

毕业设计（论文）

基于python的水果评价推荐系统设计与实现

|  |  |
| --- | --- |
| 学 院 | 管理学院 |
| 专业名称 | 信息管理与信息系统 |
| 班级学号 | 1133210 |
| 学生姓名 | 周天阔 |
| 指导教师 | 丁斌 |

**2016年5月22**日

基于python的水果评价推荐系统设计与实现

摘 要

评价推荐系统，是在消费者进行选择时的参考依据，通过协同过滤算法找到与用户喜好相似的用户，根据相似用户的爱好，进行推荐。通过评价推荐系统，可以使消费者更全面的了解商品信息，更快的帮消费者找到心仪的商品。

本文讲述了一个基于python的水果评价推荐系统的实现与设计。系统采用b/s架构，使用了了django和bootstrap框架，数据库采用的mysql，缓存使用的redis。推荐算法使用的协同过滤算法，通过基于用户的协同过滤算法，为其进行推荐。主要工作如下：

1.推荐算法的选择，使用：采用基于用户的协同过滤算法，根据用户的评价矩阵找到与其爱好相似的用户，然后据此对用户进行水果推荐。

2．数据的爬取，清洗：采用scrapy爬虫框架和Python脚本进行网站初始数据的收集工作，为网站提供数据以及图片的基础。

3.数据库的表结构设计：通过mysql数据库对网站用户以及水果信息进行保存，方便使用。

4.网页前端的样式设计以及用户交互设计：前端采用bootstrap样式集进行设计，友好的交互设计与美观的视图，提升网站的用户体验。

5.网页后端功能：用户可以在网站上进行登陆，注册，发帖，搜索感兴趣的水果，进行点赞评论，和获取个性化推荐。

关键词：协同过滤，推荐系统，数据爬取，python

The design and implementation of the fruit evaluation and recommendation system based on python

Author：Zhou Tiankuo

Tutor：Ding Bin

Abstract

Evaluation and recommendation system is the important reference when consumer choose.By Collaborative filtering algorithm, we can find users whose hobby is similar to the target user, the evaluation and recommend system give a suggestion to the target user based on them.

The paper describes the implementation and design of the evaluation and recommendation system of fruit. The system adopt b/s architecture,it uses the Django and bootstrap framework.the database adopts mysql and redis.The recommend algorithm uses Collaborative filtering algorithm,by the Collaborative filtering algorithm based on users,the system give a recommendation to the target user. Main work of this paper is listed as follows:

1.the selection and use of the recommend algorithm: using the Collaborative filtering algorithm based on users, we can find users whose hobby is similar to the target user by the evaluation matrix and hava a fruit recommendation for user based on them.

2.the crawing and clean of the data:using the scrapy crawler framework and Pyhton scripts to collect the site’s initial data,providing the site with data and image.

3.the design of the structure of the table in database:saving the information of user and product in mysql database,it is convient when data is being used.

4.the design of the front of the web: the front of the web adopts the bootstrap framework.the friendly interaction design and wonderful view,enhence the experience of the users of the web site.

5. Web backend function: users can login, register, post a message, search their favorite fruit, deliever some comments, and get personalized recommendations on the website.

**Key Words:** Collaborative filtering algorithm, Recommendation System, The Collect of Data,Python

目录

[1 绪 论 1](#_Toc483432691)

[1.1 课题研究的背景及意义 1](#_Toc483432692)

[1.2 国内推荐系统的研究现状 1](#_Toc483432693)

[1.2.1 国内研究现状 1](#_Toc483432694)

[1.2.2 国外研究现状 3](#_Toc483432695)

[1.3研究内容 4](#_Toc483432696)

[1.4 论文结构 5](#_Toc483432697)

[2 相关技术介绍 6](#_Toc483432698)

[2.1 Python Web相关技术 6](#_Toc483432699)

[2.1.1 B/S架构 6](#_Toc483432700)

[2.1.2 Django框架及MTV模式 6](#_Toc483432701)

[2.1.3 Bootstrap框架介绍 7](#_Toc483432702)

[2.1.4 Ajax介绍 7](#_Toc483432703)

[2.1.5 d3.js介绍 8](#_Toc483432704)

[2.2数据库相关技术介绍 8](#_Toc483432705)

[2.2.1 Mysql数据库介绍 8](#_Toc483432706)

[2.2.2 Redis数据库介绍 9](#_Toc483432707)

[2.3 推荐算法综述 9](#_Toc483432708)

[2.4 本章小结 9](#_Toc483432709)

[3 系统需求分析 10](#_Toc483432710)

[3.1 用户偏好的收集和表示 10](#_Toc483432711)

[3.2 计算用户间的相似度 11](#_Toc483432712)

[3.3 寻找相似近邻 12](#_Toc483432713)

[3.4 产生推荐结果 13](#_Toc483432714)

[4 数据的爬取和清洗过程 15](#_Toc483432715)

[4.1 爬虫的简单介绍 15](#_Toc483432716)

[4.2 scrapy和beautifulsoup4介绍 15](#_Toc483432717)

[4.3 数据爬取过程 16](#_Toc483432718)

[4.4 数据清洗过程 17](#_Toc483432719)

[5 数据库表设计 18](#_Toc483432720)

[5.1 mysql数据库表的设计与建立 18](#_Toc483432721)

[5.2 redis的介绍和使用 18](#_Toc483432722)

[6 网站前端页面的设计 19](#_Toc483432723)

[6.1 网站模块介绍 19](#_Toc483432724)

[6.2 Bootstrap和Ajax介绍和使用 21](#_Toc483432725)

[6.3 利用d3.js的数据可视化实现 22](#_Toc483432726)

[7 网站后端功能实现 25](#_Toc483432727)

[7.1 django介绍 25](#_Toc483432728)

[7.2 django-pagination的介绍和应用 26](#_Toc483432729)

[结 论 27](#_Toc483432730)

[致 谢 28](#_Toc483432731)

[参考文献 29](#_Toc483432732)

[附 录 31](#_Toc483432733)

[附录A 31](#_Toc483432734)

1 绪 论

这一章主要对推荐系统的研究背景，意义进行介绍。引出水果评价推荐系统的需求场景以及存在的一些问题和难点，最后对全文的结构以及内容编排进行简要阐述。

1.1 课题研究的背景及意义

信息技术的飞速发展使得各式各样的互联网技术以及互联网+服务走向了人们的生活当中，为大家日常的衣食住行提供了便利。但是我们又该如何从愈加丰富的数据中找到我们所需要的信息呢，这也就引出了推荐系统所常用的应用场景，利用大数据来提高我们的效率。推荐系统通过各式各样的推荐算法，通过分析商品的信息或者对用户以往的行为进行分析，为其进行一个推荐，尽可能地提升用户找到满意商品的效率，提高消费者购物的效率，从而提升其购物体验以及满意程度。

而在现在如今的超市中，采取的推荐方式大部分不具有针对性，特别是同类商品中，大部分都是有由消费者自行比较品牌，价格等因素，对商品作出衡量再进行购买。而对一些老年消费者，其购买商品主要依靠于自己的购买习惯。针对这样的超市购物环境，我产生了设计一个超市水果评价推荐系统的想法。通过这个系统，消费者在超市购物的时候，能够更快地了解到当季水果的热门程度，以及各种水果的营养成分含量，使消费者有更好的消费体验，更快地找到自己心仪的水果。

1.2 国内推荐系统的研究现状

1.2.1 国内研究现状

近几年，国内推荐系统的应用正在越来越广泛，例如，你在京东和淘宝购物时，都会根据你的消费习惯或者搜索记录，向你推荐一些相关的你可能喜欢的商品；又或者你在使用一些音乐播放器时，播放器也会像你推荐一些你可能喜欢的音乐；这些都是推荐系统在我们生活中的应用。但是总体来说，国内对推荐系统的相关研究以及应用与国外相比还是稍显不足的，主要有以下几点：1）国内有关推荐系统的研究主要沿袭国外的研究理念及成果，缺乏创新；2）国内有关的理论研究较国外相比较少，各个推荐系统所运用模型相似，缺乏多样性；

虽然如此，但是国内对于推荐系统的研究也存在一些建设性的指导意见，此处进行参考文献综述。

随着互联网的飞速发展，我国的电子商务英语以及个性化推荐领域已经逐步和国际接轨，呈现蓬勃发展之势，目前国内不少论坛，电商平台和音乐播放器对个性化推荐特别是协同过滤算法方面的研究都十分深入，其中包括：豆瓣网（www.douban.com），网易云音乐等。

豆瓣网是国内最早使用个性化推荐系统功能的网站之一，它提供个性化推荐功能，如果1.1所示，主要使用协同过滤算法为用户推荐用户可能喜欢的内容，包括图书，电影，音乐活动等等。豆瓣的个性化推算法是基于显性和隐形两部分，显性体现在用户可能会对某一部电影或者某一首歌曲进行打分或者评价，这显性的表明了用户的喜好。而隐形体现在用户的喜欢，收藏，关注等行为。通过这些行为，每天推送给用户其最有可能感兴趣的内容，该项服务已经在图书，影音，豆瓣fm等多产品线上不断地进行尝试。



图1.1 豆瓣图书推荐界面

网易云音乐，是由网易公司出品的一款当下比较热门的音乐播放器，其之所以能够在几近饱和的音乐市场占领到一席之地，离不开它较好的用户体验和完善的推荐算法。网易云音乐主要采用潜在因子（Latent Fator）算法，之所以采用这种算法，与其产品特性离不开关系。与观影和购买图书不同，大部分听音乐的用户没有对音乐进行打分的习惯，因为用户普遍喜好不同，喜欢的音乐类型不同，所以从用户的分数直接得到用户的喜好在音乐播放器上实现起来不太现实。而网易云音乐的潜在因子算法，其实也就是协同过滤的一种。它可能用以下方法来定义一首音乐对用户的分数：单曲循环=5，表明用户非常喜欢这首歌，分享=4，比最喜欢低一个等级，收藏-=3，听完=1，跳过=-2，拉黑=-5.这样子就可以得出用户对听过每首歌的一个评价。然后将用户与其他用户进行对比，找到与之相近的同一类人，为其推荐他喜欢的音乐。一个对用户喜好判断准确的推荐系统，无疑会赢得用户的好感。如图1.2所示：



图1.2 网易云音乐每日歌曲推荐界面

1.2.2 国外研究现状

随着协同过滤算法对个性化推荐系统产生巨大的影响，该项技术在在线学习，电子商务，信息服务网站等领域一直保持较高研究热度。近年来，出现了很多的研究机构，也涌现了一大批有关协同过滤的学术论文。国外有较深入研究的单位有：纽约大学，明尼苏达州立大学的GroupLens研究小组，卡内基梅隆大学、以及微软研究院等。

在最早期有关协同过滤推荐系统中，系统需要用户指定兴趣相似的近邻，只有在用户了解了彼此之间的兴趣爱好之后才能做出推荐，所以，这种模式只适用于规模很小，并且用户之间熟悉度高的小型用户组。之后出现了以GroupLens为代表的自动化的协同过滤推荐系统，它允许用户使用评分等方式表达自己的兴趣偏好信息，系统根据兴趣相似的用户的评分信息为当前用户进行个性化推荐。自动化的系统过滤推荐系统不需用户指定相似用户，不需要用户之间相互了解，也不需要分析系统项目的内容，它是通过自动虚招兴趣相似的用户，发现用户潜在兴趣，具有较高的个性化和自动化程度。也正因为此，协同过滤算法收到了广泛关注。

1.3研究内容

尽管现在协同过滤算法已经在电子商务，个性化推荐等诸多领域有着广泛地实际应用，但随着系统规模的不断扩大，应用场景的更加多元化，协同过滤算法也逐渐显露了一些缺点和不足，严重影响着推荐结果的质量。本文主要研究协同过滤推荐算法在水果推荐系统中的应用，以及中间需要克服的相关问题，例如：

1. 稀疏性问题

协同过滤算法进行推荐采用的主要依据是用户对项目的评分，用户评分越多，推荐效果越好。然而在实际的推荐系统中，用户的精力是有限的，是不会对所有的项目都进行评价的，所以总会出现大量的项目没有被用户评分，导致用户-项目评分矩阵上成为一个高维稀疏矩阵，根据此高维稀疏矩阵计算所得到的用户间或项目间的相似度准确性并不高，会导致推荐结果的质量下降。

2. 冷启动问题

冷启动问题主要包括新用户冷启动问题和新项目冷启动问题两种情况。其中，新用户冷启动问题是实际生产应用中推荐系统面对的严峻挑战之一，它可能导致用户对推荐系统的不信任从而引发用户的大量流失。

新用户冷启动问题是指当新用户第一次使用推荐系统时，推荐系统中并没有新用户的相关评价数据，协同过滤算法很难准确地计算新用户与其他用户之间的相似度。在这种情况下，协同过滤算法很难准确地想新用户推荐其感兴趣的项目。然而，如果新用户对推荐系统最初的推荐结果不满意，那么用户就可能会不信任这个系统，因此，提高新用户个性化推荐效果，提升新用户对系统的满意度，防止用户流失。

新项目冷启动问题与新用户冷启动问题类似，是指当一个新项目刚被加入系统，在初始阶段，因为新项目刚上线，对其评价过的用户很少，所以在评价达到一定数量之前，它很难获得到推荐。但是另一方面，如果它一直得补到推荐，它也就更不容易被用户所评价，导致其进入恶性循环。

最后本文详细描述了一个基于协同过滤推荐算法的推荐系统前后端乃至数据库的设计与实现。

1.4 论文结构

本文的其余章节按如下结构组织：

第二章 首先分析推荐系统的的形式化定义，对当今热门的推荐系统算法进行一些简单的描述和比较。

第三章 详细描述协同过滤算法的实现原理，分析其在超市水果评价推荐系统中的应用场景。

第四章 讲解数据的爬取和清洗过程。

第五章 描述mysql中表结构的设计以及redis的使用，讲解网站中所需用到的数据的存储方式及原因。

第六章 分析整个系统的前端样式设计以及用户交互体验的设计。展现系统的前端样式，以及bootstrap框架的使用。

第七章 描述网站后台功能的业务逻辑实现，讲解每个功能点如何实现，网站访问过程中可能存在的问题，在设计过程中是如何设计去解决的。

第八章 对全文进行总结，对现有工作中存在的不足进行归纳，并且提出未来有待改进和解决的几个问题与想法。

2 相关技术介绍

2.1 Python Web相关技术

2.1.1 B/S架构

B/S架构即浏览器/服务器架构，用户通过浏览器对服务器应用进行访问，服务器根据用户的请求在服务器上进行后台逻辑的运算之后将结果通过html页面呈现给用户。B/S架构将主要的负担交给了server端，将操作界面交给了borrow端，在这一点上B/S与C/S端有很大的不同，大大的减少了客户端的负担。

水果评价推荐系统应用B/S端的优点在于方便，高效，交互性强。方便：使用B/S架构，用户无需下载安装单独的客户端，不需要复杂的环境配置，只需要拥有web浏览器既即可以访问应用。高效：通过浏览器进行访问，快速并且客户端压力较小，可以给用户良好的体验。交互性强：基于html的页面访问，使用户的使用过程中具有高交互性。

2.1.2 Django框架及MTV模式

Python Web在快速web开发中，正运用的越来越多，而django框架也是当下比较成熟的Python Web框架之一，使用django框架可以提高我们web的开发速度并且减少我们所需要的代码量，所以在这次网站开发中，我们选用了django框架作为开发基础。

相信对大多数开发人员来说，mvc模式都是十分熟悉的，在这个模式中，model代表数据存取层，view代表的是系统中选择显示什么和怎么显示的部分，controller指的是系统中根据用户输入并视需要访问模型，以决定使用哪个视图的那部分。而django框架也紧紧地遵循这种mvc模式，衍生了自己的mtv框架，如图7-1所示。

M代表数据模型（Model），即数据存取层，该层处理与数据相关的所有事务：如何存取，如何验证有效性，包含哪些行为以及数据之间的关系等。

T代表模板（template），即表现层，该层处理与表现相关的决定：如何在页面或其他类型文档中进行显示。

V代表视图（view），即业务逻辑层。该层包含存取模型及调取恰当末班的相关逻辑。你可以把它看做模型模板之间的桥梁。



图7-1 django的mtv模式

2.1.3 Bootstrap框架介绍

Bootstrap，是Twitter开发使用的一套前端框架，目前使用范围很广。Bootstrap是基于HTML，CSS，JAVASCRIPT的，它简洁，灵活，美观，通过使用bootstrap，我们可以快速的获取一些样式集，使得我们在web前端的开发过程中能够更加的方便快捷，其适用于开发响应式布局，尤其在移动设备上的web项目上运用效果更佳。

2.1.4 Ajax介绍

Ajax由HTML，JavaScript技术DHTML和DOM组成，通过ajax可以将普通的web界面转化成交互性的Ajax应用程序。AJAX=Asynchronous JavaScript and XML(异步的Javascript和xml)，聊到这里，我们还需要明白一个概念，即同步和异步。他们有什么区别。

同步的定义：发送方在发出数据后，等待接收方回响应之后才发下一个数据包的通讯方式。

异步的定义：发送方在发出数据后，不等接收方发回响应，紧接着发送下个数据包的通讯方式。在B/S结构中，

同步:提交请求—等待服务器处理-处理完毕返回 这个期间客户浏览器不能干任何事

异步：请求通过事件触发-服务器处理（这时浏览器还可以作其他事情）-处理完毕

再举个现实生活中的例子同步就是你叫我去吃饭，我听到了就和你去吃饭；如果没听到，你就不停地叫，知道我告诉你我听到了，才一起去吃饭。而异步就是你叫我，然后自己去吃饭，而我得到消息后可能立即走，也可能等到下班才去吃饭。

Ajax就是这么一种异步的方法，它的最大优点就是在不重新加载整个页面的情况下，可以与服务器交换数据并更新部分网页内容。Ajax不需要任何浏览器插件，但需要用户允许javascript在浏览器上执行。通过ajax可以使网页的用户体验更好，在不用跳转和刷新整个页面的条件上完成浏览器和服务器的数据响应。

在此次系统开发中，水果评论功能使用了ajax技术，在点击评论按钮后，会将输入框的内容通过post方法传递到后台view层中，但是采用的是异步io，所以，无论评论是否成功，都会先进行响应，避免影响用户其他功能的体验。

2.1.5 d3.js介绍

d3.js是一个Javascript库，它可以通过数据来操作文档。D3可以通过使用HTM，svg和css等把数据以一种更加鲜活形象地方式展现出来。D3严格地遵循web标准，因而可以让你的程序的兼容性可以得到足够的保障。它利用javascript语言，为数据生成图表，在本项目中，我们主要用来生成柱状图和雷达图。通过柱状图使消费者能够更直观的感受到某种水果的营养含量的多少，而通过雷达图可以使消费者对两种水果的水果

的优势和劣势了解的更清楚。

2.2数据库相关技术介绍

2.2.1 Mysql数据库介绍

Mysql作为一种免费的开源的关系型数据库产品，由于其体积小，速度快，成本低的特点，被广大开发者所青睐。而在我们此次项目开发中，我们也是考虑到此其显著的优点，所以选用Mysql数据库作为我们系统主要的数据存储。

Mysql主要特点和作用：

Mysql允许多用户并发进行访问，能够多线程地实现数据库的操作，使系统的数据库的操作效率更高，用户得到响应的时间也会越短。

Mysql支持多种数据类型，使数据库表的字段建立更加灵活，存储的信息的多样性更高。

Mysql支持不同的存储引擎，可以根据业务的需要选择不同的引擎。例如：支持事务处理的InnoDB存储引擎提供了具有提交、回滚和崩溃恢复能力的事务安全。

2.2.2 Redis数据库介绍

系统中还使用了redis作为数据库的缓存体系，redis是一个开源的使用ANSI C语言编写的日志型，KEY-Value数据库，它和memcached类似，支持包括string（字符串），list（链表），set（集合），zset（有序集合）和hash（哈希类型）。Reids的出现，很大程度上弥补了memcached这类key-value存储的不足之处。在系统中，redis中存储了水果的点赞数据，通过redis作为缓存减少了mysql服务器的压力。通过redis作为缓存，用户进行点赞操作的时候，直接操作redis数据库，操作效率更高，避免了在高并发的情况下，mysql服务器压力承载不足的情况。

2.3 推荐算法综述

推荐系统算法是决定一个推荐系统是否成功的核心所在，选择一个合适的推荐算法对推荐系统的成功尤为重要。目前比较常用的推荐算法有:

基于内容的推荐算法：基于内容的推荐算法是一种不考虑用户评价信息的推荐算法，它把关注点集中于商品的属性上。根据用户之前的搜索记录，找到属性类似的商品，进行推荐。

协同过滤推荐算法：协同过滤推荐算法是现在一种比较常用的推荐算法，其逻辑实现比较简单，举个例子：在我们生活中，如果我们想买某个东西，都会问同龄段和我们爱好相同的人，而不会问那些和我们爱好不同的人。协同过滤算法就是这么一个道理，它会首先找到和用户爱好相似的用户，然后进行推荐。

基于关联规则的推荐：基于关联规则的推荐，是依靠曾经购买过该商品的人，还购买过其他什么商品作为推荐依据。例如:如果一个人曾经购买过牙膏，那么他就有很大的概率需要购买牙刷。

组合推荐：由于不同的场景可能应该使用不同的推荐算法，所以对一些复杂的场景我们可能使用一些加权，变换等策略来综合多种推荐算法，以达到最好的推荐效果。

2.4 本章小结

本章节主要介绍本次系统开发中主要用到了关键性的技术和相关的理论基础。首先介绍了Python Web相关技术：B/S架构，Django框架及MTV模式，Bootstrap框架，Ajax技术和d3.js。然后介绍了数据库中使用的相关技术，包括为什么使用mysql和redis。最后介绍了一些常用的推荐算法，以及其推荐原理。

3 系统需求分析

需求分析是软件开发生命周期的第一阶段，弄清楚一个系统的需求是对系统进行开发的首要问题，然而这个开头并不简单。因为开发与用户之间是缺少交流的，而开发只能根据自己的想法去揣测用户的需要，完成需求获取的过程。所以当我们进行需求分析时，一定要以更多的视角去思考问题，考虑到更多可能的因素，才能与用户的需求达成一致。确认各个业务功能达到理想的效果。

3.1 系统目标

“基于python的水果评价推荐系统”在本质上是一个推荐系统，旨在提升用户的挑选水果时的体验。在用户购买水果时，可以不再拘泥于个人习惯，可以更加全面的了解相关水果的信息，被推荐到一些自己平常很少食用但自己可能会喜欢的水果。并且系统中设有讨论区，供用户交流一些养生心得，以及一些对水果食用的想法，建议。

本网站系统应实现以下主要业务目标：

1）用户能够在网站上进行水果信息的查询，比较，获取推荐等基本功能。因此网站设置有水果广场，在这里用户可以搜索水果的信息，获取水果各个营养成分含量，和评价，对不同水果之间进行比较。另外设有个人推荐，利用用户之前的行为，分析用户的喜好，为用户推荐其可能喜欢的水果。

2）我认为一个网站不应该拘泥其基本功能，丰富多彩的功能可以提升用户的使用体验。所以网站还设有讨论区，营养食谱，卡路里计算器等模块。使用户和网站之间能有更多的交互，能够在网站上体验更多有趣的功能，并且把网站当成一个交流的地方，和其他用户进行分享。

3.2可行性分析

3.3 网站系统规划

体验。在用户购买水果时，可以不再拘泥于个人习惯，可以更加全面的了解相关水果的信息，被推荐到一些自己平常很少食用但自己可能会喜欢的水果。并且系统中设有讨论区，供用户

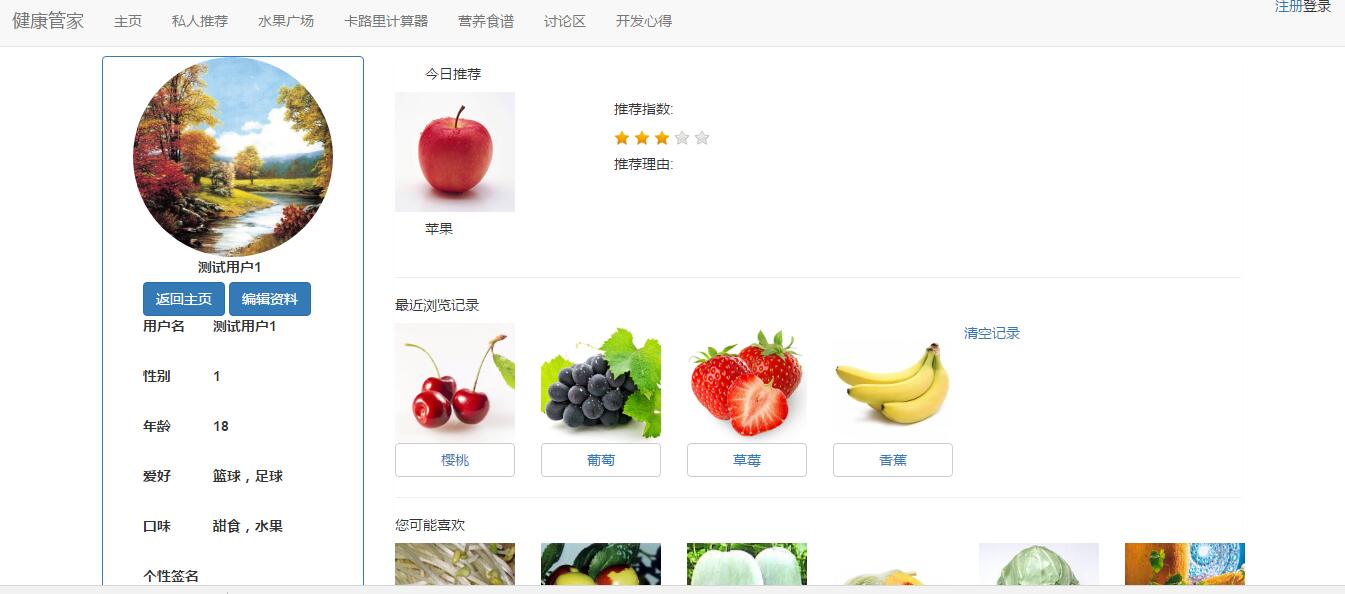
本网站的开发目标主要有三方面：

1. 为用户提供水果方面的资讯，健康常识等一些有趣的内容；
2. 利用用户的行为分析，为用户进行水果推荐；
3. 提供一个讨论区，供用户进行交流，给我们的网站提出一些建设性的意见；

本网站系统提供水果和健康养生的综合资讯，水果信息查询，比较，推荐，多样的菜谱和卡路里计算器，还有讨论区功能供大家交流，为了我们更加健康的生活护航。在栏目设置上有如下规划：

1) 首页模块主要提供用户浏览一些健康咨询，展示每天的水果热度，可以让用户了解到当季比较热门的水果是什么，为用户的水果挑选提供参考。

2) 私人推荐：提供用户的个性化推荐，可以编辑用户信息，查看每天的水果推荐，用户的浏览记录，以及根据用户的数据向用户推荐其可能喜欢的水果。如图6-2展示



3) 水果广场：提供查询系统，用户可以查询到各种水果的营养参数，并且有柱状图和雷达图可以生动形象的向用户展现水果的营养成分。如图6-3展示

****

图6-3水果广场模块

4) 卡路里计算器：用户可以使用卡路里计算器，计算自己每天的卡路里摄入量，让自己的饮食能够营养更加均衡。如图6-4所示：

5) 营养食谱：这个版块里展示了一些营养食谱，用户可以通过查阅网页上的食谱，尝试新的菜品。如图6-5所示：



图6-5营养食谱模块

6) 讨论区：用户可以在讨论区里面发帖，回复，进行交流。如图6-6所示：



图6-6讨论区模块

7) 管理员入口：管理员通过此入口进入管理网站，为网站水果信息的增删改，用户权限的增删改，网站资讯的更新，提供一个便利的渠道。

3.4 系统功能需求

通过分析用户对水果评价推荐系统的需求，得出网站应该包括的主要功能如图所示：

3.4.1网站管理员用例分析

3.4.2游客用例分析

3.4.3 注册用户用例分析

3.5 系统业务流程

3.5 本章小结

本章从系统的需求的角度来进行论述的，从网站的系统目标，系统的功能需求，可行性分析，系统规划，系统的业务流程等几方面来进行论述

4 系统设计

4.1 系统总体设计

4.1.1 系统体系结构设计

4.1.2 系统功能模块设计

4.2 系统主要功能模块详细

4.3 数据库设计

4.3.1 概念设计

4.3.2 逻辑设计

4.3.3 物理设计

4.4 系统E-R图

4.5 本章小结

5 系统实现

本章主要从系统网站的实现角度进行系统实现的详细介绍，从环境配置开始，到各个功能模块的具体实现，以及数据库的实现，下面将介绍系统的详细的实现过程：

5.1 系统环境配置

系统开发工具：

Sublime text3

MySql5.5

Redis

Mysql-front

系统运行环境：

Python2.7

5.2 系统功能模块实现

5.3 数据库实现

5.4 本章小结

结 论

在本次系统开发中，虽然项目整体难度不大，但是我也遇到了不少问题，使用了很多自己以前没有尝试过的新技术，第一次尝试用python做web，第一次使用d3.js进行图表的展示，数据的可视化，第一次进行数据分析，数据挖掘，了解到什么是协同过滤算法，其是如何实现的等。在解决很多问题的时候，我都踌躇了很久，判断方法的可行性。例如在使用cookies来保存用户的浏览记录时，我就在思考这里是否使用cookies保存比较好，如果用session或者直接存入数据库中是否会有更好的效果。不过考虑到用户的隐私性和访问效率。我觉得使用cookies是更合适，一方面，用户可以自己清理浏览器cookies来保护自己的隐私。另一方面，通过cookies，也能减轻数据库的读取压力，在一定程度上提高了用户访问网页的效率。总体来说，感觉经过这次系统开发，自己有了不少的收获，无论是程序的语法逻辑，还是对系统的架构理解，都有不少的进步。

经过数月的学习与实践，本设计最终按照预定的方案完成了整个系统的设计与实现，系统采用b/s架构，基于python的django框架，数据库使用的mysql和redis，mysql负责大部分数据的存储，redis负责存储大量读写的数据，例如点赞数。

本设计主要完成的工作如下：

（1）用户可以在网站上注册，登录，浏览自己感兴趣的水果信息，并且获取自己的个性化推荐结果。

（2）用户可以在讨论区发帖，与网站的其他用户进行交流和沟通。

由于时间和精力的原因，本系统尚为得到完全开发，有以下功能可供以后扩展：

（1）对推荐结果的验证性和修改：此次系统中只设计了根据用户喜好进行分析，向其提供个性化推荐，却没有对推荐结果是否与用户想法一致进行验证和分析。推荐结果的质量还有很大的提升空间。

（2）网站的移动端兼容设计：本次网站的前端采用bootstrasp框架，对移动端的兼容十分简单，但由于时间比较仓促，很多前端样式代码编写存在问题，在移动端的显示不是很美观，会在后期的工作进行修改。

致 谢

写到这里，关于系统的介绍和有关工作的展望已经阐述完毕了，作为全文的结尾，借此机会，我要向那些在我毕设和论文期间对我提供过帮助的同学和老师表示感谢。

首先，我要感谢我的毕业论文指导老师，丁斌老师，在研究过程中，通过和他的探讨，使我更加清楚了自己的研究方向，了解了自己有哪些问题需要解决，有哪些方向的知识需要去了解。

使自己对整个系统的架构和数据库的结构都有了清晰的认识。

然后，我要感谢那些给过我建议和帮助的同学，通过他们，我了解到了我的系统有那些不足，我应该从哪些方向着手去改进，因为站在自己的视角去思考问题，总会有一些局限性，有一些不足之处可能会很明显，但是站在自己的视角上，却很难看清楚。所以，我要感谢那些给我建议的同学，他们对这个系统的完善起到了很重要的作用。

最后，感谢自己给予自己鼓励的同学，在系统的设计和实现过程中，自己遇到了很多的困难，也曾经觉得很难完成这个系统。不过同学们一直给予我鼓励，让我相信自己，并且能一直坚持做下去。我很感谢他们，如果没有他们，这个系统可能只是我头脑中的一个想法，而无法实现。是他们，让我完成了这个系统，让我克服了这些困难，从而得到技术的提升与进步。

# 参考文献

1. 冷亚军,陆青梁,昌勇. 协同过滤推荐技术综述[J]. 模式识别与人工智能, 2014-08-15.
2. 蔡强,韩东梅,李海生,胡耀光,陈谊. 基于标签和协同过滤的个性化资源推荐[J]. 计算机科学.2014(01).
3. 荣辉桂,火生旭,胡春华,莫进侠 基于用户相似度的协同过滤推荐算法[J]．通信学报2014-02-25.
4. 夏培勇. 个性化推荐技术中的协同过滤算法研究[D]. 中国海洋大学, 2011-03-30.
5. 孔维梁. 协同过滤推荐系统关键问题研究[D].华中师范大学, 2013-09-01.
6. 孙小华. 协同过滤系统的稀疏性与冷启动问题研究[D]. 浙江大学,2005-12-01.
7. 徐蕾; 杨成; 姜春晓; 任勇. 协同过滤推荐系统中的用户博弈[J].计算机学报, 2015-07-23 .
8. 刘青文. 基于协同过滤的推荐算法研究[D]. 中国科学技术大学, 2013-05-01.
9. 孙慧峰. 基于协同过滤的个性化Web推荐[D]. 北京邮电大学, 2012-05-11.
10. 程淑玉. 基于聚类协同过滤的个性化推荐系统[J]. 宜宾学院学报, 2013-05-28.
11. 段玮. 基于协同过滤的个性化推荐算法研究[D]. 华中科技大学, 2009-11-01.
12. 王国霞; 刘贺平. 个性化推荐系统综述[J]. 计算机工程与应用, 2012-03-01
13. 任磊．推荐系统关键技术研究[D]．华东师范大学，2012-03-01
14. 梁昌勇; 冷亚军; 王勇胜; 戚筱雯. 电子商务推荐系统中群体用户推荐问题研究[J].中国管理科学, 2013-06-15.
15. 许海玲; 吴潇; 李晓东; 阎保平. 互联网推荐系统比较研究[J].软件学报, 2009-02-15.
16. Gediminas Adomavicius,Alexander Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions[J]. IEEE Transactions on Knowledge and Data Engineering . 2005
17. Jun Wang,Stephen Robertson,Arjen P. Vries,Marcel J. T. Reinders. Probabilistic relevance ranking for collaborative filtering[J] . Information Retrieval . 2008 (6)
18. Le Hoang Son. HU-FCF: A hybrid user-based fuzzy collaborative filtering method in Recommender Systems[J] . Expert Systems With Applications . 2014

附 录

附录A

Amazon.com Recommendations Item-to-Item Collaborative Filtering

Recommendation algorithms are best known for their use on e-commerce Web sites,1 where they use input about a cus-

tomer’s interests to generate a list of recommend- ed items. Many applications use only the items that customers purchase and explicitly rate to rep- resent their interests, but they can also use other attributes, including items viewed, demographic data, subject interests, and favorite artists.

At Amazon.com, we use recommendation algo- rithms to personalize the online store for each cus- tomer. The store radically changes based on cus- tomer interests, showing programming titles to a software engineer and baby toys to a new mother. The click-through and conversion rates — two important measures of Web-based and email advertising effectiveness — vastly exceed those of untargeted content such as banner advertisements and top-seller lists.

E-commerce recommendation algorithms often operate in a challenging environment. For example:

• A large retailer might have huge amounts of data, tens of millions of customers and millions of distinct catalog items.

• Many applications require the results set to be returned in realtime, in no more than half a second, while still producing high-quality rec- ommendations.

• New customers typically have extremely limit- ed information, based on only a few purchases or product ratings.

• Older customers can have a glut of information, based on thousands of purchases and ratings.

• Customer data is volatile: Each interaction pro- vides valuable customer data, and the algorithm must respond immediately to new information.

There are three common approaches to solving the recommendation problem: traditional collabora- tive filtering, cluster models, and search-based methods. Here, we compare these methods with our algorithm, which we call item-to-item collab- orative filtering. Unlike traditional collaborative filtering, our algorithm’s online computation scales independently of the number of customers and number of items in the product catalog. Our algo- rithm produces recommendations in realtime, scales to massive data sets, and generates high- quality recommendations.

Recommendation Algorithms

Most recommendation algorithms start by finding a set of customers whose purchased and rated items overlap the user’s purchased and rated items.2 The algorithm aggregates items from these similar customers, eliminates items the user has already purchased or rated, and recommends the remaining items to the user. Two popular versions of these algorithms are collaborative filtering and cluster models. Other algorithms — including search-based methods and our own item-to-item collaborative filtering — focus on finding similar items, not similar customers. For each of the user’s purchased and rated items, the algorithm attempts to find similar items. It then aggregates the simi- lar items and recommends them.

Traditional Collaborative Filtering

A traditional collaborative filtering algorithm rep- resents a customer as an N-dimensional vector of items, where N is the number of distinct catalog items. The components of the vector are positive for purchased or positively rated items and nega- tive for negatively rated items. To compensate for Amazon.com Recommendations best-selling items, the algorithm typically multi- plies the vector components by the inverse fre- quency (the inverse of the number of customers who have purchased or rated the item), making less well-known items much more relevant.3 For almost all customers, this vector is extremely sparse.

The algorithm generates recommendations based on a few customers who are most similar to the user. It can measure the similarity of two cus- tomers, A and B, in various ways; a common method is to measure the cosine of the angle between the two vectors: 4

Unfortunately, all these methods also reduce recommendation quality in several ways. First, if the algorithm examines only a small customer sample, the selected customers will be less similar to the user. Second, item-space partitioning restricts recommendations to a specific product or subject area. Third, if the algorithm discards the most popular or unpopular items, they will never appear as recommendations, and customers who have purchased only those items will not get rec- ommendations. Dimensionality reduction tech- niques applied to the item space tend to have the same effect by eliminating low-frequency items.Dimensionality reduction applied to the customerspace effectively groups similar customers into

The algorithm can select recommendations from the similar customers’ items using various meth- ods as well, a common technique is to rank each item according to how many similar customers purchased it.

Using collaborative filtering to generate recom- mendations is computationally expensive. It is O(MN) in the worst case, where M is the number of customers and N is the number of product cat- alog items, since it examines M customers and up to N items for each customer. However, because the average customer vector is extremely sparse, the algorithm’s performance tends to be closer to O(M + N). Scanning every customer is approxi- mately O(M), not O(MN), because almost all cus- tomer vectors contain a small number of items, regardless of the size of the catalog. But there are a few customers who have purchased or rated a significant percentage of the catalog, requiring O(N) processing time. Thus, the final performance of the algorithm is approximately O(M + N). Even so, for very large data sets — such as 10 million or more customers and 1 million or more catalog items — the algorithm encounters severe perfor- mance and scaling issues.

It is possible to partially address these scaling issues by reducing the data size.4 We can reduce M by randomly sampling the customers or discarding customers with few purchases, and reduce N by dis- carding very popular or unpopular items. It is also possible to reduce the number of items examined by a small, constant factor by partitioning the item space based on product category or subject classi- fication. Dimensionality reduction techniques such as clustering and principal component analysis can reduce M or N by a large factor.5

clusters; as we now describe, such clustering can also degrade recommendation quality.

Cluster Models

To find customers who are similar to the user, clus- ter models divide the customer base into many seg- ments and treat the task as a classification problem. The algorithm’s goal is to assign the user to the seg- ment containing the most similar customers. It then uses the purchases and ratings of the customers in the segment to generate recommendations.

The segments typically are created using a clus- tering or other unsupervised learning algorithm, although some applications use manually deter- mined segments. Using a similarity metric, a clus- tering algorithm groups the most similar customers together to form clusters or segments. Because optimal clustering over large data sets is imprac- tical, most applications use various forms of greedy cluster generation. These algorithms typi- cally start with an initial set of segments, which often contain one randomly selected customer each. They then repeatedly match customers to the existing segments, usually with some provision for creating new or merging existing segments.6 For very large data sets — especially those with high dimensionality — sampling or dimensionality reduction is also necessary.

Once the algorithm generates the segments, it computes the user’s similarity to vectors that sum- marize each segment, then chooses the segment with the strongest similarity and classifies the user accordingly. Some algorithms classify users into multiple segments and describe the strength of each relationship.7

Cluster models have better online scalability and performance than collaborative filtering3 because they compare the user to a controlled number of segments rather than the entire cus-

Industry Report

Figure 1. The “Your Recommendations” feature on the Amazon.com homepage. Using this feature, customers can sort recommendations and add their own product ratings.

Figure 2. Amazon.com shopping cart recommendations. The recom- mendations are based on the items in the customer’s cart: The Pragmatic Programmer and Physics for Game Developers.

tomer base. The complex and expensive clustering computation is run offline. However, recommen- dation quality is low.1 Cluster models group numerous customers together in a segment, match a user to a segment, and then consider all cus- tomers in the segment similar customers for the purpose of making recommendations. Because the similar customers that the cluster models find are not the most similar customers, the recommenda- tions they produce are less relevant. It is possible to improve quality by using numerous fine- grained segments, but then online user–segment classification becomes almost as expensive as find- ing similar customers using collaborative filtering.

Search-Based Methods

Search- or content-based methods treat the rec- ommendations problem as a search for related items.8 Given the user’s purchased and rated items, the algorithm constructs a search query to find other popular items by the same author, artist, or director, or with similar keywords or subjects. If a customer buys the Godfather DVD Collection, for example, the system might recom- mend other crime drama titles, other titles star- ring Marlon Brando, or other movies directed by Francis Ford Coppola.

If the user has few purchases or ratings, search- based recommendation algorithms scale and perform well. For users with thousands of purchases, however, it’s impractical to base a query on all the items. The algorithm must use a subset or summa- ry of the data, reducing quality. In all cases, rec- ommendation quality is relatively poor. The rec- ommendations are often either too general (such as best-selling drama DVD titles) or too narrow (such as all books by the same author). Recom- mendations should help a customer find and dis- cover new, relevant, and interesting items. Popu- lar items by the same author or in the same subject category fail to achieve this goal.

Item-to-Item Collaborative Filtering

Amazon.com uses recommendations as a targeted marketing tool in many email campaigns and on most of its Web sites’ pages, including the high- traffic Amazon.com homepage. Clicking on the “Your Recommendations” link leads customers to an area where they can filter their recommendations by product line and subject area, rate the recommended products, rate their previous purchases, and see why items are recommended (see Figure 1).

As Figure 2 shows, our shopping cart recom- mendations, which offer customers product sug- gestions based on the items in their shopping cart. The feature is similar to the impulse items in a supermarket checkout line, but our impulse items are targeted to each customer.

Amazon.com extensively uses recommendation algorithms to personalize its Web site to each cus- tomer’s interests. Because existing recommendation algorithms cannot scale to Amazon.com’s tens of millions of customers and products, we developed our own. Our algorithm, item-to-item collaborative filtering, scales to massive data sets and produces high-quality recommendations in real time.中文译文

亚马逊基于商品到商品的协同过滤算法

推荐算法是被电子商务网站广泛使用的，网站使用消费者的兴趣作为输入生成一个商品推荐清单。大量的网站仅仅使用用户购买过的商品去推断消费者目前的兴趣，但是他们也可以使用其他的属性，包括被浏览过的项目，人工评分，主题兴趣和他们最喜欢的艺术家。

在亚马逊公司，我们使用algo-rithms推荐算法为网上商城的每个消费者进行个性化推荐。网上商城完成了质的改变基于消费者的兴趣，展现了程序对于软件工程师就好像小孩的玩具对于母亲一样。点击量和转发率——基于网络和电子邮件广告效果的两项重要措施——大大超过了横幅广告和畅销书排行榜等无针对性内容。

电子商务推荐算法通常在具有挑战性的环境中运行。例如：

•大型零售商可能有大量的数据、数千万的客户和数百万个不同的目录项。

•许多应用程序要求在不超过半秒的时间内重新返回结果集，同时仍然生成高质量的推荐结果。

•新用户通常有极其有限的信息，仅基于少量购买和商品评分。

•根据成千上万的购买和评级，资历较老的用户可能会有过多的信息。

•客户数据是变化极快的:每个交互都支持有价值的客户数据，而且算法必须立即响应新信息。

有三种常见的方法解决推荐系统问题:传统的协同过滤、聚类算法和基于搜索的方法。在这里，我们将这些方法与我们的算法进行比较，我们称之为商品对商品的协同过滤。与传统的协同过滤不同，我们的算法在线计算独立于产品目录中的客户数量和项目数量。我们的算法在实时、规模和海量数据集上生成建议，并生成高质量的推荐。

推荐系统

大多数推荐算法首先要找到一组曾经购买的商品和评分过的商品和目标用户相似的用户。算法从这些相似的客户中收集项目，消除用户已经购买或评估的项目，并向用户推荐剩余的项目。这些算法的两个流行版本是协同过滤和聚类模型。其他的算法——包括基于搜索的方法和我们自己的商品到商品的协同过滤——专注于寻找类似的项目，而不是类似的客户。对于每个用户购买的和评分的商品，该算法试图找到类似的项目。然后将相似的商品聚集起来并推荐它们。

传统的协同过滤算法

传统的协同过滤算法将客户视为一个N维向量，其中N是不同的目录项的数量。矢量的组成部分对购买或积极评价的物品有积极的作用，对负面评价的物品有负面影响。为了弥补Amazon.com的推荐算法，最畅销的商品，这种算法通常是由相反的fre - quency(即购买或评价该项目的客户数量的倒数)组成的矢量组件，使得不那么知名的项目变得更加相关。对于几乎所有的客户，这个向量是非常稀疏的。

该算法根据一些与用户最相似的客户来生成建议。它可以用不同的方法来测量两个角的相似性;一个常用的方法是测量两个向量夹角的余弦值:



该算法还可以从相似的用户的物品中选择推荐，也可以使用不同的方法，一种常用的技术是根据购买的相似顾客的数量来对每个商品进行排名。

使用协同过滤来生成推荐列表在计算上是非常昂贵的。在最坏的情况下是O(MN)，其中M是客户的数量，N是产品目录的数量，因为它检查了M的客户和每个客户的N项。然而，由于平均客户矢量非常稀疏，算法的性能趋向于接近O(M + N)，扫描每一个客户都是接近(M)，而不是O(MN)，因为几乎所有的用户向量都只包含少量的项，而不考虑目录的大小。但是有一些客户已经购买或评估了很大比例的目录，需要O(N)处理时间。因此，算法的最终性能大约是O(M + N)，即使如此，对于非常大的数据集——例如1000万或更多的客户和100万或更多的目录项——算法遇到了严重的性能和缩放问题。

通过减少数据大小，可以部分解决这些问题。我们可以通过随机抽样消费者或丢弃一些消费者的购买数据，并减少N通过去掉非常受欢迎的或不受欢迎的项目。还可以通过根据产品类别或主题的划分来划分项目空间，从而减少由一个小的常数因子所检查的项数。量纲化还原技术如聚类和主成分分析可以通过一个大因子来减少M或N

不幸的是，所有这些方法也在几个方面降低了推荐质量。首先，如果算法只检查一个小的客户样本，那么所选的客户将与用户不太相似。其次，项目空间分区限制对特定产品或主题区域的推荐。第三，如果该算法丢弃最受欢迎的或不受欢迎的项目，它们将永远不会出现作为建议，而只购买那些项目的客户将不会得推荐列表。量纲化还原技术应用于项目空间可以通过消除低频项来产生同样的效果。就如我们所知，这样的聚类也会降低推荐列表的质量。

聚类模型

为了找到与用户相似的用户，聚类模型将客户基础划分为多个分区，并将任务视为分类问题。该算法的目标是将用户分配给包含最相似的用户的分区。然后它使用客户的购买和评级来生成推荐。

这些分区通常是使用聚类算法或其他非监督学习算法创建的，尽管一些应用程序使用人工监督的部分。使用相似性度量，一个聚类算法将最相似的客户组合在一起形成集群或片段。由于大型数据集上的最佳聚类是一种非系统的，大多数应用程序使用各种形式的贪婪集群。这些算法都是由一组初始的片段开始的，通常包含一个随机选择的客户。然后，他们不断地将客户与现有的部分匹配，通常是创建新的或合并现有的部分。对于非常大的数据集——特别是那些具有高维度的数据集——采样或维数减少也是必要的。

一旦该算法生成了片段，它计算用户与向量的相似性，并对每个部分进行求和，然后选择具有最强烈相似性的片段，并相应地对用户进行分类。一些算法将用户分为多个部分，并描述每个关系的强度

聚类模型具有更好的在线可伸缩性和性能协同过滤相比，因为它们将用户与受控制的部分相比，而不是整个用户群。复杂而昂贵的集群计算是离线运行的。然而，推荐的质量很低。聚类模型将众多客户聚集在一起，将用户与一个片段匹配，然后考虑所有的用户在这段时间内的类似客户，目的是提出建议。因为聚类模型发现的类似客户不是最相似的客户，他们所生产的推荐产品就不那么重要了。通过使用大量细粒度的片段来提高质量是可能的，但是在线用户段分类的成本几乎和使用协作过滤的相似客户一样昂贵。

基于搜索的方法

搜索或基于内容的方法将推荐问题视为对相关项的搜索。根据用户的购买和评价项目，该算法构造一个搜索查询，以找到其他受欢迎的物品，由同一作者、艺术家、导演，或者有类似的关键字或主题。例如，如果一个客户购买教父DVD的收藏，系统可能会修复其他犯罪剧的标题，其他的标题明星——环马龙·白兰度，或者由弗朗西斯·福特·科波拉导演的其他电影。

如果用户没有购买或评级，基于搜索的推荐算法的规模和每一个形式。然而，对于有上千种购买行为的用户来说，对所有项进行查询是不切实际的。该算法必须使用数据的子集或一部分，从而降低质量。在大部分情况下，推荐质量相对较差。通常情况下(例如畅销的戏剧DVD)或太窄(例如同一作者的所有书籍)。推荐系统应该帮助客户发现和-包括新的、相关的和有趣的项目。同一作者或同一主题类别中流行的项目未能实现这个目标。

商品到商品的协同过滤

亚马逊网站在很多电子邮件活动和大多数网站的网页上都使用推荐作为营销工具，包括高流量的亚马逊网站主页。点击“你的推荐”链接，客户就会进入一个区域，在那里他们可以通过产品线和主题区域过滤他们的建议，对推荐的产品进行评估，对他们之前的购买进行评估，并了解为什么推荐产品(见图1)。

如图2所示，我们的购物车recom - mendations，它为客户提供基于购物车中的商品的产品sug。该特性类似于超市收银台的脉冲项，但我们的脉冲项是针对每个客户的。

亚马逊公司广泛使用推荐算法将其网站的个性化设置为每个人的利益。由于现有的推荐算法无法扩展到亚马逊的数千万客户和产品，我们开发了自己的产品。我们的算法，逐项的协同过滤，扩展到大量的数据集，并实时产生高质量的建议。