电影推荐系统分析报告

王凯

2020 年 *5* 月

1 前言

如今，智能推荐服务在日常生活中无处不在，这给我们的生活带来了极大的便利，而要产生这些推荐服务，最重要的就是“数据”。4G 和 5G 等网络技术的快速发展与普及，给互联网行业注入了源源不断地新的活力。对于互联网而言，它们最为重要的特点就是数据，通过数据来驱动工作，然后将其反馈给用户，在这一过程中，用户既是数据的生产者，同时又是数据的消费者。随着网络技术的不断发展和普及，互联网用户的数量不断攀升，随之产生的数据也不断增加。并且在 2010 年以后，智能手机的快速普及，使得人们上网更加方便也更加普遍，这更使得互联网信息呈现出了爆炸增长的态势。根据有关部门资料显示，近些年在互联网产生的数据占到了所有互联数据的 80% 以上，而且其还正在高速增长，并没有出现减缓的迹象，而我们也真正迎来了“大数据时代”。这爆炸似的信息增长给互联网行业带来了前所未有的机遇，通过挖掘数据中的信息并对其进行处理，可以将之服务于用户，给用户带来更好的体验。然而这海量的数据也给数据处理技术提出了严峻的挑战，其最大的挑战就是“信息过载”问题，即用户没有办法从大量的信息中获取到自己需要的信息的问题。为了解决这个问题，信息检索技术和信息过滤技术应运而生。

目前，最好的信息检索技术以搜索引擎为代表，用户只需要输入关键词就可以检索出与之相匹配的信息，谷歌、必应和百度等互联网公司就凭借其强大而全面的搜索引擎获得了极大的成功。然而搜索引擎并不能解决所有问题，当用户自身都无法对自己需求做出描述时，搜索引擎强大的搜索能力就无用武之地了。这时候以推荐系统为代表的信息过滤技术就可以很好的解决这个问题，信息过滤技术并不需要用户提供关键词用以搜索，它只需要对海量的用户行为进行挖掘分析，得到用户可能会感兴趣的需求，然后主动将其推送给用户。本文为了实现电影的智能推荐服务，选用基于协同过滤算法对网民的观影评分数据进行了分析，建立电影推荐模型。

2 模型介绍

2.1 推荐系统

推荐系统是通过分析用户的历史行为，例如兴趣特点、购买行为，然后根据其历史行为给他推荐他可能会产生兴趣的项目。在一般的推荐系统中，“项目”被定义给用户推荐的物品。推荐系统的主要作用是在用户无法描述其自身的需求或者无法从海量信息中找寻到满足自己要求的信息时，主动为用户提供参考意见的信息过滤手段。例如我们在淘宝购物时会有商品推荐、在今日头条阅读时会有新闻推荐以及在抖音看视频时会有视频推荐等等，这些推荐的内容都是根据用户的

浏览记录由推荐系统生成，并主动推荐给用户的。然而对于每个用户而言，其兴趣爱好是不尽相同的，因此其收到的推荐结果往往是个性化的不同结果。在众多的网络应用中，它们的推荐结果大多以一个排序表的方式呈现，其排序顺序是由推荐系统预测出来的用户可能对该项目感兴趣的程度（以相似度来衡量）决定的。如果要预测用户对某个项目的感兴趣程度，我们首先需要收集用户的相关信息以及他们的历史偏好，最终进行推荐。在电子商务越来越发达的今天，商品货物变得越来越琳琅满目，对于大部分用户而言，需要从大量商品中寻找到最适合自己的商品变得愈发困难，这使推荐系统的出现将会极大缓解这个问题，在最大程度上给出推荐意见等。近些年来，推荐系统成功地应用在各行各业当中，而这也表明它在解决信息过载问题有其不可比拟的优势；而且如果推荐系统给用户带来了高品质的推荐服务，这将极大增强顾客的粘合度，并获得顾客的信任感。而顾客的信任反过来又可以丰富和扩展数据库的内容，从而可以将以后的推荐内容精准化，使之形成一个良性循环。

2.2 协同过滤算法

“协同过滤” 算法最开始是在 90 年代中期开发推荐系统时被提出来的，它是一种基于邻域的算法，它通过集体的智慧做出最优的推荐。常见的协同过滤算法有两种，一是基于用户的协同过滤；二是基于物品的协同过滤。

2.2.1 基于用户的协同过滤

基本思想：首先计算用户之间的相似度，然后根据用户之间的相似度给用户推荐那些和他有共同兴趣爱好的用户所喜欢的物品。

2.2.2 基于物品的协同过滤

基本思想：计算物品之间的相似度，然后根据物品的相似度和用户的历史行为给用户生成推荐列表。

2.3 相似度计算

2.3.1 夹角余弦

$$\mathrm{sim}_{lm} = \frac{\sum_{k=1}^{n} x_{k1} x_{km}}{\sqrt{\sum_{k=1}^{n} x_{k1}^2}}$$

夹角余弦相似度的取值范围为 [-1,1]，当余弦值的绝对值越接近于 1 时，表明相似度程度越高；越接近于 0 时，相似程度越低。

2.3.2 杰卡德相似系数

$$J\left(A_1, A_m\right) = \frac{\left|A_1 \cap A_m\right|}{\left|A_1 \cup A_m\right|}$$

杰卡德相似系数主要用于计算分类数据的相似度，其中$A_1 \cup A_m$ 表示喜欢物品 1 与喜欢物品 M 的用户总数，$A_1 \cap A_m$ 表示同时喜欢物品 1 和物品 M 的用户数，它的取值范围为 [0,1]。

2.3.3 相关系数

$$\mathrm{sim}_{lm} = \frac{\sum_{k=1}^{n}\left(x_{k1}-A_1\right)\left(x_{km}-A_m\right)}{\sqrt{\sum_{k=1}^{n}\left(x_{k1}-\overline{A_1}\right)^2}\sqrt{\sum_{k=1}^{n}\left(x_{km}-\overline{A_m}\right)^2}}$$

相关系数的取值也为 [-1,1]，相关系数的绝对值越大，表明相似度越高。

2.4 推荐系统建模的流程和步骤

1. 从业务系统中获得原始数据
2. 通过数据抽取技术获得用户访问日志
3. 对数据进行探索分析和预处理（数据去重、数据变换以及数据分类）
4. 将清洗好的数据进行建模，常用的推荐算法有协同过滤推荐、热点推荐以及随机推荐等
5. 模型建立完毕，进行模型评价，如果模型满足要求，则可以进行测试；如果模型不满足要求，则需要进行模型优化和重构

3 构建智能推荐系统

3.1 数据来源及介绍

为了实现电影的智能推荐服务，本案例选用了电影评分数据集，该数据集来自 IMDB 数据库，包含了 99415 条记录，其中详细记录了 943 名用户对 1665 部电影的评分情况，分值都在 1-5 分之间。

3.2 数据探索性分析和预处理

首先我们先预览一下数据的大致结构，通过 R 代码来实现：

```
#读入数据
data <- read.csv("D:/Desktop/dataMovieLense.csv")
head(data)
```

```
##   user            movie score
## 1    1 Toy.Story..1995.     5
## 2    2 Toy.Story..1995.     4
## 3    5 Toy.Story..1995.     4
## 4    6 Toy.Story..1995.     4
## 5   10 Toy.Story..1995.     4
## 6   13 Toy.Story..1995.     3
```

通过查看原始数据的前六行我们发现，第一列为用户的 ID，第二列为电影的名称，第三列为用户对该电影的评分情况。接下来我们查看一下电影评分的分布情况，通过 R 代码来实现：

```
summary(data$score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    1.00    3.00    4.00    3.53    4.00   12.00       7
```

```
table(data$score)
```

```
##
##     1     2     3     4     5     6     7     8     9    11    12
##  6059 11308 27003 33947 21077     2     3     4     3     1     1
```

通过分析我们发现，电影评分的分值并不是都在 1-5 分之间，还存在一些异常值和缺失值，故接下来我们要对异常值和缺失值进行清理。

删除异常值和缺失值，通过 R 代码来实现：

```
#直接删除存在缺失值的样本
newdata <- na.omit(data)
# 得到异常值的样本下标索引，然后将其删除
index1 <- newdata$score>5
newdata <- newdata[-which(index1),]
```

经过上述的操作之后，我们得到了清洗好、符合建模要求的初始数据，故接下来我们需要进一步的操作。

3.3 数据结构转换

在建立推荐系统模型时，我们需要计算用户之间的相似度和物品之间的相似度，故我们需要将目前的数据格式转换为用户 -电影评分矩阵，通过 R 代码来实现：

```
library(reshape)
newdata <- cast(newdata,user~movie,value='score')
# 查看目前的数据格式类型
class(newdata)
```

```
## [1] "cast_df"    "data.frame"
```

```
#只保留格式dataframe
class(newdata) <- 'data.frame'
#将数据转化为矩阵
newdata <- as.matrix(newdata)
#转化为用户电影评分矩阵-
library(recommenderlab)
newdata <- as(newdata,"realRatingMatrix")
newdata
```

```
## 943 x 1665 rating matrix of class 'realRatingMatrix' with 100335
    ratings.
```

```
# 查看第二号用户的前两部电影评分情况
as(newdata,'list')[[2]][1:3]
```

```
##                  user Absolute.Power..1997.  Air.Force.One..1997.
##                     2                       3                     4
```

通过分析结果可知：转化后的数据变成了 943 名用户对 1665 部电影评分的矩阵，接下来我们将根据这个评分矩阵建立推荐系统模型。

3.4 划分数据集

在正式建立推荐系统之前，我们需要对数据集进行划分，其目的是为了方便后续进行模型评价。由于该数据集较大，故我们直接使用留出法划分数据，其中 75% 作为训练集，25% 作为测试集，通过 R 代码来实现：

```
# 划分数据集
model_eval <- evaluationScheme(newdata,method='split',train=0.75,given
    =15,goodRating=5)
print(model_eval)
```

```
## Evaluation scheme with 15 items given
## Method: 'split' with 1 run(s).
## Training set proportion: 0.750
## Good ratings: >=5.000000
## Data set: 943 x 1665 rating matrix of class 'realRatingMatrix' with
    100335 ratings.
```

3.5 建立推荐系统模型

为了实现更好的电影推荐服务，本案例构建了四个不同的推荐模型，分别是随机推荐算法、流行推荐算法、基于物品的协同过滤算法以及基于用户的协同过滤算法。其中随机推荐是指随机给用户推荐物品，流行推荐是将流行的物品推荐给用户，当然推荐的物品都有一个前提条件，那就是物品在推荐之前用户并不了解。接下来，我们通过 R 代码来实现这四个推荐模型（相似度默认为夹角余弦相似度）。

```
# 建立随机推荐模型：model_random
model_random <- Recommender(getData(model_eval,'train'),method='RANDOM')
# 建立流行推荐模型：model_popular
model_popular<-Recommender(getData(model_eval,'train'),method='POPULAR')
# 建立基于物品的协同过滤推荐模型：model_ibcf
model_ibcf <- Recommender(getData(model_eval,'train'),method='IBCF')
# 建立基于用户的协同过滤推荐模型：model_ubcf
model_ubcf <- Recommender(getData(model_eval,'train'),method='UBCF')
```

3.6 模型预测及评价

模型的建立往往是建模的开始，而不是结束。在建立模型之后，我们需要利用测试数据对模型进行评价，然后根据评价结果确定模型是否符合要求。如果有多个模型时，那么我们要从中选择一个评价结果最优的模型。由于本案例是数值型数据，那么我们采用 RMSE、MSE 和 MAE 指标来进行评价，通过 R 代码来实现。

```
# 根据每个模型预测测试集的电影评分情况
predict_random <- predict(model_random,getData(model_eval,'known'),type=
    'ratings')
predict_popular <- predict(model_popular,getData(model_eval,'known'),
    type='ratings')
predict_ibcf <- predict(model_ibcf,getData(model_eval,'known'),type='
    ratings')
predict_ubcf <- predict(model_ubcf,getData(model_eval,'known'),type='
    ratings')
```

```
# 计算预测误差
error <- rbind(calcPredictionAccuracy(predict_random,getData(model_eval,'unknown')),
          calcPredictionAccuracy(predict_popular,getData(model_eval,'unknown')),
          calcPredictionAccuracy(predict_ibcf,getData(model_eval, 'unknown')),
          calcPredictionAccuracy(predict_ubcf,getData(model_eval, 'unknown')))
rownames(error) <- c('Random','Popular','IBCF','UBCF')
error
```

```
##           RMSE      MSE    MAE
## Random  60.145 3617.394 11.944
## Popular 26.305  691.948  8.344
## IBCF     1.646    2.711  1.245
## UBCF    27.267  743.486  6.504
```

通过对模型评价结果进行分析，我们发现如果采用 RMSE 和 MSE 指标进行模型评价的话，最优的推荐模型是 Popular 推荐算法；但是如果采用 MAE 指标进行评价的话，那么最优的推荐模型是基于物品的协同过滤算法。然而为了更好的保证模型的稳定性，我们综合 RMSE、MSE 和 MAE 这三个指标进行评价，故认为基于用户的协同过滤算法（UBCF）在本案例中表现更为优异。

3.7 分析模型结果

通过基于用户的协同过滤算法构建的推荐系统，我们得到了第 901-903 号用户可能感兴趣的前三个电影，通过 R 代码来实现。

```
movies_predict <- predict(model_ubcf,newdata[901:903],n=3)
as(movies_predict,'list')
```

```
## $`901`
## [1] "A.Chef.in.Love..1996."              "A.koldum.klaka..Cold.Fever...1994."
## [3] "Above.the.Rim..1994."
##
## $`902`
## [1] "A.Chef.in.Love..1996."              "A.koldum.klaka..Cold.Fever...1994."
## [3] "Above.the.Rim..1994."
##
## $`903`
## [1] "A.Chef.in.Love..1996."              "A.koldum.klaka..Cold.Fever...1994."
## [3] "Above.the.Rim..1994."
```