搜索引擎推荐算法综述

3150101354 梁宇坤

引言

如今已经进入了一个数据爆炸的时代，随着 Web 2.0 的发展， Web 已经变成数据分享的平台，那么，如何让人们在海量的数据中想要找到他们需要的信息将变得越来越难。在这样的情形下，搜索引擎（Google，Bing，百度等等）成为大家快速找到目标信息的最好途径。在用户对自己需求相对明确的时候，用搜索引擎很方便的通过关键字搜索很快的找到自己需要的信息。但搜索引擎并不能完全满足用户对信息发现的需求，那是因为在很多情况下，用户其实并不明确自己的需要，或者他们的需求很难用简单的关键字来表述。又或者他们需要更加符合他们个人口味和喜好的结果，因此出现了推荐系统，与搜索引擎对应，大家也习惯称它为推荐引擎。随着推荐引擎的出现，用户获取信息的方式从简单的目标明确的数据的搜索转换到更高级更符合人们使用习惯的信息发现。如今，随着推荐技术的不断发展，推荐引擎已经在电子商务 (E-commerce，例如 Amazon，当当网) 和一些基于 social 的社会化站点 (包括音乐，电影和图书分享，例如豆瓣，Mtime 等) 都取得很大的成功。这也进一步的说明了，Web2.0 环境下，在面对海量的数据，用户需要这种更加智能的，更加了解他们需求，口味和喜好的信息发现机制。

推荐引擎的分类

推荐引擎可以根据很多指标来分类：

推荐引擎是不是为不同的用户推荐不同的数据

根据这个指标，推荐引擎可以分为基于大众行为的推荐引擎和个性化推荐引擎

- 根据大众行为的推荐引擎，对每个用户都给出同样的推荐，这些推荐可以是静态的由系统管理员人工设定的，或者基于系统所有用户的反馈统计计算出的当下比较流行的物品。
- 个性化推荐引擎，对不同的用户，根据他们的口味和喜好给出更加精确的推荐，这时，系统需要了解需推荐内容和用户的特质，或者基于社会化网络，通过找到与当前用户相同喜好的用户，实现推荐。

这是一个最基本的推荐引擎分类，其实大部分人们讨论的推荐引擎都是将个性化的推荐引擎，因为从根本上说，只有个性化的推荐引擎才是更加智能的信息发现过程。

根据推荐引擎的数据源

其实这里讲的是如何发现数据的相关性，因为大部分推荐引擎的工作原理还是基于物品或者用户的相似集进行推荐。那么参考图 1 给出的推荐系统原理图，根据不同的数据源发现数据相关性的方法可以分为以下几种：

- 根据系统用户的基本信息发现用户的相关程度，这种被称为基于人口统计学的推荐（Demographic-based Recommendation）
- 根据推荐物品或内容的元数据，发现物品或者内容的相关性，这种被称为基于内容的推荐（Content-based Recommendation）
- 根据用户对物品或者信息的偏好，发现物品或者内容本身的相关性，或者是发现用户的相关性，这种被称为基于协同过滤的推荐（Collaborative Filtering-based Recommendation）。

根据推荐模型的建立方式

可以想象在海量物品和用户的系统中，推荐引擎的计算量是相当大的，要实现实时的推荐务必需要建立一个推荐模型，关于推荐模型的建立方式可以分为以下几种：

- 基于物品和用户本身的，这种推荐引擎将每个用户和每个物品都当作独立的实体，预测每个用户对于每个物品的喜好程度，这些信息往往是用一个二维矩阵描述的。由于用户感兴趣的物品远远小于总物品的数目，这样的模型导致大量的数据空置，即我们得到的二维矩阵往往是一个很大的稀疏矩阵。同时为了减小计算量，我们可以对物品和用户进行聚类， 然后记录和计算一类用户对一类物品的喜好程度，但这样的模型又会在推荐的准确性上有损失。
- 基于关联规则的推荐（Rule-based Recommendation）：关联规则的挖掘已经是数据挖掘中的一个经典的问题，主要是挖掘一些数据的依赖关系，典型的场景就是“购物篮问题”，通过关联规则的挖掘，我们可以找到哪些物品经常被同时购买，或者用户购买了一些物品后通常会购买哪些其他的物品，当我们挖掘出这些关联规则之后，我们可以基于这些规则给用户进行推荐。
- 基于模型的推荐（Model-based Recommendation）：这是一个典型的机器学习的问题，可以将已有的用户喜好信息作为训练样本，训练出一个预测用户喜好的模型，这样以后用户在进入系统，可以基于此模型计算推荐。这种方法的问题在于如何将用户实时或者近期的喜好信息反馈给训练好的模型，从而提高推荐的准确度。

其实在现在的推荐系统中，很少有只使用了一个推荐策略的推荐引擎，一般都是在不同的场景下使用不同的推荐策略从而达到最好的推荐效果，例如 Amazon 的推荐，它将基于用户本身历史购买数据的推荐，和基于用户当前浏览的物品的推荐，以及基于大众喜好的当下比较流行的物品都在不同的区域推荐给用户，让用户可以从全方位的推荐中找到自己真正感兴趣的物品。

推荐机制介绍

基于人口统计学的推荐

基于人口统计学的推荐机制（Demographic-based Recommendation）是一种最易于实现的推荐方法，它只是简单的根据系统用户的基本信息发现用户的相关程度，然后将相似用户喜爱的其他物品推荐给当前用户。例如A和B的个人信息(如年龄，性别等)相同，那么系统就可以认为两者的爱好是近似

的，从而将A喜欢的物品推荐给B。在这样的系统中，系统对每个用户都有一个用户 Profile 的建模，其中包括用户的基本信息，例如用户的年龄，性别等等；然后，系统会根据用户的 Profile 计算用户的相似度，用户 A 的 Profile 和用户 B 一样或非常相似，那么系统会认为用户 A 和 B 是相似用户，在推荐引擎中，可以称他们是“邻居”；最后，基于“邻居”用户群的喜好推荐给当前用户一些物品，这些推荐通常是“邻居”的喜欢物品。

对于这样的推荐机制，有点在于：

- 因为不使用当前用户对物品的喜好历史数据，所以对于新用户来讲没有“冷启动（Cold Start）”的问题。
- 这个方法不依赖于物品本身的数据，所以这个方法在不同物品的领域都可以使用，它是领域独立的（domain-independent）。

该方法的主要缺点在于粒度过粗,在一些领域可能无法得到较好的推荐效果。同时可能需要一些无关却又较为敏感的个人信息，而这些信息通常难以获取或获取到虚假信息。

基于内容的推荐

基于内容的推荐是在推荐引擎出现之初应用最为广泛的推荐机制，它的核心思想是根据推荐物品或内容的元数据，发现物品或者内容的相关性，然后基于用户以往的喜好记录，推荐给用户相似的物品。该方法相当于针对用户对物品的选择进行推荐。相对于之前的机制，该方法相对而言推荐更加精准，但是也存在一些问题：

- 需要对物品进行分析和建模，推荐的质量依赖于对物品模型的完整和全面程度。在现在的应用中我们可以观察到关键词和标签（Tag）被认为是描述物品元数据的一种简单有效的方法。
- 物品相似度的分析仅仅依赖于物品本身的特征，这里没有考虑人对物品的态度。
- 因为需要基于用户以往的喜好历史做出推荐，所以对于新用户有“冷启动”的问题。

基于协同过滤的推荐

基于用户的协同过滤推荐的基本原理是，根据所有用户对物品或者信息的偏好，发现与当前用户口味和偏好相似的“邻居”用户群，在一般的应用中是采用KNN算法；然后，基于这 K 个邻居的历史偏好信息，为当前用户进行推荐。例如：假设用户 A 喜欢物品 A，C，用户 B 喜欢物品 B，用户 C 喜欢物品 A ， C ， D；从这些用户的历史喜好信息中，我们可以发现用户 A 和用户 C 的口味和偏好是比较类似的，同时用户 C 还喜欢物品 D，那么我们可以推断用户 A 可能也喜欢物品 D，因此可以将物品 D 推荐给用户 A。

与上面讲的类似，基于项目的协同过滤推荐和基于内容的推荐其实都是基于物品相似度预测推荐，只是相似度计算的方法不一样，前者是从用户历史的偏好推断，而后者是基于物品本身的属性特征信息。

这之外还有一个问题：协同过滤，在基于用户和基于项目两个策略中应该如何选择呢？其实基于项目的协同过滤推荐机制是 Amazon 在基于用户的机制上改良的一种策略，因为在大部分的 Web 站点中，物品的个数是远远小于用户的数量的，而且物品的个数和相似度相对比较稳定，同时基于项目的

机制比基于用户的实时性更好一些。但也不是所有的场景都是这样的情况，可以设想一下在一些新闻推荐系统中，也许物品，也就是新闻的个数可能大于用户的个数，而且新闻的更新程度也有很快，所以它的形似度依然不稳定。所以，其实可以看出，推荐策略的选择其实和具体的应用场景有很大的关系。

基于模型的协同过滤推荐

基于模型的协同过滤推荐就是基于样本的用户喜好信息，训练一个推荐模型，然后根据实时的用户喜好的信息进行预测，计算推荐。

基于协同过滤的推荐机制是现今应用最为广泛的推荐机制，它有以下几个显著的优点：

- 它不需要对物品或者用户进行严格的建模，而且不要求物品的描述是机器可理解的，所以这种方法也是领域无关的。
- 这种方法计算出来的推荐是开放的，可以共用他人的经验，很好的支持用户发现潜在的兴趣偏好

而它也存在以下几个问题：

- 方法的核心是基于历史数据，所以对新物品和新用户都有“冷启动”的问题。
- 推荐的效果依赖于用户历史偏好数据的多少和准确性。
- 在大部分的实现中，用户历史偏好是用稀疏矩阵进行存储的，而稀疏矩阵上的计算有些明显的问题，包括可能少部分人的错误偏好会对推荐的准确度有很大的影响等等。
- 对于一些特殊品味的用户不能给予很好的推荐。
- 由于以历史数据为基础，抓取和建模用户的偏好后，很难修改或者根据用户的使用演变，从而导致这个方法不够灵活。
- 模型解释性不足，不能通过推荐机制解释用户爱好与用户自身的联系。

混合的推荐机制

在现行的 Web 站点上的推荐往往都不是单纯只采用了某一种推荐的机制和策略，他们往往是将多个方法混合在一起，从而达到更好的推荐效果。关于如何组合各个推荐机制，这里讲几种比较流行的组合方法。

- 加权的混合（Weighted Hybridization）: 用线性公式（linear formula）将几种不同的推荐按照一定权重组合起来，具体权重的值需要在测试数据集上反复实验，从而达到最好的推荐效果。
- 切换的混合（Switching Hybridization）：前面也讲到，其实对于不同的情况（数据量，系统运行状况，用户和物品的数目等），推荐策略可能有很大的不同，那么切换的混合方式，就是允许在不同的情况下，选择最为合适的推荐机制计算推荐。
- 分区的混合（Mixed Hybridization）：采用多种推荐机制，并将不同的推荐结果分不同的区显示给用户。其实，Amazon，当当网等很多电子商务网站都是采用这样的方式，用户可以得到很全面的推荐，也更容易找到他们想要的东西。
- 分层的混合（Meta-Level Hybridization）: 采用多种推荐机制，并将一个推荐机制的结果作为另一个的输入，从而综合各个推荐机制的优缺点，得到更加准确的推荐。

总结

在网络数据爆炸的年代，如何让用户更快的找到想要的数据，如何让用户发现自己潜在的兴趣和需求，无论是对于电子商务还是社会网络的应用都是至关重要的。推荐引擎的出现，使得这个问题越来越被大家关注。但对大多数人来讲，也许还在惊叹它为什么总是能猜到你到底想要些什么。推荐引擎的魔力在于你不清楚在这个推荐背后，引擎到底记录和推理了些什么。

其实推荐引擎只是默默的记录和观察你的一举一动，然后再借由所有用户产生的海量数据分析和发现其中的规律，进而慢慢的了解你，你的需求，你的习惯，并默默的无声息的帮助你快速的解决你的问题，找到你想要的东西。其实有时候，机器比你更了解你自己。