电影评分预测

一.任务说明

1.预测用户对某一部电影的评分情况

2.实现手段: 朴素贝叶斯算法，协同过滤算法，聚类，矩阵分解

3.使用电影评分数据集 `movieslens`

二.数据集说明

2.1 100k数据集

使用100k大小的 `movieslens` 数据集ml-100k.zip , 该数据集包含943个用户对1682部电影的10万个评分数据。每个用户至少进行了20次评分。

1.原始数据集

- u.data: 格式为 user id | item id | rating | timestamp
- u.user: 格式为 user id | age | gender | occupation | zip code
- u.item: 格式为 movie id | movie title | release date | video release date | IMDb URL | unknown | Action | Adventure | Animation | Children's | Comedy | Crime | Documentary | Drama | Fantasy | Film-Noir | Horror | Musical | Mystery | Romance | Sci-Fi | Thriller | War | Western |

2.各属性取值范围

- 评分

  `user_id` 1~943, `item_id` 1~1682, `rating` 1~5

- 用户

  `user_id` 1~943, `age` 10~70, `gender` F/M

  `occupation` 字符串 {administrator，artist，doctor,，educator，engineer，entertainment，executive，healthcare，homemaker，lawyer，librarian，marketing，none，other，programmer，retired，salesman，scientist，student，technician，writer}

- 电影

  `item_id` 1~1682

  `类型` 通过独热码表示，一部电影可属于多个类型

3.划分后的数据集

- 为进行5折交叉检验的划分

  -将评分文件`u.data`划分为训练集`u1.base`和测试集`u1.test`，划分比例为8:2

  -这样的操作进行5次，总共得到5组(训练集+测试集)

  -这样划分是为了便于进行5折交叉检验

- 均匀划分

  -将评分文件`u.data`划分为训练集`ua.base`和测试集`ub.test`，其中对于每个文件，每个用户的评分条目恰好为10条

  -这样的操作进行2次，总共得到2组(训练集+测试集)

2.2 1m数据集

该数据集包含6040个用户对3952部电影的1,000,209个评分数据。每个用户至少进行了20次评分。

1.原始数据集

- ratings.dat: 格式为 UserID::MovieID::Rating::Timestamp
- users.dat: 格式为 UserID::Gender::Age::Occupation::Zip-code
- movies.dat: 格式为 MovieID::Title::Genres

2.各属性取值范围

- 评分

  `UserID` 1~6040, `MovieID` 1~3952, `Rating` 1~5

- 用户

  `UserID` 1~6040, `Gender` F/M,

  `Age`

  - 1: "Under 18"
  - 18: "18-24"
  - 25: "25-34"
  - 35: "35-44"
  - 45: "45-49"
  - 50: "50-55"
  - 56: "56+

  `Occupation`

  - 0: "other" or not specified
  - 1: "academic/educator"
  - 2: "artist"
  - 3: "clerical/admin"
  - 4: "college/grad student"
  - 5: "customer service"
  - 6: "doctor/health care"
  - 7: "executive/managerial"
  - 8: "farmer"
  - 9: "homemaker"
  - 10: "K-12 student"
  - 11: "lawyer"
  - 12: "programmer"
  - 13: "retired"
  - 14: "sales/marketing"
  - 15: "scientist"
  - 16: "self-employed"

    - 17: "technician/engineer"
    - 18: "tradesman/craftsman"
    - 19: "unemployed"
    - 20: "writer"
- 电影 2,3,6 6,14,16 2,13 2,4 2

  `MovieID` 1~3952

  `Genres` 字符串

    - Action
    - Adventure
    - Animation 3
    - Children's 6
    - Comedy 2
    - Crime
    - Documentary
    - Drama 4
    - Fantasy
    - Film-Noir
    - Horror
    - Musical
    - Mystery
    - Romance 13
    - Sci-Fi
    - Thriller
    - War
    - Western

三.评测标准说明

Ω为测试集，y_{ij}为用户i对物品j的实际评分，y_{ij}*为预测评分

- RMSE (均方根误差)

$$RMSE = \sqrt{\frac{\sum_{\Omega}(y_{ij} - y_{ij}^*)^2}{\Omega}} RMSE = \sqrt{\frac{\sum_{\Omega}(y_{ij} - y_{ij}^*)^2}{\Omega}}$$

- MAE (平方绝对误差)

$$MAE = \frac{\sum_{\Omega}|y_{ij} - y_{ij}^*|}{\Omega}$$

本项目中以RMSE为标准进行评测

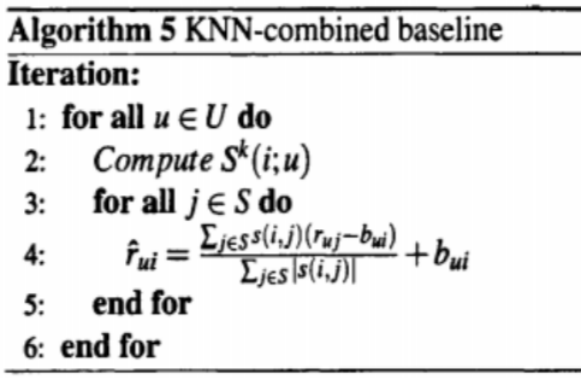
四.基于领域协同过滤

基于领域的协同过滤又可分为 `基于用户的协同过滤` 和 `基于项目的协同过滤`

4.1 基于用户的协同过滤(User-Based)

User-Based协同过滤原理: 如果用户A和用户B对一些项的评分相似，则他们对其他项的评分也相似。

1.算法说明

- 使用基于baseline的KNN
- 
  Algorithm 5 KNN-combined baseline
  Iteration:
  1: **for all** $u \in U$ **do**
  2: *Compute* $S^k(i;u)$
  3: **for all** $j \in S$ **do**
  4: $\hat{r}_{ui} = \frac{\sum_{j \in S} s(i,j)(r_{uj} - b_{ui})}{\sum_{j \in S} |s(i,j)|} + b_{ui}$
  5: **end for**
  6: **end for**
- 本课设中k取全体邻居
- 相似性度量:余弦相似度

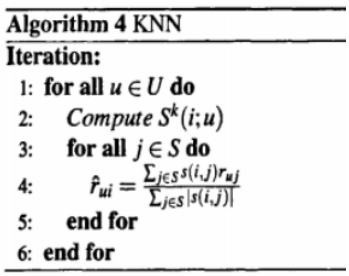
$$\rho(i,j) = \cos(\vec{i}, \vec{j}) = \frac{\vec{i} \cdot \vec{j}}{\|\vec{i}\|_2 * \|\vec{j}\|_2} \qquad \text{(Cosine-based Similarity)}$$

2.评测结果

train:test	RMSE
6:4	3.35
7:3	3.29
8:2	3.23

4.2 基于项目的协同过滤

1.算法说明

- 使用朴素KNN
- 
  Algorithm 4 KNN
  Iteration:
  1: **for all** $u \in U$ **do**
  2: *Compute* $S^k(i;u)$
  3: **for all** $j \in S$ **do**
  4: $\hat{r}_{ui} = \frac{\sum_{j \in S} s(i,j) r_{uj}}{\sum_{j \in S} |s(i,j)|}$
  5: **end for**
  6: **end for**
- 本课设中k取全体邻居

2.评测结果

train:test	RMSE
6:4	3.58
7:3	3.55
8:2	3.53

4.3 总结

-基于领域的协同过滤存在矩阵稀疏问题和计算资源有限导致的扩展性问题，基于模型的协同过滤一定程度上可以解决这些问题。

-基于内存的CF的缺点是，它不能扩展到真实世界的场景，并且没有解决众所周知的冷启动问题，也就是当新用户或新产品进入系统时。基于模型的CF方法是可扩展的，并且可以比基于内存的模型处理更高的稀疏度，但当没有任何评分的用户或产品进入系统时，预测效果会非常差

4.4 可以改进的方案

- 相似度评价添加性别
- 相似度评价添加职业
- 相似度评价添加电影类型
- 相似度评价添加电影年份

五.基于模型协同过滤

5.1 基于卷积神经网络

1.数据预处理

- `age`: 7个年龄段 {<18, 18-24, 25-34, 35-44, 45-49, 50-55, 56+ }, 分别用数字0~6表示
- `gender`: F/M用0/1表示
- `genres`: 每个电影种类用一个数字表示，比如喜剧用2表示。因为共有18个种类，故对于一部具体的电影，用一个18维的向量来表示它的种类。如果一部电影是戏剧，则用18维向量（2, Y, Y, ..., Y）来表示它的种类。Y是用来填充维度的数字
- `title`: 处理方法同`genres`, 每部电影的名字用一个15维向量表示。

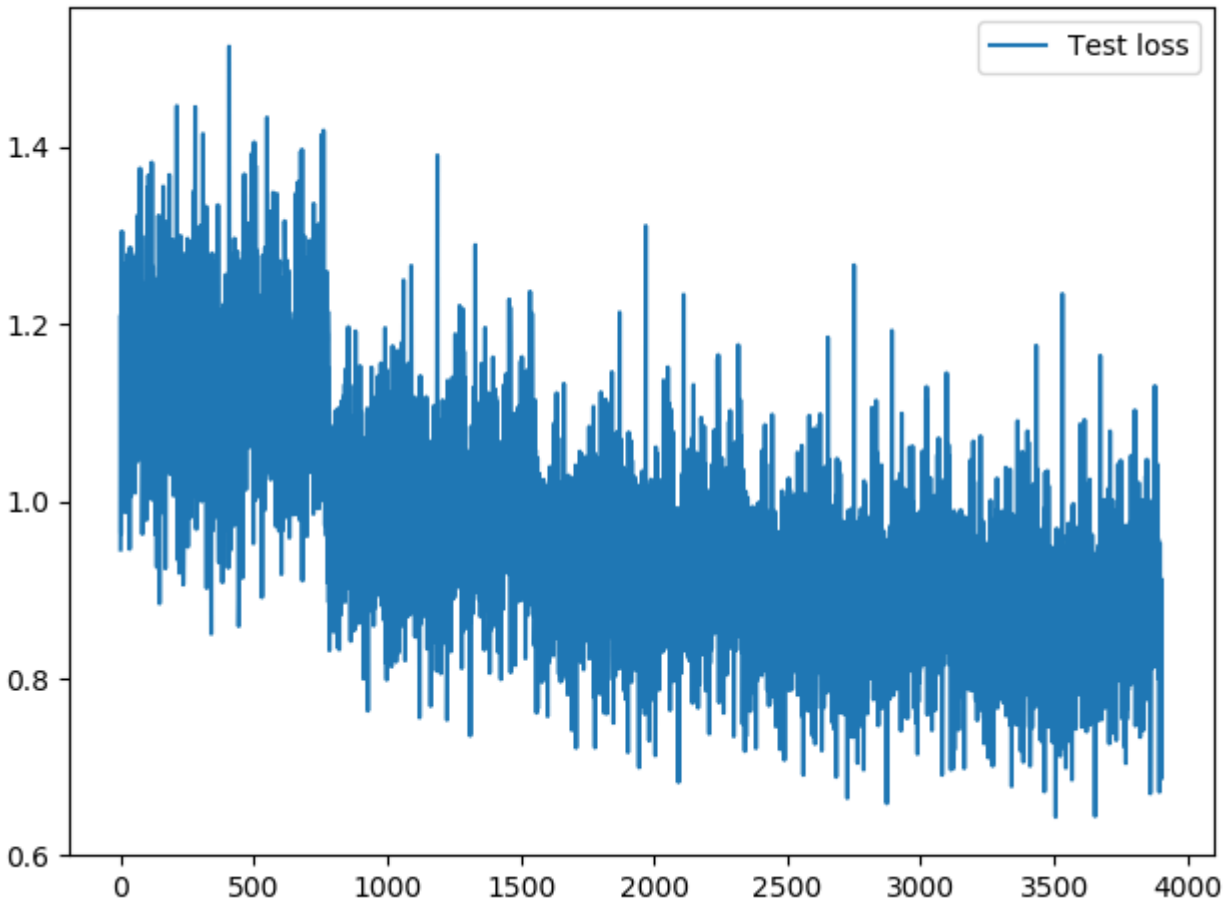
预处理后得到如下数据:

- title_count：Title字段的长度（15）
- title_set：Title文本的集合
- genres2int：电影类型转数字的字典
- features：是输入X
- targets_values：是学习目标y
- ratings：评分数据集的Pandas对象
- users：用户数据集的Pandas对象

- movies：电影数据的Pandas对象
- data：三个数据集组合在一起的Pandas对象
- movies_orig：没有做数据处理的原始电影数据
- users_orig：没有做数据处理的原始用户数据

**2.算法说明

3.评测结果



4.总结

- CNN已经成为众多科学领域的研究热点之一，特别是在模式分类领域，由于该网络避免了对图像的复杂前期预处理，可以直接输入原始图像，因而得到了更为广泛的应用
- 部分字段是类型性变量，特征工程上可以采用 `one-hot` 编码，但是对于 UserID、MovieID 这样非常稀疏的变量，如果使用 `one-hot`，那么数据的维度会急剧膨胀，对于这份数据集来说是不合适的
- CNN不足：由于一个新的用户在刚开始的时候并没有任何行为记录，所以系统会出现冷启动的问题；由于神经网络是一个黑盒子过程，我们并不清楚在反向传播的过程中的具体细节，也不知道每一个卷积层抽取的特征细节，所以此算法缺乏一定的可解释性；一般来说，在工业界，用户的数据量是海量的，而卷积神经网络又要耗费大量的计算资源，所以进行集群计算是非常重要的。但是由于本课程所做实验环境有限，还是在单机上运行，所以后期可以考虑在服务器集群上全量跑数据，这样获得的结果也更准确。

5.2 基于矩阵分解

基于模型的协同过滤技术中尤为矩阵分解(MF)技术最为普遍和流行，因为它的可扩展性极好并且易于实现。原理是通过一定数量的因子来描述各个用户的喜好和各个物品的属性。

1.算法介绍

- 构建用户-物品的评分矩阵$R_{m,n}$
- 对R作矩阵分解$R_{m,n} = P_{m,k} X Q_{k,n}$,k为待调节的参数, 通常为10-100
- 定义损失函数，评价分解的好坏
- 确定优化算法, 比如梯度下降。迭代计算得到P,Q矩阵
- 得到预测值 P

$$P_{ui} = \sum_{k} P_{uk} * P_{ki}$$

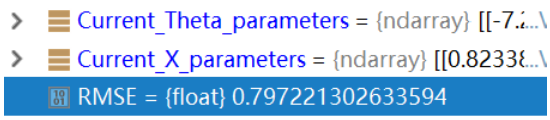
2.实现说明

- 数据预处理得到(user_id, movie_id, rating)三元组, 构建2个评分矩阵(训练和测试), 行号为user_id, 列号为movie_id, 矩阵内容为评分数值。并进行规范化, 算出均值, 评分在原评分基础上减去均值
- 设定维度k=10，随机初始化矩阵P 和 Q
- 调用tensorflow的 `optimizer = tf.train.AdamOptimizer()` 优化方法，迭代3000次得到收敛结果
- 得到预测值并进行评估

3.总结

- 矩阵分解可扩展性好, 可与其他算法做结合
- 矩阵分解是协同过滤模型中经典的方法，性能十分优良。但存在数据稀疏与冷启动问题。结合外部丰富的信息可以缓解上述问题

4.评测结果



六.扩展

- 16年一篇文章将矩阵分解(MF)与卷积神经网络(CNN)做了结合

算法	RMSE
user-based	3.35
item-based	3.53
CNN	0.87
MF	0.80

【ref】

- 基于用户/项目的协同过滤
- 卷积神经网络
- 基于矩阵分解的推荐系统