推荐系统解决了什么问题

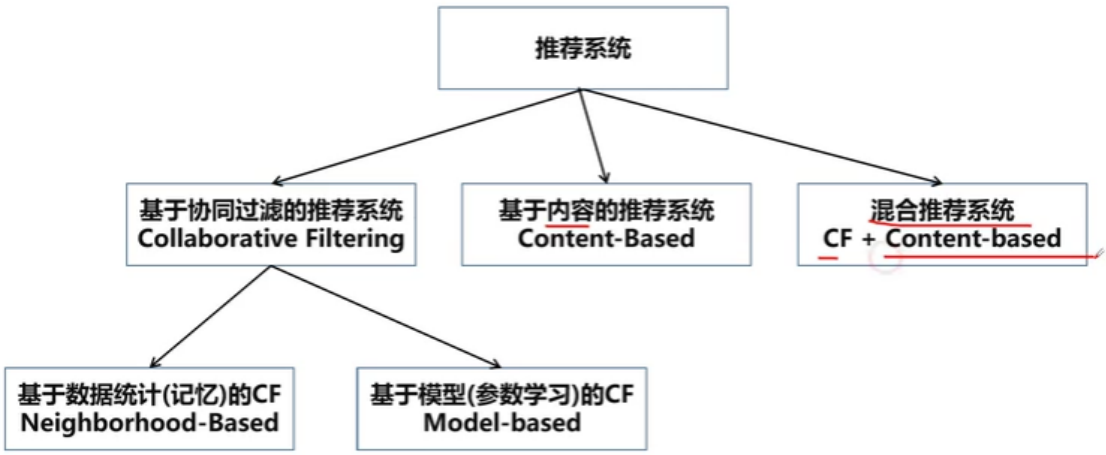
1. 信息过载，用户需求不明确
2. 挖掘长尾，增加商业价值

搜索 vs 推荐

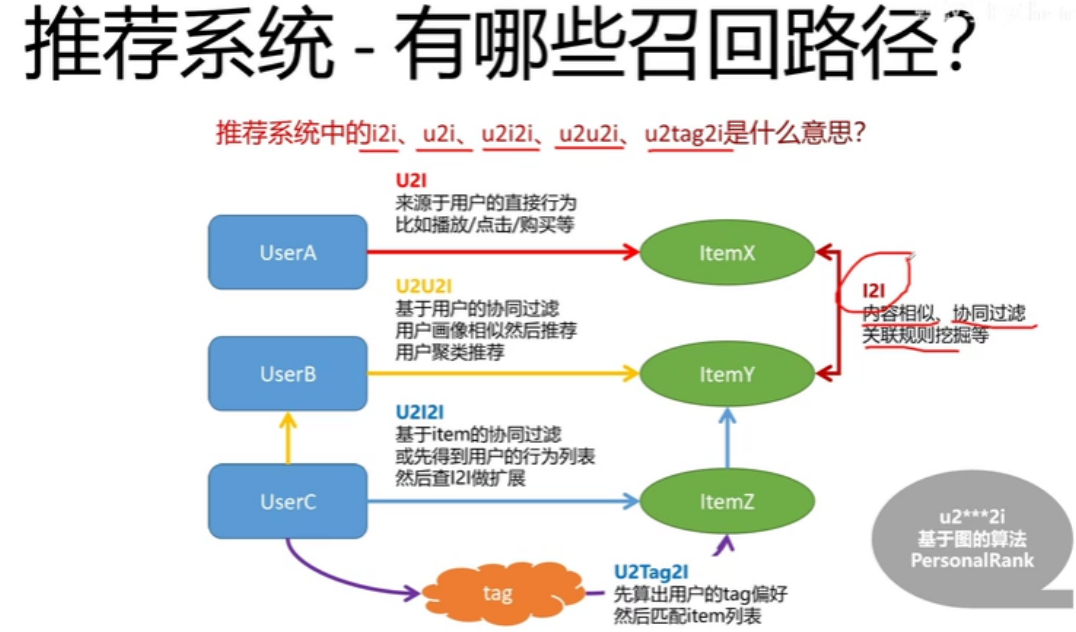
	搜索	推荐
行为方式	主动	被动
意图	明确	模糊
个性化	弱	强
流量分布	马太效应	长尾效应
目标	快速满足	持续服务
评估指标	简明	复杂

推荐系统包含哪些环节？

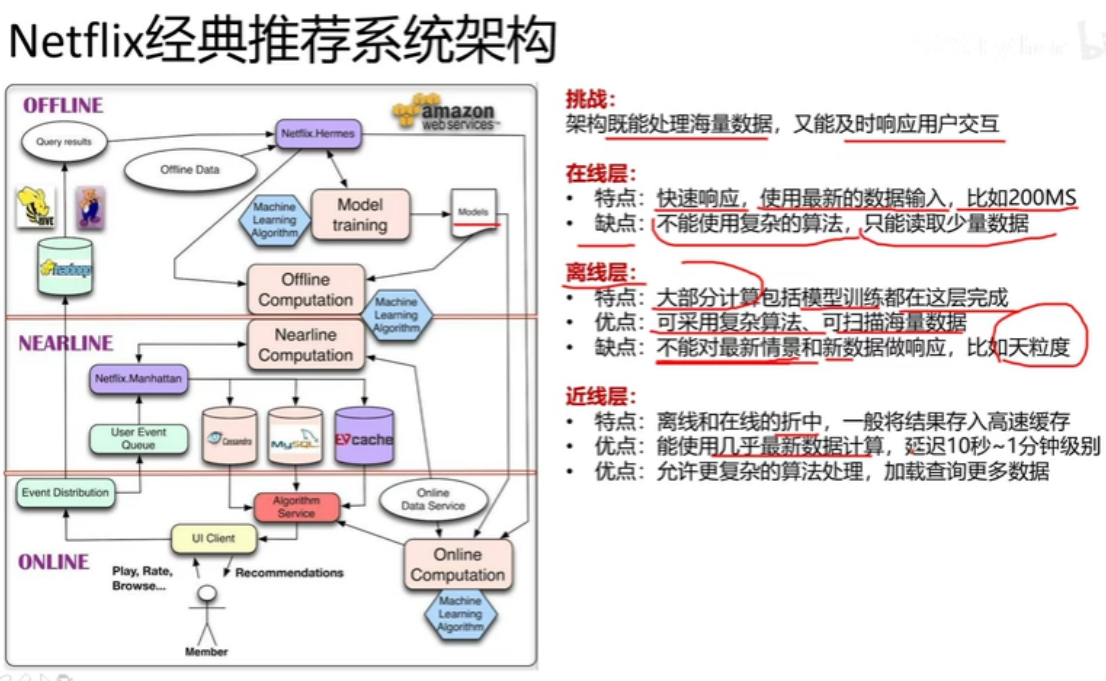
推荐系统的分类



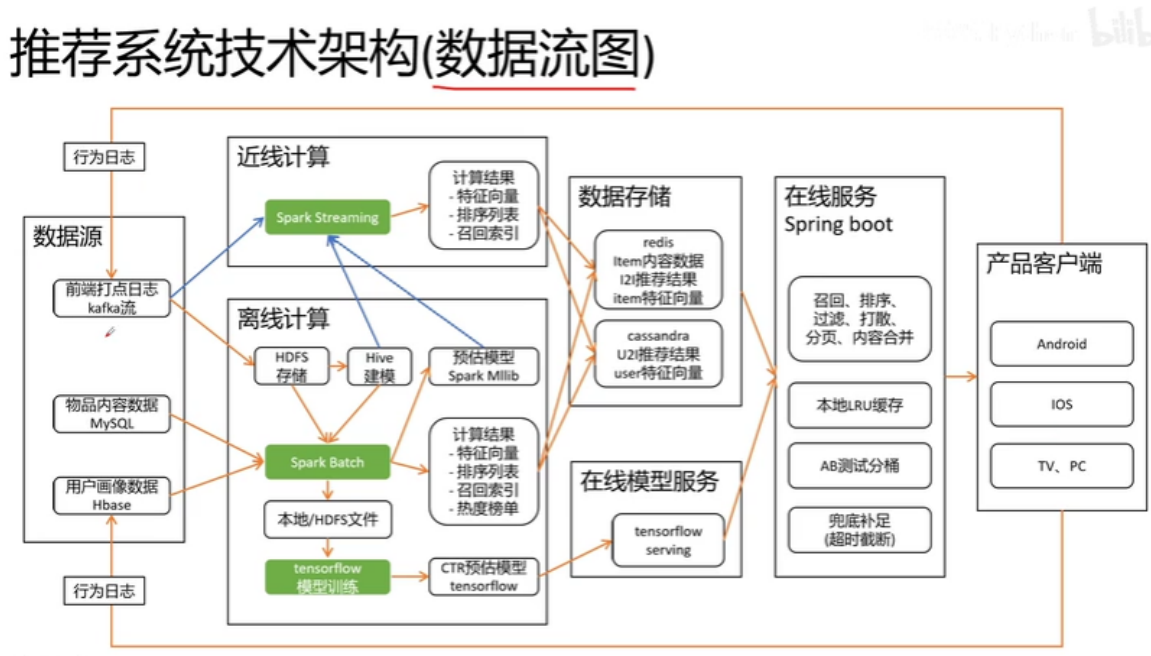
推荐系统有哪些召回路径?



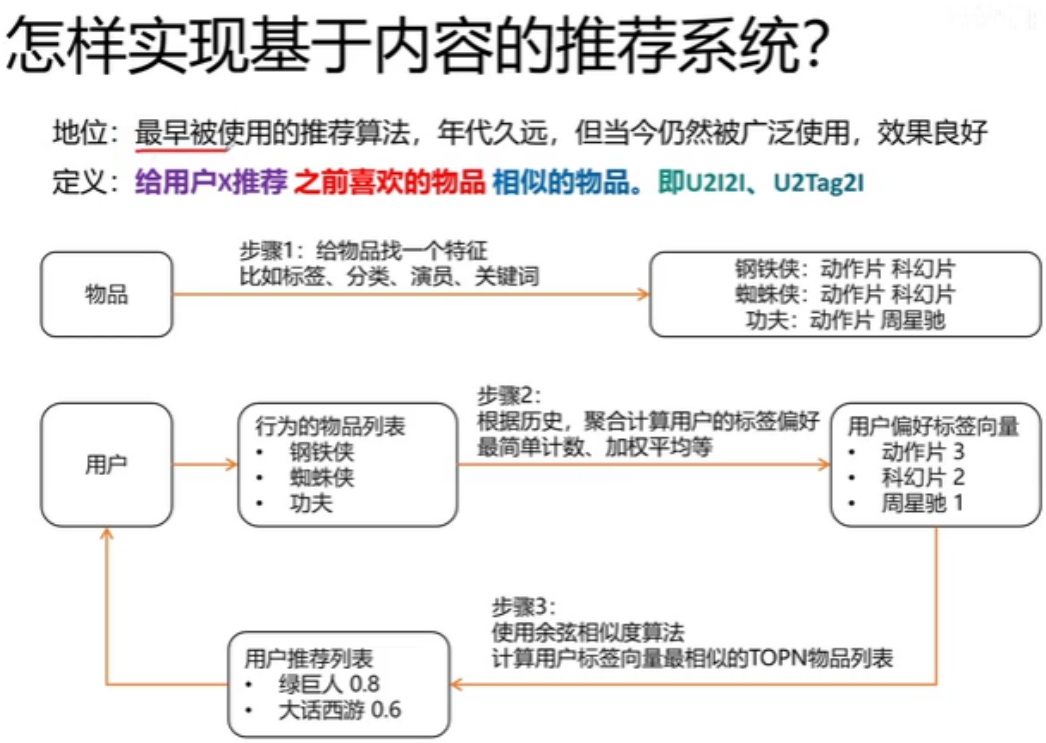
netflix技术架构



通用技术架构



基于内容的推荐系统



怎样实现用户向量和物品向量的相似度计算?

$$\cos\theta = \frac{\sum_{i=1}^{n}(A_i \times B_i)}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

用户向量：[(动作片, 3), (科幻片, 2), (周星驰, 1)]
物品向量：绿巨人[(动作片, 1), (科幻片, 1), (周星驰, 0)]

分子：3*1 + 2*1 + 1*0 = 5
分母：根号(3*3+2*2+1*1) * 根号(1*1 + 1*1 + 0*0) = 5.29

相似度为：5/5.29 = 0.94

即用户向量和物品向量的相似度值为0.94

优缺点

优点:

- 不需要其他用户的数据
- 能给具备独特口味的用户推荐
- 可以推荐最新的、冷门的物品
- 容易做推荐结果的解释

缺点:

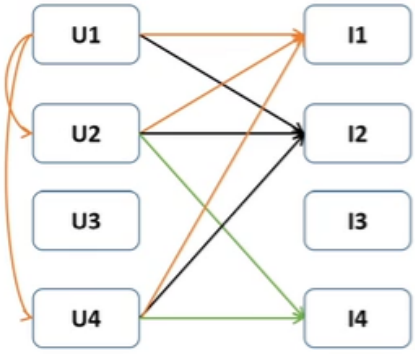
- 很难找到能表达物品的“标签”，有时候需要人工打标签
- 过于局限于自己的世界，无法挖掘出用户的潜在兴趣
- 新用户如果没有行为，没法做推荐

协同过滤

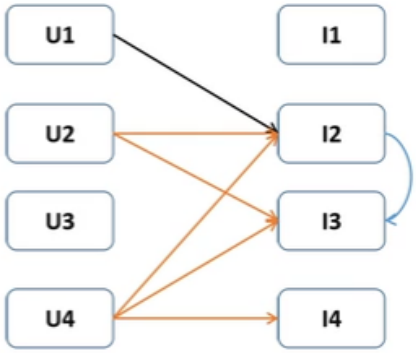
协同过滤 Collaborative Filtering

使用行为数据，利用集体智慧推荐

基于用户的协同过滤-U2U2I：
和你兴趣相投的人也喜欢 xxx



基于物品的协同过滤-U2I2I：
喜欢这个物品的人也喜欢 xxx



实例：基于用户的协同过滤

步骤1：搜索最相似的用户；步骤2：计算u和新item的相似度

		I1	I2	I3	I4	I5	I6	I7
你自己	A	4			5	1		
	B	5	5	4				
	C				2	4	5	
	D		3					3

相似度计算方法：

Jaccard相似度：缺点没有考虑评分

- $sim(A,B) = |r_A \cap r_B| / |r_A \cup r_B|$
- sim(A,B) = 1/5; sim(A,C) = 2/4
  - sim(A,B) < sim(A,C)

余弦相似度：缺点认为缺失值是0

- $sim(A,B) = \cos(r_A, r_B)$
- sim(A,B) = 0.38, sim(A,C) = 0.32

Sim(A,B)>sim(A, C)

皮尔逊相关系数：缺失值是平均值

$$r = \frac{\sum_{i=1}^{n}(X_i-\overline{X})(Y_i-\overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i-\overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i-\overline{Y})^2}}.$$

- $sim(A,B) = \cos(r_A, r_B)$ = 0.09; sim(A,C) = -0.56
  - sim(A,B) > sim(A,C)

实例：基于用户的协同过滤

步骤1：搜索最相似的用户；**步骤2：计算u和新item的相似度**

		I1	I2	I3	I4	I5	I6
你自己	A	4		5	1		
	B	5	5	4		3	
	C			2	4	5	
	D		2				3

$r_{xi} = 1/k \sum_{y \in N} r_{yi}$

方法1：直接求平均
sim(A,I5) = (3+5)/2

$r_{xi} = \sum_{y \in N} s_{xy} r_{yi} / \sum_{y \in N} s_{xy}$

where $s_{xy} = sim(x,y)$

方法2：加权平均
sim(A, I5) = (3*0.38+ 5*0.32) / (0.38 + 0.32)

计算出A和I2/I5/I6的相似度，排序取TOPN就是推荐列表

多路召唤融合

问题：多路召回怎样融合排序?

背景：

- 一般会使用多个召回策略，互相弥补不足，效果更好，三个臭皮匠顶个诸葛亮
- 每个策略之间毫不相关，一般可以编写并发多线程同时执行

问题：怎样将多个召回列表融合成一个有序列表?

几种多路召回结果融合的方法

举例，几种召回策略返回的列表(Item-ID、权重)分别为：

召回策略X：A 0.9，B 0.8，C 0.7

召回策略Y：B 0.6，C 0.5，D 0.4

召回策略Z：C 0.3，D 0.2，E 0.1

每种召回源CTR计算方法：
展现日志-带召回源：X,Y,Z,X,Z,Y
点记日志-带召回源：点击X

则每种召回的点击数/展现数=CTR

策略：

1、按顺序展示：比如实时>购买数据召回>播放数据召回，则直接展示A、B、C、D、E

2、平均法：分母为召回策略个数，分子为权重加和，C为(0.7+0.5+0.3) / 3，B为(0.8+0.6)/3，

3、加权投票：比如三种策略自己指定权重为0.4、0.3、0.2，则B的权重为(0.4*0.8+0.6*0.3+0*0.2)/(0.4+0.3+0.2)

4、动态加权法：计算X/Y/Z三种召回策略的CTR，作为每天更新的动态加权

5、机器学习权重法：逻辑回归LR分类模型预先离线算好各种召回的权重，然后做加权召回

效果依次变好，按照成本进行选择

AB测试

为什么需要AB Test?

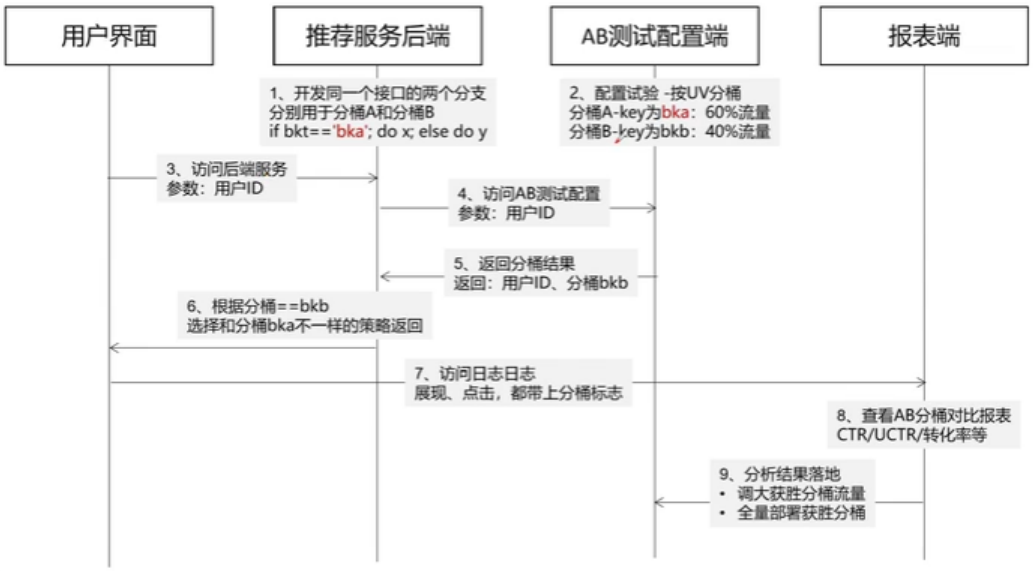
定义：

- A B测试是一种向产品的不同受众展示同一内容的2个或多个变体，并比较哪个变体带来了更多转化的做法。
- AB测试是转化率优化过程的重要方法之一，使用它来收集定性和定量的用户见解，来了解潜在客户并根据该数据优化转化渠道。

为什么需要?

- 想要数据驱动，重点是做AB对比实验，然后模型、策略、设计等不断的迭代更新；
- 进行低风险的修改，先在小流量测试，如果没有问题再调大流量；
- 实现数据统计上的重大改进，降低人工猜测、直觉决策的不确定性；
- 怎样证明自己做的好？工程开发职位和算法职位的重大区分，后者更能用对比数据说话

怎样实现AB测试?



AB测试中的常见错误

不要同时运行太多测试：

要确定测试的优先级，一起测试太多的元素很难确定**哪个元素对测试的成功或失败影响最大。**

实验的流量大小：

流量样本的数量过小，实验结论不能使人信服

测试持续时间不能太短：

运行测试时间过短会导致测试失败或产生不重要的结果

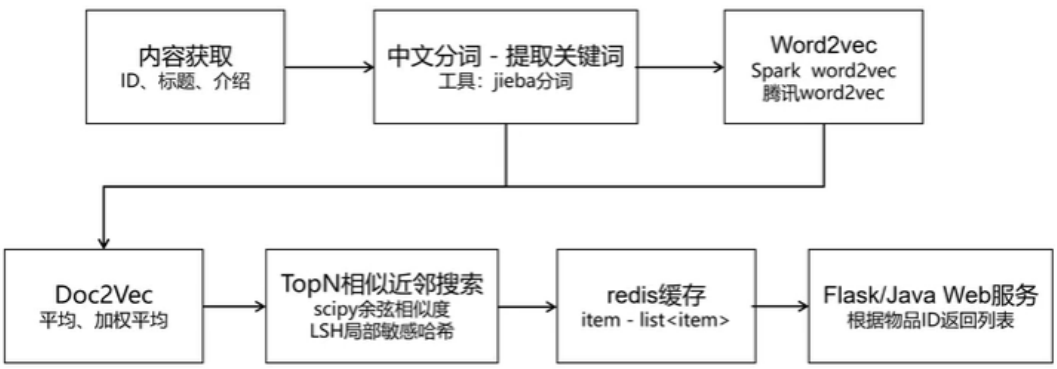
无法遵循迭代过程：

A / B测试是一个迭代过程，每个测试都基于先前测试的结果，不管当前成功或失败，都不要停止继续AB测试

内容相似推荐

怎样实现内容相似推荐

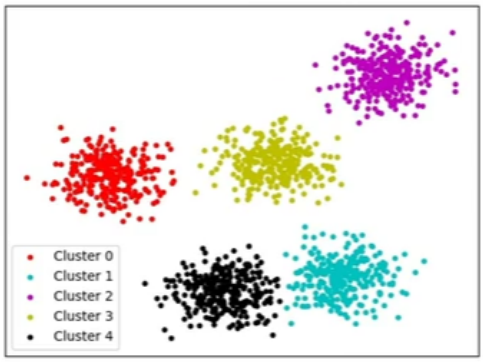
计算物品最相似的其它物品列表，直接用于I2I相似推荐，或者U2I2I扩展推荐



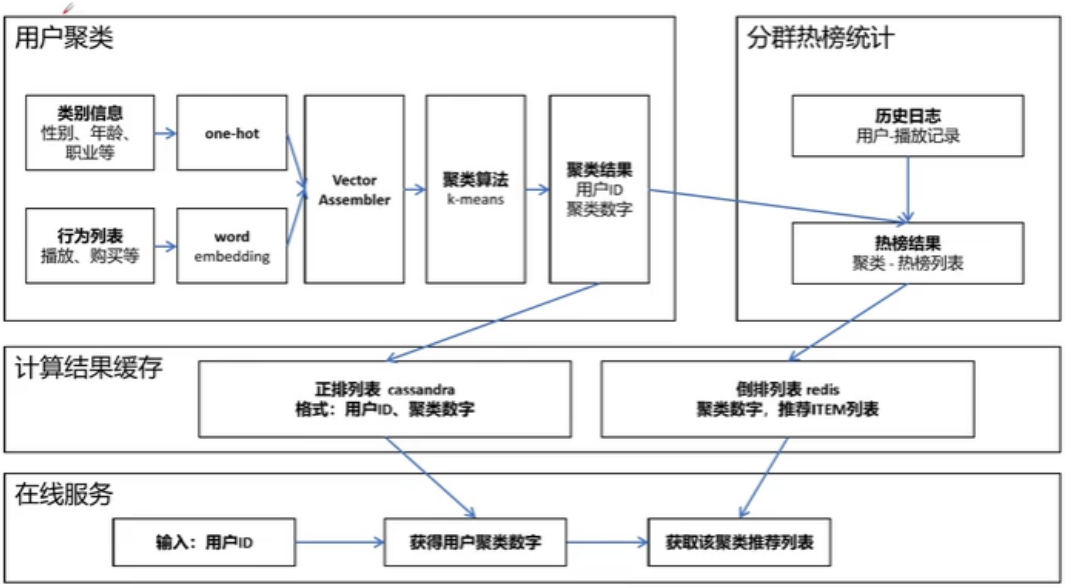
用户聚类推荐

聚类分析（英语：Cluster analysis）亦称为群集分析，是一种数据点分组的机器学习技术。给定一组数据点，可以用聚类算法将每个数据点分到特定的组中。

推荐思路：将用户进行聚类，给每个聚类推荐该人群喜欢的内容。



技术实现流程



优缺点

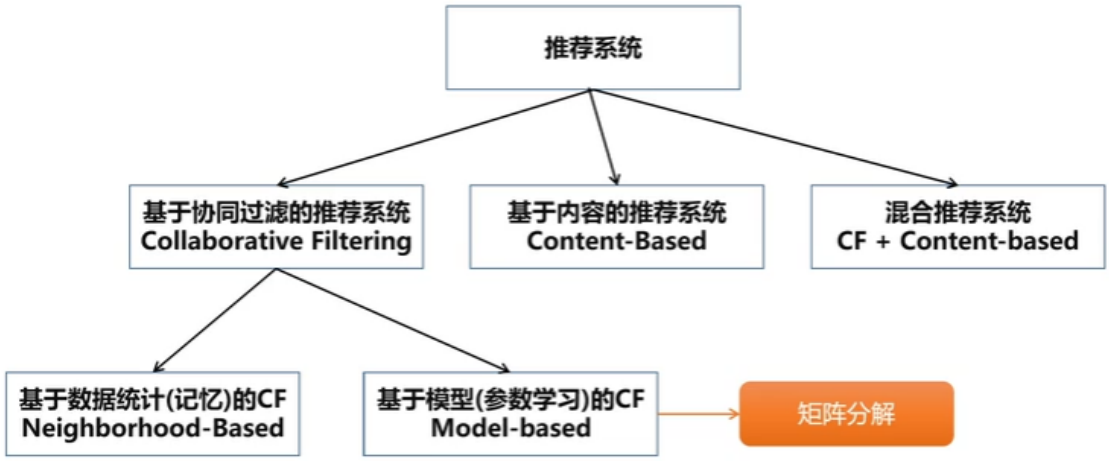
优点：

- 实现简单，spark、sklearn均有现成接口，数据结果存储量很小；
- 可用于新用户冷启动，使用用户注册信息、从站外获取用户信息、行为列表，做聚类即可个性化推荐

缺点：

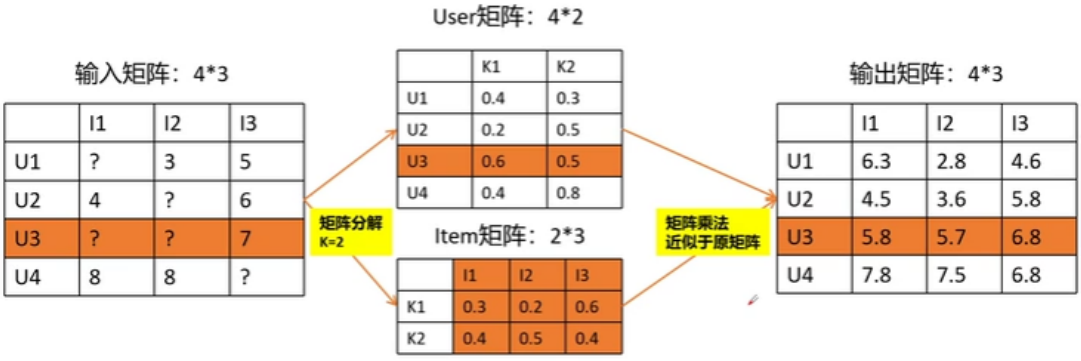
- 精度不高，群体喜欢的内容，并不一定个人喜欢，不够“个性化”

推荐系统分类



矩阵分解

矩阵分解：指将一个矩阵分解成两个或者多个矩阵的乘积，意义层面的解释为通过隐含特征（latent factor）将user兴趣与item特征联系起来；



	I1	I2	I3
U1	?	3	5
U2	4	?	6
U3	?	?	7
U4	8	8	?

	K1	K2
U1	0.4	0.3
U2	0.2	0.5
U3	0.6	0.5
U4	0.4	0.8

	I1	I2	I3
K1	0.3	0.2	0.6
K2	0.4	0.5	0.4

	I1	I2	I3
U1	6.3	2.8	4.6
U2	4.5	3.6	5.8
U3	5.8	5.7	6.8
U4	7.8	7.5	6.8

- 模型训练的目标，是输出矩阵和输入矩阵之间的误差最小
- 输出矩阵的所有单元格都有值，缺失值的填充代表用户评分的预测值
- 模型训练的输出是用户向量和物品向量，都是K维度，代表K个不同的隐含兴趣点
- 用户*用户、用户*物品、物品*物品分别都可以计算彼此的相似度

矩阵分解 - 真实数据流动

用户ID	物品ID	播放量
10001	301	3
10002	302	5
10003	303	8

用户ID	向量
10001	[0.8, 0.7, 0.9]
10002	[0.7, 0.5, 0.6]
10003	[0.3, 0.8, 0.2]

物品ID	向量
301	[0.4, 0.2, 0.4]
302	[0.6, 0.3, 0.2]
303	[0.4, 06, 07]

用户ID	推荐物品列表
10001	302, 303
10002	301, 303
10003	301, 302

物品ID	最相似的物品列表
301	302, 303
302	301, 303
303	301, 302

矩阵分解的优缺点

优点：

- 将高维的矩阵映射成两个低维矩阵的乘积，解决了数据稀疏的问题；
- 预测的精度比较高，高于基于领域的协同过滤以及基于内容等方法；

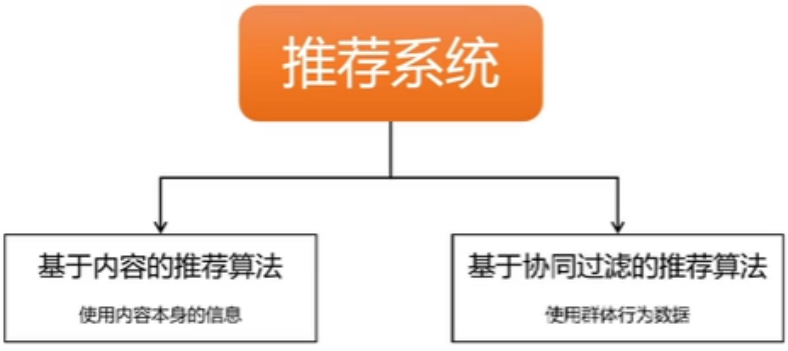
缺点：

- 推荐结果无法解释，其隐空间中的维度无法与现实中的概念对应起来；
- 模型训练比较费时，比如只能天粒度离线训练；

物品冷启动问题

怎样解决物品冷启动问题

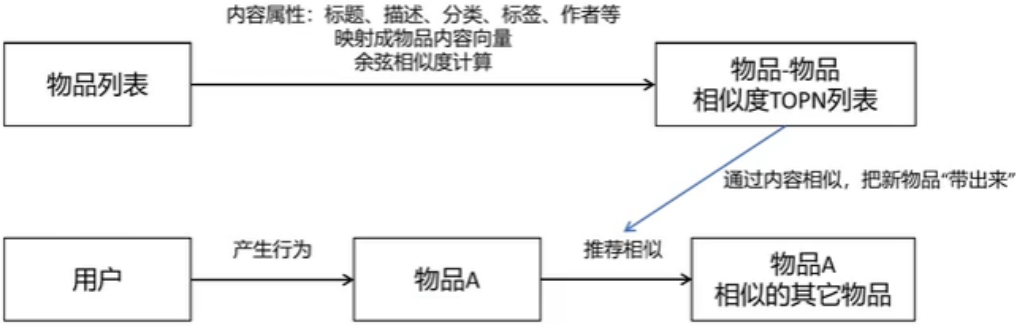
物品冷启动：新加入系统的物品，因缺少行为数据而无法被扩散推荐；
在注重时效性的场景是问题，比如新闻类应用



方法1：物品相似U2I2I　　方法2：抖音的多级流量池算法

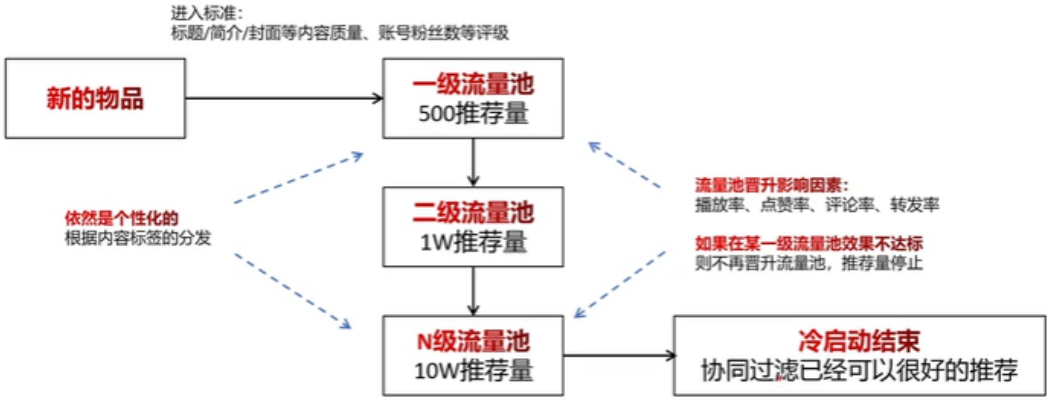
解决物品冷启动问题

方法1：基于物品相似算法的U 2 I 2 I，类似功能：看了还看、相关推荐



解决物品冷启动问题

方法2：抖音内容推荐算法，多级流量池机制，实质上是基于行为方法的试探



API接口

API接口要完成的任务

给某个确定的用户，在他某个使用场景下，推荐他最喜欢的列表

function(用户信息、环境信息、物品信息) = 推荐概率

- 前端传入：用户ID、环境信息，需要API接口传入
- 后端返回：返回的是按概率排序的TOPN物品
- 后端计算：根据用户ID查询用户信息、物品ID查询物品信息、计算用户对每个物品的喜欢概率

API接口设计

整个推荐系统只有一个大接口，通过HTTP+JSON的方式对外提供服务

入参：

- 用户ID
  - 设备号、手机号、微信号、QQ号
- 场景ID
  - 代表某个页面的某个推荐区块
  - 比如首页猜你喜欢、详情页相关推荐
- 物品ID
  - 相关推荐/相似推荐需要
- 环境信息
  - 设备平台、网络类型Wifi/4G、地理位置
- 分页参数
  - pageidx、size

返回：

物品列表

物品信息，比如标题、价格、封面图

每个Item的召回算法，可以多个，reltime@cf

分桶ID

代表某个推荐算法，比如协同过滤、内容推荐、混合推荐等

前端行为上报打点需要带上该ID，跟踪分桶效果

推荐eventid

跟踪单次推荐的展现、点击等效果

前端行为上报打点需要带上该ID

分页信息