**SSS网络学习平台大数据推荐算法应用**

# 一、背景简介

网络学习平台每时每刻都在产生海量数据，利用大数据分析可以对网络学习平台产生的数据进行分析、预测，为教学、学习等提供科学的趋势分析和数据支撑。由于在网络学习平台中，大数据最显著的作用在于其推荐算法，可以向学生推荐其感兴趣的课程等。因此，本文主要针对大数据推荐算法展开介绍。

# 二、推荐算法应用

个性化推荐是根据用户的兴趣特点和购买行为，向用户推荐用户感兴趣的信息和商品。随着电子商务规模的不断扩大，商品个数和种类快速增长，顾客需要花费大量的时间才能找到自己想买的商品。这种浏览大量无关的信息和产品过程无疑会使淹没在信息过载问题中的消费者不断流失。为了解决这些问题，个性化推荐系统应运而生。个性化推荐系统是建立在海量数据挖掘基础上的一种高级商务智能平台，以帮助电子商务网站为其顾客购物提供完全个性化的决策支持和信息服务。

在网络学习平台中运用推荐系统可以帮助学生选择合适的课程，全方面提升学生知识储备量，同时还能有效解决部分新开设的优秀课程无人问津的窘境。

## 协同过滤

1. **协同过滤背景**

我们知道要想了解商品、技术或某课程的推荐信息，最简单的方法莫过于向知情的朋友们询问，我们也知道这其中有一部分人的品味会比其他人高，通过观察这些人是否和我们喜欢同一类的东西，我们可以逐渐对这种情况有所了解。不过随着选择越来越多，要想通过询问一小群人来确定我们想要的东西变得越来越不切实际，因为他们可能并不了解所有的选择。这就是为什么人们要发展出一套被称为协同过滤的技术。

1. **协同过滤基本思想**

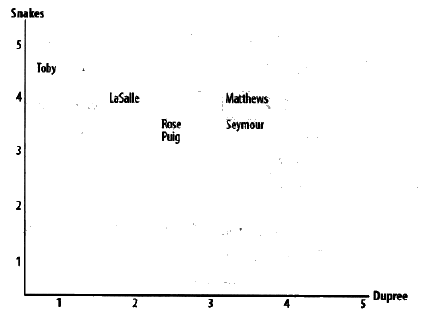
一个协同过滤算法的基本做法是对一大群人进行搜索，并从中找出与我们品味相近的一小群人。算法会对这些人所偏爱的其他内容进行考察，并将它们组合起来构造出一个经过排名的推荐列表。

* 1. 相似度判断

我们需要有一种方法来确定人们在品味方面(或物品与物品之间)的相似度。我们可以将每个人与其他所有人进行对比，并计算它们之间的相似度评价值。有多种方法可以计算相似度评价值。

(1) 欧几里德距离

计算相似度的最简单的方法就是使用欧几里德距离评价方法。它以经过人们一致评价的物品作为坐标轴，然后将参与评价的人绘制到图上，并考察他们彼此间的距离远近。



两人在偏好空间中的距离越近，他们的兴趣偏好就越相似。上例是二维空间下的情况，需要注意此方法可以推广到任意维度。

核心代码如下：

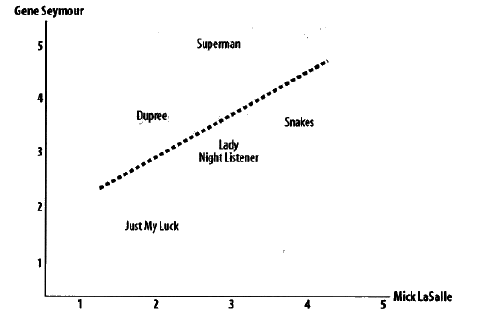
sum\_of\_squares= sum([pow(prefs[person1][item]-prefs[person2][item],2)

for item in prefs[person1] if item in prefs[person2]])

return 1/(1+sqrt(sum\_of\_squares))

(2) 皮尔逊相关度

此方法判断两组数据与某一直线的拟合程度，其计算公式比欧几里德距离评价的计算公式要复杂，它在数据不是很规范时也会给出较好的结果。



由于存在一些用户总是倾向于评分过高或过低（相对平均值），这时虽然两用户兴趣相似，但是这种情况并不能通过欧几里德距离方法计算出来。而皮尔逊相关度算法则修正了夸大分值的情况，即使某人倾向于给出更高的评分，但是因为两者之间有着相对近似的偏好，所以最终得到的直线仍然是拟合的。如果两者的评分差值始终保持一致，那么他们依然可能会存在很好的相关性。

核心代码如下：

# 所有偏好之和

sum1 = sum([prefer[person1][item] for item in sim]) #1.sum([1,4,5,,,]) 2.list的灵活生成方式!(Python灵活)

sum2 = sum([prefer[person2][item] for item in sim]) #!!!!写成person1，导致程序一直runtime error!!!!

#求平方和

sum1Sq = sum( [pow(prefer[person1][item] ,2) for item in sim] )

sum2Sq = sum( [pow(prefer[person2][item] ,2) for item in sim] )

#求乘积之和 ∑XiYi

sumMulti = sum([prefer[person1][item]\*prefer[person2][item] for item in sim])

#Pearson系数计算 http://baike.baidu.com/view/3891263.htm，计算错误

#num1 = n\*sumMulti - sum1\*sum2 #分子

#num2 = sqrt(n\*sum1Sq-pow(sum1,2))\*sqrt(n\*sum2Sq-pow(sum2,2))

num1 = sumMulti - (sum1\*sum2/n)

num2 = sqrt( (sum1Sq-pow(sum1,2)/n)\*(sum2Sq-pow(sum2,2)/n))

if num2==0:

return 0

return num1/num2

* 1. 算法应用实例

测试数据集为MovieLens-1M。

代码详见MLRecommendationTest.py， Recommendation.py，Simlarity.py

运行结果如下图：



程序求得了目标用户与其他用户之间的相似度矩阵，然后为目标用户给出推荐电影列表并按预计评分从高到低排序。