毕业论文初稿

2019 年 11 月 6 日

目录

1 绪论 **1**

1.1 研究背景及意义 1

1.2 国内外研究现状 3

1.3 研究问题与内容 4

1.4 本文的组织结构 5

2 推荐系统与协同过滤算法简介 **5**

2.1 推荐系统的基本组成 5

2.2 推荐算法的分类 6

2.2.1 基于内容的推荐 7

2.2.2 基于规则的推荐 7

2.2.3 混合推荐 7

2.3 协同过滤算法 8

2.3.1 基于内存的协同过滤 8

2.3.2 基于模型的协同过滤 9

2.4 推荐系统的评价指标 10

1 绪论

1.1 研究背景及意义

随着互联网的快速普及和信息技术的高速发展，信息资源正呈现着爆炸式的增长，不论何时何地，人们都可以非常方便地通过互联网来获取信

息。2018年天猫“双11”活动，每秒创建订单峰值高达惊人的49.1万笔；今年8月，网易的二季度财报披露网易云音乐的总用户数已经超过了8亿，而这一数字在两年前是4亿，三年前仅是2亿。很明显地，人们正在从原先的信息匮乏时代，逐渐地步入信息过载的时代。而这个信息过载的时代，不论是对于信息的生产方，还是对于我们这样普通的信息消费方，都带来了巨大的挑战。一方面，对于信息的消费方普通用户而言，面对这海量的数以万计甚至百万计的物品，倘若没有一个明确的需求目标，想要找到对我们有用的，或是我们感兴趣的物品会变得十分困难。而另一方面，对于信息的生产提供方商家而言，如何从这海量的商品库存中，为用户准确地推荐可能符合他们需求的物品，从而自己能够获得收益，这也是十分困难的。在这样的大环境下，推荐系统（Recommender Systems）便应运而生。它主要解决的就是上面所提到的用户-商家双向问题，既可以让用户找到心仪的，对自己有价值的物品，又可以让商家准确的把对应物品展现在用户面前，从而最终实现用户和商家的互利共赢。

个性化推荐系统，和我们熟知的搜索引擎有些类似，都属于帮助用户发现挖掘信息的一样工具。不过这两者最大的区别在于是否需要用户进行明确输入。搜索引擎需要用户输入关键词，然后才能进行搜索，如果我想要的东西，并不能很明确的通过一个或多个关键词进行描述的话，搜索引擎就很可能没办法很好的使用了。而推荐系统不需要用户提供显式的输入，它的核心原理是挖掘用户的行为模式，通过对海量的用户历史行为数据进行分析，然后建立模型，通过模型来预测每个用户的兴趣爱好，最终给不同的用户推荐不同的商品，这也体现了“个性化”这三个字的含义。不过也正是因为这种工作原理，推荐系统通常不会像搜索引擎一样独立的成为一个网站，如百度、Google等等，而是会存在于每个网站的不同应用或是模块的背后，为这些应用和模块的正常工作而服务。

目前，推荐系统已经在国内外许许多多的领域内有着非常广泛而成功的应用：电子商务网站，如淘宝、京东的商品个性化推荐页面；视频网站，如Youtube首页的推荐视频；社交网络，如新浪微博的热门推荐；音乐电台，如网易云音乐的每日歌曲推荐/私人FM等等。毫不夸张的说，只要是联网的应用，涉及到信息资源的获取，就都会有推荐系统的身影存在。另一方面，在这些领域内，推荐系统的实际应用也为公司带来了庞大的商业价值和经济收益。著名的北美在线视频服务提供商Netflix，其自己估计每年通过个性化推荐系统为自身业务节省了约10亿美金；著名电子商务网

站Amazon也曾披露，其销售额有20%-30%是来自推荐系统。正因如此，对推荐系统的研究在学术界也是越来越受到学者们的关注，并且成为了一个比较热门的方向。同时由于其和不同的学科研究领域，比如数据挖掘、机器学习、人工智能等都有着一定的联系，这种交叉学科的性质使得推荐系统正在飞速的向前发展。在当下这个信息爆炸的时代，研究推荐系统是很有意义的，对互联网生态的发展也有着一定的意义。

1.2 国内外研究现状

最早的推荐系统大概可以追溯到1994年，明尼苏达大学的GroupLens研究组设计了第一个自动化的新闻推荐系统GroupLens[1]。这个系统不但首次提出了协同过滤的思想，而且为后世的推荐问题建立了一个形式化的范式。顺便一提这个研究组后来创建了大名鼎鼎的MovieLens推荐网站。1997年，Resnick 等人[2]首次提出推荐系统（Recommender System，RS）一词，自此，推荐系统一词被广泛引用，并且推荐系统开始成为一个重要的研究领域。1998年，著名的个性化商城Amazon推出了基于项目的协同过滤算法(item-based Collaborative Filtering)，在此之前所普遍使用的协同过滤算法都是基于用户的(user-based CF)，而Amazon提出的这个新算法的实际效果非常好。2003年，Amazon在IEEE Internet Computing[3]上公开了这个item-CF算法，带来了广泛的关注和使用，包括YouTube、Netflix等。2005年Adomavicius等人的综述论文[4] 将推荐系统分为3个主要类别，即基于内容的推荐、基于协同过滤的推荐和混合推荐的方法，并提出了未来可能的主要研究方向。

到了2006年，一个大事件将推荐系统的研究推向了快速发展的高潮阶段：Netflix宣布了一项竞赛，第一个能将现有推荐算法的准确度提升10%以上的参赛者将获得100万美元的奖金。这个比赛在学术界和工业界引起了很大的关注，吸引了来自186个国家和地区的超过4万支队伍参赛。而在之后的那几年，许多经典的推荐算法被提出，比如：Koren等人[5]对利用矩阵分解（Matrix Factorization, MF）实现协同过滤的现有技术进行了综述，包括基本MF原理，以及包含隐反馈、时序动态等特殊元素的MF等；Steffen Rendle[6]等人在2009年提出了一种“基于贝叶斯后验优化”的个性化排序算法BPR，它是采用pairwise训练的一种纯粹的排序算法；

近年来随着人工智能的火爆，机器学习和深度学习技术也被广泛运用到了推荐系统的排序场景中。Google在2016年最新发布的模型Wide&Deep[9]

将Logistic Regression和forward DNN结合在一起，既发挥了逻辑回归的优势，又将 dnn和embedding的自动特征组合学习和强泛化能力进行了补充。这个模型在Google play的推荐场景取得了应用并且获得良好的效果；Guo, Huifeng[10]等人于2017年提出了DeepFM模型，将传统的因子分解机FM和深度神经网络DNN结合在一起，用于解决CTR预估中，挖掘构造有用的高级交叉特征的问题。

而除了基本的推荐召回算法与排序算法外，关于推荐系统的一些其他环节，比如冷启动问题、缺失数据的填充、隐反馈信息的利用等等，也涌现出不少优秀的研究成果。Hu Y, Koren Y, Volinsky C.（2008）采用隐语义模型（LFM）来解决隐性反馈数据的协同过滤问题；Ren Y, et al.（2012）基于缺失得分应该和现有得分相似的假设，利用完整数据的协同过滤来预测缺失的得分信息；Steck H.（2010）则是认为用户u对某一物品i的得分缺失，隐式地表明了用户u对物品i不怎么喜欢，基于新提出的一种评价指标，论证了为缺失得分填充为 2 分是比较合理的。Zhang Z P, et al.（2019）针对 item-CF 中的完全物品冷启动问题，提出了一种非常简单的通过挖掘关联属性的方法，很好的处理了冷启动问题。

1.3 研究问题与内容

真实世界的用户行为数据集通常存在以下几个问题：

1. 数据的稀疏性。由于用户和物品的庞大基数，导致每个用户只会对极小部分的物品有过行为，如给电影打分，购买物品等等，而对于剩下的大量的其他物品，用户都没有过行为。这样导致我们生成的用户-物品行为矩阵就是一个稀疏矩阵，这对一些推荐算法的实现是一个阻碍。

2. 有关用户的显性行为和隐性行为。显性行为指的是可以明确反映用户对物品的喜好程度的行为，最有代表性的的显性行为就是打分，打分高说明用户喜欢这个物品，打分低则说明用户不喜欢这个物品。与之对应的，隐性行为指的就是不能明确反映用户喜好程度的行为，以电子商务网站为例，之前说的给商品打分是显性行为，而诸如用户对商品的浏览、点击、收藏等行为就属于隐性行为。在真实数据集中，隐性行为要远多于显性行为，有效的利用隐性行为特征可以提高推荐的效果。

3. 冷启动问题，具体地又分为用户的冷启动，以及物品的冷启动。其中，用户的冷启动指的是如何对新用户进行推荐，因为新用户是没有历史行为数据可供我们的推荐系统进行挖掘分析的；而物品的冷启动主要解决的是如何把一个新的物品推荐给用户，因为这个新的物品在所有的历史行为数据中肯定是没有出现过的。

本文选取了一个网上公开的真实世界中的数据集，这个数据集也同时有着上面提到的这些问题。基于该数据集的基础上，首先探讨比较了多种召回算法的效果与优缺点；然后将机器学习技术应用于推荐模型的排序阶段，对召回结果进行精排，比较精排后与精排前的效果；最后针对上面提到的三个问题，提出一些相应的解决方案，并对比改进前后的效果。具体地，对于数据的稀疏性，采用KNN填充和基于邻域的评分预测的加权组合方法；对于隐性反馈行为，采用隐反馈特征和显反馈特征相结合的方法来生成样本标签，通过交叉验证来选择具体的组合方式；对于冷启动问题，采用基于KMeans聚类的方法。

1.4 本文的组织结构

先不写。。

2 推荐系统与协同过滤算法简介

2.1 推荐系统的基本组成

在不同的实际业务情形下，推荐系统的设计也会有所差异，但是一些必要的组件都是共通的。一个普通的推荐系统最主要的组成模块有下面三个：数据采集层、召回层和排序层。

- 数据采集层：推荐系统是在做用户的行为模式挖掘，而这个挖掘是通过分析用户对物品的行为数据来进行的，所以离不开大量的数据，数据采集层主要就是用来上报业务数据、日志数据等等，作为推荐算法的数据来源。对于用户行为数据，应当尽可能地覆盖用户可能涉及到的业务流程。而对于物品相关的属性数据，也应当尽可能地覆盖更多的属性维度。用户信息和物品的信息越全，模型最后作出的推荐结果也会越准确。

- 召回层：通常推荐模型的计算开销会比较大，而我们可供推荐的物品集大小往往不下于上万种。如果完全依赖模型进行推荐的话，就要对着至少万级的物品一个一个排序打分，显然成本太高。这时候就需要设计召回策略，将用户可能感兴趣的物品从全量的物品库中事先提取出来，作为我们推荐内容的候选集，最终为每个用户推荐的结果都会是这个候选集的某一个子集，这个候选集的大小一般不会很大。召回层的算法种类很多，比如有热门推荐、基于内容/标签等的召回、协同过滤召回、主题模型召回等等。通常来说，单一的召回算法得出的结果会比较难以满足业务需求，所以实际情况中往往采用多路召回的策略，即：每一路召回尽量采取一个不同的策略，召回K条物料，这个K值可以根据各路召回算法的实际表现来分配不同的大小。

- 排序层：排序层主要是针对上面多路召回的结果候选集，利用排序算法进行逐一打分和重排序，以更好的反映用户的偏好。通过排序优化用户对召回集的点击行为后，将用户更可能喜欢的物品取出作为最终的推荐列表推荐给用户，提升用户体验。通常会选取模型预测打分最高的前N个物品，这也被称为Top-N推荐。排序算法的种类也很丰富，比如有LR，GBDT等机器学习模型，有BPR等排序模型，还有Wide & Deep, DNN等深度学习模型等。

2.2 推荐算法的分类

Adomavicius G , Tuzhilin A .(2005)的经典综述论文中曾对推荐系统的问题有一个范式化的定义：设所有用户集合为C，所有可能被推荐的物品集合为S，这两个集合都可以非常大。令u表示一个效用函数，度量了某物品s对某用户c的价值：$u : C \times S \to R$，这里的R是一定区间内的非负实数。则对于用户集合C中的任意一个用户c，我们想要找到物品集合S中那个使得效用最大化的物品，即：

$$\forall u \in C, s_c^{'} = \underset{s \in S}{argmax}\, u(c, s)$$

在推荐系统中，这个效用函数通常被表示为一个打分函数，表示了用户对物品的喜好程度。

根据数据源或是推荐策略的不同，推荐系统算法大体上可以分成下面几大类：基于内容的推荐、基于规则的推荐、协同过滤推荐以及混合推荐。

其中，协同过滤算法是目前最主流，应用也是最广泛的一类算法，稍后会单独作一个介绍。

2.2.1 基于内容的推荐

基于内容的推荐算法（Content-based Recommendation）起源于信息检索和信息过滤研究，该方法中，效用函数$u(c,s)$是基于$u(c,s_i)$来估计的，其中$s_i \in S$表示和物品s相似的物品。换句话说，这种算法会将和用户过去历史上喜欢的物品相似的物品推荐给这个用户。具体地，通过对用户历史行为信息的分析，提取出可以表示这个用户的兴趣爱好的特征向量，然后通过匹配用户特征向量和物品特征向量的相似度，来预测用户对该物品的评分，最后将分高的物品推荐给用户。当前许多基于内容的推荐都把重心放在推荐含有文本信息的物品上，比如文档、新闻、网站等等，而这类推荐会依赖于一些自然语言处理的知识，用关键词来作为表征物品的特征，同时利用TF-IDF等NLP技术来对特征重要性进行加权处理，转化成物品的特征向量。

2.2.2 基于规则的推荐

基于规则的推荐属于大众型的推荐，比如最多用户点击、最多用户收藏等等，类似那些排名榜单的推荐。这种推荐方法不属于“个性化推荐”，但是很适合应对一些冷启动问题。

2.2.3 混合推荐

混合推荐（Hybrid Methods），是将协同过滤方法（CF）和基于内容的推荐方法结合在一起，大体上可以分成下面几类：

- 分别执行协同过滤推荐和基于内容的推荐，然后把两个预测结果直接合并
- 把一些基于内容的特征加入到协同过滤中去
- 把一些协同过滤的特征加入到基于内容的推荐方法中去
- 构建一个统一的模型，把两种方法的特征都整合到一起去

2.3 协同过滤算法

相比于上面的几类算法，协同过滤算法（Collaborative Filtering）就显得复杂许多，同时也是应用最多的一类算法。协同过滤的意思是：通过集体的智慧，找到用户可能喜欢的物品，并过滤掉一些不值得推荐的物品，比如评分较低的物品或是用户曾经观看过/购买过的物品。根据是否使用机器学习的思想，可以进一步划分为基于内存的协同过滤（Memory-based CF）和基于模型的协同过滤（Model-based CF）。

2.3.1 基于内存的协同过滤

Memory-based CF主要采用启发式的方法进行推荐，关键的步骤在于选取合适的相似度度量函数。根据维度的不同，分为基于用户的协同过滤（User-based CF）和基于物品的协同过滤（Item-based CF）

1. 基于用户的协同过滤 User-based CF的思想是：当某个用户u需要个性化推荐的时候，寻找和他兴趣相似的其他一些用户$v_1, v_2, \ldots, v_k$，然后把这些用户喜欢的，但是该用户没有接触过的物品推荐给u。对于用户间的相似度计算，User-based CF采用如下简单的余弦相似度：

$$sim(u,v) = \frac{|N(u) \cap N(v)|}{\sqrt{|N(u)| \cdot |N(v)|}}$$

   其中记号$N(u)$表示用户u喜欢过的物品集合。得到了用户间的相似度矩阵后，就可以对指定用户对指定物品的感兴趣程度进行预测：

$$r_{ui} = \sum_{v \in N(i) \cap B(u,k)} sim(u,v) \cdot r_{vi}$$

   上式中，$B(u,k)$表示和用户u相似度最高的k个其他用户，$N(i)$是对物品i有过行为的用户集合，r_{vi}是用户v对物品i的兴趣程度（一般是具体的打分）

2. 基于物品的协同过滤 Item-based CF的思想是：当某个用户u需要个性化推荐的时候，寻找该用户历史上曾经喜欢过的物品列表$j_1, j_2, \ldots, j_n$，然后将和这些物品比较相似的其他物品推荐给用户u。对于物品间的相似度计算，Item-based CF的做法和User-based CF相仿：

$$sim(i,j) = \frac{|N(i) \cap N(j)|}{\sqrt{|N(i)| \cdot |N(j)|}}$$

其中记号$N(i)$表示喜欢物品i的用户集合。得到了物品间的相似度矩阵后，就可以对指定用户对指定物品的感兴趣程度进行预测：

$$r_{ui} = \sum_{j \in N(u) \cap B(i,k)} sim(i,j) \cdot r_{uj}$$

上式中，$B(i,k)$表示和物品i相似度最高的k个其他物品，$N(u)$是用户u喜欢过的物品集合，r_{vi}是用户v对物品i的兴趣程度（一般是具体的打分）

2.3.2 基于模型的协同过滤

Model-based CF应用了一些机器学习的思想，常见的一些方法有：矩阵分解算法、分类回归算法、图算法等等。

- 矩阵分解算法：将数据集变换成一个UI打分矩阵$M_{m \times n}$，共m行n列，表示m个用户对n个物品的打分情况。由于稀疏性的问题，这个UI矩阵大部分元素都等于零。这种情况下，如果能估计出矩阵的每个位置元素，就可以实现推荐预测了。以简单但有效的FunkSVD算法为例，我们期望将UI矩阵M进行分解：

  $$M_{m \times n} = P^T_{m \times k} Q_{k \times n}$$

  对于每一个元素m_{ij}，通过矩阵分解后的对应表示应该是：

  $$m_{ij} = e_i^T P^T Q e_j = p_i^T q_j$$

  用均方差作为损失函数，对于全部的用户和物品组合，我们期望最小化的目标函数就是：

  $$\underset{p_i, q_j}{argmin} \sum_{i,j} \left(m_{ij} - p_i^T q_j \right)^2$$

  通过最优化上式，得到分解的矩阵P, Q，进而用于推荐预测。

- 分类回归算法：把推荐问题转化成传统的机器学习分类/回归问题解决，分类就是输出概率值然后根据概率值大小排序推荐；回归就是预测用户对物品的打分，根据打分高低排序推荐。

2.4 推荐系统的评价指标

这里主要讨论实际应用中最常见的top-N推荐（因为后面的实验中采用的也是top-N推荐而非评分推荐）。记系统对用户u推荐的N个物品组成的列表为$R(u)$，用户u在测试集上实际喜欢的物品组成的集合为$T(u)$，测试集全体用户集合为$\tilde{U}$。下面列出了在后文中用到的一些评价指标：

- 精确率：

$$Precision = \frac{\sum_{u \in \tilde{U}} |T(u) \cap R(u)|}{\sum_{u \in \tilde{U}} |R(u)|}$$

- 召回率：

$$Recall = \frac{\sum_{u \in \tilde{U}} |T(u) \cap R(u)|}{\sum_{u \in \tilde{U}} |T(u)|}$$

- 覆盖率：

$$Coverage = \frac{\left| \cup_{u \in \tilde{U}} R(u) \right|}{|I|}$$

- 准确率(命中率)：

$$hit\_rate = \frac{\sum_{u \in \tilde{U}} 1_{\{T(u) \cap R(u) \neq \emptyset\}}}{50}$$

类比机器学习中二分类问题的评价指标，在top-N推荐中，也有对应的精确率（Precision）和召回率（Recall）。Precision体现了结果推荐列表中有多少比例的物品的确被用户产生了行为；而Recall体现了被用户实际产生行为的物品中有多少比例被包含在了最终的推荐结果中。除了Precision和Recall以外，Coverage可以反映算法挖掘长尾的能力，如果覆盖率越高，说明算法越能够将长尾中的物品推荐给用户，而不是集中推荐一些热门物品。命中率有些类似机器学习里的$Accuracy$，这里也是作为一个辅助的参考指标。