分解

✓ 矩阵分解

✎ 目标其实就是得到一个大表

✎ 分解后的矩阵还原回这个大表即可

✎ 数值即表示对当前商品喜好程度

✎ 方法蛮简单，具体怎么做呢？

| | 成都 | 董小姐 | 安河桥 | 洗白白 | 抓泥鳅 | 小白兔 | 西海情歌 | 青藏高原 | 呼伦贝尔 |
|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 用户1 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 |
| 用户2 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 |
| 用户3 | 0.88 | 0.69 | 0.69 | -0.01 | -0.13 | -0.13 | -0.01 | -0.13 | -0.13 |
| 用户4 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 |
| 用户5 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 |
| 用户6 | 0.88 | 0.69 | 0.69 | -0.01 | -0.13 | -0.13 | -0.01 | -0.13 | -0.13 |
| 用户7 | 0.64 | 0.46 | 0.46 | 0.14 | 0.00 | 0.00 | 0.14 | 0.00 | 0.00 |
| 用户8 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 |
| 用户9 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 |
| 用户10 | -0.01 | -0.13 | -0.13 | 0.88 | 0.69 | 0.69 | -0.01 | -0.13 | -0.13 |
| 用户11 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 |
| 用户12 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 | 0.10 | -0.07 | -0.07 |
| 用户13 | -0.01 | -0.13 | -0.13 | 0.88 | 0.69 | 0.69 | -0.01 | -0.13 | -0.13 |
| 用户14 | 0.14 | 0.00 | 0.00 | 0.64 | 0.46 | 0.46 | 0.14 | 0.00 | 0.00 |
| 用户15 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 |
| 用户16 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 |
| 用户17 | -0.01 | -0.13 | -0.13 | -0.01 | -0.13 | -0.13 | 0.88 | 0.69 | 0.69 |
| 用户18 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 |
| 用户19 | 0.10 | -0.07 | -0.07 | 0.10 | -0.07 | -0.07 | 0.93 | 0.70 | 0.70 |
| 用户20 | -0.01 | -0.13 | -0.13 | -0.01 | -0.13 | -0.13 | 0.88 | 0.69 | 0.69 |
| 用户21 | 0.14 | 0.00 | 0.00 | 0.14 | 0.00 | 0.00 | 0.64 | 0.46 | 0.46 |
| 用户22 | 0.72 | 0.43 | 0.43 | 0.72 | 0.43 | 0.43 | 0.25 | -0.01 | -0.01 |
| 用户23 | 0.72 | 0.43 | 0.43 | 0.25 | -0.01 | -0.01 | 0.72 | 0.43 | 0.43 |
| 用户24 | 0.25 | -0.01 | -0.01 | 0.72 | 0.43 | 0.43 | 0.72 | 0.43 | 0.43 |

矩阵分解

✓ 目标函数

✎ 跟回归方程很像：
$$\min_{X,Y} \sum_{r_{ui} \neq 0} (r_{ui} - x_u^T y_i)^2 + \lambda (\sum_u \|x_u\|_2^2 + \sum_i \|y_i\|_2^2)$$

✎ 用户矩阵： $X = [x_1, x_2, \dots, x_N]$ 商品矩阵： $Y = [y_1, y_2, \dots, y_M]$

✎ N为用户个数，M为商品个数，还需注意隐向量维度

✎ 其中还额外引入了正则化惩罚项

矩阵分解

✓ 后续的改进

✎ 如果用户就特别刁钻，评分都会很低；如果商品本身就很好，评分都较高

✎ 这里还需要注意的就是用户与商品的本身属性信息，之前公式中未有涉及

✎ 在原公式中 $\min_{X,Y} \sum_{r_{ui} \neq 0} (r_{ui} - x_u^T y_i)^2 + \lambda (\sum_u \|x_u\|_2^2 + \sum_i \|y_i\|_2^2)$ 分别加入用户与商品偏置项

✎ 例如bu表示用户偏置，bi表示商品偏置

矩阵分解

✓ 隐式情况分析

✎ 用户-商品的评分矩阵做起来非常直接，但是哪有那么正好的事啊

✎ 通常收集的数据都是用户的行为：观看时间，点击次数等指标

✎ 这种数据该怎么求解呢？首先定义置信度： $c_{ui} = 1 + \alpha r_{ui}$

✎ 置信度默认为1，表示用户没有产生行为的商品；行为越多，置信度越大

矩阵分解

✓ 隐式情况分析

✎ 重新定义评分: $p_{ui} = \begin{cases} 1 & r_{ui} > 0 \\ 0 & r_{ui} = 0 \end{cases}$ (有行为的则评分为1)

✎ 新的优化目标: $G(x_*, y_*) = \left(\sum_{u,i} c_{ui} (p_{ui} - x_u^T y_i)^2 \right) + \lambda \left(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \right)$

✎ 总结起来就是置信度越大的你得预测的越准, 要不损失就大了

✎ 求解过程依旧交替使用最小二乘法, 固定Y优化X, 再固定X优化Y

矩阵分解

✓ Embedding的作用

✍ 无处不在的Embedding，Ai的核心其实就是让计算机能更懂我！

✍ NLP,CV领域做得太多啦，推荐中也不例外，Embedding做好啦一切都解决了！

