# 推荐系统介绍

## 什么是推荐系统

## 推荐系统的应用

# 推荐系统算法

# 推荐系统问题

## 冷启动问题

推荐系统需要根据用户的历史行为和兴趣预测用户未来的行为和兴趣，因此大量的用户行为数据就成为推荐系统的重要组成部分和先决条件。

对于很多像百度、当当这样的网站来说，这或许不是个问题，因为它们目前已经积累了大量的用户数据。但是对于很多做纯粹推荐系统的网站 （比如Jinni和Pandora），或者很多在开始阶段就希望有个性化推荐应用的网站来说，如何在没有大量用户数据的情况下设计个性化推荐系统并且让用户对推荐结果满意从而愿意使用推荐系统，就是冷启动的问题。

### 冷启动问题介绍

冷启动问题（cold start）主要分3类。 

* **用户冷启动** 用户冷启动主要解决如何给新用户做个性化推荐的问题。当新用户到来时， 我们没有他的行为数据，所以也无法根据他的历史行为预测其兴趣，从而无法借此给他做个性化推荐。 
* **物品冷启动** 物品冷启动主要解决如何将新的物品推荐给可能对它感兴趣的用户这一问题。
* **系统冷启动** 系统冷启动主要解决如何在一个新开发的网站上（还没有用户，也没有用 户行为，只有一些物品的信息）设计个性化推荐系统，从而在网站刚发布时就让用户体验到个性化推荐服务这一问题。

对于这3种不同的冷启动问题，有不同的解决方案。一般来说，可以参考如下解决方案：

* **提供非个性化的推荐** 非个性化推荐的最简单例子就是热门排行榜，我们可以给用户推荐热门排行榜，然后等到用户数据收集到一定的时候，再切换为个性化推荐。 
* 利用用户注册时提供的年龄、性别等数据**做粗粒度的个性化**。 
* **利用用户的社交网络账号登录**（需要用户授权），导入用户在社交网站上的好友信息，然后给用户推荐其好友喜欢的物品。 
* **要求用户在登录时对一些物品进行反馈**，收集用户对这些物品的兴趣信息，然后给用户推荐那些和这些物品相似的物品。 
* **对于新加入的物品，可以利用内容信息**，将它们推荐给喜欢过和它们相似的物品的用户。
* **在系统冷启动时，可以引入专家的知识**，通过一定的高效方式迅速建立起物品的相关度表。

### 利用用户注册信息

### 选择合适的物品启动用户的兴趣

### 利用物品的内容信息

物品冷启动需要解决的问题是如何将新加入的物品推荐给对它感兴趣的用户。物品冷启动在 新闻网站等时效性很强的网站中非常重要，因为那些网站中时时刻刻都有新加入的物品，而且每 个物品必须能够在第一时间展现给用户，否则经过一段时间后，物品的价值就大大降低了。

### 发挥专家的作用

很多推荐系统在建立时，既没有用户的行为数据，也没有充足的物品内容信息来计算准确的物品相似度。那么，为了在推荐系统建立时就让用户得到比较好的体验，很多系统都利用专家进行标注。这方面的代表系统是个性化网络电台Pandora和电影推荐网站Jinni。

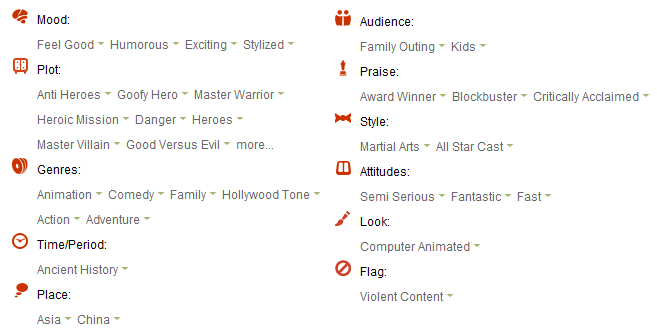
Pandora是一个给用户播放音乐的个性化电台应用。众所周知，计算音乐之间的相似度是比较困难的。首先，音乐是多媒体，如果从音频分析入手计算歌曲之间的相似度，则技术门槛很高，而且也很难计算得令人满意。其次，仅仅利用歌曲的专辑、歌手等属性信息很难获得令人满意的歌曲相似度表，因为一名歌手、一部专辑往往只有一两首好歌。

为了解决这个问题，Pandora雇用了一批懂计算机的音乐人进行了一项称为音乐基因的项目①。他们听了几万名歌手的歌，并对这些歌的各个维度进行标注。最终，他们使用了400多个特征②（Pandora称这些特征为基因）。标注完所有的歌曲后，每首歌都可以表示为一个400维的向量，然后通过常见的向量相似度算法可以计算出歌曲的相似度。

* ① “About The Music Genome Project”，<http://www.pandora.com/mgp.shtml>
* ② <http://en.wikipedia.org/wiki/List_of_Music_Genome_Project_attributes>。

和Pandora类似，Jinni也利用相似的想法设计了电影基因系统，让专家给电影进行标注。Jinni网站对电影基因项目进行了介绍③。下图是Jinni中专家给《功夫熊猫》标注的基因。

* ③ 参见<http://www.jinni.com/movie-genome.html>。



可以看到，这里的基因包括如下分类。

* **心情（Mood）**　表示用户观看电影的心情，比如对于《功夫熊猫》观众会觉得很幽默，很兴奋。
* **剧情（Plot）**　包括电影剧情的标签。
* **类别（Genres）**　表示电影的类别，主要包括动画片、喜剧片、动作片等分类。
* **时间（Time/Period）**　电影故事发生的时间。
* **地点（Place）**　电影故事发生的地点。
* **观众（Audience）**　电影的主要观众群。
* **获奖（Praise）**　电影的获奖和评价情况。
* **风格（Style）**　功夫片、全明星阵容等。
* **态度（Attitudes）**　电影描述故事的态度。
* **画面（Look）**　电脑拍摄的画面技术，比如《功夫熊猫》是用电脑动画制作的。
* **标记（Flag）**　主要表示电影有没有暴力和色情内容。

Jinni在电影基因工程中采用了半人工、半自动的方式。首先，它让专家对电影进行标记，每个电影都有大约50个基因，这些基因来自大约1000个基因库。然后，在专家标记一定的样本后，Jinni会使用自然语言理解和机器学习技术，通过分析用户对电影的评论和电影的一些内容属性对电影（特别是新电影）进行自己的标记。同时，Jinni也设计了让用户对基因进行反馈的界面，希望通过用户反馈不断改进电影基因系统。总之，Jinni通过专家和机器学习相结合的方法解决了系统冷启动问题。

# 推荐系统&其他因素

## 结合上下文信息

## 结合社交网络

# 推荐系统实践

# 推荐系统评测

## 章节目标：

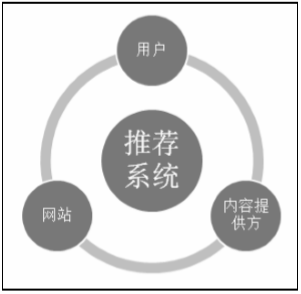
* 了解推荐系统 --- 评测实验方法；
* 了解推荐系统 --- 各种评测指标；
* 用评测指标去评价推荐系统各方面的性能，并作出相关优化；

什么才是好的推荐系统？这是推荐系统评测需要解决的首要问题。一个完整的推荐系统一般存在3个参与方：用户、物品提供者和提供推荐系统的网站。

以图书推荐为例：

* 首先，推荐系统需要满足用户的需求，给用户推荐那些令他们感兴趣的图书。
* 其次，推荐系统要让各出版社的书都能够被推荐给对其感兴趣的用户，而不是只推荐几个大型出版社的书。
* 最后，好的推荐系统设计，能够让推荐系统本身收集到高质量的用户反馈，不断完善推荐的质量，增加用户和网站的交互，提高网站的收入。

因此在评测一个推荐算法时，需要同时考虑三方的利益， 一个好的推荐系统是能够令三方共赢的系统。



## 实验方法

在推荐系统中，主要有3种评测推荐效果的实验方法，即离线实验（offline experiment）、用户调查（user study） 和在线实验（online experiment）。

### 离线实验

离线实验的方法一般由如下几个步骤构成：

(1) 通过日志系统获得用户行为数据，并按照一定格式生成一个标准的数据集；

(2) 将数据集按照一定的规则分成训练集和测试集；

(3) 在训练集上训练用户兴趣模型，在测试集上进行预测；

(4) 通过事先定义的离线指标评测算法在测试集上的预测结果。

从上面的步骤可以看到，推荐系统的离线实验都是在数据集上完成的，也就是说它不需要一个实际的系统来供它实验，而只要有一个从实际系统日志中提取的数据集即可。

|  |  |
| --- | --- |
| 优点 | 缺点 |
| 1）不需要有对实际系统的控制权；  2）不需要真实用户参与实验；  3）速度快，可以测试大量算法； | 1）无法计算商业上关心的指标，如点击率、转化率等；  2）离线实验的指标和商业指标存在差距； |

### 用户调查

离线实验的指标和实际的商业指标存在差距，比如预测准确率和用户满意度之间就存在很大差别，高预测准确率不等于高用户满意度。因此，如果要准确评测一个算法，需要相对比较真实的环境。最好的方法就是将算法直接上线测试，但在对算法会不会降低用户满意度不太有把握的情况下，上线测试具有较高的风险，所以在上线测试前一般需要做一次称为用户调查的测试。

用户调查需要有一些真实用户，让他们在需要测试的推荐系统上完成一些任务。在他们完成任务时，我们需要观察和记录他们的行为，并让他们回答一些问题。最后，我们需要通过分析他们的行为和答案了解测试系统的性能。

用户调查是推荐系统评测的一个重要工具，很多离线时没有办法评测的与用户主观感受有关的指标都可以通过用户调查获得。比如，如果我们想知道推荐结果是否很令用户惊喜，那我们最好直接询问用户。

但是，用户调查也有一些缺点。首先，用户调查成本很高，需要用户花大量 时间完成一个个任务，并回答相关的问题。有些时候，还需要花钱雇用测试用户。因此，大多数情况下很难进行大规模的用户调查，而对于参加人数较少的用户调查，得出的很多结论往往没有统计意义。因此，我们在做用户调查时，一方面要控制成本，另一方面又要保证结果的统计意义。

此外，测试用户也不是随便选择的。需要尽量保证测试用户的分布和真实用户的分布相同，比如男女各半，以及年龄、活跃度的分布都和真实用户分布尽量相同。

此外，用户调查要尽量保证是双盲实验，不要让实验人员和用户事先知道测试的目标，以免用户的回答和实验人员的测试受主观成分的影响。

|  |  |
| --- | --- |
| 优点 | 缺点 |
| 1）可以获得很多体现用户主观感受的指标；  2）相对在线实验风险很低，出现错误后很容易弥补； | 1）招募测试用户代价较大；  2）很难组织大规模的测试用户，测试结果的统计意义不足；  3）双盲实验设计非常困难；  4）测试环境下的用户行为在真实环境上很难重现； |

### 在线实验

在完成离线实验和必要的用户调查后，可以将推荐系统上线做AB测试，将它和旧的算法进行比较。

AB测试是一种很常用的在线评测算法的实验方法。它通过一定的规则将用户随机分成几组， 并对不同组的用户采用不同的算法，然后通过统计不同组用户的各种不同的评测指标比较不同算法，比如可以统计不同组用户的点击率，通过点击率比较不同算法的性能。

对AB测试感兴趣的 读者可以浏览一下网站http://www.abtests.com/，该网站给出了很多通过实际AB测试提高网站用户 满意度的例子，从中我们可以学习到如何进行合理的AB测试。

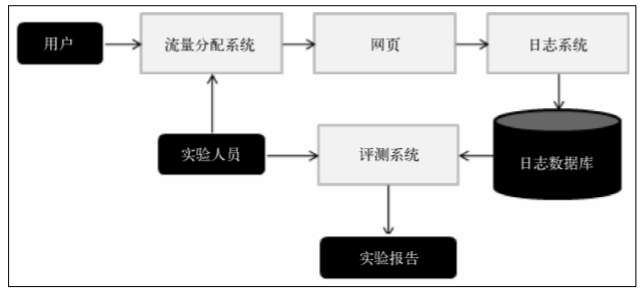
|  |  |
| --- | --- |
| 优点 | 缺点 |
| 1）可以公平获得不同算法实际在线时的性能指标，包括商业上关注的指标； | 1）周期比较长，必须进行长期的实验才能得到可靠的结果； |

因此一般不会用 AB测试测试所有的算法，而只是用它测试那些在离线实验和用户调查中表现很好的算法。

其次，一个大型网站的AB测试系统的设计也是一项复杂的工程。一个大型网站的架构分前端和后端， 从前端展示给用户的界面到最后端的算法，中间往往经过了很多层，这些层往往由不同的团队控制，而且都有可能做AB测试。如果为不同的层分别设计AB测试系统，那么不同的AB测试之间往 往会互相干扰。

比如，当我们进行一个后台推荐算法的AB测试，同时网页团队在做推荐页面的界面AB测试，最终的结果就是你不知道测试结果是自己算法的改变，还是推荐界面的改变造成的。因此，切分流量是AB测试中的关键，不同的层以及控制这些层的团队需要从一个统一的地方获得自己AB测试的流量，而不同层之间的流量应该是正交的。

下图是一个简单的AB测试系统。用户进入网站后，流量分配系统决定用户是否需要被进行AB测试，如果需要的话，流量分配系统会给用户打上在测试中属于什么分组的标签。然后用户浏览网页，而用户在浏览网页时的行为都会被通过日志系统发回后台的日志数据库。此时，如果用户有测试分组的标签，那么该标签也会被发回后台数据库。在后台，实验人员的工作首先是配置流量分配系统，决定满足什么条件的用户参加什么样的测试。其次，实验人员需要统计日志数据库中的数据，通过评测系统生成不同分组用户的实验报告，并比较和评测实验结果。



### 小结

一般来说，一个新的推荐算法最终上线，需要完成上面所说的3个实验。 

* 首先，需要通过离线实验证明它在很多离线指标上优于现有的算法；
* 然后，需要通过用户调查确定它的用户满意度不低于现有的算法；
* 最后，通过在线的AB测试确定它在我们关心的指标上优于现有的算法。

## 评测指标

为了全面评测推荐系统对三方利益的影响，我们常从不同角度出发，提出不同的指标。这些指标包括**准确度、覆盖度、新颖度、惊喜度、信任度、透明度**等，可用于评价推荐系统各方面的性能。

这些指标有些可以定量计算，有些只能定性描述，有些可以通过离线实验计算，有些需要通过用户调查获得，还有些只能在线评测。

### 用户满意度

用户作为推荐系统的重要参与者，其满意度是评测推荐系统的最重要指标。但是，用户满意度没有办法离线计算，只能通过**用户调查**或者**在线实验**获得。

用户调查获得用户满意度主要是通过**调查问卷**的形式。用户对推荐系统的满意度分为不同的层次。

GroupLens曾经做过一个论文推荐系统的调查问卷，该问卷的调查问题是：

请问下面哪句话最能描述你看到推荐结果后的感受？

* 推荐的论文都是我非常想看的。
* 推荐的论文很多我都看过了，确实是符合我兴趣的不错论文。
* 推荐的论文和我的研究兴趣是相关的，但我并不喜欢。
* 不知道为什么会推荐这些论文，它们和我的兴趣丝毫没有关系。

由此可以看出，这个调查问卷不是简单地询问用户对结果是否满意，而是从不同的侧面询问用户对结果的不同感受。比如，如果仅仅问用户是否满意，用户可能心里认为大体满意，但是对某个方面还有点不满，因而可能很难回答这个问题。因此在设计问卷时需要考虑到用户各方面的感受，这样用户才能针对问题给出自己准确的回答。

在在线系统中，用户满意度主要通过一些对**用户行为的统计**得到。比如在电子商务网站中， 用户如果购买了推荐的商品，就表示他们在一定程度上满意。因此，我们可以利用**购买率**度量用户的满意度。此外，有些网站会通过设计一些**用户反馈界面**收集用户满意度。比如在视频网站 Hulu的推荐页面和豆瓣网络电台中，都有对推荐结果满意或者不满意的反馈按钮，通过统计两种按钮的单击情况就可以度量系统的用户满意度。更一般的情况下，我们可以用**点击率**、**用户停留时间**和**转化率**等指标度量用户的满意度。

### 预测准确度

**预测准确度**度量一个推荐系统或者推荐算法预测用户行为的能力。这个指标是最重要的推荐系统**离线评测指标**，从推荐系统诞生的那一天起，几乎99%与推荐相关的论文都在讨论这个指标。 这主要是因为该指标可以通过离线实验计算，方便了很多学术界的研究人员研究推荐算法。

* 首先，在计算该指标时需要有一个离线的数据集，该数据集包含用户的历史行为记录；
* 然后，将该数据集通过时间分成训练集和测试集；
* 最后，通过在训练集上建立用户的行为和兴趣模型，预测用户在测试集上的行为，并计算预测行为和测试集上实际行为的重合度作为预测准确度；

由于离线的推荐算法有不同的研究方向，因此下面将针对不同的研究方向介绍它们的预测准确度指标。

#### （1）评分预测

很多提供推荐服务的网站都有一个让用户给物品打分的功能。那么，如果知道了用户对物品的历史评分，就可以从中习得用户的兴趣模型，并预测该用户在将来看到一个他没有评过分的物品时，会给这个物品评多少分。预测用户对物品评分的行为称为评分预测。



评分预测的预测准确度一般通过均方根误差（RMSE）和平均绝对误差（MAE）计算。关于两者的定义见[**[附录:误差]**](#_误差)

假设我们用一个列表records存放用户评分数据，令records[i] = [u,i,rui,pui]，其中rui是用户u对物品i的实际评分，pui是算法预测出来的用户u对物品i的评分，那么下面的代码分别实现了RMSE和MAE的计算过程。

**import** math

**def** RMSE(records):

**return** math.sqrt(sum([(rui - pui) \* (rui - pui) **for** u, i, rui, pui **in** records])/float(len(records)))

**def** MAE(records):

**return** sum([abs(rui - pui) **for** u, i, rui, pui **in** records])/ float(len(records))

**if** \_\_name\_\_ == '\_\_main\_\_':

    records = [["u1","v1",0.5,0.3],

               ["u1", "v2", 0.5, 0.5],

               ["u2", "v1", 0.5, 0.5],

               ["u2", "v2", 0.5, 0.3]

               ]

**print**(RMSE(records))  # 0.14142135623730953

**print**(MAE(records))  # 0.1

关于RMSE和MAE这两个指标的优缺点， Netflix认为RMSE加大了对预测不准的用户物品评 分的惩罚（平方项的惩罚），因而对系统的评测更加苛刻。研究表明，如果评分系统是基于整数 建立的（即用户给的评分都是整数），那么对预测结果取整会降低MAE的误差。

#### （2）TopN推荐

网站在提供推荐服务时，一般是给用户一个个性化的推荐列表，这种推荐叫做TopN推荐。TopN推荐的预测准确率一般通过准确率（precision）/召回率（recall）度量。 关于两者的定义见[**[附录:混淆矩阵]**](#_混淆矩阵)**。**

下面的Python代码同时计算出了一个推荐算法的准确率和召回率：

**def** Recommend(user,N):

    #模拟获得推荐

    recommendDict = {"user1": {"item1","item2","item3","item4","item5"},

                     "user2": {"item1","item2","item3","item4","item5"},

                     "user3": {"item1","item2","item3","item4","item5"}

            }

**return** recommendDict[user]

**def** PrecisionRecall(test, N):

    hit = 0

    n\_recall = 0

    n\_precision = 0

**for** user, items **in** test.items():

        rank = Recommend(user, N)

        hit += len(rank & items)

        n\_recall += len(items)

        n\_precision += N

**print**(hit) #3

**print**(n\_precision) #6

**print**(n\_recall) #6

**return** [hit / (1.0 \* n\_recall), hit / (1.0 \* n\_precision)]

**if** \_\_name\_\_ == '\_\_main\_\_':

    test = {"user1": {"item1","item2"},

            "user2": {"item6", "item5"},

            "user3": {"item7", "item8"}

            }

**print**(PrecisionRecall(test,2)) # [0.5, 0.5]

有的时候，为了全面评测TopN推荐的准确率和召回率，一般会选取不同的推荐列表长度N， 计算出一组准确率/召回率，然后画出准确率/召回率曲线（precision/recall curve）。

* **关于评分预测和TopN推荐的讨论**

评分预测一直是推荐系统研究的热点，绝大多数推荐系统的研究都是基于用户评分数据的评分预测。这主要是因为，一方面推荐系统的早期研究组GroupLens的研究主要就是基于电影评分 数据MovieLens进行的，其次，Netflix大赛也主要面向评分预测问题。因而，很多研究人员都将 研究精力集中在优化评分预测的RMSE上。

对此，亚马逊前科学家Greg Linden有不同的看法。2009年, 他在Communications of the ACM 网站发表了一篇[文章[1]](#_4.相关文献)，指出电影推荐的目的是找到用户最有可能感兴趣的电影，而不是预测用户看了电影后会给电影什么样的评分。因此，TopN推荐更符合实际的应用需求。也许有一部电影用户看了之后会给很高的分数，但用户看的可能性非常小。因此，预测用户是否会看一部电影， 应该比预测用户看了电影后会给它什么评分更加重要。

### 覆盖率

覆盖率（coverage）描述一个推荐系统对物品长尾的发掘能力。覆盖率有不同的定义方法， 最简单的定义为推荐系统能够推荐出来的物品占总物品集合的比例。假设系统的用户集合为U，推荐系统给每个用户推荐一个长度为N的物品列表R(u)。那么推荐系统的覆盖率可以通过下面的公式计算：

公式解释：

* 分子：对所有用户的推荐列表求并集
* 分母：总物品集合

### 多样性

用户的兴趣是广泛的，在一个视频网站中，用户可能既喜欢看《猫和老鼠》一类的动画片， 也喜欢看成龙的动作片。那么，为了满足用户广泛的兴趣，推荐列表需要能够覆盖用户不同的兴趣领域，即推荐结果需要具有多样性。多样性推荐列表的好处用一句俗话表述就是“不在一棵树 上吊死”。

尽管用户的兴趣在较长的时间跨度中是一样的，但具体到用户访问推荐系统的某一刻， 其兴趣往往是单一的，那么如果推荐列表只能覆盖用户的一个兴趣点，而这个兴趣点不是用户这个时刻的兴趣点，推荐列表就不会让用户满意。反之，如果推荐列表比较多样，覆盖了用户绝大多数的兴趣点，那么就会增加用户找到感兴趣物品的概率。因此给用户的推荐列表也需要满足用户广泛的兴趣，即具有多样性。

多样性描述了推荐列表中物品两两之间的不相似性。因此，多样性和相似性是对应的。假设 定义了物品和之间的相似度，那么用户的推荐列表的多样性定义如下：

公式解释：

* 分子：用户u的推荐列表中，两两物品的相似度之和
* 分母：即 ,表示列表中任取两个的组合个数
* 1-平均相似度，即为多样性

而推荐系统的整体多样性可以定义为所有用户推荐列表多样性的平均值：

从上面的定义可以看到，不同的物品相似度度量函数 可以定义不同的多样性。如果用内容相似度描述物品间的相似度，我们就可以得到内容多样性函数，如果用协同过滤的相似度函数描述物品间的相似度，就可以得到协同过滤的多样性函数。

关于推荐系统多样性最好达到什么程度，可以通过一个简单的例子说明：

假设用户喜欢动作片和动画片，且**用户80%的时间在看动作片，20%的时间在看动画片**。那么，可以提供4种不同的 推荐列表：

A列表中有10部动作片，没有动画片；

B列表中有10部动画片，没有动作片；

**C列表中有8部动作片和2部动画片；**

D列表有5部动作片和5部动画片；

在这个例子中，**一般认为C列表是最好的**，因为它具有一定的多样性，但又考虑到了用户的主要兴趣。A满足了用户的主要兴趣， 但缺少多样性，D列表过于多样，没有考虑到用户的主要兴趣。B列表即没有考虑用户的主要兴趣，也没有多样性，因此是最差的。

### 新颖性

新颖的推荐是指给用户推荐那些他们以前没有听说过的物品。在一个网站中实现新颖性的最简单办法是，把那些用户之前在网站中对其有过行为的物品从推荐列表中过滤掉。

比如在一个视频网站中，新颖的推荐不应该给用户推荐那些他们已经看过、打过分或者浏览过的视频。但是，有些视频可能是用户在别的网站看过，或者是在电视上看过，因此仅仅过滤掉本网站中用户有过行为的物品还不能完全实现新颖性。

O’scar Celma在博士[论文[2]](#_3.混淆矩阵)中研究了新颖度的评测。**评测新颖度的最简单方法是利用推荐结果的平均流行度，因为越不热门的物品越可能让用户觉得新颖。因此，如果推荐结果中物品的平均热门程度较低，那么推荐结果就可能有比较高的新颖性。**

但是，用推荐结果的平均流行度度量新颖性比较粗略，因为不同用户不知道的东西是不同的。 因此，要准确地统计新颖性需要做用户调查。 最近几年关于多样性和新颖性的研究越来越受到推荐系统研究人员的关注。ACM的推荐系统会议在2011年有一个专门的研讨会讨论推荐的多样性和新颖性。该研讨会的组织者认为，通过牺牲精度来提高多样性和新颖性是很容易的，而困难的是如何在不牺牲精度的情况下提高多样性和新颖性。关心这两个指标的读者可以关注一下这个研讨会最终发表的[论文[3]](#_3.混淆矩阵)。

### 惊喜度

惊喜度（serendipity）是最近这几年推荐系统领域最热门的话题。但什么是惊喜度，惊喜度与新颖性有什么区别是首先需要弄清楚的问题。注意，这里讨论的是惊喜度和新颖度作为推荐指标在意义上的区别，而不是这两个词在中文里的含义区别（因为这两个词是英文词翻译过来的， 所以它们在中文里的含义区别和英文词的含义区别并不相同），所以我们首先要摒弃大脑中关于这两个词在中文中的基本含义。

可以举一个例子说明这两种指标的区别。假设一名用户喜欢周星驰的电影，然后我们给他推 荐了一部叫做《临歧》的电影（该电影是1983年由刘德华、周星驰、梁朝伟合作演出的，很少有 人知道这部有周星驰出演的电影），而该用户不知道这部电影，那么可以说这个推荐具有新颖性。但是，这个推荐并没有惊喜度，因为该用户一旦了解了这个电影的演员，就不会觉得特别奇怪。 但如果我们给用户推荐张艺谋导演的《红高粱》，假设这名用户没有看过这部电影，那么他看完 这部电影后可能会觉得很奇怪，因为这部电影和他的兴趣一点关系也没有，但如果用户看完电影 后觉得这部电影很不错，那么就可以说这个推荐是让用户惊喜的。

这个例子的原始版本来自于 Guy Shani的[论文[4]](#_3.混淆矩阵)，他的基本意思就是，如果推荐结果和用户的历史兴趣不相似，但却让用户觉得满意,那么就可以说推荐结果的惊喜度很高，而推荐的新颖性仅仅取决于用户是否听说过这个推荐结果。

目前并没有什么公认的惊喜度指标定义方式，这里只给出一种定性的度量方式。上面提到， 令用户惊喜的推荐结果是和用户历史上喜欢的物品不相似，但用户却觉得满意的推荐。那么，定义惊喜度需要首先定义推荐结果和用户历史上喜欢的物品的相似度，其次需要定义用户对推荐结果的满意度。前面也曾提到，用户满意度只能通过问卷调查或者在线实验获得，而推荐结果和用户历史上喜欢的物品相似度一般可以用内容相似度定义。也就是说，如果获得了一个用户观看电影的历史，得到这些电影的演员和导演集合A，然后给用户推荐一个不属于集合A的导演和演员创作的电影，而用户表示非常满意，这样就实现了一个惊喜度很高的推荐。因此提高推荐惊喜度 需要提高推荐结果的用户满意度，同时降低推荐结果和用户历史兴趣的相似度。

惊喜度的问题最近几年获得了学术界的一定关注，但这方面的工作还不是很成熟。相关工作 可以参考Yuan Cao Zhang等的[论文[5]](#_3.混淆矩阵)和Tomoko Murakami等的[论文[6]](#_3.混淆矩阵)。

### 信任度

如果你有两个朋友，一个人你很信任，一个人经常满嘴跑火车，那么如果你信任的朋友推荐 你去某个地方旅游，你很有可能听从他的推荐，但如果是那位满嘴跑火车的朋友推荐你去同样的地方旅游，你很有可能不去。这两个人可以看做两个推荐系统，尽管他们的推荐结果相同，但用户却可能产生不同的反应，这就是因为用户对他们有不同的信任度。

对于基于机器学习的自动推荐系统，同样存在信任度（trust）的问题，如果用户信任推荐系统，那就会增加用户和推荐系统的交互。特别是在电子商务推荐系统中，让用户对推荐结果产生信任是非常重要的。同样的推荐结果，以让用户信任的方式推荐给用户就更能让用户产生购买欲，而以类似广告形式的方法推荐给用户就可能很难让用户产生购买的意愿。

度量推荐系统的信任度只能通过**问卷调查**的方式，询问用户是否信任推荐系统的推荐结果。

提高推荐系统的信任度主要有两种方法。首先需要**增加推荐系统的透明度**（transparency）[[8]](#_3.混淆矩阵)， 而增加推荐系统透明度的主要办法是**提供推荐解释**。只有让用户了解推荐系统的运行机制，让用户认同推荐系统的运行机制，才会提高用户对推荐系统的信任度。其次是考虑**用户的社交网络信息**，利用用户的好友信息给用户做推荐，并且用好友进行推荐解释。这是因为用户对他们的好友一般都比较信任，因此如果推荐的商品是好友购买过的，那么他们对推荐结果就会相对比较信任。

关于推荐系统信任度的研究[[9]](#_3.混淆矩阵)主要集中在评论网站Epinion的推荐系统上。这是因为Epinion创建了一套用户之间的信任系统来建立用户之间的信任关系，帮助用户判断是否信任当前用户对某一个商品的评论。

如图所示，当用户在Epinion上浏览一个商品时，他会通过用户评论判断是否购买该商品。Epinion为了防止垃圾评论或者广告评论影响用户的决策，在每条用户评论的右侧都显示了评论作者的信息，并且让用户判断是信任该评论人还是将他加入黑名单。如果网站具有Epinion的用户信任系统，那么可以在给用户做推荐时，尽量推荐他信任的其他用户评论过的物品。

### 实时性

在很多网站中，因为物品（新闻、微博等）具有很强的时效性，所以需要在物品还具有时效性时就将它们推荐给用户。比如，给用户推荐昨天的新闻显然不如给用户推荐今天的新闻。因此，在这些网站中，推荐系统的实时性就显得至关重要。

推荐系统的实时性包括两个方面。**首先，推荐系统需要实时地更新推荐列表来满足用户新的行为变化**。比如，当一个用户购买了iPhone，如果推荐系统能够立即给他推荐相关配件，那么肯定比第二天再给用户推荐相关配件更有价值。很多推荐系统都会在离线状态每天计算一次用户推荐列表，然后于在线期间将推荐列表展示给用户。这种设计显然是无法满足实时性的。与用户行为相应的实时性，可以通过**推荐列表的变化速率**来评测。如果推荐列表在用户有行为后变化不大， 或者没有变化，说明推荐系统的实时性不高。

**实时性的第二个方面是推荐系统需要能够将新加入系统的物品推荐给用户**。这主要考验了推荐系统处理物品冷启动的能力。关于如何将新加入系统的物品推荐给用户，而对于新物品推荐能力，我们可以利用用户推荐列表中有多大比例的物品是当天新加的来评测。

### 健壮性

任何一个能带来利益的算法系统都会被人攻击，这方面最典型的例子就是搜索引擎。搜索引 擎的作弊和反作弊斗争异常激烈，这是因为如果能让自己的商品成为热门搜索词的第一个搜索结果，会带来极大的商业利益。推荐系统目前也遇到了同样的作弊问题，而健壮性（即robust,[鲁棒性](#_鲁棒性)）指标衡量了一个推荐系统抗击作弊的能力。

2011年的推荐系统大会专门有一个关于推荐系统健壮性的[教程[7]](#_3.混淆矩阵)。作者总结了很多作弊方法， 其中最著名的就是行为注入攻击（profile injection attack）。众所周知，绝大部分推荐系统都是通过分析用户的行为实现推荐算法的。比如，亚马逊有一种推荐叫做“购买商品A的用户也经常购买的其他商品”。它的主要计算方法是统计购买商品A的用户购买其他商品的次数。那么，我们 可以很简单地攻击这个算法，让自己的商品在这个推荐列表中获得比较高的排名，比如可以注册很多账号，用这些账号同时购买A和自己的商品。还有一种攻击主要针对评分系统，比如豆瓣的电影评分。这种攻击很简单，就是雇用一批人给自己的商品非常高的评分，而评分行为是推荐系统依赖的重要用户行为。

**算法健壮性的评测主要利用模拟攻击**。首先，给定一个数据集和一个算法，可以用这个算法 给这个数据集中的用户生成推荐列表。然后，用常用的攻击方法向数据集中注入噪声数据，然后 利用算法在注入噪声后的数据集上再次给用户生成推荐列表。最后，通过比较攻击前后推荐列表 的相似度评测算法的健壮性。如果攻击后的推荐列表相对于攻击前没有发生大的变化，就说明算法比较健壮。

在实际系统中，提高系统的健壮性，除了选择健壮性高的算法，还有以下方法。

* 设计推荐系统时尽量使用代价比较高的用户行为。比如，如果有用户购买行为和用户浏览行为，那么主要应该使用用户购买行为，因为购买需要付费，所以攻击购买行为的代价远远大于攻击浏览行为。 
* 在使用数据前，进行攻击检测，从而对数据进行清理。

### 商业目标

很多时候，网站评测推荐系统更加注重网站的商业目标是否达成，而商业目标和网站的盈利模式是息息相关的。一般来说，最本质的商业目标就是平均一个用户给公司带来的盈利。不过这种指标不是很难计算，只是计算一次需要比较大的代价。因此，很多公司会根据自己的盈利模式设计不同的商业目标。

不同的网站具有不同的商业目标。比如电子商务网站的目标可能是销售额，基于展示广告盈利的网站其商业目标可能是广告展示总数，基于点击广告盈利的网站其商业目标可能是广告点击总数。因此，设计推荐系统时需要考虑**最终的商业目标**，而网站使用推荐系统的目的除了满足用户发现内容的需求，也需要利用推荐系统加快实现商业上的指标。

### 小结

|  |  |  |  |
| --- | --- | --- | --- |
|  | 离线实验 | 问卷调查 | 在线实验 |
| 用户满意度 | × | √ | ○ |
| 预测准确度 | √ | √ | × |
| 覆盖率 | √ | √ | √ |
| 多样性 | ○ | √ | ○ |
| 新颖性 | ○ | √ | ○ |
| 惊喜度 | × | √ | × |

对于可以离线优化的指标，我个人的看法是应该在给定覆盖率、多样性、新颖性等限制条件 下，尽量优化预测准确度。用一个数学公式表达，离线实验的优化目标是：

最大化预测准确度

使得 覆盖率 > A

多样性 > B

新颖性 > C

其中，A、B、C的取值应该视不同的应用而定。

## 评测维度

在评测系统中还需要考虑评测维度，比如一个推荐算法， 虽然整体性能不好，但可能在某种情况下性能比较好，而增加评测维度的目的就是知道一个算法 在什么情况下性能最好，这样可以为融合不同推荐算法取得最好的整体性能带来参考。

一般来说，评测维度分为如下3种。 

* **用户维度** 主要包括用户的人口统计学信息、活跃度以及是不是新用户等
* **物品维度** 包括物品的属性信息、流行度、平均分以及是不是新加入的物品等。 
* **时间维度** 包括季节，是工作日还是周末，是白天还是晚上等。

如果能够在推荐系统评测报告中包含不同维度下的系统评测指标，就能帮我们全面地了解推荐系统性能，找到一个看上去比较弱的算法的优势，发现一个看上去比较强的算法的缺点。

# 各个推荐产品介绍

## 亚马逊的个性化产品推荐

## Netflix的视频和DVD推荐

## Pandora的音乐推荐

## Facebook的好友推荐

## Google Reader的个性化阅读

## 各种个性化广告

# 附录

## 信息熵

## 误差

### 均方根误差

### 平均绝对误差

## 混淆矩阵

## 长尾效应

## 马太效应

## 鲁棒性

## 相关工具

代码高亮：<http://www.planetb.ca/syntax-highlight-word>

## 相关文献

[1] 亚马逊前科学家Greg Linden：[“What is a Good Recommendation Algorithm？”](http://cacm.acm.org/blogs/blog-cacm/22925-what-is-a-good-%20recommendation%20-algorithm/fulltext)

[2] O’scar Celma博士论文[“Music Recommendation and Discovery in the Long Tail”](http://mtg.upf.edu/static/media/PhD_ocelma.pdf)

[3] ACM的2011研讨会论文：[“International Workshop on Novelty and Diversity in Recommender Systems”](http://ir.ii.uam.es/divers2011/)

[4] Guy Shani和 Asela Gunawardana的“Evaluating Recommendation Systems”

[5] Yuan Cao Zhang、 Diarmuid Ó Séaghdha、 Daniele Quercia和 Tamas Jambor的“Auralist: introducing serendipity into music recommendation.”。

[6] Tomoko Murakami、 Koichiro. Mori和Ryohei Orihara的“ Metrics for evaluating the serendipity of recommendation lists”。

[7] Neil Hurley的“ Tutorial on Robustness of Recommender System”（ACM RecSys 2011）

[8] Henriette Cramer、Vanessa Evers、 Satyan Ramlal、 Maarten van Someren、Lloyd Rutledge、 Natalia Stash、Lora Aroyo和Bob Wielinga的“ The effects of transparency on trust in and acceptance of a content-based art recommender”。

[9] Paolo Massa和 Paolo Avesani的“Trust-aware recommender systems”。

## 准确率、精确率和召回率

例如某个班级有男生80人、女生20人，共计100人 ，目标是找出所有女生。现在某人挑选出50个人，其中20人是女生，另外还错误的把30个男生也当作女生挑选出来了。那么怎样评估他的工作？

准确率（accuracy）--- 正确分类的item数与总数之比

A=（20+50）/100 =70%

精确率（precision）--- 所有被检索到的item中，“应该被检索到”的item占的比例

P = 20/（20+30） =40%

召回率（recall） --- 所有检索到的item占所有“应该检索到的item”的比例

R = 20/(20+0) =100%