

硕士专业学位论文

基于混合推荐模式的汽车类文章推荐系统的设计与实现

Design and Implementation of Automotive Article Recommendation System based on Mixed Recommendation Mode

作者：申玉聪

导师：孔令波

北京交通大学

2018年3月

学位论文版权使用授权书

本学位论文作者完全了解北京交通大学有关保留、使用学位论文的规定。特授权北京交通大学可以将学位论文的全部或部分内容编入有关数据库进行检索，提供阅览服务，并采用影印、缩印或扫描等复制手段保存、汇编以供查阅和借阅。同意学校向国家有关部门或机构送交论文的复印件和磁盘。学校可以为存在馆际合作关系的兄弟高校用户提供文献传递服务和交换服务。

（保密的学位论文在解密后适用本授权说明）

学位论文作者签名： 导师签名：

签字日期： 年 月 日 签字日期： 年 月

学校代码：10004 密级：

北京交通大学

**专业硕士学位论文**

基于混合推荐模式的汽车类文章推荐系统的设计与实现

Design and Implementation of Automotive Article Recommendation System based on Mixed Recommendation Mode

作者姓名：申玉聪 学 号：16126192

导师姓名：孔令波 职 称：讲师

工程领域：软件工程 学位级别：硕士

北京交通大学

2018年3月

摘要

**关键词：**

ABSTRACT

**KEYWORDS：**

目录

1. 引言

本章首先对课题的研究背景，学籍信息管理信息系统的研究现状、课题的研究目的和意义、本文的研究内容等进行了阐述和分析。并在本章最后给出了论文的行文结构。

* 1. 课题背景与意义

北京无限讯奇信息技术有限公司（以下简称“无限讯奇”）是中国移动12580业务独家合作伙伴，负责开发和建设中国移动12580业务核心系统、内容采编与发布、广告销售、用户拓展与服务等运营工作[1]。

阿迪沙发北京无限讯奇信息技术有限公司（以下简称“无限讯奇”）是中国移动12580业务独家合作伙伴，负责开发和建设中国移动12580业务核心系统、内容采编与发布、广告销售、用户拓展与服务等运营工作[1]。

* 1. 国内外研究现状

就目前国内外学籍信息管理系统的发展状况来说，国外院校的重视程度较高，且由于其计算机水平起步早发展快，因此，相对起步较早，发展较快且利用率较高；国内从上世纪90年代传入国内，起步较晚，但是由于我国政府政策支持和倡导，相应的学籍信息管理信息系统发展较快，且在政府部门和企业都有了较快的发展和广泛的应用，但是针对具体不同类型的高校而言，缺乏学籍管理的针对性。

国内研究现状

就目前国内外学籍信息管理系统的发展状况来说，国外院校的重视程度较高，且由于其计算机水平起步早发展快，因此，相对起步较早，发展较快且利用率较高；国内从上世纪90年代传入国内，起步较晚，但是由于我国政府政策支持和倡导，相应的学籍信息管理信息系统发展较快，且在政府部门和企业都有了较快的发展和广泛的应用，但是针对具体不同类型的高校而言，缺乏学籍管理的针对性。

就目前国内外学籍信息管理系统的发展状况来说，国外院校的重视程度较高，且由于其计算机水平起步早发展快，因此，相对起步较早，发展较快且利用率较高；国内从上世纪90年代传入国内，起步较晚，但是由于我国政府政策支持和倡导，相应的学籍信息管理信息系统发展较快，且在政府部门和企业都有了较快的发展和广泛的应用，但是针对具体不同类型的高校而言，缺乏学籍管理的针对性。

国外研究现状

就目前国内外学籍信息管理系统的发展状况来说，国外院校的重视程度较高，且由于其计算机水平起步早发展快，因此，相对起步较早，发展较快且利用率较高；国内从上世纪90年代传入国内，起步较晚，但是由于我国政府政策支持和倡导，相应的学籍信息管理信息系统发展较快，且在政府部门和企业都有了较快的发展和广泛的应用，但是针对具体不同类型的高校而言，缺乏学籍管理的针对性。

* 1. 本人主要工作

就目前国内外学籍信息管理系统的发展状况来说，国外院校的重视程度较高，且由于其计算机水平起步早发展快，因此，相对起步较早，发展较快且利用率较高；国内从上世纪90年代传入国内，起步较晚，但是由于我国政府政策支持和倡导，相应的学籍信息管理信息系统发展较快，且在政府部门和企业都有了较快的发展和广泛的应用，但是针对具体不同类型的高校而言，缺乏学籍管理的针对性。

* 1. 论文组织结构

就目前国内外学籍信息管理系统的发展状况来说，国外院校的重视程度较高，且由于其计算机水平起步早发展快，因此，相对起步较早，发展较快且利用率较高；国内从上世纪90年代传入国内，起步较晚，但是由于我国政府政策支持和倡导，相应的学籍信息管理信息系统发展较快，且在政府部门和企业都有了较快的发展和广泛的应用，但是针对具体不同类型的高校而言，缺乏学籍管理的针对性。

* 1. 本章小结

1. 理论基础与关键技术介绍

就目前国内外学籍信息管理系统的发展状况来说，国外院校的重视程度较高，且由于其计算机水平起步早发展快，因此，相对起步较早，发展较快且利用率较高；国内从上世纪90年代传入国内，起步较晚，但是由于我国政府政策支持和倡导，相应的学籍信息管理信息系统发展较快，且在政府部门和企业都有了较快的发展和广泛的应用，但是针对具体不同类型的高校而言，缺乏学籍管理的针对性。

* 1. 开发工具

IntelliJ IDEA

Navicat

Hue

* 1. Thrift框架
  2. Django Web框架
  3. ECharts 数据可视化工具
  4. Hadoop生态系统

HDFS概述

MapReduce概述

Hive概述

* 1. 推荐系统

推荐系统定义

基于内容的推荐算法

基于关联规则的推荐算法

基于协同过滤的推荐算法

* 1. 本章小结

1. 需求分析
   1. 系统需求概述
   2. 系统功能性需求分析

文章数据同步与用户标签提取模块

基于多种推荐算法的混合推荐引擎模块

* + - 1. 相关推荐
      2. 个性化推荐
      3. 推荐解释的生成
      4. 推荐热点候补
      5. 已推荐过滤及新颖性排名

推荐点击预估模块

* + - 1. 用户兴趣建模
      2. 文章特征计算
      3. 特征工程
      4. 模型训练

推荐效果可视化模块

标签系统

* 1. 系统非功能性需求分析
  2. 本章小结

1. 推荐系统概要设计

本章主要阐述了汽车类文章推荐系统的概要设计，包括系统总体架构设计、系统模块划分、系统业务流程、数据库设计。

* 1. 系统总体架构设计

推荐系统整体对外提供Thrift服务，App客户端通过发送Thrift请求获取推荐列表。

* 1. 系统模块划分

汽车类文章推荐系统划分为原始数据同步与解析模块、基于多种推荐算法的混合推荐引擎模块、推荐点击预估模块、推荐效果统计及可视化模块。推荐系统的原始数据来源有两个，其中一种数据来源方式是通过WebService接口同步外部文章数据，另一种数据来源方式通过Kafka消息队列获取原始的用户行为日志数据。其中原始数据同步与解析模块主要负责实时同步文章数据，并将原始的日志数据按照一定的规则进行解析，并根据用户的浏览、收藏、搜索、分享等行为为用户打上相应的兴趣标签存储到数据库中为推荐引擎提供数据来源。基于多种推先算法的混合推荐引擎模块主要负责利用多种推荐算法计算用户对文章的兴趣度、文章关键词、文章内容相似度、文章的关联规则，然后根据计算结果筛选出推荐候选集列表，最终对推荐候选集进行已推荐过滤及新颖性排名操作。推荐点击预估模块主要通过机器学习的方法训练点击预估模型，利用训练好的模型预测推荐的文章被用户点击的概率，相当于使用模型预测的概率值对上一个模块产生的推荐候选集进行重排序，再选择概率最高的文章N篇文章作为最终的推荐结果列表，此外训练模型使用的特征数据的提取涉及大量的分布式离线计算，也是本模块的重点开发任务。推荐效果统计及可视化模块主要负责统计每天的推荐效果，如每种算法的PV、UV、点击率、人均阅读次数、用户留存率等并以图表的形式在网页中展示，方便我们查看推荐效果的趋势，进而帮助我们不断改进推荐算法。系统模块设计图如4-1图所示。

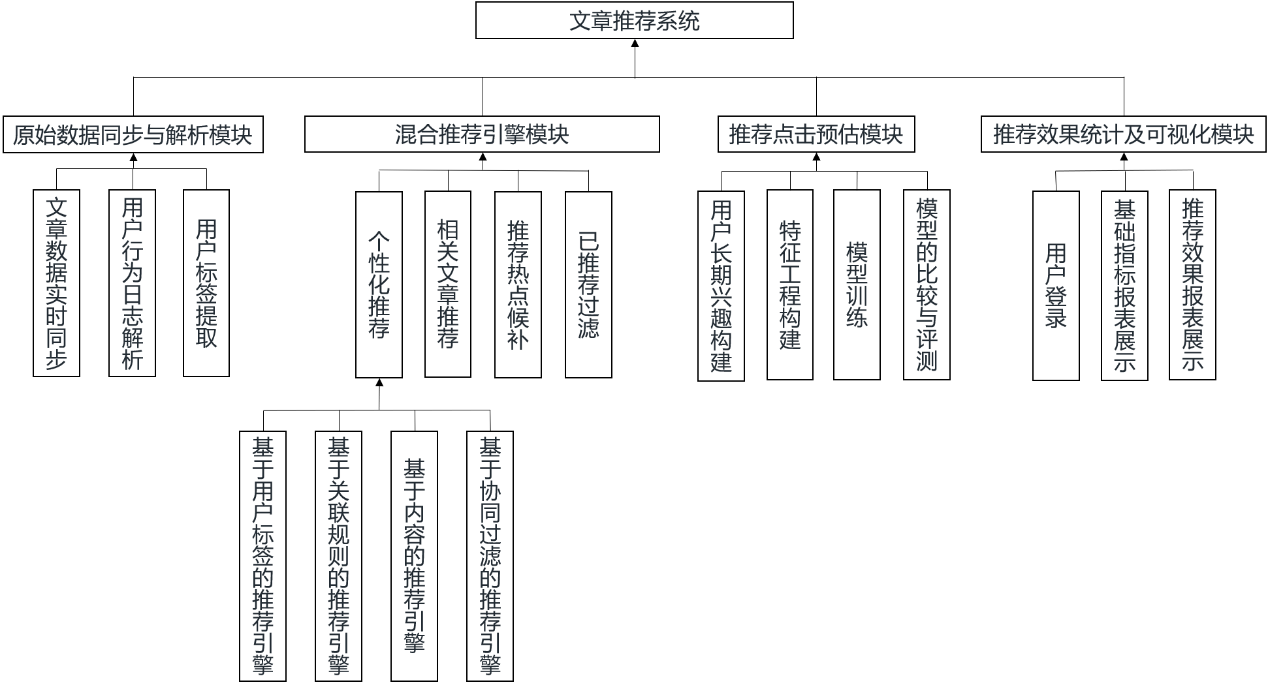
图4-1 系统模块设计图

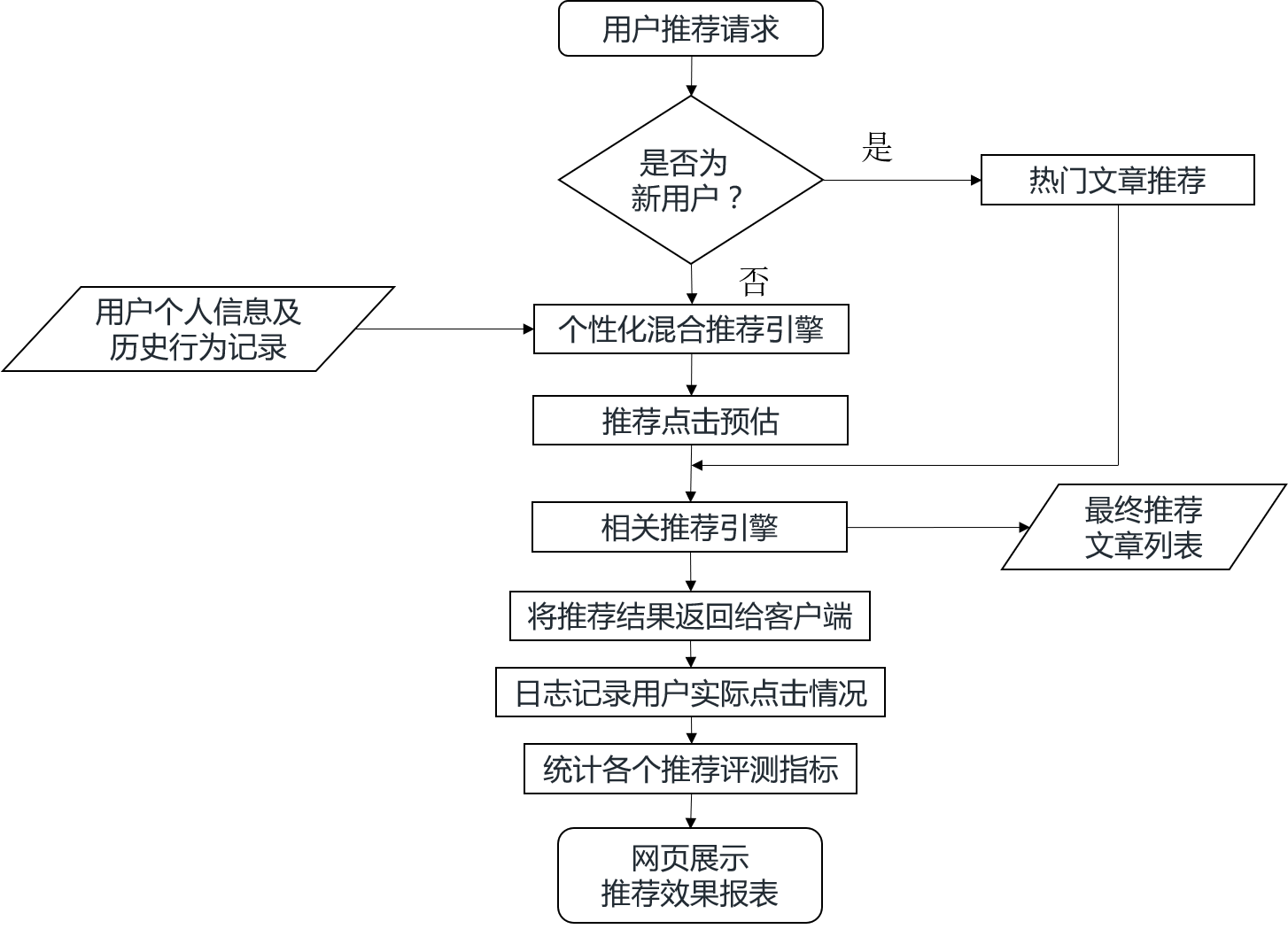
Figure 4-1 System Module Diagram

* 1. 系统业务流程

本节主要介绍推荐系统整体的业务流程，首先用户在APP客户端发出推荐请求，推荐系统服务端首先根据用户的设备ID判断用户是新用户还是老用户，如果该用户新用户，我们直接给用户推荐热门文章列表，此时推荐完成。如果该用户是老用户，那么我们只需要将老用户的个人信息输入到基于多种算法的混合推荐引擎中，即可得到该用户的推荐候选列表，为了使我们的推荐更加准确，推荐候选列表会输入到我们已经训练好的推荐点击预估模型中，对用户的点击行为进行预测，得出的结果是候选列表中每篇文章被该用户点击的概率值，选择点击预估概率最高的N篇文章作为最终的推荐结果，然后将最终的N篇文章输入到相关文章推荐引擎中得到每篇文章的相关文章，最终将个性化推荐结果和相关文章推荐结果作为最终的推荐结果返回到APP客户端，除此之外我们会对每天线上的点击率、人均阅读率等在线指标进行统计，并以报表的形式可视化地展现在网页上，方便对推荐算法进行改进。整体的系统流程如4-2图所示。

图4-2 系统流程图

Figure 4-2 System Flow Diagram



* 1. 数据库设计

本节主要说明推荐系统的数据库设计，包括E-R图和物理表结构设计。

E-R图设计

由于本次项目主要针对于与汽车相关的文章的推荐，因此汽车相关和文章相关的数据都是必不可少的，除此之外还有与推荐策略相关的表，以及数据可视化涉及到的推荐效果报表数据。表之间的联系如图4-3所示。

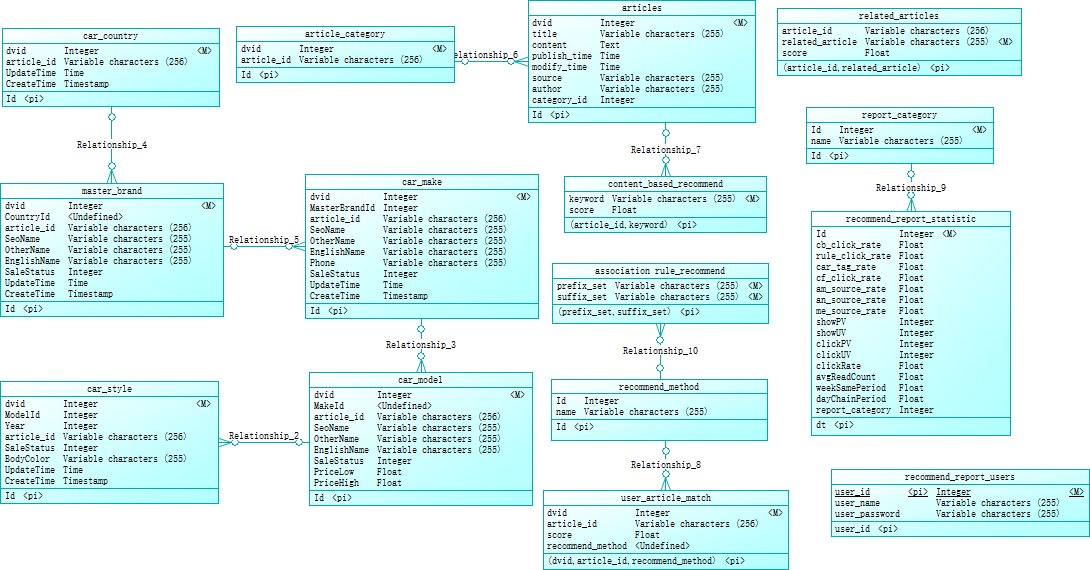


图4-3 E-R图

Figure 4-3 E-R Diagram

物理表结构设计

涉及到本次项目的表共有15张，本节将利用表格的形式说明每张表的表结构及其含义与作用，具体信息如表4-1至表4-11所示。

1. 文章基本信息表

如表4-1所示，文章基本信息表主要用于记录文章的ID、标题、内容、发表时间、修改时间、文章来源、文章作者、文章类型等内容。

表4-1 文章信息表

Table 4-1 article information sheet

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 | 备注 |
| id | varchar (12) | 文章ID | 主键 |
| title | varchar (500) | 文章标题 | 非空 |
| content | mediumtext | 文章内容 | 非空 |
| publish\_time | datetime | 发表时间 | 非空 |
| modify\_time | datetime | 修改时间 |  |
| source | varchar (100) | 文章来源 | 非空 |
| author | varchar (500) | 文章作者 | 非空 |
| category\_id | int | 文章类型 | 非空 |

表4-2 文章类别信息表

Table 4-2 article category information sheet

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 | 备注 |
| id | int | 文章类别ID | 主键 |
| name | varchar (500) | 文章类别名称 | 非空 |

表4-3 车辆国别表

Table 4-4 article category information sheet

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 | 备注 |
| id | int | 国别ID | 主键 |
| name | varchar (500) | 国别名称 | 非空 |
| update\_time | timestamp | 更新时间 | 非空 |
| create\_time | timestamp | 创建时间 | 非空 |

表4-4 车辆品牌表

Table 4-3 article category information sheet

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 | 备注 |
| id | int | 品牌ID | 主键 |
| name | varchar (100) | 品牌名称 | 非空 |
| country\_id | int | 国别ID | 外键 |
| seo\_name | varchar (100) | App显示名称 |  |
| other\_name | varchar (500) | 其他别名 |  |
| English\_name | varchar (100) | 英文名称 |  |
| Sale\_status | int | 销售状态 | 非空 |
| Update\_time | timestamp | 更新时间 | 非空 |
| Create\_time | timestamp | 创建时间 | 非空 |

表4-5 车辆制造厂商表

Table 4-5 article category information sheet

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 | 备注 |
| id | int | 厂商ID | 主键 |
| name | varchar (100) | 厂商名称 | 非空 |
| Brand\_id | int | 品牌ID | 外键 |
| seo\_name | varchar (100) | App显示名称 |  |
| other\_name | varchar (500) | 其他别名 |  |
| English\_name | varchar (100) | 英文名称 |  |
| Sale\_status | int | 销售状态 | 非空 |
| Update\_time | timestamp | 更新时间 | 非空 |
| Create\_time | timestamp | 创建时间 | 非空 |

表4-6 车型表

Table 4-6 article category information sheet

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 | 备注 |
| id | int | 车型ID | 主键 |
| name | varchar (100) | 车型名称 | 非空 |
| make\_id | int | 厂商ID | 外键 |
| seo\_name | varchar (100) | App显示名称 |  |
| other\_name | varchar (500) | 其他别名 |  |
| English\_name | varchar (100) | 英文名称 |  |
| Sale\_status | int | 销售状态 | 非空 |
| Price\_low | Double | 最低价格 | 非空 |
| Price\_high | double | 最高价格 | 非空 |
| Update\_time | timestamp | 更新时间 | 非空 |
| Create\_time | timestamp | 创建时间 | 非空 |

表4-7 车款表

Table 4-6 article category information sheet

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段类型 | 字段说明 | 备注 |
| id | int | 车款ID | 主键 |
| name | varchar (100) | 车款名称 | 非空 |
| model\_id | int | 车型ID | 外键 |
| Year | Double | 车款年份 | 非空 |
| Sale\_status | Int | 销售状态 |  |
| Body\_color | Varchar(100) | 车身颜色 |  |
| Update\_time | timestamp | 更新时间 | 非空 |
| Create\_time | timestamp | 创建时间 | 非空 |

1. 用户-文章相关表

如表4-7所示，用户-文章相关表主要用于记录用户和文章的匹配度，而表格中匹配度是通过不同的推荐策略计算而得出的结果，具体使用的推荐策略存储在推荐算法类别表中。

表4-8 用户-文章相关表

Table 4-8 article category information sheet

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字段名称 | | 字段类型 | 字段说明 | 备注 | |
| dvid | int | | 车款ID | | 联合主键 |
| Article\_id | varchar (100) | | 文章ID | | 联合主键 |
| Score | Double | | 匹配度 | | 外键 |
| Rcmd\_method | Int | | 推荐策略类别 | | 联合主键 |

1. 推荐策略类别表

如表4-9所示，推荐策略类别表中存储的是计算用户-文章匹配度所使用的多种推荐策略，推荐策略类别表中主要存储的是推荐策略ID、推荐策略名称。

表4-9 推荐策略类别表

Table 4-9 article category information sheet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | | 字段类型 | | 字段说明 | 备注 | |
| Id | int | | 推荐策略ID | | | 主键 |
| name | varchar (100) | | 推荐策略名称 | | | 非空 |

1. 文章关键词表

如表4-10所示，文章关键词表中包括文章ID、关键词、关键词得分三个字段。文章使用TF-IDF算法进行关键词的提取，并将关键词及其得分存储在文章关键词表中。

表4-10 文章关键词表

Table 4-10 article category information sheet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | | 字段类型 | | 字段说明 | 备注 | |
| Article\_id | varchar (100) | | 文章ID | | | 联合主键 |
| keyword | varchar (100) | | 文章关键词 | | | 联合主键 |
| score | double | | 关键词得分 | | | 非空 |

1. 相关文章表

如表4-11所示，相关文章表中有三个字段分别是文章ID、相关文章ID、两篇文章的相似度，其中文章ID和相关文章ID联合起来共同作为相关文章表的主键。

表4-11 相关文章表

Table 4-11 article category information sheet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | | 字段类型 | | 字段说明 | 备注 | |
| Article\_id | varchar (100) | | 文章ID | | | 联合主键 |
| Related\_article | varchar (100) | | 相关文章ID | | | 联合主键 |
| score | double | | 文章相似度 | | | 非空 |

1. 关联规则表

如表4-12所示，假设A、B、C为文章，（A,B）->C代表的含义为看过A和B文章的人也会看C文章，此时，我们认为（A,B）这个集合和C具有关联性，我们称（A,B）为前件，C为后件，关联规则表中存储的就是我们通过计算得出的文章之间的关联性，即关联规则中的前件和后件两个字段。

表4-12 关联规则表

Table 4-12 article category information sheet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | | 字段类型 | | 字段说明 | 备注 | |
| prefix\_set | varchar (100) | | 前件 | | | 联合主键 |
| suffix\_set | varchar (100) | | 后件 | | | 联合主键 |

1. 评测指标类型表

如表4-13所示，评测指标类型表中主要存储用于评测推荐效果的指标信息，如各个算法的点击率、人均阅读率等，包括两个字段，分别是指标的ID和指标的名称

表4-13 评测指标类型表

Table 4-13 article category information sheet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | | 字段类型 | | 字段说明 | 备注 | |
| Id | varchar (100) | | 报表类型ID | | | 主键 |
| Name | varchar (100) | | 报表类型名称 | | | 非空 |

1. 指标统计结果表

如表4-14所示，我们将每天统计各个指标的值存储到指标统计结果表中，这些指标用于评测推荐系统在线上的推荐效果，帮助我们对推荐算法做出改进。

表4-14 指标统计结果表

Table 4-14 article category information sheet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | 字段类型 | | | 字段说明 | 备注 | |
| dt | Datetime | | | 日期 | 联合主键 | |
| Id | | integer | 指标ID | | | 联合主键 | |
| Value | | double | 指标值 | | | 非空 | |

1. 推荐报表系统用户表

如表4-15所示，推荐报表系统用户表主要用于存储报表系统的用户信息，包括用户ID、用户名、用户密码等信息。

表4-15 推荐报表系统用户表

Table 4-14 article category information sheet

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段名称 | | 字段类型 | | 字段说明 | 备注 | |
| User\_id | | Integer | | 用户ID | 主键 | |
| User\_name | Varchar(100) | | 用户名 | | | 非空 | |
| password | Varchar(100) | | 密码 | | | 非空 | |

1. 推荐算法及策略设计
   1. 基于内容的推荐

基于内容的推荐算法主要根据用户的历史兴趣特征与文章内容的相似度来产生对该用户的文章推荐列表。基于内容的推荐算法主要包含4个步骤：用户兴趣特征构造，文章内容关键词提取，计算相似度并产生推荐。

* + 1. 基于内容的推荐流程

用户的兴趣特征主要通过分析用户在历史上看过的一些文章内容来构造出用户感兴趣的关键词信息。通过实时消费用户行为日志，记录用户看过的文章信息，然后对文章进行关键词提取操作，将文章关键词信息作为用户的特征进行记录，通过用户行为的不断更新，用户特征的权重也会不断更新，用户感兴趣的关键词权重会越来越高，为了能够更好地表达用户最近一段时间的兴趣，我们采用权重的时间衰减策略，对于长时间没有更新过权重的特征进行降权。

文章关键词的提取操作是在文章数据同步与更新时进行的，当系统收到文章更新请求时，首先将文章的基本信息进行存储，然后调用文章分词接口将文章进行分词，最终利用文章的分词结果计算TF-IDF值作为文章词语的权重值，输出权重值最高的词语作为文章最终的关键词，文章的关键词数据会被结构化地存储在数据库中供线上实时调用。

当基于内容的推荐引擎收到某一个用户推荐请求时，首先将该用户的特征数据和文章关键词数据向量化，计算该用户特征向量与候选集文章向量的余弦相似度，选择相似度最高的文章作为基于内容的推荐引擎的推荐结果列表。

* + 1. 基于内容推荐的优势与劣势

基于内容的推荐算法有如下优点：（1）用户之间是相互独立的，计算某一个用户的兴趣偏好的时候不需要依赖其他用户的行为数据，很好地解决了数据稀疏性问题。（2）当有新的文章加入到推荐候选集中，可以立即得到被推荐的机会，而且新文章被推荐的机会和老文章是一致的。（3）具有良好的推荐解释性，可以很容易地向用户提供合理的推荐理由，比如推荐的文章和用户关注的哪些关键词匹配。

基于内容的推荐算法也会有一些局限性：（1）文章特征的抽取过程中，只能抽取一些容易提取的文本类内容，而对于文章中的图片等多媒体数据的特征抽取具有一定的难度。（2）不能挖掘用户的潜在兴趣，只能为用户推荐和已有兴趣相似的文章内容。

* 1. 基于关联规则的推荐

基于关联规则的推荐算法主要是要发掘文章之间一些潜在关联性，并将这些相关性运用在我们的推荐上。举个例子，假如我们发掘出了如下一条关联规则：（A，B）->C，这条规则的含义是，同时看过文章A和文章B的文章有很大的可能性都会看文章C，根据这条规则，我们可以很容易地对用户做出推荐，为同时看过A和B文章的用户推荐C文章。由以上的例子我们可以发现，对于关联规则的挖掘是本节的重点。

* + 1. 支持度、置信度、提升度计算方式

支持度、置信度、提升度的计算和关联规则的挖掘紧密相关，是我们度量关联规则的三个重要指标，下面将用实例介绍支持度、置信度、提升度的具体计算方式。

支持度用来表示文章I1和文章I2同时被同一个用户点击的概率，如果支持度大代表两篇文章的关联程度就大，支持度具体计算方式如下所示，其中N(I1,I2)代表同时阅读过文章I1和文章I2的用户数量，N(u)代表用户总数。

置信度则表明了如果一个用户阅读过I1文章，则这个用户阅读文章I2的概率有多大，如果置信度为100%，则代表如果一个用户阅读了文章I1，则这个用户一定会阅读文章I2，置信度具体的计算方式如下所示，其中N(I1,I2)代表同时阅读过文章I1和文章I2的用户数量，N(I1)代表阅读过文章I1的用户数。

提升度表达的意思是文章I1是否被阅读能否影响文章I2被阅读，提升度具体的计算方式如下所示，其中代表置信度，即阅读文章I1后阅读文章I2的概率，P(I2)代表没有阅读文章I1但是确阅读文章I2的用户出现的概率。

* + 1. 频繁项集生成算法的比较

所谓项集，在本项目中指的是文章的组合，K-项集指的是具有K篇文章的项集，而频繁项集就是指在所有用户浏览记录中出现最频繁的文章的组合，我们要找文章之间的关联性，就要首先发掘频繁项集，目前主流的频繁项集生成算法主要有Apriori、FP-growth、Eclat算法，在本节中我们将对这三种算法进行优缺点的分析，选择最适合本项目的算法来生成频繁项集。

Apriori算法是一种经典的频繁项集算法，采用一种迭代搜索的方式，首先找到1-项集，对1-项集进行剪枝操作，将不满足最小支持度的项集删除，然后将剩余1-项集中前缀相同的项集进行连接组合生成2-项集。根据上述方法一直迭代下去，直至不能再生成满足最小支持度的项集。整个过程看成是剪枝，连接两个步骤不断地重复迭代，虽然算法简洁明了，但是这种算法的缺点也是显而易见的：（1）每次进行剪枝操作时都需要遍历一次整个数据集，大量的I/O操作造成了该算法的性能瓶颈。（2）连接操作过程中产生大量的无效的候选集，使得计算的空间复杂度非常高。

FP-growth算法中最大的亮点是对FP-tree（Frequent Pattern Tree）的构建，该算法首先对数据集进行遍历，将数据集中的信息全部保存到树结构中，而且保存了项集之间的关联关系，很好地避免了大量候选集的产生，构建完树结构后，然后遍历FP-tree的头表产生一系列条件模式基，每个条件模式基就是一个长度为1的频繁项集的前缀，然后继续分别对条件模式基构建条件FP-tree，最终产生频繁项集。综合来看该算法引入了树结构，并采用分而治之的策略，相对于Apriori算法优点显而易见：（1）计算的过程不需要产生大量候选集（2）只需要遍历两次数据集就可以完成整个算法。虽然FP-growth比较高效，但是在FP-tree的构建过程对内存的消耗很大，容易造成内存溢出。

* + 1. 推荐过程

具体的推荐步骤主要分为三步，分别是：根据原始数据集生成频繁项集，根据频繁项集生成强关联规则，根据强关联规则产生推荐。

关联规则算法使用的数据集是用户浏览文章的历史记录，每一条记录代表一个用户历史浏览过的文章的集合。

第一步，我们利用上一节讨论过的FP-growth算法对原始数据集进行处理，这个过程不断采用支持度进行剪枝操作，将不满足最小支持度的文章项删除，直到所有文章项集满足最小支持度，即生成频繁项集。

第二步，根据第一步产生的所有频繁项集生成强关联规则，是否为强关联规则通过计算置信度来判断，删除不满足最小置信度的关联规则，直至所有关联规则都满足最小置信度，即生成强关联规则。

第三步，计算上一步生成的所有强关联规则的提升度，当提升度小于等1时，为无效的关联规则。举个例子，若等于1，则证明I1和I2是相互独立的，也就是不管I1是否出现，对I2的出现都毫无影响，这种情况下，即使得到了很高的支持度和置信度，但是它也不是一条有效的关联规则，至此，删除所有无效关联规则，即生成最终的有效强关联规则。

第四步，生成用户推荐列表时，根据用户的历史行为记录，按照关联规则进行筛选，得到和用户浏览过的文章相关联的文章生成推荐。

由于用户行为记录数据规模很大，因此将数据集分布式地存储在HDFS上，关联规则的计算方式采用MapReduce分布式离线计算，并将离线计算的结果周期性更新到数据库中，保证线上调用时的响应速度。（离线计算的结果是用户的推荐结果还是关联规则？？？？？？？？？？？？？？）

* + 1. 关联规则优缺点
  1. 基于项目的协同过滤的推荐算法

基于项目的协同过滤推荐算法通过用户评分矩阵和文章相似度矩阵计算用户与文章之间的相似度。该算法是推荐系统中最常用的基本算法，而且早在业界中得到了广泛的应用。

该方法的基本思想就是给用户推荐的文章是与该用户之前喜欢的文章相似的文章，其中文章相似度的计算方式和基于内容的推荐算法的相似度计算方式不同，基于项目的协同过滤算法的相似度计算是通过分析用户的历史行为记录得到的，比如文章A和文章B的相似度很高，是因为喜欢文章A的用户很大一部分也喜欢文章B。

基于邻域的算法主要包括两类，一类是基于用户的协同过滤算法，另外一类是基于项目的协同过滤算法，由于随着用户数据的不断增加，使用基于用户的协同过滤算法的在计算上时间复杂度和空间复杂度都非常大，因此我们在本项目中采用基于项目的协同过滤算法。

* + 1. 文章相似度的计算及改进方式

首先根据用户的历史行为计算文章之间的相似性，计算相似性的方式为统计喜欢A文章的用户中有多少比例的用户同时也喜欢B文章，具体的计算公式为，这个公式中分母代表喜欢文章i的用户数量，分子代表同时喜欢文章i和文章j的用户数量，这样看来，这个公式是有一个缺点的，因为当喜欢文章j的用户非常多，文章j是非常热门的文章的时候，这个公式的值会趋近于1，造成的结果就是任何一篇文章和热门文章的相似度都很高，很显然我们并不希望每次推荐的结果都是热门商品，因此我们根据这个缺点对公式做出改进。

将公式改进为，改进后的这个公式对文章j的权重进行惩罚，从而减小了热门文章和其他文章的相似度。

通过以上的公式，我们可以看到，在基于项目的协同过滤算法中，两篇文章具有相似度是因为这两篇文章同时被很多用户感兴趣，也就是说，很多用户对相似度的产生都有贡献，但是由于用户的活跃度不同，因此用户对相似度的贡献也不尽相同。经过分析，我们发现有些用户虽然很活跃，但是很多行为并不是因为自身的兴趣而产生的，因此我们引入IUF （Inverse User Frequence）来对文章相似度的计算进行改进。IUF是用户活跃度的对数的倒数，改进之后的公式对活跃用户进行了惩罚，使得相似度的计算更加合理，具体的计算公式为XXXXXXXXXXXXXXXXXXXXXXXXXXX。

* + 1. 用户评分矩阵计算

用户在app上对文章的浏览、收藏等行为属于隐形反馈数据集，这些隐形反馈数据集在一定程度上能够反映用户对文章的评分，我们制订了如下规定，当用户u对文章i有收藏行为的时候，用户u对文章i的评分为2，当用户u对文章i有浏览行为的时候，用户u对文章i的评分为1，当用户u对文章i没有任何行为的时候，用户u对文章的评分为0。根据以上规定对所有用户行为进行统计可以得到m行n列的用户评分矩阵Rm×n，其中评分矩阵中第i行，第j列的元素rij表示用户i对文章j的隐性评分。

由于每个用户对文章的行为及其有限，因此评分矩阵中评分非零的元素远远小于矩阵元素的总数，是一个稀疏矩阵，因此我们在对矩阵进行存储时，为了节省空间，我们只存储非零元素，将（用户，文章，评分）三元组结构化地存储在数据库中。

* + 1. 基于项目的协同过滤推荐的基本流程

基于项目的协同过滤算法的实现主要包括三个部分，第一部分计算文章相似度矩阵，第二个部分计算用户评分矩阵，最后综合前面两部分计算用户对文章的兴趣度，产生用户的推荐列表集合。

评分矩阵和文章相似度矩阵的计算涉及对大量行为日志的处理和分布式离线计算，因此我们对评分矩阵和文章相似度矩阵进行周期性离线更新，在线上避免复杂的计算过程，从而保证了线上能够实时处理推荐请求。

项目之间的相似度计算可以使用5.3.1节介绍的经过IUF参数修正的相似度计算公式，计算完文章相似度矩阵和用户评分矩阵后，我们使用如下公式计算用户u对一个物品j的兴趣：

puj=∑i∈N(u)∩S(j,K)ruiwij

上式中N(u)表示用户感兴趣的文章的集合，S(j,K)表示和文章j相似度最高的K篇文章，wij表示文章i和文章j的相似度，rui表示用户u对于文章i的评分，该公式表明和用户以往喜欢的文章相似度越高的文章，最后在该用户的推荐列表中排名越高。

根据以上兴趣度公式的计算选择用户兴趣度最高的前N篇文章作为该用户的推荐集合，将离线计算好的推荐结果结构化地更新到数据库中，供推荐系统在线上调用。

* + 1. 基于项目的协同过滤算法的优势和不足

基于项目的协同过滤算法有如下优点：（1）用户的推荐结果和用户的行为紧密相关，如果用户有了新的行为，一定会体现在最终的推荐结果中。（2）对于汽车类的文章来说，用户的兴趣相对比较稳定，可以将复杂的计算工作离线进行，并周期性更新，这样可以保证线上的响应速度。（3）相似度的计算来源于用户的历史行为，能够合理地为用户提供推荐理由，增加用户的信任度。

基于项目的协同过滤算法也有一些不足的地方：（1）需要维护一个文章相似度矩阵，随着文章数量的增加，所需要的空间代价也随之增加。（2）不能解决冷启动问题，新用户没有历史行为记录，因此不能为新用户做出合理的推荐。（3）新文章没有被任何用户浏览过，因此也不会被推荐给用户。

* 1. 点击预估模型的设计
     1. XGBoost算法设（模型使用的什么参数）
     2. GBDT算法设计
     3. XGBoost算法与GBDT算法比较
     4. 时间复杂度
     5. 空间复杂度

1. 推荐算法设计