SHUANGLN

工作汇报

屠晓涵

2016.10.12

本周工作

1，明确选题背景

2，构建基于矩阵分解的推荐引擎

——提取有效特征

——训练推荐模型

——使用推荐模型

——对用户推荐

选题背景

- 传统的协同过滤推荐算法缺点
  - 基于历史数据，对新物品和新用户都有“冷启动”的问题。
  - 不够灵活，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
  - 对于一些特殊品味的用户不能给予很好的推荐。
- 基于内容的推荐算法缺点
  - 物品相似度的衡量标准只考虑到了物品本身，有一定的片面性
  - 物品的属性有限，很难有效的得到更多数据
- 研究目标
  - 提高推荐准确度
  - 缓解数据稀疏性
  - 解决冷启动问题

  → 混合推荐——多种推荐技术进行混合，相互弥补缺点

构建基于矩阵分解推荐引擎—SVD算法原理

基于SVD 的推荐：根据已有的评分情况，分析电影包含各个因子的程度和用户对于各个因子的喜爱程度，再反过来分析数据得到预测结果。

SVD原理：

$$\hat{r}_{ui} = \mu + bi + bu + q_i^T p_u$$

为了防止过拟合，加入参数，得到优化函数：

$$\min \sum (r_{ui} - \mu + bi + bu + q_i^T p_u)^2 + \lambda(b_i^2 + b_u^2 + \|q_i\|^2 + \|p_u\|^2)$$

效果评测：

$$\mathrm{RMSE}(P, A) = \sqrt{\frac{\sum_{i=1}^{n}\sum_{j=1}^{m} J_{ij}(A_{ij} - P_{ij})^2}{\sum_{i=1}^{n}\sum_{j=1}^{m} J_{ij}}}$$

构建基于矩阵分解推荐引擎

实验数据集：

电影评分数据集：movielens，有1k,1m,10m,20m数据量

http://www.grouplens.org/datasets/movielens

1）用户数据文件 （6040个用户）每个用户至少有20个评分

用户ID::性别::年龄::职业编号::邮编

```
[root@centos ml-1m]# tail users.dat
6031::F::18::0::45123
6032::M::45::7::55108
6033::M::50::13::78232
6034::M::25::14::94117
6035::F::25::1::78734
6036::F::25::15::32603
6037::F::45::1::76006
6038::F::56::1::14706
6039::F::45::0::01060
6040::M::25::6::11106
```

2）电影数据文件 （3883行记录）3706个电影

电影ID::电影名称::电影种类

```
[root@centos ml-1m]# tail movies.dat
3943::Bamboozled (2000)::Comedy
3944::Bootmen (2000)::Comedy|Drama
3945::Digimon: The Movie (2000)::Adventure|Animation|Children's
3946::Get Carter (2000)::Action|Drama|Thriller
3947::Get Carter (1971)::Thriller
3948::Meet the Parents (2000)::Comedy
3949::Requiem for a Dream (2000)::Drama
3950::Tigerland (2000)::Drama
3951::Two Family House (2000)::Drama
3952::Contender, The (2000)::Drama|Thriller
```

3）评

用户

```
[root@centos ml-1m]# tail ratings.dat
6040::2022::5::956716207
6040::2028::5::956704519
6040::1080::4::957717322
6040::1089::4::956704996
6040::1090::3::956715518
6040::1091::1::956716541
6040::1094::5::956704887
6040::562::5::956704746
6040::1096::4::956715648
6040::1097::4::956715569
```

构建基于矩阵分解推荐引擎

提取有效特征：

- 加载数据

```
val rawData = sc.textFile("/movielens/ml-100k/u.data")
```

- 抽取 用户id,电影id,评分　3种数据：

```
val rawRatings = rawData.map(_.split("\t").take(3))
```

- 转换为 Rating对象

```
val ratings = rawRatings.map { case Array(user, movie, rating) =>
    Rating(user.toInt, movie.toInt, rating.toDouble) }
```

```
res5: org.apache.spark.mllib.recommendation.Rating = Rating(196,242,3.0)
```

构建基于矩阵分解推荐引擎

训练推荐模型：

- val model = SVD.train(ratings, 50, 10, 0.01)
- model.userFeatures
- model.userFeatures.count

由于spark操作数延迟性的转化操作。只有当物品因子或用户因子RDD调用了执行操作时，计算才发生。

使用推荐模型：

预测：通常有2种，为某用户推荐物品，或找出与某物品相关或相似的其他物品

预测单个用户对单个电影的评分

val predictedRating = model.predict(789, 123)

构建基于矩阵分解推荐引擎

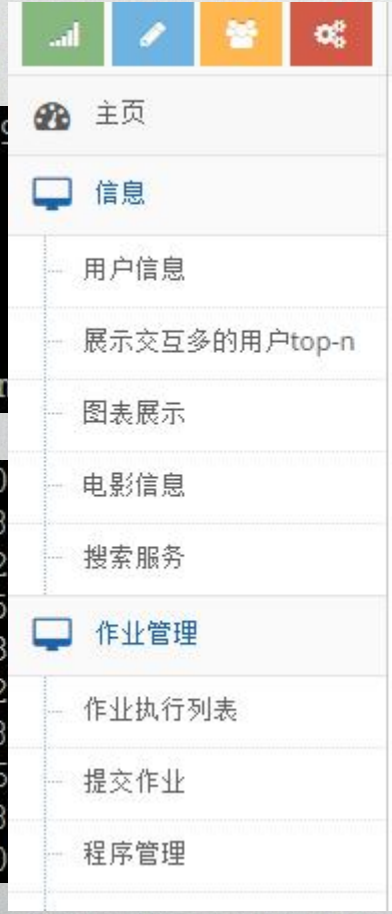
为推荐推荐电影：

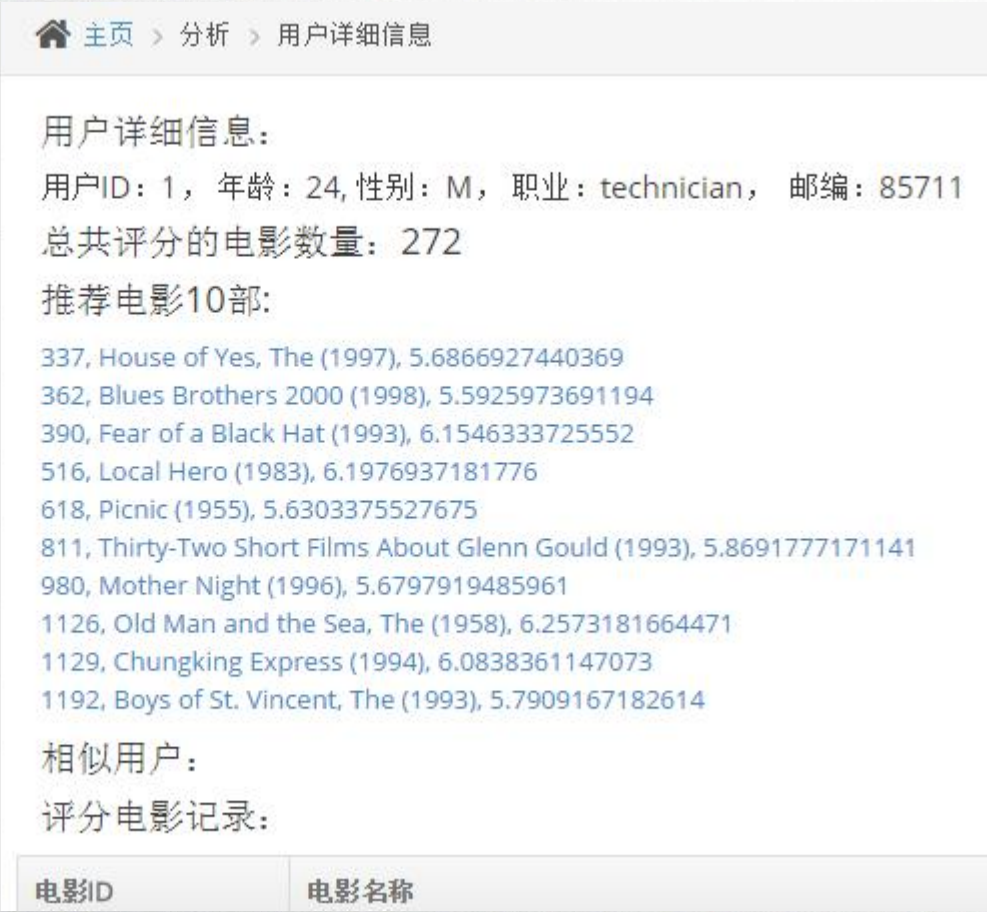
```
scala> val userId = 789
userId: Int = 789

scala> val K = 10
K: Int = 10

scala> val topKRecs = m
```

```
Rating(789,56,5.793640
Rating(789,1098,5.3348
Rating(789,789,5.30942
Rating(789,614,5.21985
Rating(789,12,5.196173
Rating(789,42,5.194872
Rating(789,708,5.09558
Rating(789,187,5.07636
Rating(789,272,5.07128
Rating(789,129,5.02280
```





构建基于矩阵分解推荐引擎

推荐效果评价指标：

（1）用户满意度

指从用户角度来看，对结果的认可程度。测评有两种方法,一是进行用户调查，二是在线评测

（2）预测准确度

- 评分预测：均方根误差RMSE、平均绝对误差MAE
- TopN推荐：准确率/召回率

```
import org.apache.spark.mllib.evaluation.RegressionMetrics
val predictedAndTrue = ratingsAndPredictions.map { case ((user, product), (actual, predicted)) => (actual, predicted)
val regressionMetrics = new RegressionMetrics(predictedAndTrue)
println("Mean Squared Error = " + regressionMetrics.meanSquaredError)
println("Root Mean Squared Error = " + regressionMetrics.rootMeanSquaredError)
// Mean Squared Error = 0.08231947642632852
// Root Mean Squared Error = 0.2869137090247319
```

下周任务

下周任务： 算法改进

相关功能实现

SHUANGLN

2016

谢谢您的观看