

架构师

ARCHITECT

| 特刊 |

推荐系统 [理论篇]



SPECIAL ISSUE

May, 2016

架构师特刊



InfoQ^{new}

作者简介

Maya Hristakeva 是 Mendeley 和 RELX 团队的首席数据科学家，致力于创建能够帮助研究者连接他们的研究和合作者的工具。她本人的研究领域是可扩展的机器学习、推荐系统和优化算法。她也对端对端的构建数据产品过程感兴趣：从算法到好的用户体验。

Kris Jack 是 Mendeley 的首席数据科学家，同时也是 RELX 团队的数据科学家的负责人。他热衷于创造能够帮助人们理解和传达复杂信息以及做出新发现的软件。他的主要研究兴趣在于推荐系统、信息检索、信息抽取、机器学习、人工智能等等。他还热衷于将技术转化为真正对用户有用的产品以及参与能够创造奇迹的团队。

陈运文，博士，达观数据 CEO；中国知名大数据技术专家，国际计算机学会（ACM）会员，中国计算机学会（CCF）高级会员，复旦大学计算机博士和杰出毕业生；在国际顶级学术期刊和会议上发表多篇 SCI 论文，多次参加 ACM 国际数据挖掘竞赛并获得冠军荣誉；曾担任盛大文学首席数据官（CDO），腾讯文学高级总监、数据中心负责人，百度核心技术研发工程师，在大数据挖掘、用户个性化建模、文本信息处理、推荐和搜索技术等方面有丰富的研发和管理经验。

目录

第 1 章 推荐算法简介

第 2 章 协同过滤推荐算法

第 3 章 基于内容的过滤算法

第 4 章 混合推荐算法

第 5 章 如何选择推荐算法

第 6 章 推荐系统和搜索引擎的关系



序言

在 Web 1.0 时代，导航网站是人们获取信息的主要方式，人们从固定的目录式资源列表中寻找自己感兴趣的东西；Web2.0 时代，搜索引擎让人们可以方便的获取整个互联网的资源，让互联网真正无缝互联。但是当互联网的资源量已经爆炸到人们无法分辨和接收时，推荐系统则从更深的层次发现用户需求，给大家带来了个性化的体验。

推荐系统作为支撑个性化的主要技术方案，已经在互联网上得到了广泛的应用：从 1998 年亚马逊利用 Item-base 协同过滤算法开始为用户进行产品推荐开始，如今我们已可以在传统电商、视频、论坛、新闻、广告、邮件、社交、阅读等领域看到推荐系统的应用；在移动端，由于用户界面的缩小，推荐系统的作用更显得突出，在应用市场、游戏市场、桌面、推送等场景，个性化推荐系统也发挥着重要的作用。

推荐算法是推荐系统的核心，了解每个推荐算法的原理和特点有助于在具体场景中算法的选择，无论是基于用户行为协同过滤算法，抑或是基于内容的推荐算法，还是基于流行度等其他算法，都各有优缺点，即使使用深度学习、社会化推荐等高级模型，现实场景中也很难用一个算法解决问题，多算法的组合是一种行之有效的方法，如何选择最合适的算法也是一个“技术活”。

推荐算法很重要，但推荐算法只是推荐系统的其中一部分，推荐系统原理看似简单，但要完整支撑一个应用场景却是一个系统工程。在一个优秀的推荐系统中，数据占 70%，产品占 30%，算法占 10%，这是一个比较广泛的共识，任何一方的短缺都无法取得较好的推荐效果。原理和实验都很难说明一个算法或推荐系统好，真正验证效果还得实践。

在实际应用中，搜索和推荐是目前人们最常用的获取资源的方式，前者体现用户最直接的需求，后者发现用户的隐性需求，两者互相配合，解决用户不同层面的个性化需求。现实的搜索引擎中大量的融合了推荐系统的思路和算法解决个性化排序的问题，推荐系统中也会借助搜索引擎的思路来达到更精确的召回效果。

不管什么技术，我们的目标是解决个性化的问题，洞察用户需求，降低用户获取优质资源的成本，提升用户体验，让用户在方方面面都体验到个性化的服务。

InfoQ 此次汇集各方智慧形成的推荐系统的电子书，可以让大家对推荐系统有一个更全面的认识，内容简洁清晰，其中每一点都是经验之谈，可以作为推荐爱好者的入门读物，也可以作为推荐系统开发者的方向指导。

雷银

百分点集团开发副总监

第 1 章 推荐算法简介

为推荐系统选择正确的推荐算法是非常重要的决定。目前为止，已经有许多推荐算法可供选择，但为你需要解决的特定问题选择一种特定的算法仍然很困难。每一种推荐算法都有其优点和缺点，当然也有其限制条件，在作出决定之前，你必须一一考量。在实践中，你可能会测试几种算法，以发现哪一种最适合你的用户，测试中你也会直观地发现它们是什么以及它们的工作原理。

推荐系统算法通常是某类推荐模型的实现，它负责获取数据，例如用户的喜好和可推荐项的描述，以及预测给定的用户组会对哪些选项感兴趣。

推荐算法通常被分为四大类（1-4）：

- 协同过滤推荐算法；
- 基于内容的推荐算法；
- 混合推荐算法；
- 流行度推荐算法。

除了这些种类以外，还有一些高级非传统的推荐算法（5）。

推荐算法综述是分为五个部分的系列文章，本文作为第一章，将会简要介绍推荐系统算法的主要种类。其中包括算法的简要描述、典型的输入、

不同的细分类型以及其优点和缺点。在第二章和第三章中，我们将会详细介绍这些算法的区别，让你能够深入理解他们的工作原理。系列文章中的一些内容参考了一篇来自 RecSys 2014 tutorial 的文章：由 Xavier Amatriain 编写的 The Recommender Problem Revisited。

协同过滤推荐算法

- 简介：通过在用户的一系列行为中寻找特定模式来产生用户特殊推荐。
- 输入：仅仅依赖于惯用数据（例如评价、购买、下载等用户偏好行为）。
- 类型：
 - 基于邻域的协同过滤（基于用户和基于项）；
 - 基于模型的协同过滤（矩阵因子分解、受限玻尔兹曼机、贝叶斯网络等）。
- 优点：
 - 需要最小域；
 - 不需要用户和项；
 - 大部分场景中能够产生足够好的结果。
- 缺点：
 - 冷启动问题；
 - 需要标准化产品；
 - 需要很高的用户和项的比例（1：10）；
 - 流行度偏见（有长尾的时候表现不够好）；
 - 难于提供解释。

基于内容的推荐算法

- 简介：向用户推荐和其过去喜欢项的内容（例如元数据、描述、

话题等等)相似的项。

- 输入: 仅仅依赖于项和用户的内容/描述(除了惯用数据)。
- 类型:
 - 信息检索(例如 tf-idf 和 Okapi BM25);
 - 机器学习(例如朴素贝叶斯、支持向量机、决策树等)。
- 优点:
 - 没有冷启动问题;
 - 不需要惯用数据;
 - 没有流行度偏见,可以推荐有罕见特性的项;
 - 可以使用用户内容特性来提供解释。
- 缺点:
 - 项内容必须是机器可读的和有意义的;
 - 容易归档用户;
 - 很难有意外,缺少多样性;
 - 很难联合多个项的特性。

混合推荐算法

- 简介: 综合利用协同过滤推荐算法和基于内容的推荐算法各自的优点同时抵消各自的缺点。
- 输入: 同时使用用户和项的内容特性与惯用数据,同时从两种输入类型中获益。
- 类型:
 - 加权;
 - 交换;
 - 混合;
 - 特性组合。

- 案列
- 特征增强
- 元层次
- 优点：
 - 由于单独使用协同过滤推荐算法和基于内容的推荐算法；
 - 没有冷启动问题；
 - 没有流行度偏见，可推荐有罕见特性的项；
 - 可产生意外，实现多样性。
- 缺点：
 - 需要通过大量的工作才能得到正确的平衡。

流行度推荐算法

- 简介：这是一种推荐流行项的方法（例如最多下载、最多看过、最大影响的项）。
- 输入：使用惯用数据和项的内容（例如类目）。
- 优点：
 - 相对容易实现；
 - 良好的基准算法；
 - 有助于解决新用户冷启动问题。
- 缺点：
 - 需要标准化产品；
 - 经常需要一些项的类型进行分类；
 - 不会推荐新项（很少有机会被观测到）；
 - 推荐列表不会改变太大。

高级非传统推荐算法

- 类型：

- 深度学习;
- 学习等级;
- Multi-armed bandits (探索/开发);
- 上下文感知推荐;
- 张量分解;
- 分解机;
- 社会推荐。
- 优点:
 - 利于勉强维持最终性能百分点;
 - 你可以说你正在使用渐进的方式。
- 缺点:
 - 难于理解;
 - 缺乏推荐工具支持;
 - 没有为你的首个推荐系统提供推荐的方式。

延伸阅读

“如何将大数据建模在商业领域玩转得风声水起”



第 2 章 协同过滤推荐算法

协同过滤（CF）推荐算法通过在用户活动中寻找特定模式来为用户产生有效推荐。它依赖于系统中用户的惯用数据，例如通过用户对其阅读过书籍的评价可以推断出用户的阅读偏好。这种算法的核心思想就是：如果两个用户对于一些项的评分相似程度较高，那么一个用户对于一个新项的评分很有可能类似于另一个用户。值得注意的是，他们推荐的时候不依赖于项的任何附加信息（例如描述、元数据等等）或者用户的任何附加信息（例如喜好、人口统计相关数据等等）。CF 的方法大体可分为两类：分别为邻域和基于模型的方法。邻域方法（即基于内存的 CF）是使用用户对已有项的评分直接预测该用户对新项的评分。与之相反，基于模型的方法是使用历史评分数据，基于学习出的预测模型，预测对新项的评分。通常的方式是使用机器学习算法，找出用户与项的相互作用模型，从而找出数据中的特定模式。

基于邻域的 CF 方法意在找出项与项之间的联系（基于项的 CF），或者用户与用户之间的联系（基于用户的 CF）。

基于用户的 CF 通过找出对项的偏好与你相似的用户从而基于他们对于新项的喜好来为你推荐。

						
	4	3			5	
	5		4		4	
	4		5	3	4	
		3				5
		4				4
			2	4		5

图1：用户对图书的评分。所有的评分范围从1到5，5代表喜欢程度最高。第一个用户（行1）对第一个图书（列1）的评分是4。空的单元格代表用户未给图书评价

基于项的 CF 会向用户推荐与用户喜欢的项相似的项，这种相似是基于项的共同出现几率（例如用户买了 X，通时也买了 Y）。

首先，在做基于项的协同过滤之前，我们先通过例子来看一下基于用户的协同过滤。

假设我们有一组用户，他们表现出了对一组图书的喜好。用户对一本图书的喜好程度越高，就会给其更高的评分，范围是从 1 到 5。我们来通过一个矩阵来展示它，行代表用户，列代表图书（图 1）。

使用基于用户的协同过滤方法，我们首先要做的是基于用户给图书作出的评价计算出用户之间的相似度。让我们从一个单一用户的角度考虑这个问题，看图 1 中的第一行。要做到这一点，常见的做法是将使用包含了用户喜好项的向量（或数组）代表每一个用户。相较于使用多样化的相似度量这种做法非常直接。在这个例子中，我们将使用余弦相似性。当我们把第一个用户和其他五个用户进行比较时，就能直观地看到他和其他用户的相似程度（图 2）。对于大多数相似度量，向量之间相似度越高，代表彼此更相似。本例中，第一个用户与两位其他用户非常相似，有两本共同

书籍，与另两位用户的相似度低一些，只有一本共同书籍，而与最后一名用户完全不相似，没有一本共同书籍。

更一般地，我们可以计算出每个用户的相似性，并且在相似矩阵中表示它们（图 3）。这是一个对称矩阵，这意味着对它进行数学计算会有一些有用的特性。单元格的背景颜色表明用户相似度的高低，更深的红色表示他们之间更相似。

现在我们使用基于用户的协同过滤方法准备好了为用户生成推荐。在

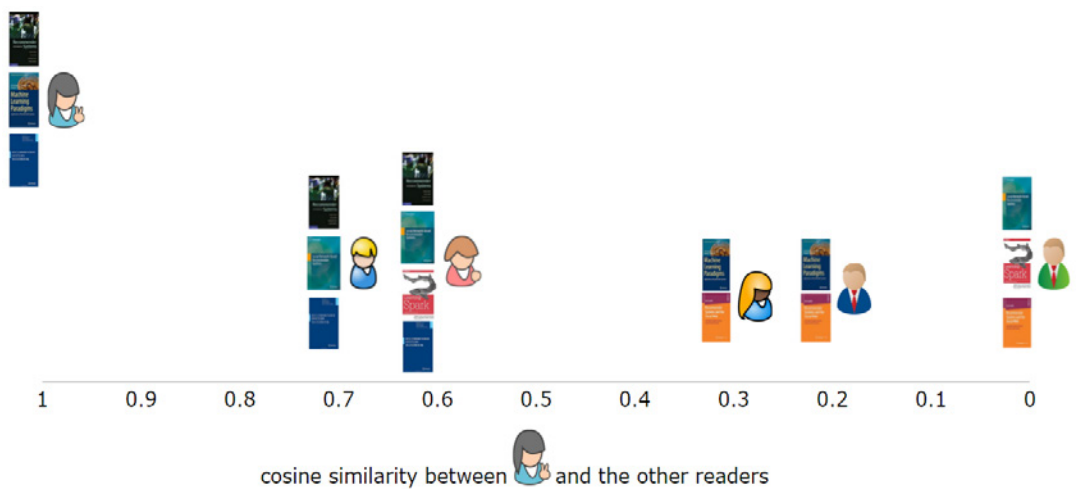


图2：第一个用于与其他用户的相似度。可以使用余弦相似性在一个单一维度绘制用户之间的相似度

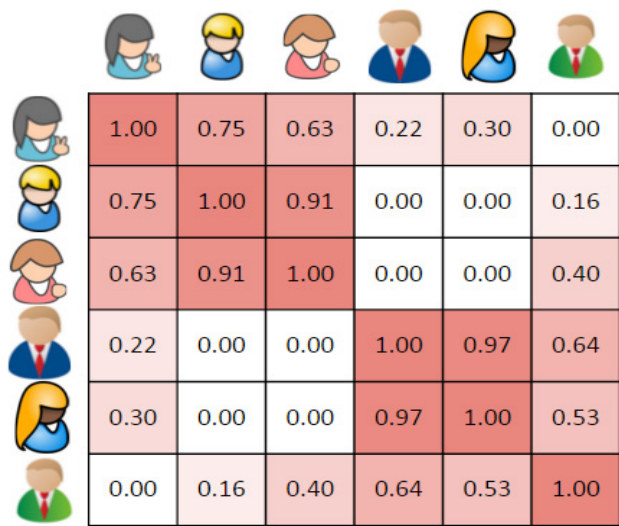


图3：用户间相似矩阵。用户之间的相似度基于用户对所读图书的评价数据的相似度

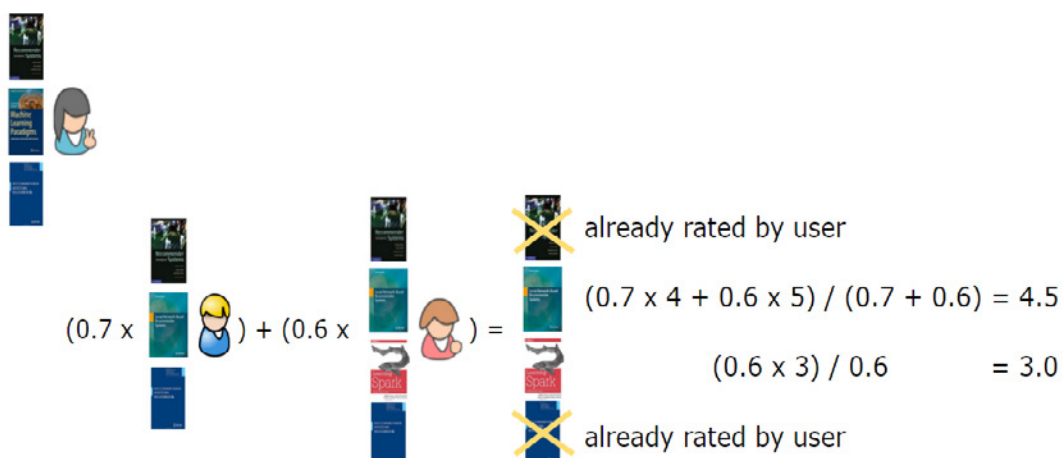


图4： 为一个用户进行推荐。我们将选取最相似的两个用户所评价的书，进行加权，然后推荐加权分数最高且目标用户未评价过的图书

一般情况下，对于一个给定的用户，这意味着找到最相似的用户，并推荐这些类似的用户欣赏的项，并根据用户相似度对其进行加权。我们先来看第一个用户，为他们生成一些推荐。首先，我们找到了与第一个用户最相似的另一用户，删除用户已经评价过的书籍，给最相似用户正在阅读的书籍加权，然后计算出总和。在这种情况下，我们计算出 $n=2$ ，表示为了产生推荐，需要找出与目标用户最相似的两个用户。这两个用户分别是 2 和 3（图 4）。然后，第一个用户已经评价了 5 和 1，所产生的推荐书是 3（4.5 分）和 4（3 分）。

通过基于用户的协同过滤让我们对协同过滤有了一定的理解。接着让我们再看一个例子，基于项的协同过滤。同样地，我们以评价过一些书籍的一组用户为基础（图 1）。

类似于基于用户的协同过滤，在基于项的协同过滤中，我们要做的第一件事也是计算相似矩阵。然而，这一次，我们要看的是项，而非用户的相似性。类似地，我们要计算出一本书和其它书的相似性，我们将使用评价过一本书的用户向量（或数组）表示这本图书，并比较他们的余弦相似性函数。在第一列的第一本书，最类似于第五列的第五本书，因为评价他

们的用户大致相同（图 5）。第三本书有两个相同的评价用户，第四和第二本书只有一个共同评价用户，而最后一本书是不认为是相似的，因为同它有没有共同的评价用户。为了更充分地显示结果，我们可以显示表示所有图书之间相似度的相似矩阵（图 6）。同样地，单元格背景颜色的深浅表示相似度的高低，颜色越深表明相似度越高。

现在，我们已经知道了图书之间的相似度，我们可以为用户进行推荐。在基于项的方法中，我们采用被用户评价过的项，推荐和被采取项最相似的项。在我们的例子中，第一个用户首先将被推荐第三本书，其次是第六

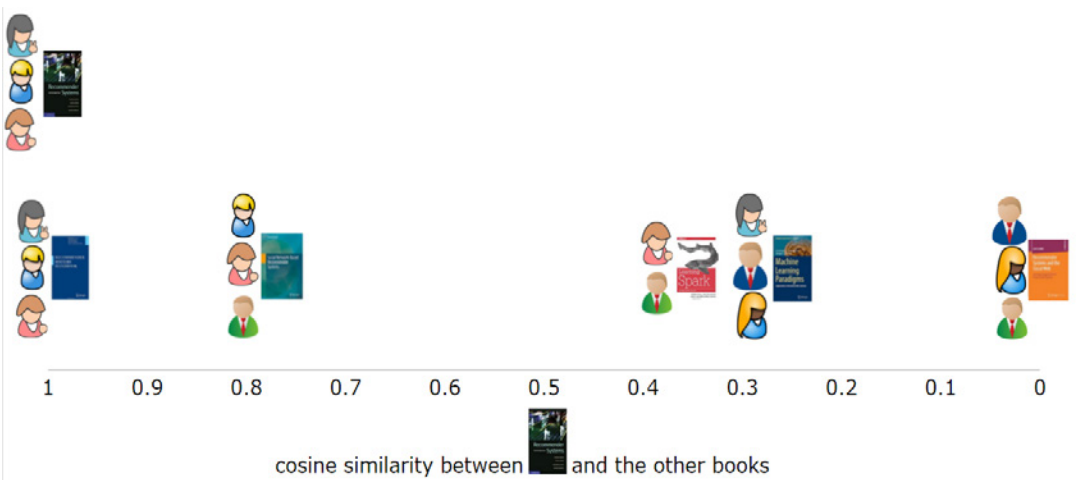


图5：第一个图书和其它图书的对比。图书用评价过它们的用户表示。使用相似值（0-1）表示相似度。两本书越相似则相似值越高

	1.00	0.27	0.79	0.32	0.98	0.00
	0.27	1.00	0.00	0.00	0.34	0.65
	0.79	0.00	1.00	0.69	0.71	0.18
	0.32	0.00	0.69	1.00	0.32	0.49
	0.98	0.34	0.71	0.32	1.00	0.00
	0.00	0.65	0.18	0.49	0.00	1.00

图6：书的相似矩阵

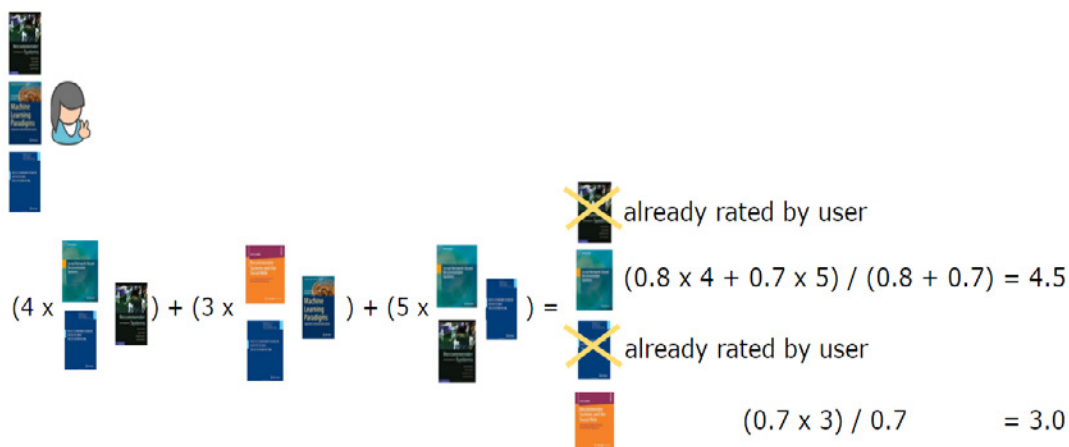


图7：为用户进行推荐。我们选取他们评价过的图书，找出与他们最相似的前两本书，进行加权，然后推荐给用户加权分最高且他没有读过的书

本书（图7）。另外，我们只推荐和用户已经评价过图书最相似的前两本书。

鉴于基于用户和基于项的协同过滤的描述听起来非常相似，有趣的是它们可以产生不同的推荐结果。即使在我们这里给出的简易的例子中，即使使用的数据相同，这两种方法产生对于同一用户产生的推荐结果也不相同。当你构建推荐系统的时候，这两种协同过滤方式都是值得考虑的。即使将这两种方式描述给非专家听，它们听起来也非常相似，在实践中，他们可以产生不同的结果，为用户提供了不同的体验。

邻域方法由于其简单性和效率具有相当的知名度，同时也是由于它们有产生准确的和个性化的推荐的能力。然而，它们也有一些可扩展性的限制，因为在用户数量和项的数量增长的情况下，它们需要一个相似度的计算（基于用户或项）。在最坏的情况下，这种计算的时间复杂度可能是 $O(m \times n)$ ，但在实践中的情况稍微好一点 $O(m+n)$ ，部分原因是由于利用了数据的稀疏度。虽然稀疏有助于可扩展性，它也对基于邻域的方法提出了一个挑战，因为我们的用户仅仅对庞大数量项中的很少一部分进行了评分。例如，在 Mendeley，我们有数以百万计的文章而一个用户可能只读了其中几百篇文章。两个读过 100 篇文章的用户有一篇相同文章的概率（共

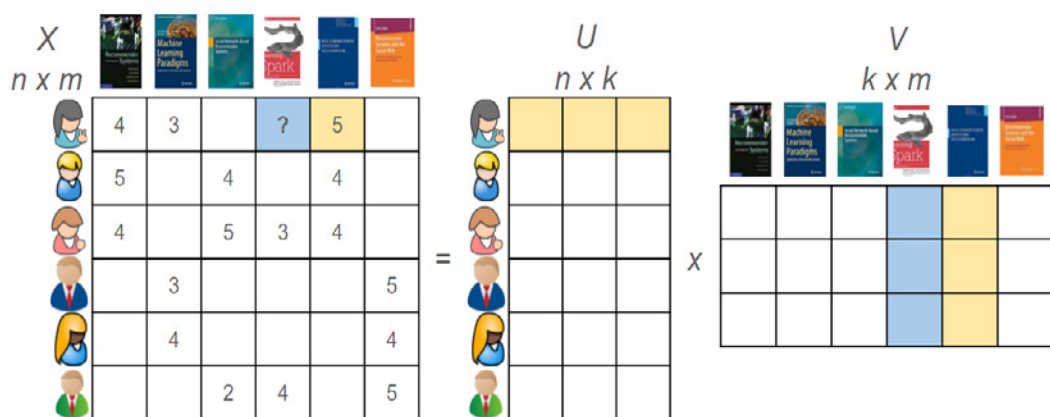


图8：矩阵分解表示。用户偏好矩阵可以被分解成一个用户主题矩阵乘以一个主题项矩阵

5000 万篇文章）是 0.0002。

基于模型的方法可以帮助克服一些基于邻域的方法的局限性。它不像基于邻域的方法，使用用户项评分直接预测新的项。基于模型的方法会在使用评分去学习预测模型的基础上，去预测新项。一般的想法是使用机器学习算法建立用户和项的相互作用模型，从而找出数据中的模式。在一般情况下，基于模型的 CF 被认为是建立 CF 推荐系统的更先进的算法。有许多不同的算法可用于构建模型并基于这些模型进行预测，例如，贝叶斯网络、聚类、分类、回归、矩阵分解、受限玻尔兹曼机等等。这些技术在为了最终赢得 Netflix 奖的解决方案中扮演了关键角色。Netflix 发起了一个竞赛，从 2006 年到 2009 年提供一百万美元奖金，颁发给产生的推荐比他们自己的推荐系统精确 10% 以上的推荐系统团队。成功获奖的解决方案是 Netflix 研发的一个集成（即混合）了超过 100 种算法模型，这些算法模型都采用了**矩阵分解和受限玻尔兹曼机**。

矩阵因子分解（如奇异值分解，奇异值分解 + +）将项和用户都转化成了相同的潜在空间，它所代表了用户和项之间的潜相互作用（图 8）。矩阵分解背后的原理是潜在特征代表了用户如何给项进行评分。给定用户和项的潜在描述，我们可以预测用户将会给还未评价的项多少评分。

在表 1 中，我们概述了基于邻域和基于模型的协同过滤方法的主要优点和缺点。由于它们仅依赖于用户的惯用数据，协同过滤方法需要最低限度专业工程的努力，以产生足够好的结果，但是，他们也有局限性。例如，CF 倾向于推荐流行的项，很难推荐给有独特口味的人（即感兴趣的项并没有产生足够多的惯用数据）。这被称为流行性偏见，它通常是用基于内容的过滤方法。CF 方法的一个更重要的限制是我们所称的“冷启动问题”，系统是不能够给没有（或非常少）惯用活动的用户进行推荐，又名曰新用户问题，或推荐新项问题。新用户的“冷启动问题”可以通过流行性和混合方法进行解决，而新项问题可以通过使用基于内容的过滤或 multi-armed bandits（即探索利用）进行解决。我们将在下一篇文章中讨论上述方法中的一些方法。

在这篇文章中，我们讨论了三种基本的协同过滤的实现方法。基于项、基于用户和基于矩阵分解之间的差异是很微妙的，很难简单明了地解释它们。了解它们之间的差异将有助于你选择最适合你的推荐算法。在接下来的文章中，我们会继续更加深入地介绍一些流行的推荐算法。

第 3 章 基于内容的过滤算法

本文是推荐算法综述的第 3 章。第 1 章主要介绍了推荐算法的主要类型。第 2 章主要涵盖了不同类型的协同过滤算法，突出他们之间的一些细微差别。本章主要介绍基于内容的过滤算法的工作原理，以及它的优点和缺点，从而让读者对其有更深入的理解。

基于内容的推荐算法总是为用户推荐那些与用户过去喜欢的 item 类似的 item。它不同于协同过滤，它是基于 item 的内容（例如标题、年份、描述）比较 item 之间的相似度。并没有考虑用户过去如何使用 item 的情况。例如，如果一个用户喜欢电影“指环王：魔戒再现”和“指环王：双塔奇兵”，然后使用电影的标题信息，推荐系统可以向用户推荐电影“指环王：王者无敌”。在基于内容的推荐中，假设可以获取到 item 的描述信息，并将其作为 item 的特征向量（例如标题、年份、描述）。这些特征向量被用于创建一个反映用户偏好的模型。各种信息检索（例如 TF-IDF）和机器学习技术（例如朴素贝叶斯、支持向量机、决策树等）可被用于创建用户模型，从而为用户产生推荐。

举个例子，假设有一些用户表达过对于一系列书籍的偏好。他们越喜欢一本书，他们对书籍的评分就会越高，通常划分为从 1 到 5 的 5 个等级。





						
	4	3			5	
	5		4		4	
	4		5	3	4	
		3				5
		4				4
			2	4		5

图1：用户对书籍的偏好。所有的偏好都分为5个等级，5表示最喜欢的。第一个用户（行1）对于第一本书的偏好给出了一个4分的评分。如果一个单元格是空的，表示用户对于该书籍的偏好没有给出







	Introduction to Recommender Systems
	Machine learning Paradigms
	Social Network-based Recommender Systems
	Learning Spark
	Recommender Systems Handbook
	Recommender Systems and the Social Web

图2：用户已经评分过的书籍的标题

可以将用户对于书籍的偏好表示为一个矩阵，其中行代表用，列表示书籍，如图 1 所示。

在基于内容的推荐中，我们想要做的第一件事是基于内容计算书籍之间的相似度。在这个例子中仅仅使用了书籍标题中的词汇，这是为了将例子进行简化，以方便理解基于内容的推荐算法的工作原理，如图 2 所示。在实际应用中，可以使用更多的属性。

首先，从内容中删除停止词（例如语法词语、常见的词语）是非常普遍的，然后将书籍用一个向量表示（或数组），代表使用了哪些词，这被称为矢量空间表示，如下图 3 所示。

给定每本书的表示之后，使用一系列的相似度度量来对书籍进行比较就变得非常简单了。在这个例子中，我们选择了余弦相似性度量。当我们把第一本书与五本其他的书进行比较时，就能得到这本书与其他书之间的相似程度，如下图 4 所示。如同大多数相似度度量一样，向量之间的相似性度量值越高，表明两个对象之间越相似。在这种情况下，第一本书与其中三本书非常类似，因为它们的表示之间有两个词汇相同（recommender 和 systems），但是其中一本书的描述的词汇最少，它与第一本书最相似，因为它有最少的多余的词汇。而与剩下两本书之间因为没有共同的描述词汇，因此可以当做一点都不相似。



图3：使用书籍标题中词汇作为描述书籍的向量表示。当相应的词汇在标题中，对应的单元格中标注1，否则为空白

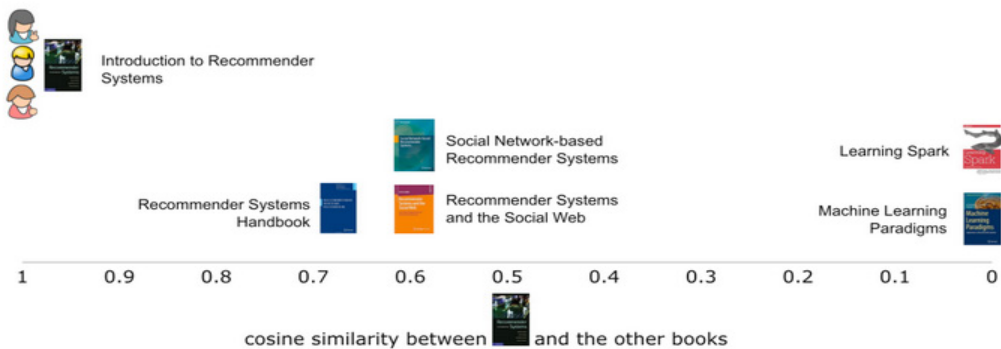


图4：使用书籍标题中词汇作为描述书籍的向量表示。当相应的词汇在标题中，对应的单元格中标注1，否则为空白



图5：书籍之间的相似度矩阵。每个相似度是基于书籍的词汇向量表示用余弦相似性度量进行计算的

更进一步，就可以在一个相似度矩阵中显示所有书籍之间的相似程度，如图 5 所示。单元格的背景颜色表示两本书之间的相似程度，红颜色越深，它们之间越相似。

在知道书籍之间有多相似之后，我们就可以对用户推荐书籍了。类似于我们在第二部分所介绍的基于 item 的协同过滤方法，我们选取一个用户此前评分过的书籍，并推荐与它们最相似的书籍。与协同过滤方法不同的是，这里的相似性度量是基于书籍的内容，在这个例子中，准确来说是标题，而不是使用用户过去的行为数据。在我们的例子中，第一个用户将会被推荐第四本书，之后是第六本书，如下图 6 所示。再次，我们仅仅选取了最相似的两本书。

基于内容的方法克服了协同过滤方法的很多不足。具体来说，基于内容的推荐算法可以克服流行度偏离和新 item 的冷启动问题，这些问题在第二部分介绍协同过滤的时候已经讨论过。然而，值得注意的是，纯粹基于内容的推荐算法的性能通常不如协同过滤算法。基于内容的推荐算法通常还存在过度专业化（over-specialisation）的问题，即用户可能会得

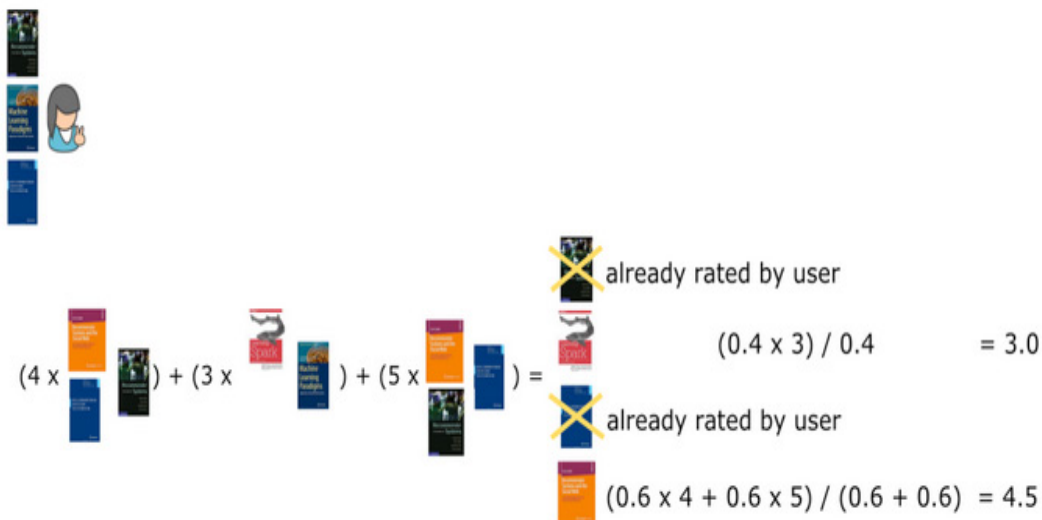


图6：针对一个具体的用户产生的推荐。我们选择用户已经评分过的书籍，然后找到两本与它们最相似的书，推荐给用户未评分过的书籍

到过多相同类型的 item（如推荐所有的“指环王”系列电影），而不会推荐其他不同的、但用户可能感兴趣的 item。最后是，基于内容的推荐算法中，仅仅使用了包含在 item 元数据中的词汇（如标题、描述年份），这限制了推荐算法的实用性，不能帮助用户探索和发现词汇之外的内容。

第 4 章 混合推荐算法

本文是推荐算法综述的第 4 章。第 1 章主要介绍了推荐算法的主要类型。第 2 章主要涵盖了不同类型的协同过滤算法，突出他们之间的一些细微差别。第 3 章详细介绍了基于内容的过滤算法。在本章中，我们将介绍混合推荐技术，它是建立在我们前面介绍过的算法之上的。我们也将简要讨论针对协同过滤算法（Collaborative Filtering, CF）和基于内容的过滤方法中存在的不足，可以如何通过融入 item 的流行度来缓解这些局限性。

混合方法组合了用户和项目内容特征以及用户的历史行为数据，从而从两种数据中提取信息。一个混合推荐系统如果结合了算法 A 和 B，那么它是希望使用算法 A 的优势来解决算法 B 的缺点。例如，协同过滤算法存在新 item 的问题，即它们不能为用户推荐所有用户没有使用过或评分过 item。但这对于基于内容的推荐算法来说并不是问题，因为当新的 item 进入系统的时候，基于内容的推荐算法可以基于 item 的内容推荐新的 item 给用户。通过提出一个混合推荐方法，让其组合协同过滤算法和基于内容的过滤算法，可以克服单个算法存在的不足，例如冷启动问题和流行度偏差问题。将两种基本的推荐技术进行组合形成一个混合推荐系统存在不同

Method	Description
Weighted	Outputs from several techniques (in the form of scores or votes) are combined with different degrees of importance to offer final recommendations
Switching	Depending on situation, the system changes from one technique to another
Mixed	Recommendations from several techniques are presented at the same time
Feature combination	Features from different recommendation sources are combined as input to a single technique
Cascade	The output from one technique is used as input of another that refines the result
Feature augmentation	The output from one technique is used as input features to another
Meta-level	The model learned by one recommender is used as input to another

表1：将两种基本的推荐技术进行组合形成一个混合推荐系统的不同方式的方式，我们对此进行了概括，如表 1 所示。

举个例子，假设有一些用户表达过对于一系列书籍的偏好。他们越喜欢一本书，他们对书籍的评分就会越高，通常划分为从 1 到 5 的 5 个等级。可以将用户对于书籍的偏好表示为一个矩阵，其中行代表用，列表示书籍，如图 1 所示。

使用相同的设置，我们在第二部分介绍过两个例子，分别展示如何使用基于用户的协同过滤技术和基于 item 的协同过滤技术来对用户产生推荐，并在第三部分，我们展示了如何使用基于内容的过滤技术来对用户进行推荐。现在，我们将组合这三种不同的算法来构建一个新的混合推荐系统。我们将使用加权方法（表 1）来组合这几种不同技术的结果，在这种情况下，通过考虑不同结果的重要性（即权重）来产生新的推荐。

让我们选择第一个用户，并为他产生一些推荐。首先，我们从第二部分中提取使用基于用户的协同过滤技术和基于 item 的协同过滤技术（CF）所得到的推荐结果，从第三部分中提取使用基于内容的过滤技术（CB）所

						
	4	3			5	
	5		4		4	
	4		5	3	4	
		3				5
		4				4
			2	4		5

图1：用户对书籍的偏好。所有的偏好都分为5个等级，5表示最喜欢的。第一个用户（行1）对于第一本书的偏好给出了一个4分的评分。如果一个单元格是空的，表示用户对于该书籍的偏好没有给出

	CF User-Based Recs	CF Item-Based Recs	Content-Based Recs
	 4.5  3.0	 4.5  3.0	 4.5  3.0

图2：分别使用基于用户的协同过滤技术、基于item的协同过滤技术和基于内容的过滤技术所产生的推荐结果用户对于该书籍的偏好没有给出

得到的推荐结果（图 2）。值得注意的是，在这个小的例子中，即使三种方法的输入相同，但它们为相同用户产生了略微不同的推荐结果。

接下来，我们使用一个加权混合推荐系统为给定的用户产生推荐，基于用户的协同过滤的权重为 40%，基于 item 的协同过滤的权重为 30%，基于内容的过滤技术的权重为 30%（图 3）。相比使用单个算法的例子中仅仅向用户推荐两本书，在这个例子中，用户将被推荐所有他们未评分过的三本书。

虽然混合方法解决 CF 方法和 CB 方法中存在的一些局限性，但它们同

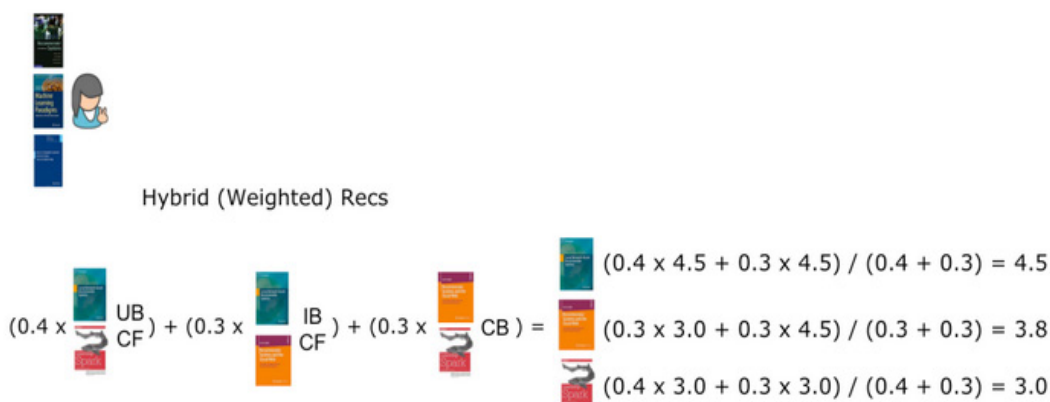


图3：使用加权混合推荐系统对用户进行推荐。基于用户的协同过滤的权重为40%，基于item的协同过滤的权重为30%，基于内容的过滤技术的权重为30%

时也需要大量的工作来获取系统中的不同算法之间的平衡。组合单个的推荐算法的另一种技术是集成方法，它需要学习一个函数（即集成器）来确定不同推荐算法组合的权重。值得注意的是，通常集成方法不仅仅结合了不同的算法，同时也组合了基于相同算法的不同变种（模型）。例如，在赢得 Netflix 竞赛的解决方案中，研究者使用了来自于超过 10 种不同算法（流行度、领域方法、矩阵因子分解、受限玻尔兹曼机、回归等）的 100 多种模型，并通过使用梯度 boosted 决策树将它们组合到一个集成器中。

值得补充的是，基于流行度的方法对于新用户的冷启动问题也是一个很好的解决方法。这种方法在对 item 进行评分时使用某种形式流行度度量，例如最多的下载次数或购买量，然后向新用户推荐这些受欢迎的 item。当你有一个好的流行度度量的时候，这是一个基本的、但功能强大的方法，而且在与其他推荐算法进行比较时提供一个好的基线。在可以切换到其他能够更好地建模用户偏好的方法（协同过滤技术和基于内容的过滤技术）之前，流行度度量本身可以作为一种算法来增强一个推荐系统，以获得足够的活跃度和使用量。流行度的模型也可以组合在混合方法中，以帮助解决推荐系统的新用户冷启动问题。

第 5 章 如何选择推荐算法

第 1 章主要介绍了推荐算法的主要类型。第 2 章主要涵盖了不同类型的协同过滤算法，突出他们之间的一些细微差别。第 3 章详细介绍了基于内容的过滤算法。第 4 章主要介绍了混合引荐技术和基于流行度的推荐方法。在这章中，我们先回顾所有基本的推荐算法，之后会介绍如何选择最合适的推荐算法。

除了我们已经介绍的一些比较传统的推荐系统（例如流行度、协同过滤、基于内容的过滤、混合方法），目前还有许多的其他方法也可以用于增强推荐系统，包括：

- 深度学习；
- 社会化推荐；
- 学习排序；
- 多臂Bandit（探索/利用）；
- 张量因子分解和因子分解（情境感知的推荐）。

这些更先进的和非传统的方法有利于将推荐系统的性能推高到一个新的水平，但实际上这些算法也存在不足，不太易于理解，而且在推荐插件中并没有很好地被支持。在实际应用中，相比一些更传统的方法而言，用

RECOMMENDER SYSTEM ALGORITHMS

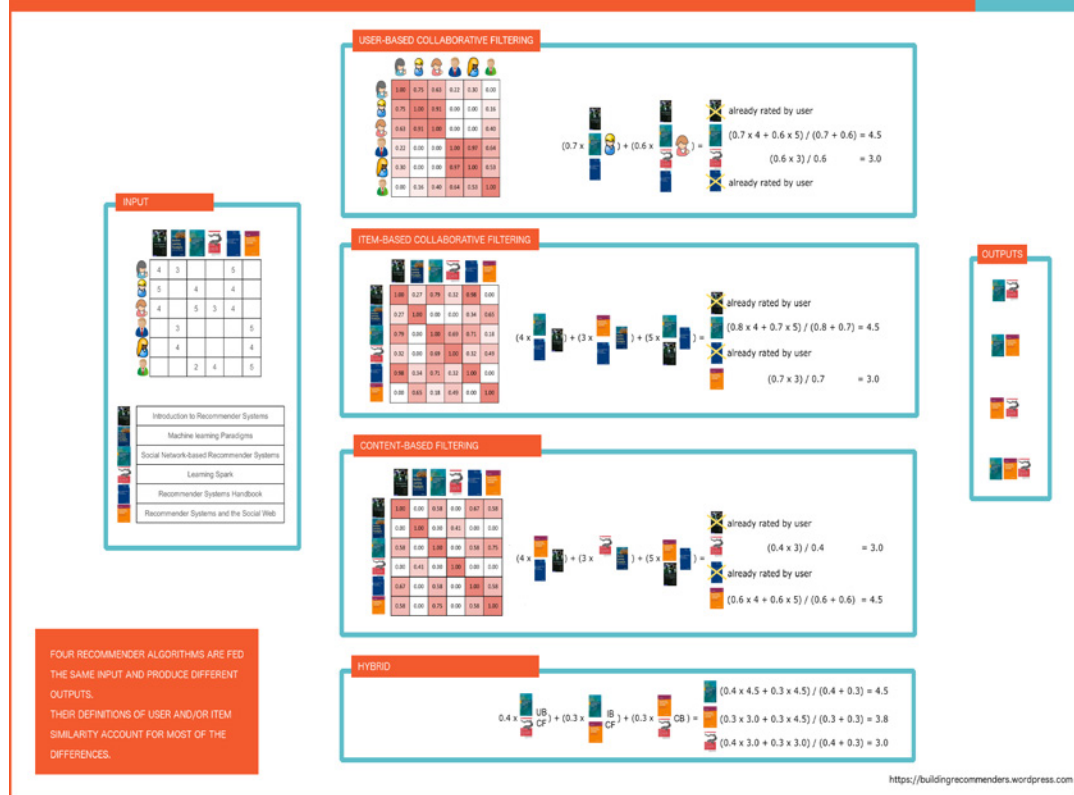


图1：四种推荐系统算法被应用到相同的数据集时所产生的不同的推荐结果。在左边，我们以矩阵的形式给出了用户对于几个item的偏好，以及要推荐的item的标题列表。在中间，我们给出了四种不同的算法为第一个用户（即用户偏好矩阵中的第一行）所产生的推荐结果。按照显示的相似度度量，它们在相似度上有不同的定义。在右边，我们看到由每个推荐算法推荐的item，从上到下按照四种算法排序

户还需要考量执行更新的方法所带来的性能提升是否值得算法所花费的开销。根据我们的经验，基本的传统算法还将在实际系统中应用很久，并还将驱动一些伟大的产品的诞生。

在这个综述的系列文章中，我们想向读者介绍一些常见的推荐算法，包括基于用户的协同过滤算法、基于 item 的协同过滤算法、基于内容的过滤算法和混合方法。在这里，我们通过举一个简单的例子，提供了一个综合的阐述，当有相同的输入数据时，这四种不同的算法将为相同的用户

产生如何不同的推荐结果（图 1）。在算法被应用到大的、真实的数据中时，这种差异会一直存在，所以在决定要使用哪种算法时需要考虑它们的优点和缺点，并且在评价它们的时候，还要考虑它们执行的好坏程度。

在实际应用中，如果你利用协同过滤算法作为你的推荐模型，一般不会出什么问题。协同过滤容易比其他算法产生更好的结果，但是它不能很好地处理新用户和新 item 的冷启动问题，如果要处理这些问题，基于内容的推荐算法是一个很好的备选。如果你有时间，那么可以将这些方法进行组合，这样你就可以同时利用协同过滤算法和基于内容的推荐算法的优点。即使需要考察更为先进的推荐算法，在此之前，先好好考虑一下这些基本的算法也不失为一个好主意。

最后，需要紧紧牢记的是，推荐模型仅仅是推荐系统五个部件中的其中一个。付出努力将推荐模型正确建立起来是非常重要的，但是对于其他的所有部件，如数据收集和处理、后处理、在线模块和用户界面，做出正确的选择同样重要。正如我们一遍又一遍所强调的，该推荐算法仅仅是推荐系统中的一部分，你的决策需要考虑整个产品。

延伸阅读

“多模型融合推荐算法——从原理到实践”



第 6 章 推荐系统和搜索引擎的关系

从信息获取的角度来看，搜索和推荐是用户获取信息的两种主要手段。无论在互联网上，还是在线下的场景里，搜索和推荐这两种方式都大量并存，那么推荐系统和搜索引擎这两个系统到底有什么关系？区别和相似的地方有哪些？本文作者有幸同时具有搜索引擎和推荐系统一线的技术产品开发经验，结合自己的实践经验来为大家阐述两者之间的关系、分享自己的体会。

主动或被动：搜索引擎和推荐系统的选择

获取信息是人类认知世界、生存发展的刚需，搜索就是最明确的一种方式，其体现的动作就是“出去找”，找食物、找地点等，到了互联网时代，搜索引擎（Search Engine）就是满足找信息这个需求的最好工具，你输入想要找的内容（即在搜索框里输入查询词，或称为 Query），搜索引擎快速的给你最好的结果，这样的刚需催生了 Google、百度这样的互联网巨头。

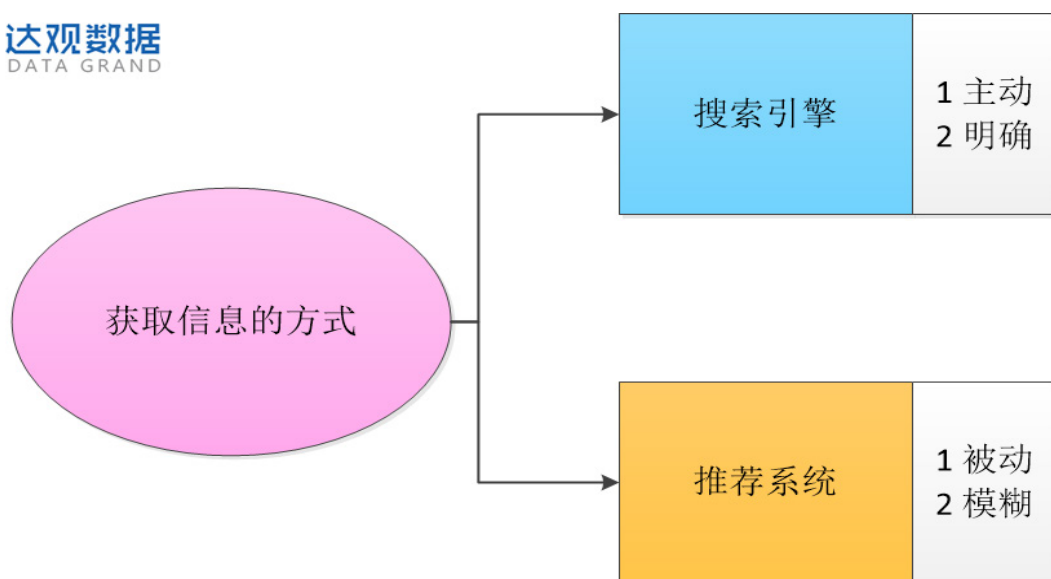


图 1：搜索引擎和推荐系统是获取信息的两种不同方式

但是获取信息的方式除了搜索外，还有另一类，称为推荐系统（Recommendation System, 简称 Recsys），推荐也是伴随人类发展而生的一种基本技能，你一定遇到这样的场景，初来乍到一个地方，会找当地的朋友打听“嗨，请推荐下附近有啥好吃好玩的地方吧！”——知识、信息等通过推荐来传播，这也是一种获取信息的方式。

搜索和推荐的区别如图 1 所示，搜索是一个非常主动的行为，并且用户的需求十分明确，在搜索引擎提供的结果里，用户也能通过浏览和点击来明确的判断是否满足了用户需求。然而，推荐系统接受信息是被动的，需求也都是模糊而不明确的。以“逛”商场为例，在用户进入商场的时候，如果需求不明确，这个时候需要推荐系统，来告诉用户有哪些优质的商品、哪些合适的内容等，但如果用户已经非常明确当下需要购买哪个品牌、什么型号的商品时，直接去找对应的店铺就行，这时就是搜索了。（见图 2）

很多互联网产品都需要同时满足用户这两种需求，例如对提供音乐、新闻、或者电商服务的网站，必然要提供搜索功能，当用户想找某首歌或某样商品的时候，输入名字就能搜到；与此同时，也要同时提供推荐功能，

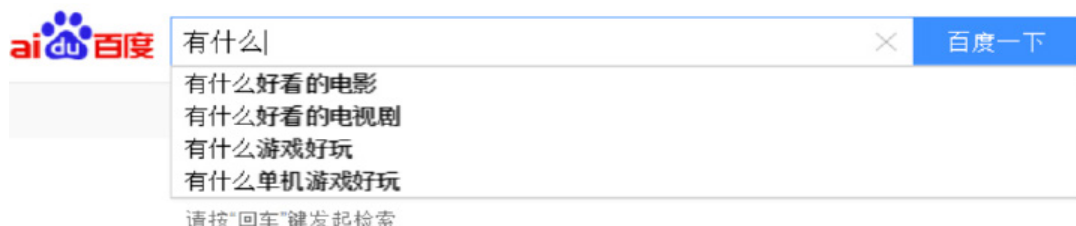


图2：从搜索词中可以看出，用户有大量个性化推荐的需求

当用户就是想来听好听的歌，或者打发时间看看新闻，但并不明确一定要听哪首的时候，给予足够好的推荐，提升用户体验。

个性化程度的高低

除了主被动外，另一个有趣的区别是个性化程度的高低之分。搜索引擎虽然也可以有一定程度的个性化，但是整体上个性化运作的空间是比较小的。因为当需求非常明确时，找到结果的好坏通常没有太多个性化的差异。例如搜“天气”，搜索引擎可以将用户所在地区的信息作补足，给出当地天气的结果，但是个性化补足后给出的结果也是明确的了。

但是推荐系统在个性化方面的运作空间要大得多，以“推荐好看的电影”为例，一百个用户有一百种口味，并没有一个“标准”的答案，推荐系统可以根据每位用户历史上的观看行为、评分记录等生成一个对当前用户最有价值的结果，这也是推荐系统有独特魅力的地方。虽然推荐的种类有很多（例如相关推荐、个性化推荐等），但是个性化对于推荐系统是如此重要，以至于在很多时候大家干脆就把推荐系统称为“个性化推荐”甚至“智能推荐”了。

快速满足还是持续服务

开发过搜索引擎的朋友都知道，评价搜索结果质量的一个重要考量指标是要帮用户尽快的找到需要的结果并点击离开。在设计搜索排序算法里，

需要想尽办法让最好的结果排在最前面，往往搜索引擎的前三条结果聚集了绝大多数的用户点击。简单来说，“好”的搜索算法是需要让用户获取信息的效率更高、停留时间更短。

但是推荐恰恰相反，推荐算法和被推荐的内容（例如商品、新闻等）往往是紧密结合在一起的，用户获取推荐结果的过程可以是持续的、长期的，衡量推荐系统是否足够好，往往要依据是否能让用户停留更多的时间（例如多购买几样商品、多阅读几篇新闻等），对用户兴趣的挖掘越深入，越“懂”用户，那么推荐的成功率越高，用户也越乐意留在产品里。

所以对大量的内容型应用来说，打造一个优秀的推荐系统是提升业绩所不得不重视的手段。

推荐系统满足难以文字表述的需求

目前主流的搜索引擎仍然是以文字构成查询词（Query），这是因为文字是人们描述需求最简洁、直接的方式，搜索引擎抓取和索引的绝大部分内容也是以文字方式组织的。

因为这个因素，我们统计发现用户输入的搜索查询词也大都是比较短小的，查询词中包含 5 个或 5 个以内元素（或称 Term）的占总查询量的 98% 以上（例如：Query“达观数据地址”，包含两个元素“达观数据”和“地址”）。

但另一方面，用户存在着大量的需求是比较难用精炼的文字来组织的，例如想查找“离我比较近的且价格 100 元以内的川菜馆”、“和我正在看的这条裙子同款式的但是价格更优惠的其他裙子”等需求。

一方面几乎没有用户愿意输入这么多字来找结果（用户天然都是愿意偷懒的），另一方面搜索引擎对语义的理解目前还无法做到足够深入；所以在满足这些需求的时候，通过推荐系统设置的功能（例如页面上设置的“相关推荐”、“猜你喜欢”等模块），加上与用户的交互（例如筛选、

排序、点击等），不断积累和挖掘用户偏好，可以将这些难以用文字表达的需求良好的满足起来。

形象的来说，推荐引擎又被人们称为是无声的搜索，意思是用户虽然不用主动输入查询词来搜索，但是推荐引擎通过分析用户历史的行为、当前的上下文场景，自动来生成复杂的查询条件，进而给出计算并推荐的结果。

马太效应和长尾理论

马太效应（matthew effect）是指强者愈强、弱者愈弱的现象，在互联网中引申为热门的产品受到更多的关注，冷门内容则愈发的会被遗忘的现象。马太效应取名自圣经《新约·马太福音》的一则寓言：“凡有的，还要加倍给他叫他多余；没有的，连他所有的也要夺过来。”

搜索引擎就非常充分的体现了马太效应——如下面的 Google 点击热图，越红的部分表示点击多和热，越偏紫色的部分表示点击少而冷，绝大部分用户的点击都集中在顶部少量的结果上，下面的结果以及翻页后的结果获得的关注非常少。这也解释了 Google 和百度的广告为什么这么赚钱，企业客户为什么要花大力气做 SEM 或 SEO 来提升排名——因为只有排到搜索结果的前面才有机会。（见图 3）

有意思的是，与“马太效应”相对应，还有一个非常有影响力的理论称为“长尾理论”。

长尾理论（Long Tail Effect）是“连线”杂志主编克里斯·安德森（Chris Anderson）在 2004 年 10 月的“长尾”（Long Tail）一文中最早提出的，长尾实际上是统计学中幂率（Power Laws）和帕累托分布特征（Pareto Distribution）的拓展和口语化表达，用来描述热门和冷门物品的分布情况。Chris Anderson 通过观察数据发现，在互联网时代由于网络技术能以很低的成本让人们去获得更多的信息和选择，在很多网站内

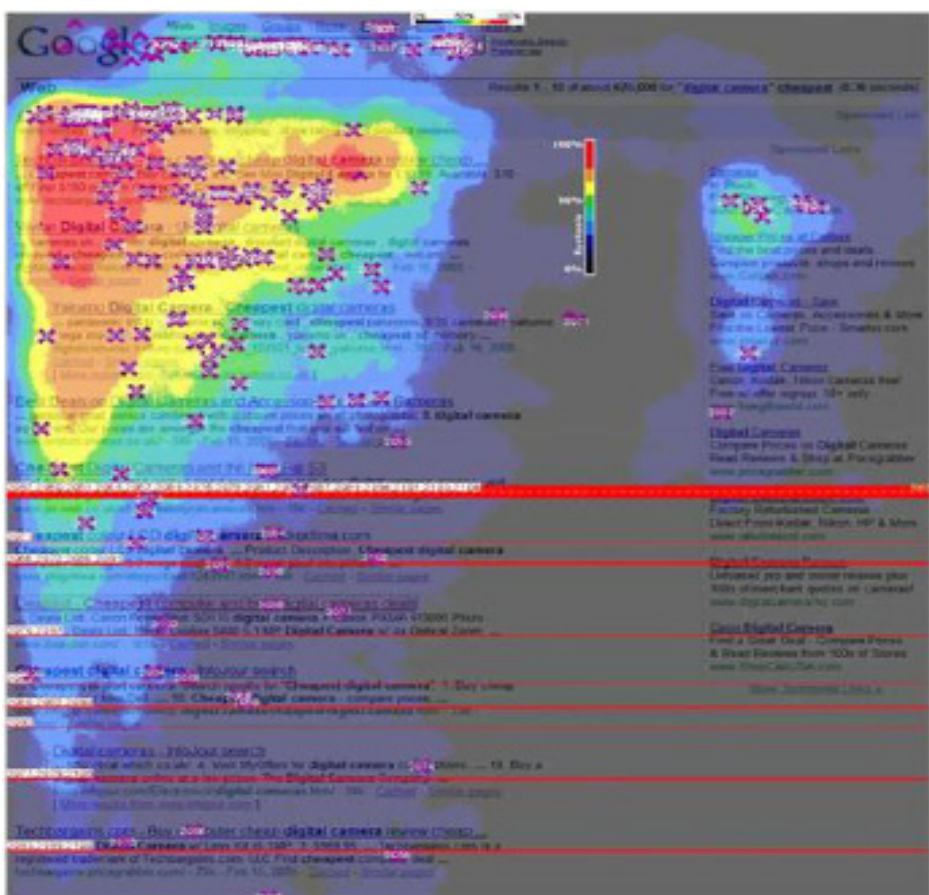


图3：搜索引擎充分体现的马太效应：头部内容吸引了绝大部分点击有越来越多的原先被“遗忘”的非最热门的事物重新被人们关注起来。事实上，每一个人的品味和偏好都并非和主流人群完全一致，Chris 指出：当我们发现得越多，我们就越能体会到我们需要更多的选择。如果说搜索引擎体现着马太效应的话，那么长尾理论则阐述了推荐系统发挥的价值。（见图4）

一个实际的例子就是亚马逊网络书店和传统大型书店的数据对比。市场上出版发行的图书种类超过了数百万，但是其中大部分图书是无法在传统大型书店上架销售的（实体店铺空间有限），而能放在书店显著位置（例如畅销书 Best Seller 货架）上的更是凤毛麟角，因此传统书店的经营模式多以畅销书为中心。但是亚马逊等网络书店的发展为长尾书籍提供了无限广阔的空间，用户浏览、采购这些长尾书籍比传统书店方便得多，于是

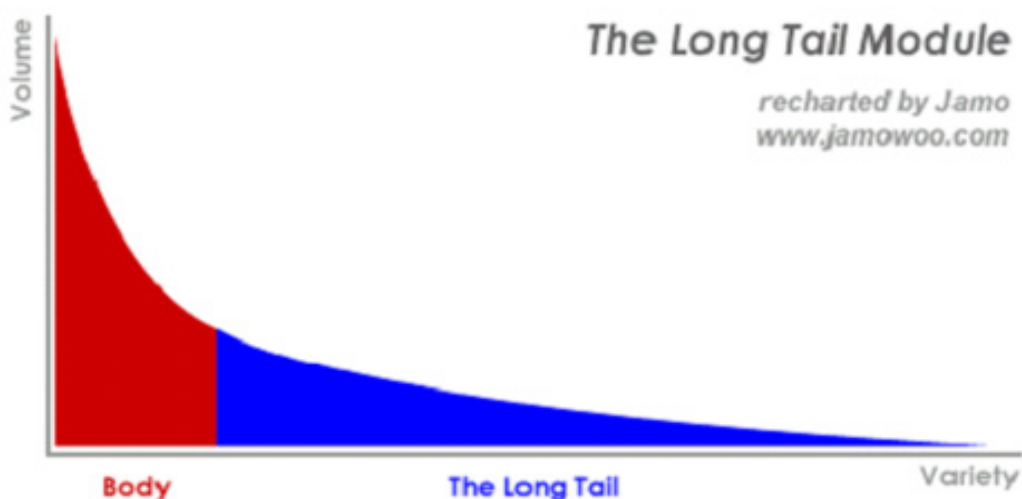


图 4：推荐系统和长尾理论

互联网时代销售成千上万的小众图书，哪怕一次仅卖一两本，但是因为这些图书的种类比热门书籍要多得多，就像长长的尾巴那样，这些图书的销量积累起来甚至超过那些畅销书。正如亚马逊的史蒂夫·凯赛尔所说：“如果我有 10 万种书，哪怕一次仅卖掉一本，10 年后加起来它们的销售就会超过最新出版的《哈利·波特》！”

长尾理论作为一种新的经济模式，被成功的应用于网络经济领域。而对长尾资源的盘活和利用，恰恰是推荐系统所擅长的，因为用户对长尾内容通常是陌生的，无法主动搜索，唯有通过推荐的方式，引起用户的注意，发掘出用户的兴趣，帮助用户做出最终的选择。

盘活长尾内容对企业来说也是非常关键的，营造一个内容丰富、百花齐放的生态，能保障企业健康的生态。试想一下，一个企业如果只依赖 0.1% 的“爆款”商品或内容来吸引人气，那么随着时间推移这些爆款不再受欢迎，而新的爆款又没有及时补位，那么企业的业绩必然会有巨大波动。

只依赖最热门内容的另一个不易察觉的危险是潜在用户的流失：因为只依赖爆款虽然能吸引一批用户（简称 A 类用户），但同时也悄悄排斥了对这些热门内容并不感冒的用户（简称 B 类用户），按照长尾理论，B 类

用户的数量并不少，并且随时间推移 A 类用户会逐步转变为 B 类用户（因为人们都是喜新厌旧的），所以依靠推荐系统来充分满足用户个性化、差异化的需求，让长尾内容在合适的时机来曝光，维护企业健康的生态，才能让企业的运转更稳定，波动更小。

评价方法的异同

搜索引擎通常基于 Cranfield 评价体系，并基于信息检索中常用的评价指标，例如 nDCG（英文全称为 normalized Discounted Cumulative Gain）、Precision-Recall（或其组合方式 F1）、P@N 等方法，具体可参见之前发表于 InfoQ 的文章《怎样量化评价搜索引擎的结果质量 陈运文》。整体上看，评价的着眼点在于将优质结果尽可能排到搜索结果的最前面，前 10 条结果（对应搜索结果的第一页）几乎涵盖了搜索引擎评估的主要内容。让用户以最少的点击次数、最快的速度找到内容是评价的核心。

推荐系统的评价面要宽泛的多，往往推荐结果的数量要多很多，出现的位置、场景也非常复杂，从量化角度来看，当应用于 Top-N 结果推荐时，MAP（Mean Average Precision）或 CTR（Click Through Rate，计算广告中常用）是普遍的计量方法；当用于评分预测问题时，RMSE（Root Mean Squared Error）或 MAE（Mean Absolute Error）是常见量化方法。

由于推荐系统和实际业务绑定更为紧密，从业务角度也有很多侧面评价方法，根据不同的业务形态，有不同的方法，例如带来的增量点击，推荐成功数，成交转化提升量，用户延长的停留时间等指标。

搜索和推荐的相互交融

搜索和推荐虽然有很多差异，但两者都是大数据技术的应用分支，存在着大量的交叠。近年来，搜索引擎逐步融合了推荐系统的结果，例如右侧的“相关推荐”、底部的“相关搜索词”等，都使用了推荐系统的产品思路和运算方法（如下图红圈区域）。

在另一些平台型电商网站中，由于结果数量巨大，且相关性并没有明显差异，因而对搜索结果的个性化排序有一定的运作空间，这里融合运用的个性化推荐技术也对促进成交有良好的帮助。

搜索引擎中融合的推荐系统元素

推荐系统也大量运用了搜索引擎的技术，搜索引擎解决运算性能的一个重要的数据结构是倒排索引技术（Inverted Index），而在推荐系统中，一类重要算法是基于内容的推荐（Content-based Recommendation），这其中大量运用了倒排索引、查询、结果归并等方法。另外点击反馈（Click Feedback）算法等也都在两者中大量运用以提升效果。

本文总结

作为大数据应用的两大类应用，搜索引擎和推荐系统既相互伴随和影响，又满足不同的产品需求。在作为互联网产品的连接器：连接人、信息、服务之间的桥梁，搜索和推荐有其各自的特点，本文对两者的关系进行了阐述，分析了异同。它们都是数据挖掘技术、信息检索技术、计算统计学等悠久学科的智慧结晶，也关联到认知科学、预测理论、营销学等相关学科，感兴趣的读者们可以延伸到这些相关学科里做更深入的了解。



版权声明

InfoQ 中文站出品

架构师特刊：推荐系统（理论篇）

©2016 北京极客邦科技有限公司

本书版权为北京极客邦科技有限公司所有，未经出版者预先的书面许可，不得以任何方式复制或抄袭本书的任何部分，本书任何部分不得用于再印刷，存储于可重复使用的系统，或者以任何方式进行电子、机械、复印和录制等形式传播。

本书提到的公司产品或者使用到的商标为产品公司所有。

如果读者要了解具体的商标和注册信息，应该联系相应的公司。

出版：北京极客邦科技有限公司

北京市朝阳区来广营容和路叶青大厦北园 5 层

欢迎共同参与 InfoQ 中文站的内容建设工作，包括原创投稿和翻译，请联系 editors@cn.infoq.com。

网 址： www.infoq.com.cn

Geekbang.

极客邦科技

整合全球优质学习资源，帮助技术人 and 企业成长

InfoQ

技术媒体

EGG

职业社交

StuQ

在线教育

Git

企业培训



扫一扫关注InfoQ