推荐算法

个性化推荐算法

- 人口属性
- 地理属性
- 资产属性
- 兴趣属性

推荐算法分支

协同过滤推荐算法

基于内容的推荐算法

混合推荐算法

流行度推荐算法

推荐算法

为推荐系统选择正确的推荐算法是非常重要的决定。目前为止，已经有许多推荐算法可供选择，但为你需要解决的特定问题选择一种特定的算法仍然很困难。每一种推荐算法都有其优点和缺点，当然也有其限制条件，在作出决定之前，你必须要一一考量。在实践中，你可能会测试几种算法，以发现哪一种最适合你的用户，测试中你也会直观地发现它们是什么以及它们的工作原理。

协同过滤算法

基于内存的协同过滤/基于邻域的协同过滤

相似统计的方法得到具有相似兴趣爱好的邻居用户

基于模型的协同过滤

先用历史数据得到一个模型，再用此模型 进行预测。基于模型的推荐广泛使用的技术包括神经网络等学习技术、潜在语义 检索(latent seman2tic indexing)和贝叶斯网络(bayesian networks).

邻域的协同过滤

UCF　　距离算法　　ICF

欧几里得距离（Euclidean Distance）以及欧式距离的标准化（Standardized Euclidean dista

马哈拉诺比斯距离（Mahalanobis Distance）

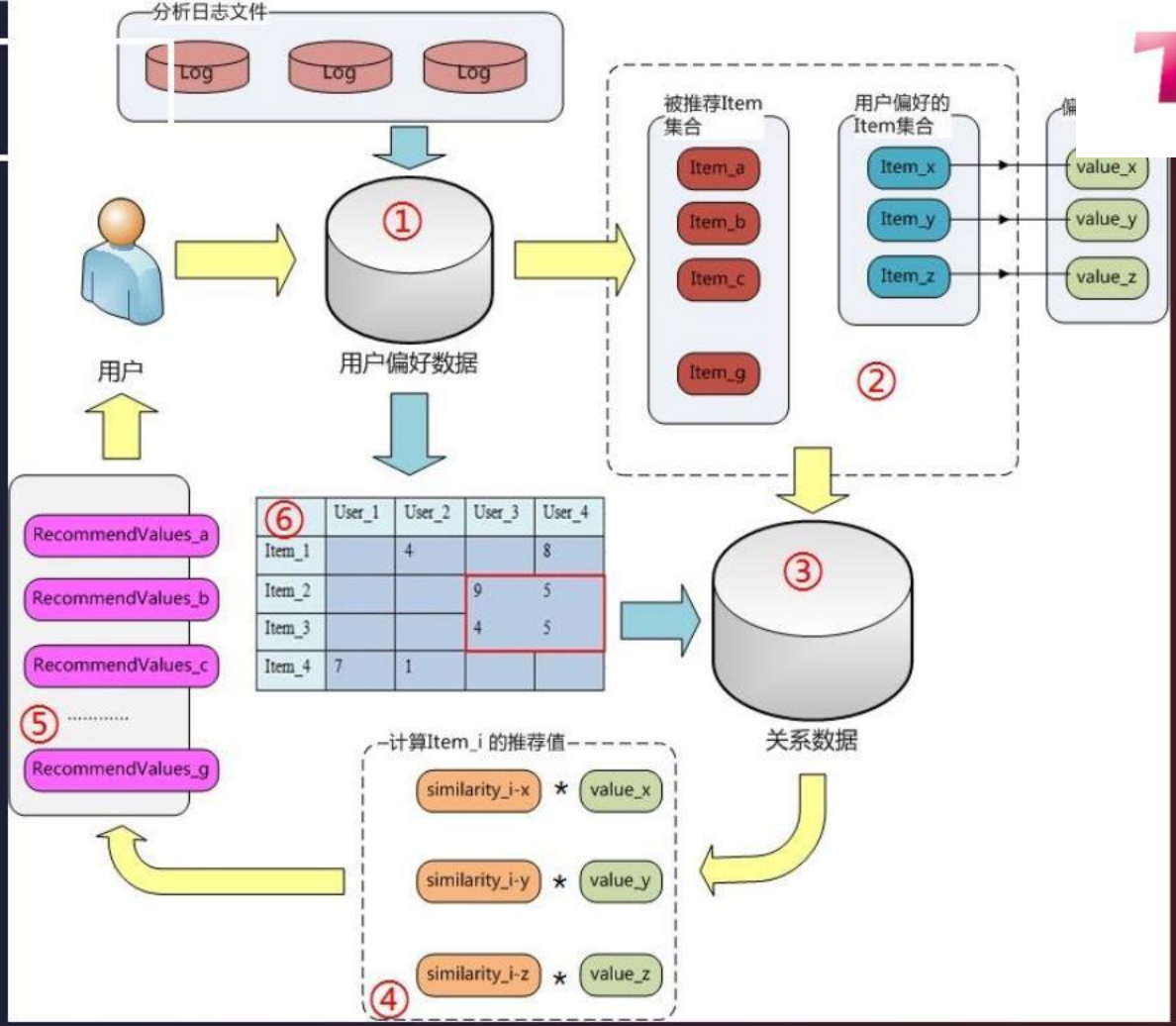
曼哈顿距离（Manhattan Distance）

切比雪夫距离（Chebyshev Distance）

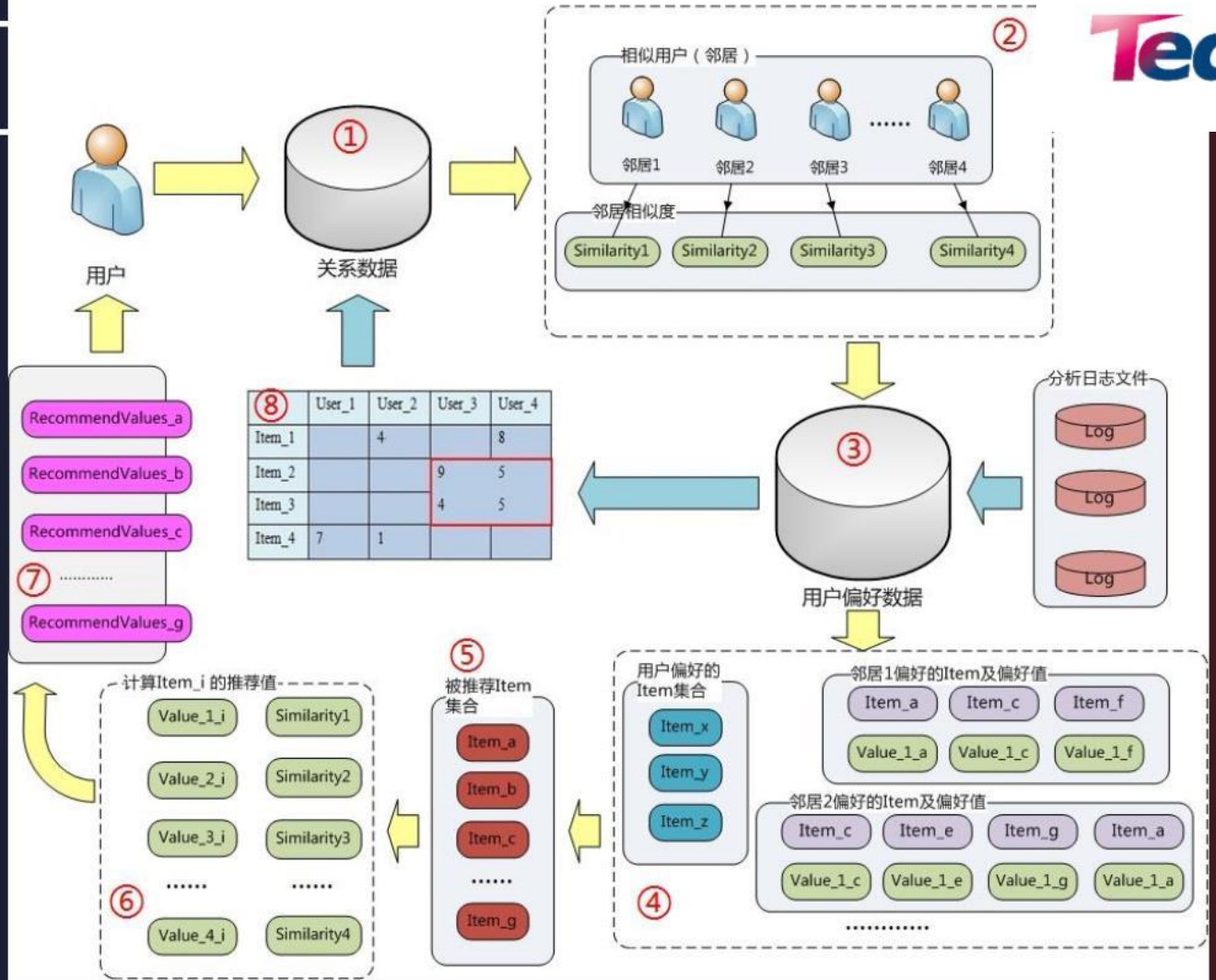
明可夫斯基距离（Minkowski Distance）

海明距离（Hamming distance）

ICF


分析日志文件
Log
Log
Log
①
用户
用户偏好数据
被推荐Item集合
Item_a
Item_b
Item_c
Item_g
用户偏好的Item集合
Item_x
Item_y
Item_z
value_x
value_y
value_z
②
RecommendValues_a
RecommendValues_b
RecommendValues_c
⑤
RecommendValues_g
⑥
User_1 User_2 User_3 User_4
Item_1 4 8
Item_2 9 5
Item_3 4 5
Item_4 7 1
③
关系数据
计算Item_i 的推荐值
similarity_i-x * value_x
similarity_i-y * value_y
similarity_i-z * value_z
④

UCF


①
关系数据
用户
②
相似用户（邻居）
邻居1
邻居2
邻居3
邻居4
邻居相似度
Similarity1
Similarity2
Similarity3
Similarity4
③
用户偏好数据
分析日志文件
Log
Log
Log
④
用户偏好的Item集合
Item_x
Item_y
Item_z
邻居1偏好的Item及偏好值
Item_a
Item_c
Item_f
Value_1_a
Value_1_c
Value_1_f
邻居2偏好的Item及偏好值
Item_c
Item_e
Item_g
Item_a
Value_1_c
Value_1_e
Value_1_g
Value_1_a
⑤
被推荐Item集合
Item_a
Item_b
Item_c
Item_g
计算Item_i 的推荐值
Value_1_i
Similarity1
Value_2_i
Similarity2
Value_3_i
Similarity3
Value_4_i
Similarity4
⑥
⑦
RecommendValues_a
RecommendValues_b
RecommendValues_c
RecommendValues_g
⑧
User_1
User_2
User_3
User_4
Item_1
Item_2
Item_3
Item_4
4
8
9
5
4
5
7
1

邻域的协同过滤

UCF 相似度 **ICF**

余弦相似度（Cosine Similarity）以及调整余弦相似度（Adjusted Cosine Similarity）

皮尔森相关系数（Pearson Correlation Coefficient）

Jaccard相似系数（Jaccard Coefficient）

Tanimoto系数（广义Jaccard相似系数）

对数似然相似度/对数似然相似率

互信息/信息增益，相对熵/KL散度

信息检索--词频-逆文档频率（TF-IDF）

词对相似度--点间互信息

基于模型的协同过滤

- 用关联算法做协同过滤
- 用聚类算法做协同过滤
- 用分类算法做协同过滤
- 用回归算法做协同过滤
- 用矩阵分解做协同过滤
- 用神经网络做协同过滤
- 用图模型做协同过滤
- 用隐语义模型做协同过滤

关联规则

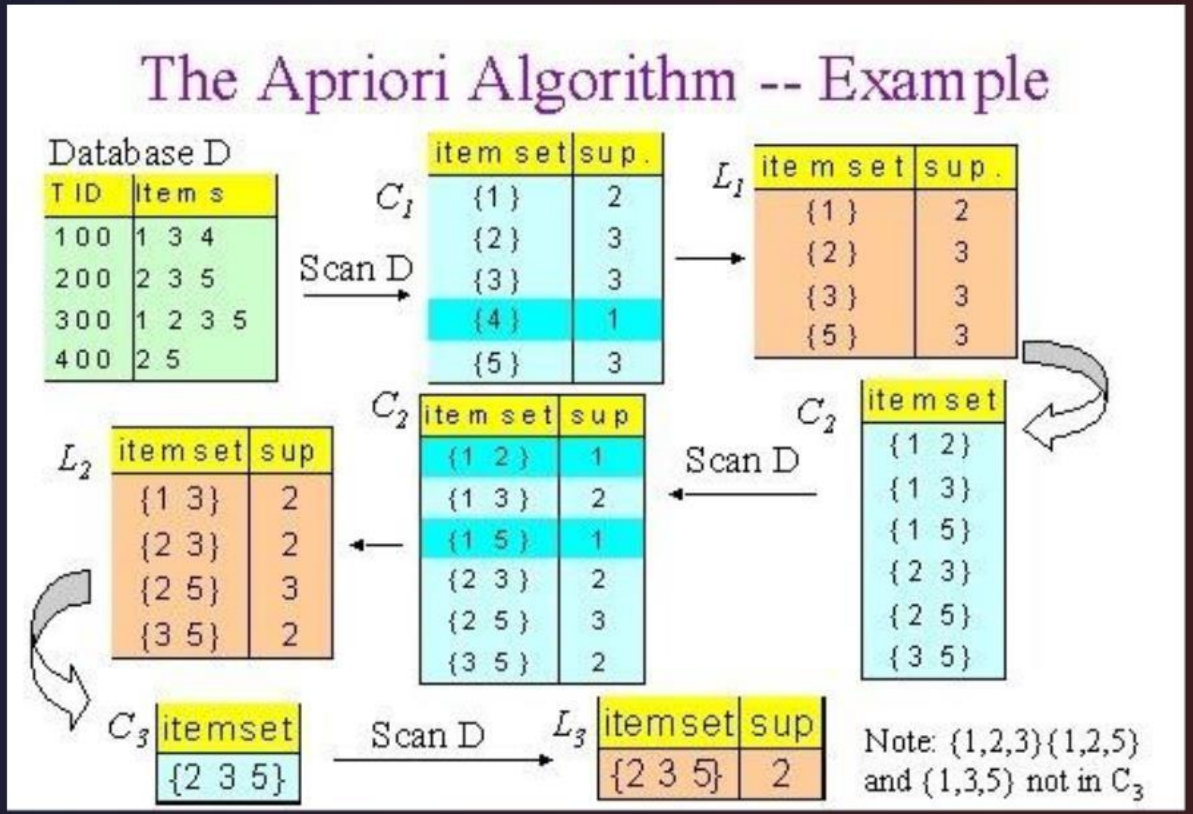
Apriori

Apriori算法是常用的用于挖掘出数据关联规则的算法，它用来找出数据值中频繁出现的数据集合，找出这些集合的模式有助于我们做一些决策。比如在常见的超市购物数据集，或者电商的网购数据集中，如果我们找到了频繁出现的数据集，那么对于超市，我们可以优化产品的位置摆放，对于电商，我们可以优化商品所在的仓库位置，达到节约成本，增加经济效益的目的。

FPGROWTH

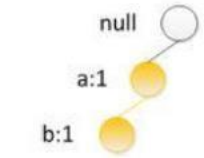
FpGrowth算法通过构造一个树结构来压缩数据记录，使得挖掘频繁项集只需要扫描两次数据记录，而且该算法不需要生成候选集合，所以效率会比较高

关联规则

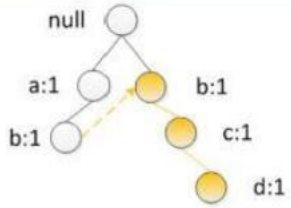

The Apriori Algorithm -- Example
Database D
TID | Items
100 | 1 3 4
200 | 2 3 5
300 | 1 2 3 5
400 | 2 5
Scan D
C1
itemset | sup.
{1} | 2
{2} | 3
{3} | 3
{4} | 1
{5} | 3
L1
itemset | sup.
{1} | 2
{2} | 3
{3} | 3
{5} | 3
C2
itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}
Scan D
C2
itemset | sup
{1 2} | 1
{1 3} | 2
{1 5} | 1
{2 3} | 2
{2 5} | 3
{3 5} | 2
L2
itemset | sup
{1 3} | 2
{2 3} | 2
{2 5} | 3
{3 5} | 2
C3
itemset
{2 3 5}
Scan D
L3
itemset | sup
{2 3 5} | 2
Note: {1,2,3}{1,2,5} and {1,3,5} not in C3

关联规则

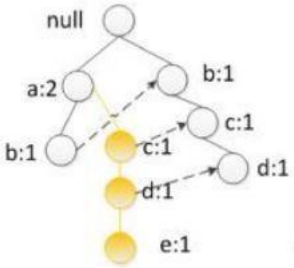
TID	Items
1	{a,b}
2	{b,c,d}
3	{a,c,d,e}
4	{a,d,e}
5	{a,b,c}
6	{a,b,c,d}
7	{a}
8	{a,b,c}
9	{a,b,d}
10	{b,c,e}



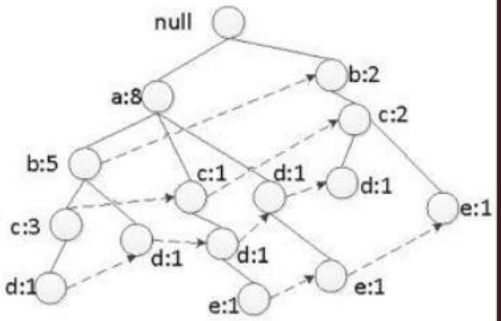
(i) After reading TID =1



(ii) After reading TID =2



(iii) After reading TID =3



(iv) After reading TID =10

Fig. 7. Construction of an FP-tree - Based on [28]

聚类

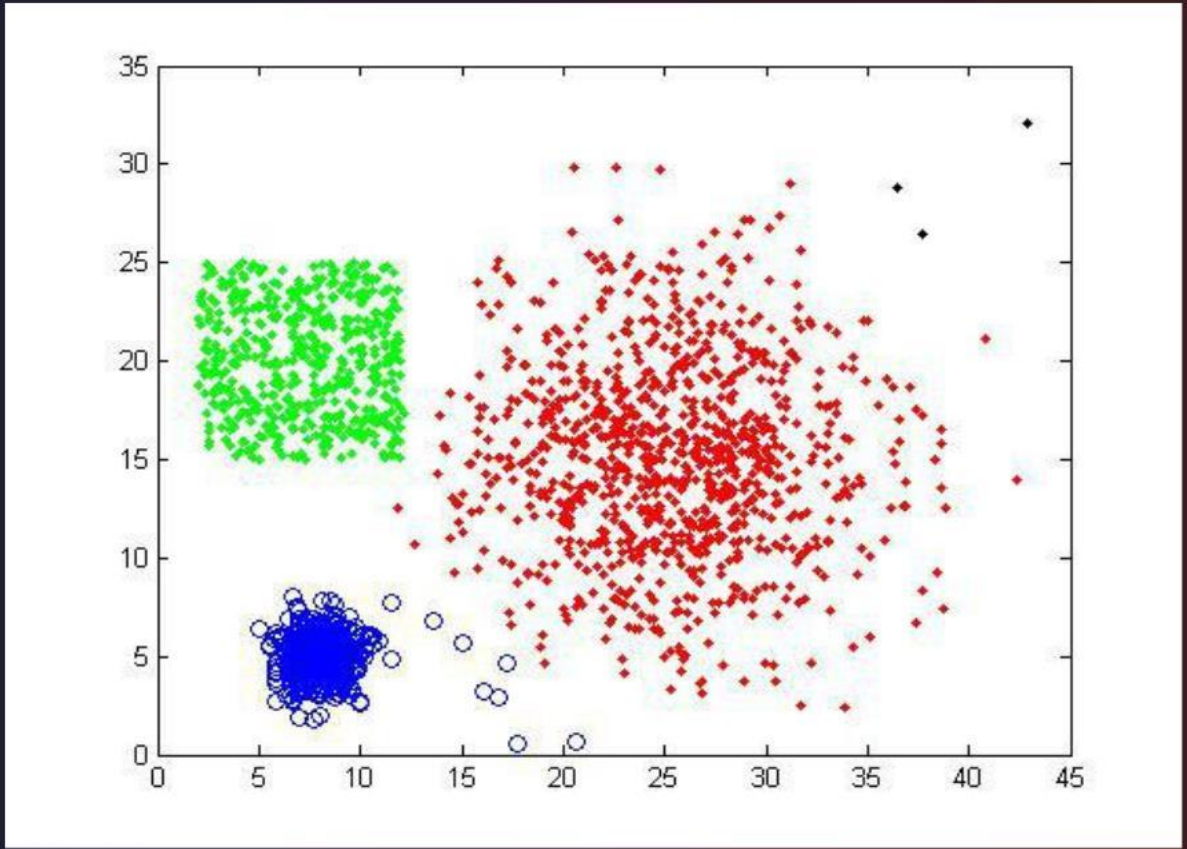
K-Means

K-Means算法是无监督的聚类算法，它实现起来比较简单，聚类效果也不错，因此应用很广泛。K-Means算法有大量的变体，本文就从最传统的K-Means算法讲起，在其基础上讲述K-Means的优化变体方法。包括初始化优化K-Means++, 距离计算优化elkan K-Means算法和大数据情况下的优化Mini Batch K-Means算法。

BIRCH

BIRCH的全称是利用层次方法的平衡迭代规约和聚类（Balanced Iterative Reducing and Clustering Using Hierarchies），名字实在是太长了，不过没关系，其实只要明白它是用层次方法来聚类和规约数据就可以了

聚类



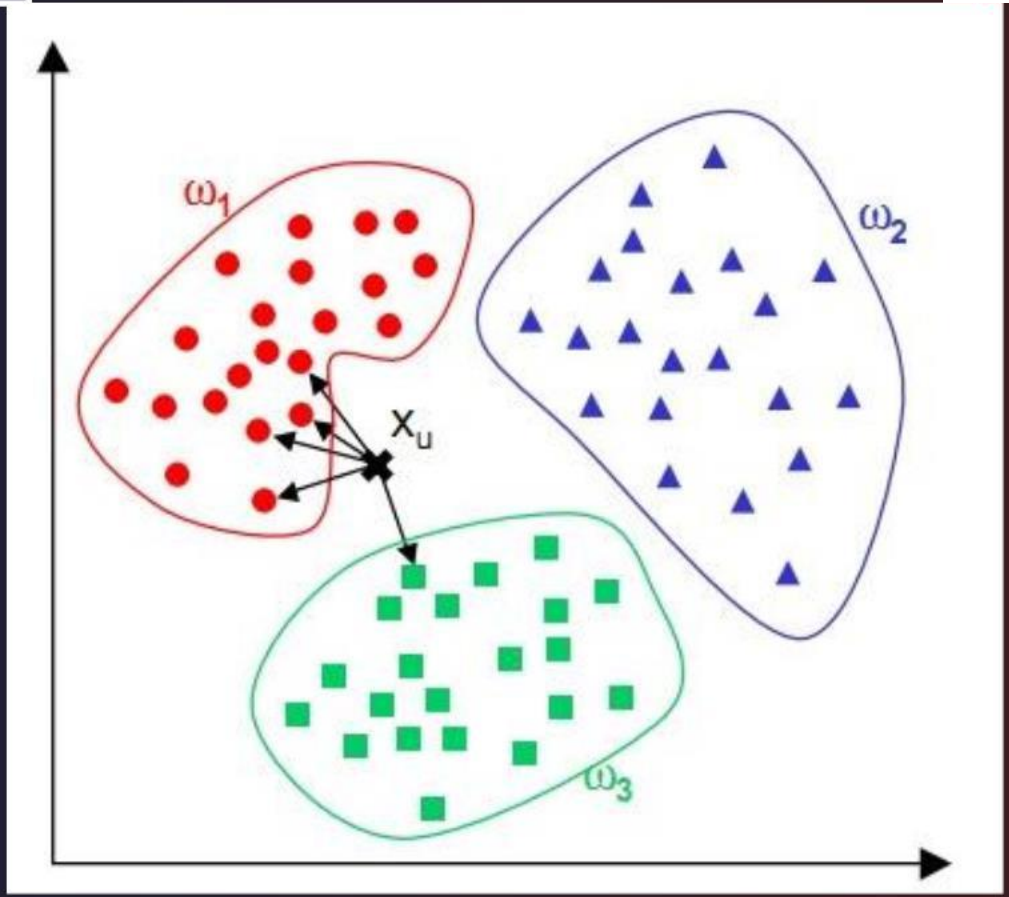
分类

逻辑回归原理

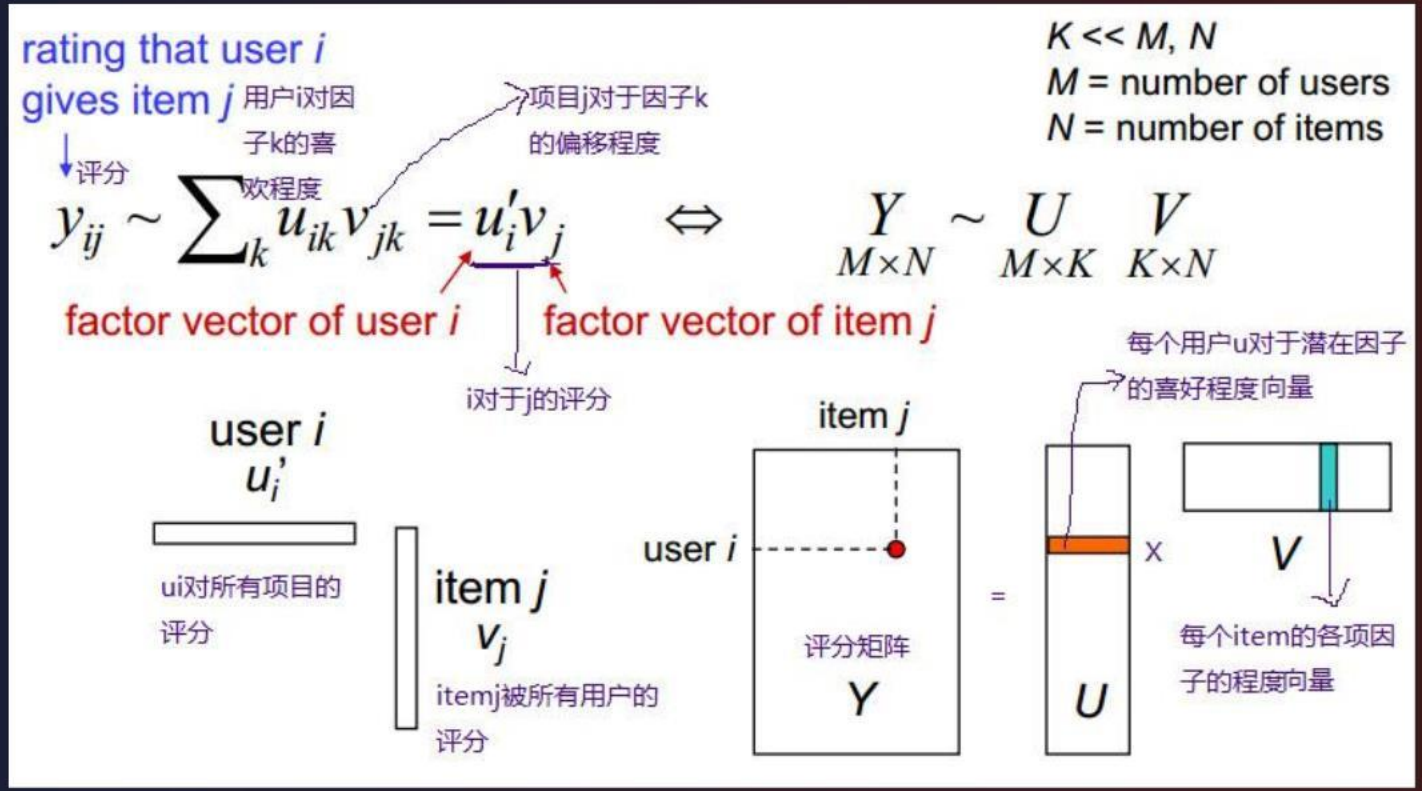
如果我们根据用户评分的高低，将分数分成几段的话，则这个问题变成分类问题。比如最直接的，设置一份评分阈值，评分高于阈值的就是推荐，评分低于阈值就是不推荐，我们将问题变成了一个二分类问题。虽然分类问题的算法多如牛毛，但是目前使用最广泛的是逻辑回归

朴素贝叶斯算法

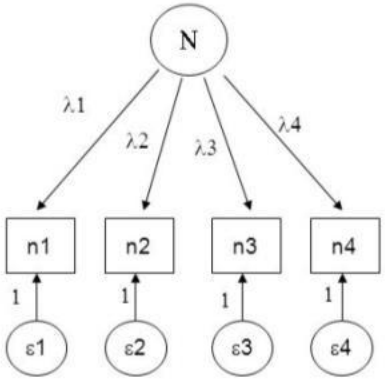
分类



矩阵分解


rating that user i
gives item j
评分
用户i对因子k的喜欢程度
项目j对于因子k的偏移程度
$y_{ij} \sim \sum_k u_{ik} v_{jk} = u_i' v_j$
$\Leftrightarrow$
$Y_{M\times N} \sim U_{M\times K} \; V_{K\times N}$
K << M, N
M = number of users
N = number of items
factor vector of user i
factor vector of item j
i对于j的评分
user i
u_i'
ui对所有项目的评分
item j
v_j
itemj被所有用户的评分
item j
user i
评分矩阵
Y
=
U
X
V
每个用户u对于潜在因子的喜好程度向量
每个item的各项因子的程度向量

矩阵分解之隐语义



Reflective indicators:
They reflect the causal action of the latent variable N

A substantive aspect of the common factor model: interpretation (you bring to the model!)

Strong realistic view of the latent variable N:

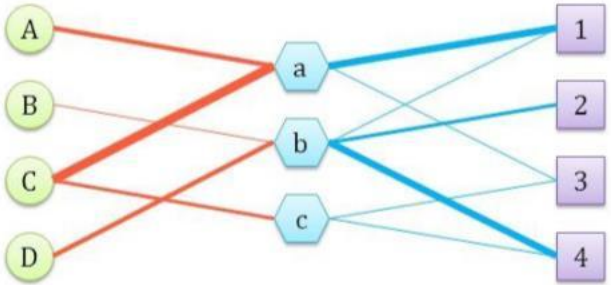
N is **a real, causal, unidimensional** source of individual differences. It **exists beyond the realm of the indicator set**, and is not dependent on any given indicator set.

Causal - part I: **The position of N determines causally the response to the items. N is the only direct cause of systematic variation in the items**. I.e., if you condition on N, then the correlations among the items are zero: local independence.

矩阵分解之隐语义

Latent Factor Model

- Users and items are connect by latent features.



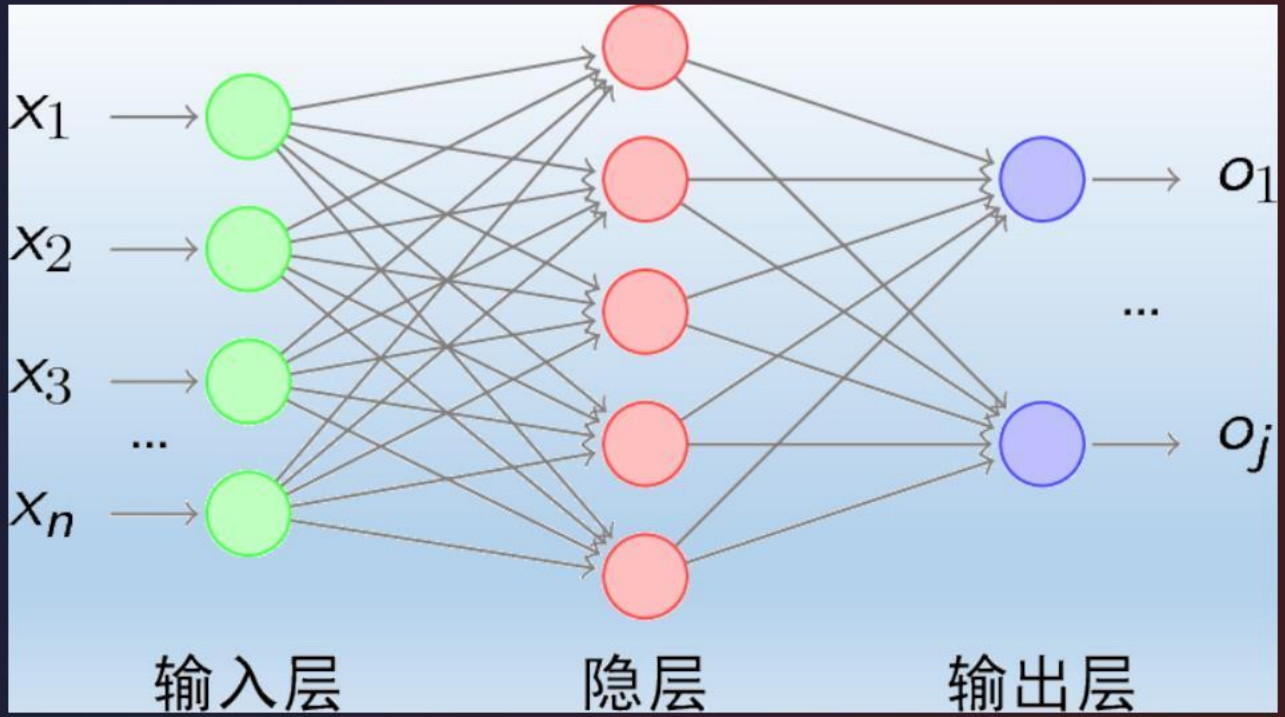
神经网络

2006年，Hinton在《Science》和相关期刊上发表了论文，首次提出了“深度信念网络”的概念。与传统的训练方式不同，“深度信念网络”有一个“预训练”（pre-training）的过程，这可以方便的让神经网络中的权值找到一个接近最优解的值，之后再使用“微调”(fine-tuning)技术来对整个网络进行优化训练。这两个技术的运用大幅度减少了训练多层神经网络的时间。他给多层神经网络相关的学习方法赋予了一个新名词--“深度学习”。

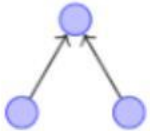
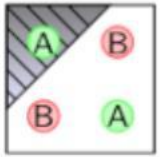
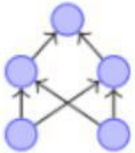
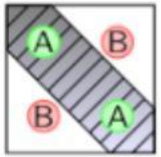
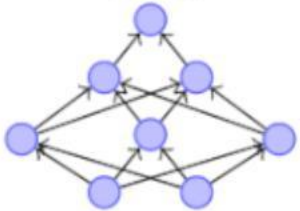
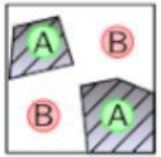
很快，深度学习在语音识别领域暂露头角。接着，2012年，深度学习技术又在图像识别领域大展拳脚。Hinton与他的学生在ImageNet竞赛中，用多层的卷积神经网络成功地对包含一千类别的一百万张图片进行了训练，取得了分类错误率15%的好成绩，这个成绩比第二名高了近11个百分点，充分证明了多层神经网络识别效果的优越性。

图29 Geoffery Hinton

神经网络



神经网络

结构	决策区域类型	区域形状	异或问题
无隐层 	由一超平面分成两个		
单隐层 	开凸区域或闭凸区域		
双隐层 	任意形状（其复杂度由单元数目确定）		

协同过滤优点

- 实现快
- 对商品和用户没有要求
- 效果有保证

协同过滤缺点

- 冷启动
- 马太效应
- 推荐解释模糊

工具

scikit-learn

Machine Learning in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license