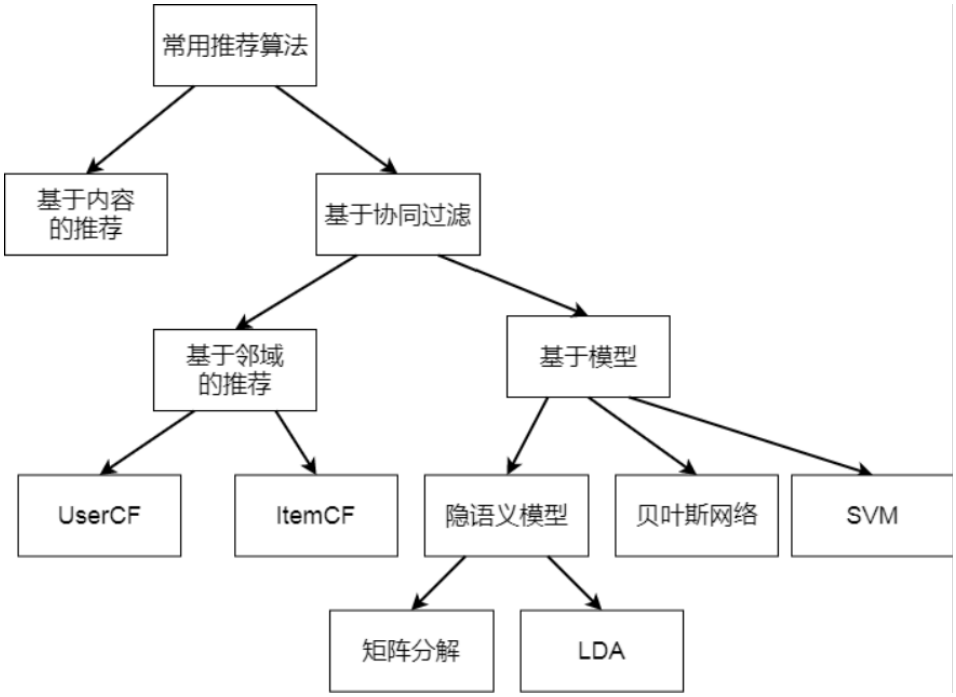
第五节课笔记 Chapter 3.1 矩阵分解ALS

https://github.com/cystanford/Recommended_System 老师的GitHub

第五节课笔记		
课堂笔记	第二次反馈	第三次反馈

讲义内容

- 推荐系统的算法都有哪些



隐语义模型（LFM, Latent Factor Model）：

- 矩阵分解（MF）
- LDA, LSA, pLSA

基于模型与基于邻域的推荐之间的区别：

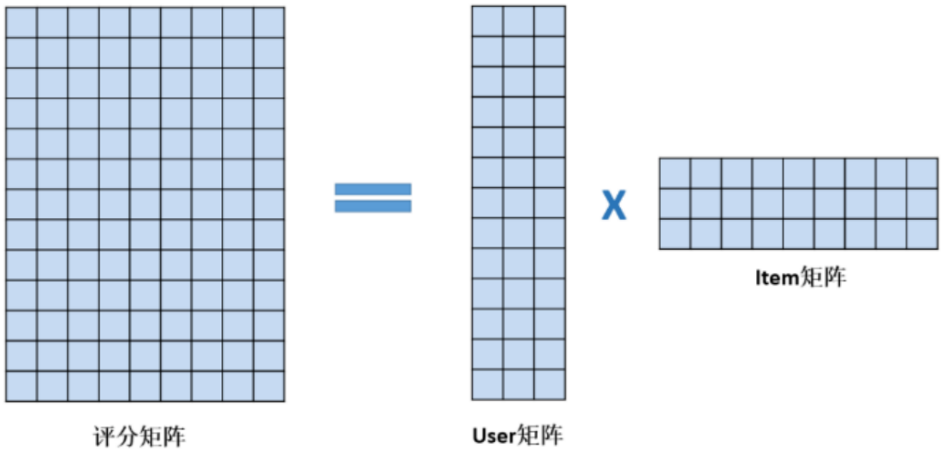
- 基于邻域的协同过滤包括UserCF, ItemCF，将用户的所有数据读入到内存中进行运算，也称之为基于内存的协同过滤（Memory-based）。数据量少的情况下，可以在线实时推荐
- 基于模型的推荐（Model-based），采用机器学习的方式，分成训练集和测试集。离线训练时间比较长，但训练完成后，推荐过程比较快。

推荐系统的两大应用场景：

- 评分预测（Rating Prediction） 主要用于评价网站，比如用户给自己看过的电影评多少分（MovieLens），或者用户给自己看过的书籍评价多少分（Douban）。矩阵分解技术主要应用于评分预测问题。

- Top-N推荐（Item Ranking） 常用于购物网站，拿不到显式评分，通过用户的隐式反馈为用户提供一个可能感兴趣的Item列表。排序任务，需要排序模型进行建模。

- 什么是矩阵分解



$$\min_{X,Y} \sum_{r_{ui}\neq 0} \left(r_{ui} - x_u^T y_i\right)^2 + \lambda\left(\sum_u \|x_u\|_2^2 + \sum_i \|y_i\|_2^2\right)$$

- 矩阵分解中的ALS算法
  - ALS, Alternative Least Square, ALS，交替最小二乘法
    - Step1，固定Y 优化X：

对目标函数 J关于x_u 求梯度，并令梯度为零，得

$$\frac{\partial J(x_u)}{\partial x_u} = -2Y_u\left(R_u - Y_u^T x_u\right) + 2\lambda x_u = 0$$

$$x_u = \left(Y_u Y_u^T + \lambda I\right)^{-1} Y_u R_u$$

=>

    - Step2，固定X 优化Y
    - 重复Step1和2，直到X 和Y 收敛。每次固定一个矩阵，优化另一个矩阵，都是最小二乘问题

加权值：

对隐式矩阵进行分解：引入置信度

当rui>0时，cui与rui线性递增

当rui=0时，cui=1，也就是cui最小值为1
目标函数：

$$\min_{X,Y} \sum_{u=1} \sum_{i=} c_{ui} \left(p_{ui} - x_u^T y_i \right)^2 + \lambda \left(\sum_u \left\| x_u \right\|_2^2 + \sum_i \left\| y_i \right\|_2^2 \right)$$

转化为矩阵形式：

$$x_u = \left(Y \Lambda_u Y^T + \lambda I \right)^{-1} Y \Lambda_u P_u$$

$$\Lambda_u = \begin{pmatrix} c_{u1} & & \\ & \ddots & \\ & & c_{uM} \end{pmatrix}$$

- ALS工具
  - spark mllib库（spark3.0版本后废弃）
    支持常见的算法，包括 分类、回归、聚类和协同过滤
    from pyspark.mllib.recommendation import ALS, Rating, MatrixFactorizationModel
  - spark ml库（官方推荐）
    功能更全面更灵活，ml在DataFrame上的抽象级别更高，数据和操作耦合度更低，使用起来像sklearn
    Spark安装很多坑，需要慢慢来
  - Python代码
    https://github.com/tushushu/imylu/blob/master/imylu/recommend/als.py

- SGD:
  - 批量梯度下降：在每次更新时用所有样本 稳定，收敛慢
  - 随机梯度下降: 每次更新时用1个样本，用1个样本来近似所有的样本 更快收敛，最终解在全局最优解附近
  - mini-batch梯度下降: 每次更新时用b个样本，折中方法 速度较快

- Baseline算法

即不直接使用用户评分，而是用针对物品的平均分+用户普遍给分高出/低于平均的分数差+该物品与普通物品的分数差

- 推荐系统工具：Surprise与LightFM
  - Surprise
    - Surprise是scikit系列中的一个推荐系统库
      pip install scikit-surprise
      文档：https://surprise.readthedocs.io/en/stable/
    - LightFM
      Python推荐算法库，具有隐式和显式反馈的多种推荐算法实现。
      易用、快速（通过多线程模型估计），能够产生高质量的结果。

Question?

1.

- Baseline算法
  基于邻域的协同过滤
  矩阵分解：SVD，SVD++，PMF，NMF
  SlopeOne 协同过滤算法
  - Surprise中的评价指标
    - to

课堂笔记

GCN

全量
offline
A/B test

Attention
模型

- SlopeOne算法
  - Step1，计算Item之间的评分差的均值，记为评分偏差（两个item都评分过的用户）
  - Step2，根据Item间的评分偏差和用户的历史评分，预测用户对未评分的item的评分
  - Step3，将预测评分排序，取topN对应的item推荐给用户

- 对MovieLens进行推荐
  - Python代码
  - surprise
  - NormalPredictor

Summary (MF)

- **MF是一种隐语义模型，它通过隐类别匹配用户和item来做推荐。**
  - **MF对原有的评分矩阵R进行了降维，分成了两个小矩阵：User矩阵和Item矩阵，User矩阵每一行代表一个用户的向量，Item矩阵的每一列代表一个item的向量。将User矩阵和Item矩阵的维度降低到隐类别个数的维度。**
- **根据用户行为，矩阵分解分为显式矩阵分解和隐式矩阵**
  - **在显式MF中，用户向量和物品向量的内积拟合的是用户对物品的实际评分**
  - **在隐式MF中，用户向量和物品向量的内积拟合的是用户对物品的偏好(0或1)，拟合的强度由置信度控制，置信度又由行为的强度决定**

Summary (ASL&SGD)

- **ALS和SGD都是数学上的优化方法，可以解决最优化问题（损失函数最小化）**
- **ALS-WR算法，可以解决过拟合问题，当隐特征个数很多的时候也不会造成过拟合**
- **ALS，SGD都可以进行并行化处理**

- **SGD方法可以不需要遍历所有的样本即可完成特征向量的求解**
- **Facebook把SGD和ALS两个算法进行了揉合，提出了旋转混合式求解方法，可以处理1000亿数据，效率比普通的Spark MLlib快了10倍**

Summary (Surprise)

- **python开源推荐系统，包含了多种经典的推荐算法**
- **官方文档：https://surprise.readthedocs.io/en/stable/**
- **数据集：可以使用内置数据集（Movielens等），也可以自定义数据集**
- **优化算法：支持多种优化算法，ALS，SGD**
- **预测算法：包括基线算法，邻域方法，矩阵分解，SlopeOne等**
- **相似性度量：内置cosine，MSD，pearson等**
- **scikit家族，可以使用GridSearchCV自动调参，方便比较各种算法结果**

- 什么是冷启动
  - 物品冷启动
  - 用户冷启动
  - 系统冷启动
  - 新用户，新商品是不断产生的过程

- 冷启动的常用方法有哪些
  - 提供非个性化的推荐(用户冷启动)
  - 利用用户的注册信息(用户冷启动、系统冷启动)
  - 基于内容做推荐(用户冷启动、系统冷启动)
  - 利用标的物的metadata信息做推荐(商品冷启动)
  - 快速试探策略(用户冷启动、商品冷启动)
  - 兴趣迁移策略(用户冷启动、系统冷启动)
  - 基于关系传递的策略(商品冷启动)

- 推荐系统中的冷启动
  - 提供非个性化的推荐(用户冷启动)
    - 热门商品
    - 人工指定策略
    - 提供多样性的选择
  - 利用用户注册信息(用户冷启动、系统冷启动)
    - 基于用户信息，比如年龄，性别，地域、学历、职业等做推荐 让用户选择兴趣点，
    - 让用户选择自己喜欢的分类标签（避免选项太多，操作复杂）
    - 利用社交关系，将好友喜欢的商品推荐给你
  - 基于内容做推荐(用户冷启动、系统冷启动)
    - 当用户行为较少时，无法使用协同过滤，可以基于item本身的内容做推荐 比如看过某个分类（体育）的内容，可以推荐这个分类的其他热门内容
  - 快速试探策略(用户冷启动、标的物冷启动)
    - 随机或者按照非个性化推荐的策略进行推荐，基于用户的点击反馈

快速发现兴趣点
        - 用于新闻、短视频领域
        - Bandit算法
    - 兴趣迁移策略(用户冷启动、系统冷启动)
        - 借鉴了迁移学习的思路，比如基于头条给抖音导流，知道用户在头条上的兴趣点，可以为其进行推荐
    - 利用商品的metadata信息做推荐（商品冷启动）
        - 利用Item与用户行为的相似性 计算新商品的特征，再计算商品特征与用户行为特征之间的相似度，从而为商品推荐和它最匹配的用户
        - 利用Item与Item的相似性 可以基于Item的属性信息来做推荐，一般新上线的商品都会有一些属性，根据这些属性找到与最相似的商品，这些相似的商品被哪些用户“消费”过，可以将该item推荐给这些消费过的用户。 淘宝提出使用基于side information（辅助信息）的图嵌入学习方法，称为Graph Embedding with Side information (GES)

- Paper Reading List 怎样阅读更高效 课程相关Paper

- ALS-WR Paper Reading

- 简历Updating
    - 银行产品购买预测：采用Item-based CF方法，对Santandery银行的用户产品购买数据进行分析，并对未来可能购买的产品进行预测https://github.com/xxx/Santandery
    - 电影推荐算法：基于矩阵分解的协同过滤算法（ALS，SVD，SVD++，FunkSVD） 给Netflix网站进行推荐算法，RMSE降低到0.9111 https://github.com/xxx/netflix
    - CTR广告点击率预测：采用基于神经网络的DeepFM算法，对DSP公司Avazu的网站的广告转化率进行预测，项目中使用了线性模型及非线性模型，并进行了对比分析 https://github.com/xxx/avazu-ctr-prediction
    - 房屋价格走势预测引擎：通过时间序列算法，分析北京、上海、广州过去4年（2015.8-2019.12）的房屋历史价格，预测未来6个月（2020.1-2020.6）不同区的价格走势 https://github.com/xxx/house-price-prediction
    - 邮件数据分析：通过PageRank算法分析邮件中的人物关系图谱，并针对邮件数量较大的情况筛选出重要的人物，进行绘制：https://github.com/xxx/PageRank
    - 电影数据集关联规则挖掘：采用Apriori算法，分析电影数据集中的导演和演员信息，从而发现导演和演员之间的频繁项集及关联规则：https://github.com/xxx/Apriori
    - 信用卡违约率分析：针对台湾某银行信用卡的数据，构建一个分析信用卡违约率的分类器。采用Random Forest算法，信用卡违约率识别率在80%左右：https://github.com/xxx/credit_default
    - 信用卡欺诈分析：针对欧洲某银行信用卡交易数据，构建一个信用卡交易欺诈识别器。采用逻辑回归算法，通过数据可视化方式对混淆矩阵进行展示，统计模型的精确率，召回率和F1值，F1值为0.712，并绘制了精确率

和召回率的曲线关系：https://github.com/xxx/credit_fraud

- 比特币走势分析：分析2012年1月1日到2018年10月31日的比特币价格数据，并采用时间序列方法，构建自回归滑动平均模型（ARMA模型），预测未来8个月比特币的价格走势。预测结果表明比特币将在8个月内降低到4000美金左右，与实际比特币价格趋势吻合（实际最低降到4000美金以下）：https://github.com/xxx/bitcoin

总结Summary

- 记录作业内容

1. Thinking 1：高德地图中的路径规划原理是怎样的?
2. Thinking 2：football.gml 美国大学生足球联赛，包括115支球队，被分为12个联盟。为什么使用LPA标签传播进行社区发现，只发现了11个社区?
3. Thinking 3：微博采用了类似FaceBook的EdgeRank算法，如果你给微博的信息流做设计，你会如何设计?
4. Action 1：使用Python模拟下面的PageRank计算过程，求每个节点的影响力（迭代100次）简化模型 随机模型
5. Action 2：使用TextRank对新闻进行关键词提取，及文章摘要输出新闻文本见：news.txt