数据科学导论第 11 讲—— 推荐算法

王小宁

中国传媒大学数据科学与智能媒体学院

2021 年 06 月 10 日

目录

推荐算法

推荐算法

- 推荐算法是计算机专业中的一种算法，通过一些数学算法，推测出用户可能喜欢的东西，目前应用推荐算法比较好的地方主要是网络，其中淘宝、头条、京东、拼多多等电商或互联网平台中都有应用。
- 推荐算法就是利用用户的一些行为，通过一些数学算法，推测出用户可能喜欢的东西。
- 推荐算法的研究起源于 20 世纪 90 年代，由美国明尼苏达大学 GroupLens 研究小组最先开始研究，他们想要制作一个名为 Movielens 的电影推荐系统，从而实现对用户进行电影的个性化推荐。

问题的提出

例子

- 我们在上网购物、看小说、买电影票的时候，都会遇到各种各样的推荐，给我们推荐一些我们曾经买过或收藏过的同类型产品，或者是推荐一些我们看过的小说题材相同的小说。那这些产品推荐都是如何实现的呢?
- 常用的推荐算法有：协同过滤推荐算法（Collaborative Filtering ）、内容推荐算法（Content-based)、相似性推荐算法（Similarity ）、关联规则推荐算法（Association Rule)。
- 不同的算法都有不同的应用场景，合理的应用这些算法，能够为我们带来更多的经济效益。

关联规则

- “尿布与啤酒”的故事是营销界的神话，一直为人们津津乐道。
- 按常规思维，尿布与啤酒风马牛不相及，若不是借助**数据挖掘**技术对大量交易数据进行挖掘分析，沃尔玛是不会发现隐藏于其中的关联规律的。
- 数据关联是数据库中存在的一类重要的可被发现的知识。若两个或多个变量的取值之间存在某种规律性，就称为关联规则(association rules, AR)。

关联规则

- 关联规则可分为简单关联规则、时序关联规则、因果关联规则。
- 关联规则分析的目的是找出数据库中隐藏的关联网。关联规则挖掘就是在事务数据库、关系数据库、交易数据库等信息载体中，发现存在于大量项目集合或者对象集合中有趣或有价值的关联或相关关系。
- Agrawal,R 等于 1993 年在分析购物篮问题时首先提出了关联规则挖掘，以后又经过诸多研究人员对其进行研究和发展，到目前它已经成为数据挖掘领域最活跃的分支之一。
- 关联规则的应用十分广发，比如在购物篮分析（Market Basket Analysis），网络连接（Web Link Analysis）和基因分析等领域均有所应用，具体应用场景包括优化货架商品摆放、交叉销售和捆绑销售、异常识别等。

基本概念

- 项与项集
- 设 $I = \{I_1, I_2, \cdots, I_m\}$ 是 m 个不同项目的集合，$I_p(p = 1, 2, \cdots, m)$ 称为数据项 (Item)，也称项。
- 数据项集合 I 称为数据项集（Itemset），简称项集。
- 项集其实就是不同属性取不同值的组合，不会存在相同属性、不同属性值的项集。例如，二项集 $\{$性别 $=$ 男，性别 $=$ 女$\}$ 中两项是互斥的，现实中不会存在

事务与事务集

- 关联挖掘的事务集记为 D，$D = \{T_1, T_2, \cdots, T_n\}$，$T_k(k = 1, 2, \cdots, n)$ 是项集 I 的非空子集，称为事务(Transaction)。
- 每一个事物有且仅有一个标识符，称为 TID(Transaction ID)。
- 事务集 D 包含的事务数记为 count(D)，事务集 D 中包含项集 X 的事务数目称为项集 X 的支持数，记为 occur(X)，则项集 X 的支持度（Support）定义为：

$$supp(X) = \frac{occur(X))}{count(D)} \times 100$$

- 若 supp(X) 大于等于预定义的最小支持度阈值，即 supp(X)≥min.supp，则称 X 为频繁项集，否则称 X 为非频繁项集。

项集支持度与频繁项集

- 一般情况下，给定最小支持数与给定最小支持度的效果是相同的，甚至给定最小支持数会更加直接且方便。假如需要寻找最小支持数为 3 的（频繁）项集，那么支持数小于 3（0,1,2）的项集就不会被选择。
- **定理**：设 X、Y 是事务集 D 中的项集，假定 $X \subseteq Y$，则：

1. $\text{supp}(X) \geq \text{supp}(Y)$
2. Y 是频繁项集 $\Longrightarrow$ X 是频繁项集
3. X 是频繁项集 $\Longrightarrow$ Y 是频繁项集

关联规则

- 若 $X, Y \subseteq I$ 且 $X \cap Y = \emptyset$, 蕴含式 $R : X \Longrightarrow Y$ 称为关联规则。其中项集 X,Y 分别是该规则的先导（antecedent 或 left-hand-side, LHS）和后继（consequent 或 right-hand-side, RHS）
- 项集 $X \cup Y$ 的支持度称为关联规则 $R : X \Longrightarrow Y$ 的支持度，记作 $supp(R)$:

$$supp(R) = supp(X \cup Y) = \frac{occur(XandY))}{count(D)}$$

- 关联规则 $R : X \Longrightarrow Y$ 的置信度（Confidence）定义为:

$$conf(R) = \frac{X \cup Y}{supp(X)} = P(Y|X)$$

支持度和置信度

- 支持度描述了项集 $X \cup Y$ 在事务集 D 中出现的概率的大小，是对关联规则重要性的衡量，支持度越大，关联规则越重要；
- 置信度是测度关联规则正确率的高低。
- 置信度高低与支持度大小之间不存在简单的指示性联系，有些关联规则的置信度虽然很高，但是支持度却很低，说明该关联规则的实用性很小，因此不那么重要。
- 任意给出事务集中两个项目集，它们之间必然存在关联规则，这样的关联规则将有无穷多种，为了在这无穷多种的关联规则中找出有价值规则，也为了避免额外的计算和 I/O 操作，一般给定两个阈值：**最小支持度和最小支持度。**

最小支持度/置信度

- 关联规则必须满足的支持度的最小值称为最小支持度 (Minimum Support)
- 关联规则必须满足的置信度的最小值称为最小置信度 (Minimum Confidence)
- 如果满足最小支持度阈值和最小置信度阈值，则认为该关联规则是有趣的。
- 支持度和置信度都不宜太低，如果支持度太低，说明规则在总体中占据的比例较低，缺乏价值；如果置信度太低，则很难从 X 关联到 Y，同样不具有实用性。

提升度

- 最小支持度与最小置信度的给定需要有丰富的经验做参考，且带有很强的主观性，为此我们可以引入另外一个量，即提升度 (Lift)，以度量此规则是否可用。
- 对于规则 $R: X \Longrightarrow Y$，其提升度计算方式为：

$$lift(R) = \frac{conf(X \Longrightarrow Y)}{supp(Y)} = \frac{supp(X \Longrightarrow Y)}{supp(X) * supp(Y)}$$

- 提升度显示了关联规则的左边和右边关联在一起的强度，提升度越高，关联规则越强。换而言之，提升度描述的是相对于不用规则，使用规则（效率、价值等）可以提高多少。有用的规则的提升度大于 1。

例子

表1 某超市顾客购物记录数据库D

TID	Date	Items
T100	6/6/2010	{面包，麦片}
T200	6/8/2010	{面包，牛奶，果酱}
T300	6/10/2010	{面包，牛奶，麦片}
T400	6/13/2010	{面包，牛奶}
T500	6/14/2010	{牛奶，麦片}
T600	6/15/2010	{面包，牛奶，果酱，麦片}

事务1、2、3、4、6中包含{面包}，事务2、3、4、6中包含{面包，牛奶}，支持度 $\sup p(R)=4/6\approx 0.67$，置信度 $conf(R)=4/5=0.8$，若给定支持度阈值 $\min.\sup p=0.5$，置信度阈值 $\min.conf=0.8$，那么关联规则{面包}$\Rightarrow${牛奶}是有用的，即购买面包和购买牛奶之间存在强关联。

例子

表1 某超市顾客购物记录数据库D

TID	Date	Items
T100	6/6/2010	{面包，麦片}
T200	6/8/2010	{面包，牛奶，果酱}
T300	6/10/2010	{面包，牛奶，麦片}
T400	6/13/2010	{面包，牛奶}
T500	6/14/2010	{牛奶，麦片}
T600	6/15/2010	{面包，牛奶，果酱，麦片}

计算提升度，知事务2、3、4、6中包含牛奶，则 $lift(R)=(4/5)/(4/6)=1.2$，意即对买了面包的顾客推荐牛奶，其购买概率是对随机顾客推荐牛奶的1.2倍，因此该规则是有价值的。

分类

- 关联规则可以分为布尔型关联规则（boolean association rules）和数值型关联规则（quantitative association rules）。
- 布尔型关联规则处理的变量是离散的、种类化的，它显示这些变量之间的关系
- 数值型关联规则可以处理数值型变量，将其进行动态分割，数值型关联规则中也可以包含种类变量
- 如考虑的是规则中数据是否出现，即为布尔型关联规则
- 如涉及的年龄是数值型变量，即为数值型关联规则，另外，此处一般都将数量离散化为区间

分类（维度/层次）

- 如果关联规则各项或属性只涉及一个维度，则它是单维关联规则(single-dimensional association rules)
- 在多维关联规则（multidimensional association rules）中，要处理的数据将涉及多个维度。
- 关联规则可以分为单层关联规则（single-level association rules）和广义关联规则（generalized association rules）
- 在单层关联规则中，所有的数据项或属性只涉及同一个层次而在广义关联规则中，将会充分考虑现实数据项或属性的多层次性。

Apriori 算法

- 有效建立规则的过程主要分为两个阶段，首先产生满足指定最小支持度的（频繁）项集，然后从每一个（频繁）项集中寻找满足指定最小置信度的规则。
- Apriori 是一种挖掘产生 0-1 布尔型关联规则所需频繁项集的基本算法，也是目前最具影响力的关联规则挖掘算法之一。这种算法因利用了有关频繁项集性质的先验知识而得名，其核心是基于两阶段频集思想的递推算法，在分类上属于单维、单层、布尔型关联规则。

Apriori 算法

- 基本思想: 使用逐层搜索的迭代方法，用频繁的（k-1）-项集探索生成候选的频集 k-项集（如果集合不是频繁项集，那么包含的更大的集合也不可能是频繁项集），再用数据库扫描和模式匹配计算候选集的支持度，最终得到有价值的关联规则。

1. 发现频繁项集
2. 由频繁项集产生关联规则

Apriori 算法

- 优点：（1）简单（2）易理解（3）数据要求低
- 缺点：（1）可能需要重复扫描数据库（2）可能产生大量的候选集搜索每个 L_k 需要扫描一次数据库，即如果最长模式为 n，那么就需要扫描数据库 n 遍，这无疑需要很大的 I/O 负载

协同过滤算法

协同过滤算法

- 基于协同过滤的推荐算法往往在预测效果上表现更好，因此是一种应用广泛的推荐算法。它是通过输入用户的历史行为信息来预测用户对未接触的商品可能采取的行为。

1. 获得稀疏的用户—项目评分（或购买、点击）矩阵，该矩阵包含了用户的偏好信息；
2. 通过算法预测空缺部分的评分，这里的算法可分为：
   (a) 基于邻居的协同过滤算法，它包括：
      （a1）基于用户（User-based）的协同过滤算法；
      （a2）基于物品（Item-based）的协同过滤算法；
   (b) 基于模型（Model- based）的协同过滤算法；
3. 为用户推荐其对应空缺部分中评分靠前的项目。

基于邻居的协同过滤算法

- 基于邻居的协同过滤又分为着眼于用户之间的关系（基于用户的协同过滤）和着眼于物品之间的关系（即基于物品的协同过滤）两种情况。
- 假设有 N 个用户，M 种物品，评分矩阵记为 $R_{N\times M}$,γ_{ui} 表示第 u 个用户对第 i 个物品的评分，一般地，在评分矩阵中会存在很多缺失值，表示用户并未使用某些物品或者未对某些物品进行评分。

基于用户的协同过滤算法

- 基于用户的协同过滤基于这样的一个假设：跟你喜好相似的人喜欢的东西你也很有可能喜欢。这个算法的原理是通过用户对物品的评分向量（即评分矩阵的行），计算不同用户之间的相似度，然后给每个用户推荐和他兴趣相似的其他用户（我们称之为邻居）喜欢的物品。
- 以第 u 个用户为例，假设用户 u 对物品 i 尚未评分，现在我们就通过用户 u 的邻居当中，评价过物品 i 的用户来预测用户 u 对物品 i 的评分 γ_{ui}。

基于用户的协同过滤算法

- ❶ 计算第 u 个用户与其他各个用户的相似度，找到所有评价过物 i 的用户中 u 的邻居，记为 $N_i(u)$.
- ❷ 假设用户 v 是 $N_i(u)$ 的一员，用 w_{uv} 表示用户 u、v 之间相似度的大小，则可以用加权平均来对 $\hat{\gamma}_{ui}$ 来进行预测。
- ❸ 若需要调整评分尺度，则可以引入标准化函数 h 进行预测。

基于用户的协同过滤算法

- 优点: 简单方便
- 缺点：存在性能上的瓶颈，即当用户数量越来越多时，寻找最近邻居的复杂度也将大幅度增加，因此无法满足及时推荐的要求。
- 基于物品的协同过滤可以解决上述这个问题。

基于物品的协同过滤算法

- 假设：能够使用户感兴趣的物品，必定与其之前给出的高评分物品相似。
- 原理：通过各个物品的用户评分向量（即评分矩阵的列），计算不同物品之间的相似度，然后向用户推荐与其偏好的物品相似的其他物品（称之为邻居）。
- 以第 i 个物品为例，假设用户 u 对物品 i 尚未评分，现在就通过物品 i 的邻居当中，用户 u 评价过的物品来预测用户 u 对物品 i 的评分 $\hat{\gamma}_{ui}$。

基于物品的协同过滤算法

- 缺点：没有考虑用户间的差异，因此精度比较差。
- 优点：不需要用户的历史数据，或是进行用户识别。对于物品来说，它们之间的相似性更加稳定，因此可以离线完成大量相似性的计算，从而降低了在线计算量，提高推荐效率，尤其是在用户多于物品的情形下这种方法的优势尤为显著。

基于邻居的预测的三要素（一）

- 邻居的确定: 邻居是基于邻居的协同过滤算法中至关重要的因素，我们需要确定的问题有两个：首先，如何定义邻居；其次，如何选取邻居。第一个问题我们在前面有提到，是用相似度的大小来定义邻居，相似度越大，就认为两个用户（或物品）越相邻。
- 对于第二个问题一般采用以下三个标准进行选择:
  1. Top-N filtering：保留相似度最大的前 N 个;
  2. Threshold filtering：保留相似度（绝对值）大于一个给定阈值的用户(或物品);
  3. Negative filtering：去掉不像的用户（或物品）。

基于邻居的预测的三要素 (二)

- 相似度的计算：相似度在基于邻居的协同过滤算法中既是确定邻居的依据，又包含在计算当中，因此起着非常大的作用。
  1. Pearson 相关系数
  2. 余弦相似度（Cosine Vector，CV）
  3. 评分标准化，均值中心化，Z-评分归一化

基于模型的协同过滤算法

- 基于邻居的协同过滤算法存在的缺点是不能处理稀疏数据集或数据量很大的数据集，因此就发展出了以模型为基础的协同过滤方法，它可以克服基于邻居方法的限制。
- 基于模型的协同过滤算法，是利用用户的历史信息，如购买、点击和评分，来构建机器学习模型，进而用此模型预测未发生的购买、点击和评分情况。基于不同的模型，又可将其细分为更多的算法。其中，基于 SVD 的算法是一种实现简单、容易改进，且效果很好的方法，受到了广泛的关注和应用。

基于 SVD 模型的协同过滤算法

- 原理：建立用户和项目——对应的评分矩阵（其为稀疏矩阵，因为用户并未对所有物品都进行过评分）后，通过机器学习算法将将该评分矩阵分解成两个矩阵相乘的形式，并利用分解得到的两个矩阵相乘来拟合评分矩阵，该拟合矩阵不仅与原评分矩阵的评分接近，而且还填补了原评分矩阵缺失的评分，最后就可以根据该拟合矩阵来预测用户未评价过的某一物品的评分。
- 这种矩阵因式分解背后的原理在于：假设用户和物品之间存在数量不多的潜在特征，这些潜在特征能代表用户如何对物品进行评分，所以根据用户与物品的潜在表现，我们就可以预测用户对未评分的物品的喜爱程度。

SVD 模型建模过程

- 对任意大小为的评分矩阵 R，可将其通过 SVD 分解成：

$$R_{N\times M} = P_{n\times K}Q_{K\times M}$$

- 其中，N 为用户数量，M 为项目数量，K 为分解得到的用户或项目的潜在特征的数量。
- 对于矩阵 R 内的任一已知的评分 γ_{ui}(即能观测到的评分部分)，其预测值（拟合值）就可以用 P、Q 的元素表示成：

$$\hat{\gamma}_{ui} = p_u^{'} q_i$$

- 计算每个评分的预测误差和误差平方和 SSE
- 定义损失函数为上述 SSE，则通过最小化损失函数，即可以求出矩阵 P、Q.

R 语言实现

- R 的 recommenderlab 包可以实现协同过滤算法。这个包中有许多关于推荐算法建立、处理及可视化的函数。
- 选用 recommenderlab 包中内置的 MovieLense 数据集进行分析，该数据集收集了网站 MovieLens（movielens.umn.edu）从 1997 年 9 月 19 日到 1998 年 4 月 22 日的数据，包括 943 名用户对 1664 部电影的评分。

本周作业

- 学习 R 软件包 recommenderlab，并对 MovieLense 数据集进行分析，提交作业（6.23）

  要求：生成 *word* 文档或 *pdf*，构建推荐算法，进行分析说明，同时有余力对变量数据进行描述分析，文件打包名字 (全部文件打包一个 *zip* 文件)，打包文件名：学号 + 姓名 + 第 *4* 次作业

作业提交

谢　谢!