大数据高端人才专项计划

东北大学 Northeastern University

HORIZON 昊宸科技

目录

推荐系统出现的原因

随着信息技术和互联网的发展，人们逐渐从信息匮乏的时代走入了信息过载的时代。在这个时代无论信息消费者还是生产者都遇到了极大的挑战：

对于信息消费者，从大量信息中找到自己感兴趣的信息是一件很困难的事；

对于信息生产者，让自己生产的信息脱颖而出，受到广大用户的关注，也是一件非常困难的事情。

推荐系统的基本任务是联系用户和物品，解决信息过载的问题

推荐系统与搜索引擎的异同

- 相同点：

  都是一种帮助用户快速发现有用信息的工具

- 不同点：

  搜索引擎需要用户主动提供准确的关键词来寻找信息

  推荐系统不需要用户提供明确的需求，而是通过分析用户的历史行为给用户的兴趣建模。

- 从某种意义上说，推荐系统和搜索引擎对于用户来说是两个互补的工具

  搜索引擎满足了用户有明确目的时的主动查找需求

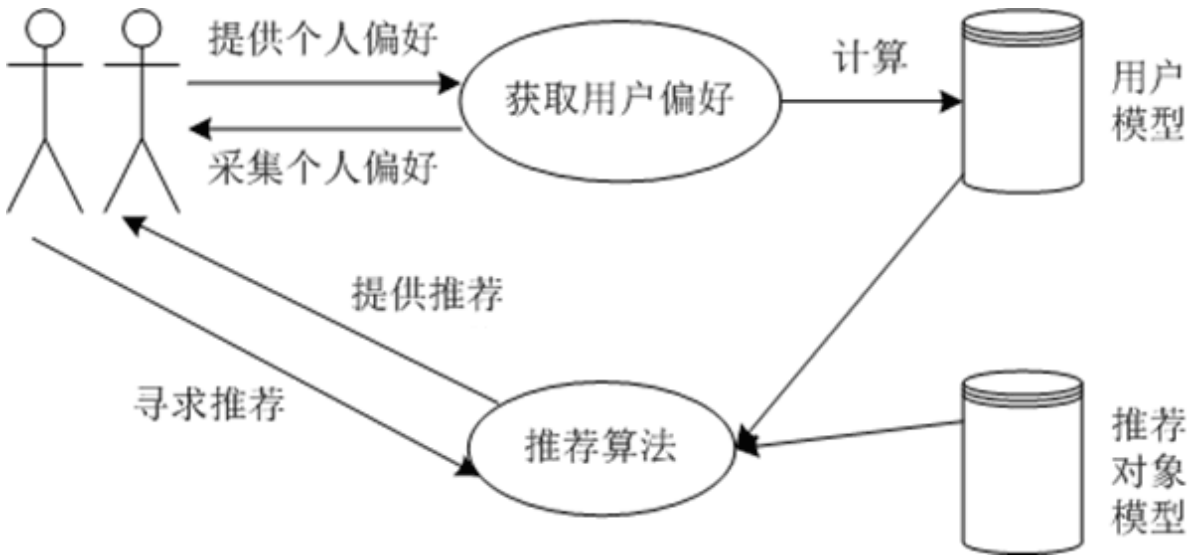
  推荐系统能够在用户没有明确目的时候帮助他们发现感兴趣的内容

推荐系统联系用户和物品的方式

推荐系统联系用户和物品的方式

一个好的推荐系统要给用户提供个性化的、高效的、准确的推荐，那么推荐系统应能够获取反映用户多方面的、动态变化的兴趣偏好。

推荐系统有必要为用户建立一个用户模型 ，该模型能获取、表示、存储和修改用户兴趣偏好，能进行推理对用户进行分类和识别，帮助系统更好地理解用户特征和类别，理解用户的需求和任务， 从而更好地实现用户所需要的功能。



用户行为分析

- 显性反馈行为：用户明确表示对物品喜好的行为
- 隐性反馈行为：不能明确反应用户喜好的行为

	显性反馈数据	隐性反馈数据
用户兴趣	明确	不明确
数量	较少	庞大
存储	数据库	分布式文件系统
实时读取	实时	有延迟
正负反馈	都有	只有正反馈
	显性反馈	隐性反馈
视频网站	用户对视频的评分	用户观看视频的日志、浏览视频页面的日志
电子商务网站	用户对商品的评分	点击、收藏、加购、购买日志
门户网站	用户对新闻的评分	阅读新闻日志
音乐网站	用户对音乐/歌手、专辑的评分	听歌日志

显性反馈和隐性反馈的比较

用户行为分析

➢ 长尾理论

美国《连线》杂志主编Chris Anderson在2004年发表了“The Long Tail”（长尾）一文并于2006年出版了《长尾理论》一书。该书指出，传统的80/20原则（80%的销售额来自于20%的热门品牌）在互联网的加入下会受到挑战。互联网条件下，由于货架成本极端低廉，电子商务网站往往能出售比传统零售店更多的商品。虽然这些商品绝大多数都不热门，但与传统零售业相比，这些不热门的商品数量极其庞大，因此这些长尾商品的总销售额将是一个不可小觑的数字，也许会超过热门商品（即主流商品）带来的销售额。

主流商品往往代表了绝大多数用户的需求，而长尾商品往往代表了一小部分用户的个性化需求。因此，如果要通过发掘长尾提高销售额，就必须充分研究用户的兴趣，而这正是个性化推荐系统主要解决的问题。推荐系统通过发掘用户的行为，找到用户的个性化需求，从而将长尾商品准确地推荐给需要它的用户，帮助用户发现那些他们感兴趣但很难发现的商品。

用户行为分析

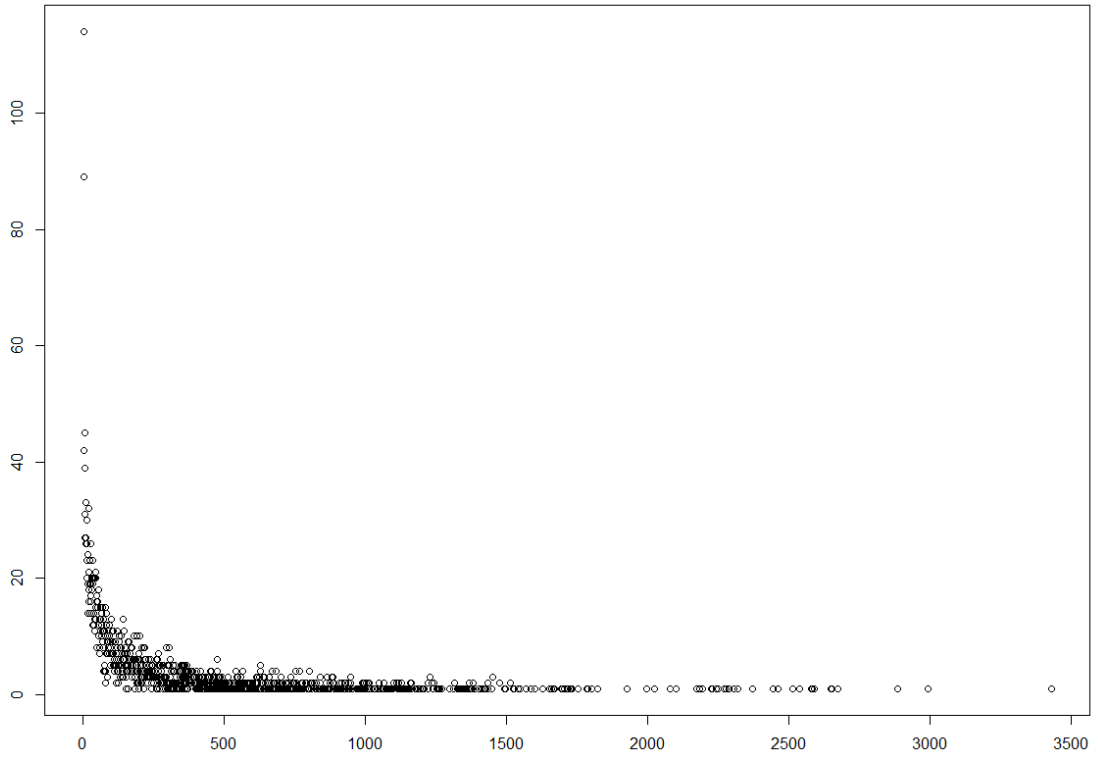
➢ 长尾分布

用户行为数据也蕴含着这种规律。令 $f_u(k)$为对 k 个物品产生过行为的用户数，令 $f_i(k)$为被 k个用户产生过行为的物品数。那么，$f_u(k)$和 $f_i(k)$都满足长尾分布。也就是说：

$$f_i(k) = \alpha_i k^{\beta_i}$$

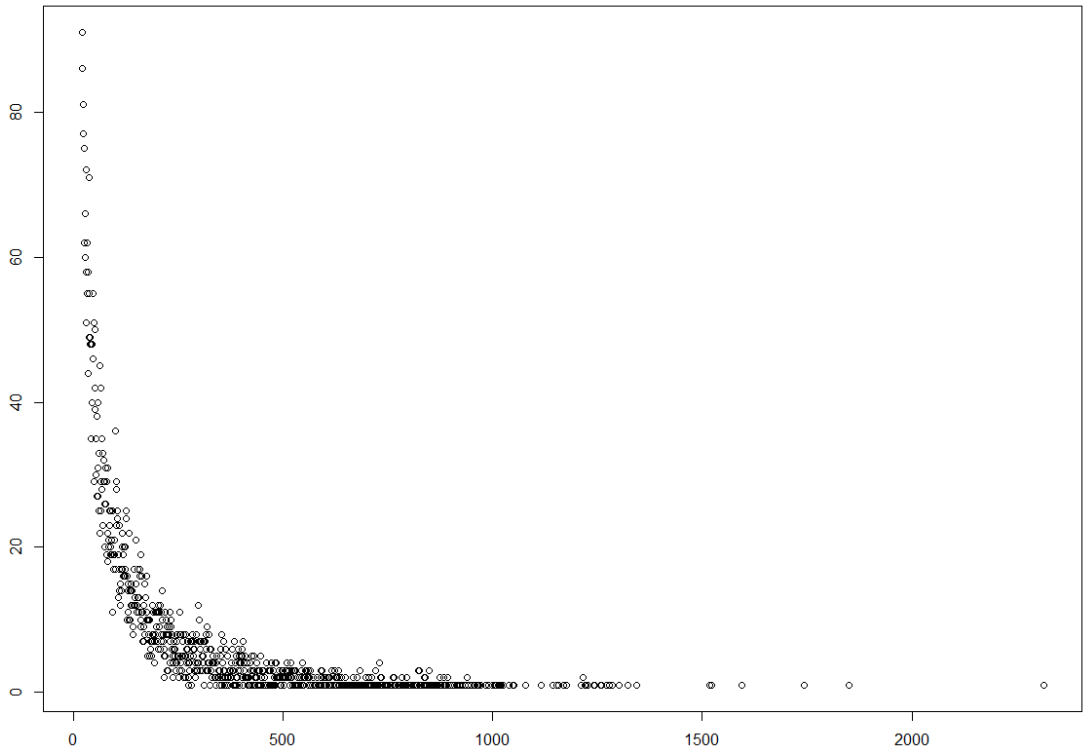
$$f_u(k) = \alpha_u k^{\beta_u}$$

右图展示了movielens数据集中物品流行度的分布曲线。横坐标是物品的流行度K，纵坐标是流行度为K的物品的总数。这里，物品的流行度指对物品产生过行为的用户总数。

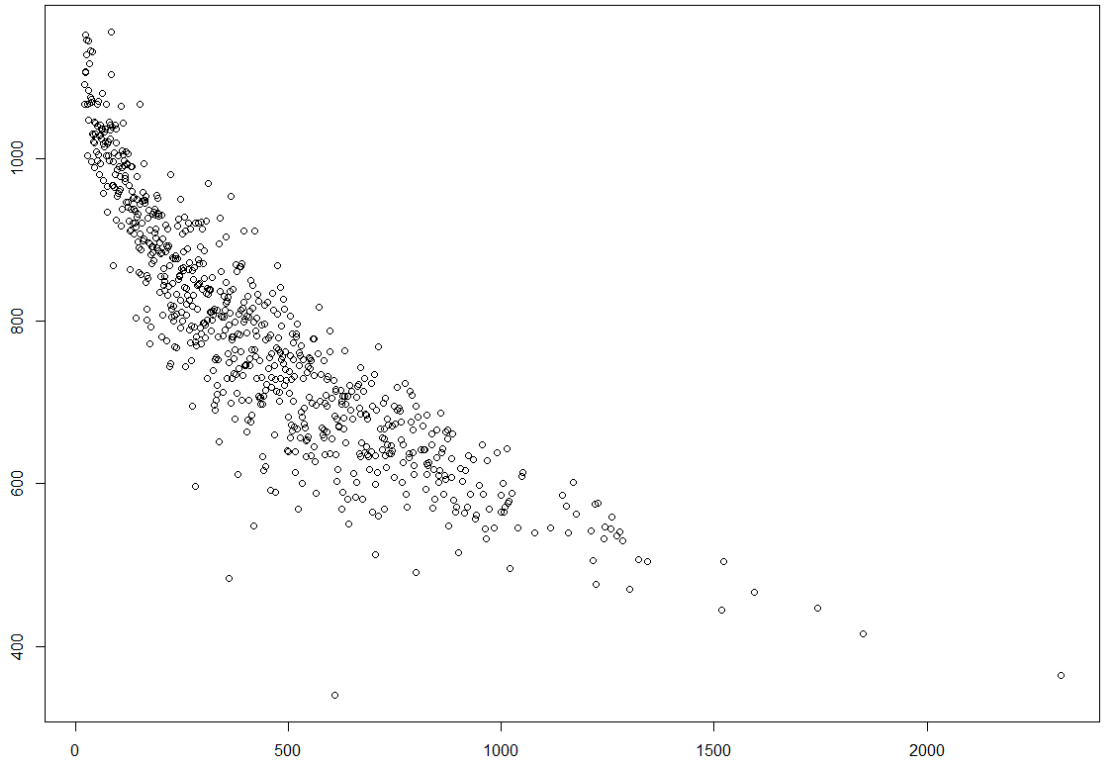


用户行为分析

下图展示了movielens数据集中用户活跃度的分布曲线。横坐标是用户的活跃度K，纵坐标是活跃度为K的用户总数。这里，用户的活跃度为用户产生过行为的物品总数。



下图展示了MovieLens数据集中用户活跃度和物品流行度之间的关系，其中横坐标是用户活跃度，纵坐标是具有某个活跃度的所有用户评过分的物品的平均流行度。如图所示，图中曲线呈明显下降的趋势，这表明用户越活跃，越倾向于浏览冷门的物品。



协同过滤Collaborative Filtering

仅仅基于用户行为数据设计的推荐算法一般称为协同过滤算法。学术界对协同过滤算法进行了深入研究，提出了很多方法，比如基于邻域的方法（neighborhood-based）、隐语义模型（latent factor model）、基于图的随机游走算法（random walk on graph）等。在这些方法中，最著名的、在业界得到最广泛应用的算法是基于邻域的方法，而基于邻域的方法主要包含下面两种算法。

基于用户的协同过滤算法：这种算法给用户推荐和他兴趣相似的其他用户喜欢的物品。

基于物品的协同过滤算法：这种算法给用户推荐和他之前喜欢的物品相似的物品。

实验设计和算法评测

数据集：Movielens1M 数据集
https://grouplens.org/datasets/movielens/
MovieLens是一组从20世纪90年末到21世纪初用户提供的电影评分数据。这些数据中包括电影评分、电影元数据（风格类型和年代）以及关于用户的人口统计学数据（年龄、邮编、性别和职业等）。
MovieLens 1M数据集含有来自约6000名用户对约4000部电影的100多万条评分数据。将该数据从zip文件中解压出来之后共有三个表，分别为用户信息、电影信息和评分。
users.dat,movies.dat,ratings.dat

```
1::F::1::10::48067
2::M::56::16::70072
3::M::25::15::55117
4::M::45::7::02460
5::M::25::20::55455
6::F::50::9::55117
```

users.dat，表头为'user_id', 'gender', 'age', 'occupation', 'zip'

```
1::Toy Story (1995)::Animation|Children's|Comedy
2::Jumanji (1995)::Adventure|Children's|Fantasy
3::Grumpier Old Men (1995)::Comedy|Romance
4::Waiting to Exhale (1995)::Comedy|Drama
5::Father of the Bride Part II (1995)::Comedy
6::Heat (1995)::Action|Crime|Thriller
```

movies.dat，表头为'movie_id', 'title', 'genres'

```
1::1193::5::978300760
1::661::3::978302109
1::914::3::978301968
1::3408::4::978300275
1::2355::5::978824291
1::1197::3::978302268
```

ratings.dat，表头为'user_id', 'movie_id', 'rating', 'timestamp'

实验设计和算法评测

交叉验证：
将用户行为数据集按照均匀分布随机分成M份（本章取M=8），挑选一份作为测试集，将剩下的M-1份作为训练集。然后在训练集上建立用户兴趣模型，并在测试集上对用户行为进行预测，统计出相应的评测指标。为了保证评测指标并不是过拟合的结果，需要进行M次实验，并且每次都使用不同的测试集。然后将M次实验测出的评测指标的平均值作为最终的评测指标。

评测指标：
对用户u推荐N个物品（记为R(u)），令用户u在测试集上喜欢的物品集合为T(u)，然后可以通过**准确率/召回率**评测推荐算法的精度：

$$\text{Recall} = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |T(u)|} \qquad \text{Precision} = \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |R(u)|}$$

召回率描述有多少比例的用户—物品评分记录包含在最终的推荐列表中，而准确率描述最终的推荐列表中有多少比例是发生过的用户—物品评分记录。

基于用户的协同过滤算法

流程：

（1）找到和目标用户兴趣相似的用户集合

（2）找到这个集合中的用户喜欢的，且目标用户没有交互过的物品推荐给目标用户

步骤(1)的关键就是计算两个用户的兴趣相似度。这里，协同过滤算法主要利用行为的相似度计算兴趣的相似度。给定用户u和用户v，令N(u)表示用户u曾经有过正反馈的物品集合，令N(v)为用户v曾经有过正反馈的物品集合。那么，我们可以通过如下的Jaccard公式简单地计算u和v的兴趣相似度

$$w_{uv} = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}$$

或者通过余弦相似度计算：

$$w_{uv} = \frac{|N(u) \cap N(v)|}{\sqrt{|N(u)||N(v)|}}$$

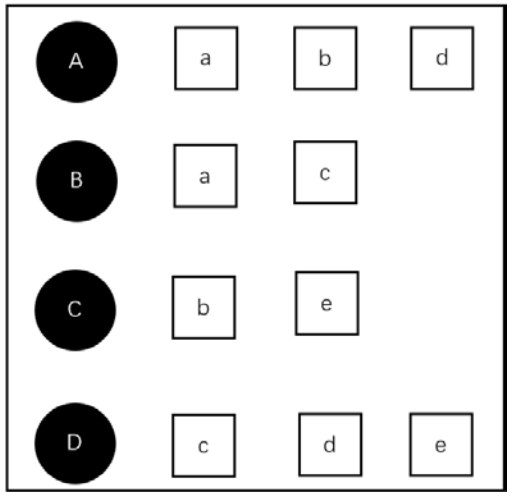
举例说明UserCF计算用户兴趣相似度。用户A对物品{a, b, d}有过行为，用户B对物品{a, c}有过行为，利用余弦相似度公式计算用户A和用户B的兴趣相似度为：

$$w_{AB} = \frac{|\{a,b,d\} \cap \{a,c\}|}{\sqrt{|\{a,b,d\}||\{a,c\}|}} = \frac{1}{\sqrt{6}}$$

基于用户的协同过滤算法

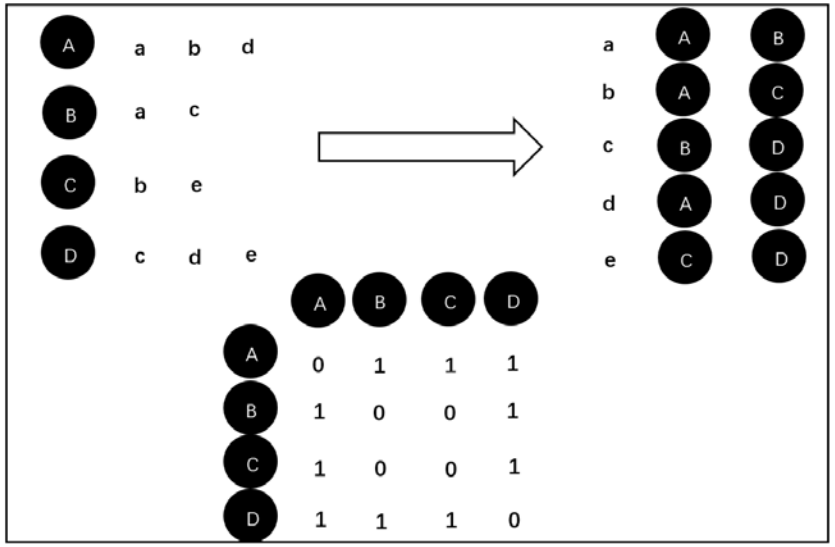
如果对两两用户都利用余弦相似度计算相似度，这种方法的时间复杂度为 $O(|U|*|U|)$，这在用户数量很大时非常耗时。事实上，很多用户相互之间并没有对同样的物品产生过行为，即很多时候$|N(u)\cap N(v)|=0$。换一个思路，我们可以首先计算出$|N(u)\cap N(v)|\neq 0$ 的用户对，然后再对这种情况除以分母。为此，可以首先建立物品到用户的倒查表，对每个物品都保存对该物品产生过行为的用户列表。令稀疏矩阵$C[u][v]=N(u)\cap N(v)$ 。那么，假设用户u和用户v同时属于倒查表中K个物品对应的用户列表，就有$C[u][v]=K$。从而，可以扫描倒查表中每个物品对应的用户列表，将用户列表中的两两用户对应的$C[u][v]$ 加1，最终就可以得到所有用户之间不为0的 $C[u][v]$。

基于用户的协同过滤算法



用户行为记录举例

用户—物品倒查表



以左图的用户行为为例解释上面的算法。首先需要建立物品-用户的倒查表。然后，建立一个4x4的用户相似度矩阵W，对于物品a，将$W[A][B]$和$W[B][A]$加1，对于物品b，将$W[A][C]$和$W[C][A]$加1，以此类推。扫描完所有物品后，我们可以得到最终的W矩阵。这里的W是余弦相似度中的分子部分，然后将W除以分母可以得到最终的用户兴趣相似度。

基于用户的协同过滤算法

得到用户之间的兴趣相似度后，给用户推荐和他兴趣最相似的K个用户喜欢的物品。如下的公式度量了用户u对物品i的感兴趣程度：

$$p(u,i)=\sum_{w\in S(u,K)\cap N(i)} w_{uv}r_{vi}$$

其中，$S(u,K)$包含和用户u兴趣最接近的K个用户，N(i)是对物品i有过行为的用户集合，w_{uv} 是用户u和用户v的兴趣相似度，r_{vi} 代表用户v对物品i的兴趣，因为使用的是单一行为的隐反馈数据，所以所有的$r_{vi}=1$ 。

- 算法要求：

  算法只有一个参数K，即为每个用户选出K个和他兴趣最相似的用户，然后推荐那K个用户感兴趣的物品。

  离线实验测量不同K值下算法的性能指标。

THANKS