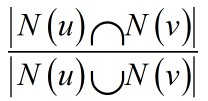
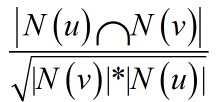
一个人想看电影的时候常常会思考要看什么电影呢。这个时候他可能会问周围爱好的人求推荐。现在社会每天都会产生海量的信息。面对这么多信息好多人都不知道什么信息是自己需要的。推荐系统正是起了这么一个作用。推荐系统的应用随处可见。网络购物是一个典型的例子，电子商务的运营商往往会根据用户在网站的行为推荐用户可能会购买的商品。豆瓣FM是做的一个非常好的电台，这个电台能够根据用户的历史行为学习出用户喜欢歌曲的类型，从而给用户推荐歌曲。除此之外还有电影推荐、新闻推荐、博客推荐等等。

　　本文主要介绍协同过滤算法。本博客以后会逐渐介绍一些其他的推荐算法。

      协同过滤算法的出现标志着推荐系统的产生。协同过滤算法包含基于用户的协同过滤算法和基于物品的协同过滤算法。

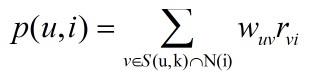
      基于用户的协同过滤算法基于这样一个事实，如果A和B在电影方面的喜好相同，那么把B喜欢的的电影推荐给A是有道理的。 根据这个事实，基于用户的协同过滤算法出现了。根据这个事实需要求出两个用户的相似度。这个相似度可以是公式1(jaccd)或者公式2(余弦公式)。 如果想计算每两个用户的相似度需要的时间复杂度为O(n\*n\*d)。n为用户数目，d为商品的数目。

                                            （公式1）

                                          （公式2）

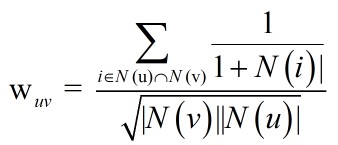
      通过公式1或者2我们能得到一个相似度矩阵。然而在很多应用中这个相似度矩阵式非常稀疏的。也就是很多用户相互之间没有对相同的商品产生行为。如果我们直接先把相似度不为0的用户对数求出来，然后只计算这些不为0的用户对，这样子会剩很多复杂度。 用数组C[u][v]记录用户u和v有相同商品行为的数目。首相建立一个倒排表。每个物品都保存被产生过行为的用户。然后对于每个物品所有的用户对数（u,v）,C[u][v]加1。这样结束以后就可以只利用相似度不为0的用户对数了。

　　得到相似度矩阵后利用公式3预测用户u对物品i的感兴趣程度。其中S(u,k)表示与用户u最接近的k个用户N(i)表示对物品i有过行为的用户集合。 ***w****uv* 表示用户u和v的相似度，***r****vi*表示用户v对物品i的感兴趣程度。

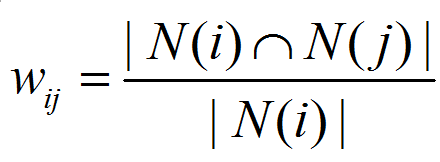
                             （公式3）

　　到此基于用户的系统过滤算法基本就介绍完了。 公式1和公式2衡量用户的用户相似度其实上是比较粗糙的。举个例子，小时候基本每个人都会买《新华字典》，其实并不能据此说明他们的兴趣相似，然而如果两个人都买了《模式识别》，那么就基本可以肯定他们的兴趣是比较相似的。《新华字典》与《模式识别》的区别在于一个是火爆的物品，一个相对不火爆。因此相似度的计算方式可以修改为公式4。

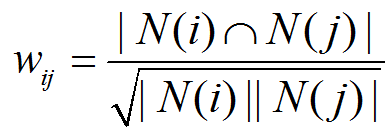
1/(1+N(i) )惩罚了火爆的物品。

                             （公式4）

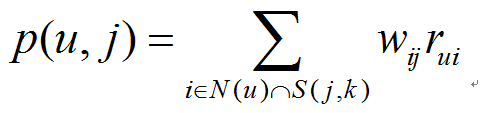
       基于物品的协同过滤算法在应用中更频繁。基于物品的协同过滤算法主要基于这样的思想： 如果用户x购买了物品A，那么她很可能会购买与A很相似的物品B（比如A是面膜，B是洗面奶）。这样的话就需要计算物品间的相似度。

                                 （公式5）

       i和j分别表示两个物品， 其中*N(i)* 表示喜欢物品i的人，*N(j)* 表示喜欢物品j的人。这个公式可以理解购买了i的用户中有多少人购买了j。 这个公式其实有一个问题： 如果物品j是一个特别热门的物品， 那么物品j跟很多物品的相似度都会很高， wij 很可能都会接近1。因此为了避免热门物品造成的这种影响， 我们把公式修改为：

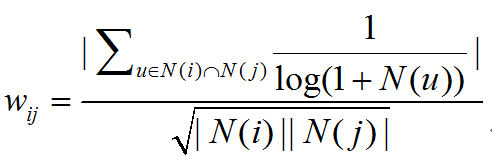
                                   （公式6）

        在得到了物品相似度后， 我们要计算用户u对物品j的兴趣程度：

                       （公式7）

        其中N(i) 是用户u喜欢的物品集合， S(j,k)是与物品j最相似的k个物品的集合。 wij是物品i和j的相似度。 rui是用户u对物品i的兴趣度。基本思想就是，如果要计算用户u与物品j的兴趣度， 先找到与j最相似的k个物品，再看用户u与这些物品的感兴趣程度，加权得到用户u对物品j的兴趣程度。

　　　这里需要提出一个问题，如果用户u是一个书商，那么在计算物品相似度的时候，这个书上对他购买书中的任何两本书计算相似的时候都会做出贡献。 其实这事不合理的。 如果他购买了《数据挖掘导论》和一本《红楼梦》，这个信息其实意义不大的， 我们并不能因此而直接地增加这两本书的相似度。 这里我们需要对用户的热度做惩罚。

               （公式8）

          公式8是对公式6的修正， 惩罚了热门的用户。

　　　  上述讨论只是用了无下文的隐性数据。 在计算物品相似度和用户相似度的时候有很多公式。实际应用中发现，基于物品的协同的过滤利用皮尔逊相关系数效果比较好。而基于物品的协同过滤利用余弦相似度效果会比较好。