建模部分详细内容：

选取suprise库，简单易用，同时支持多种推荐算法（基础算法，基于近邻方法，矩阵分解方法）

建模流程

一：载入movielens数据集(即：song\_id ,song\_name, artist, popularity)

1. 指定文件所在路径
2. 指定文本格式
3. 加载数据
4. 手动切分成5折（方便交叉验证）

二：K折交叉验证（k=3）

第一：（测试数据：歌曲名&歌曲id）

1. 用算法计算相互间的相似度
2. 获取歌曲名到歌曲id 和 歌曲id到歌曲名的映射
3. 任意取出一部歌曲对应的item\_id
4. 找到最近的10个邻居
5. 从临近的id映射回歌曲名称.

第二：利用歌单来查找最近的user

1.导入歌单信息（歌单名字，歌单id）

2.取出对应的内部user id -> to\_inner\_uid

3.把歌曲id转换成歌曲名字

三：分解SVD矩阵

1. 初始化几个bias项，求出bias向量、user向量和item向量
2. 按照实际公式对user向量和item向量求内积
3. 加上偏置项和正则化项
4. 选择合适的optimizer做优化

四：测试效果

1.载入数据

2.分别对数据进行NormalPredictor、BaselineOnly、基础版协同过滤、均值协同过滤、协同过滤baseline和SVD推荐算法，求出各自的RMSE和MAE值

3.选择效果最好的算法进行音乐推荐系

五：输出结果

1. 将输出的数据集movielens保存至本地
2. 提取movielens数据集中的音乐id和音乐名

## 用户兴趣预测问题：每个人的兴趣都是有时效性的。把每个用户喜欢（收藏）过的歌，沿着时间轴排好，同时由近到远给不同的衰减因子（比如最近一首歌是1，前一首是0.98，再前一首是0.982，以此类推…），同时我们针对不同的歌曲热度，给定不同的推荐因子（比如热度100的是1，热度80的是0.9…），每一首歌都可以拿回一个song2vec的推荐列表和对应的相似度，对相似度以时间衰减因子和热度权重进行加权，最后的结果排序后，展示给用户