推荐系统项目

一、工作流程

1. 同步RDS的业务数据和埋点数据至ODPS中
2. 数据清洗和用户岗位过滤
3. 基于物品协同过滤算法构建用户对岗位的评分矩阵
4. 计算岗位之间的相似度
5. 根据用户历史行为和岗位相似度给用户生成推荐列表
6. 将推荐结果同步至RDS
7. 业务方API接口调用

二、详细说明

- 数据同步：将用户信息、兼职岗位信息表、数据埋点表和用户报名表等业务表数据同步至ODPS中。第一次同步全量同步，以后每天增量同步前一天的数据。
- 数据清洗：过滤三个月活跃用户；岗位要求与用户基本信息过滤；分类目统计用户历史行为数据；用户埋点数据信息统计处理等。
- 构建评分矩阵：利用用户历史对岗位的行为信息，按照不同权重值计算用户对兼职岗位的评分数据。
- 利用余弦相似度公式计算岗位之间的相似度，公式有改进
- 得到岗位之间的相似度后，结合用户历史行为计算每个用户对每个类目下所有岗位的分数作为最终推荐结果
- 将计算完的推荐结果同步至RDS

三、算法说明

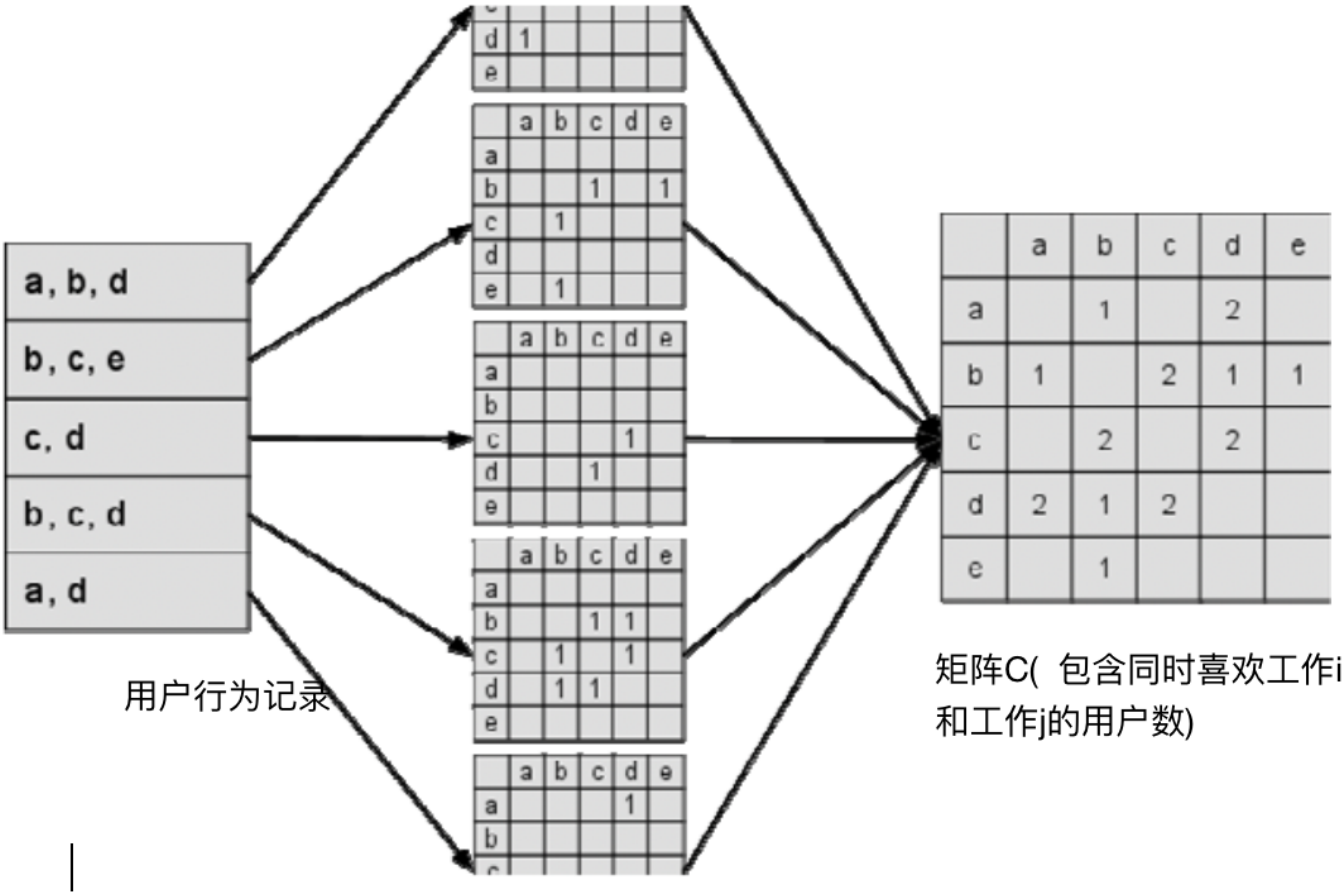
关于协同过滤的一个最经典的例子就是看电影，有时候不知道哪一部电影是我们喜欢的或者评分比较高的，那么通常的做法就是问问周围的朋友，看看最近有什么好的电影推荐。在问的时候，都习惯于问跟自己口味差不 多的朋友，这就是协同过滤的核心思想。 要实现协同过滤，需要进行如下几个步骤

1）收集用户偏好

2）找到相似的用户或者物品

3）计算并推荐

构建评分矩阵



图中最左边是输入的用户行为记 录，每一行代表一个用户感兴趣的岗位集合。然后，对于每个岗位集合，我们将里面的工作两两加一，得到一个矩阵。最终将这些矩阵相加得到上面的C矩阵。其中C[i][j]记录了同时喜欢岗位i 和岗位j的用户数。最后，将C矩阵归一化可以得到岗位之间的余弦相似度矩阵W。 根据我们的业务场景是按照不同权重值（报名：10，点击：1，收藏：2）计算用户对该岗位的评分

用户相似度计算的改进

协同过滤算法主要利用行为的相似度计算兴趣的相似度。这里我们使用基于物品的协同过滤算法，对于物品i和j，令N(i)表示喜欢物品i的用户数，令N(j)为喜欢物品j的用户数。那么余弦相似度计算为：

$$w_{ij} = \frac{\left|N(i) \cap N(j)\right|}{\sqrt{\left|N(i)\right|\left|N(j)\right|}}$$

公式中分子是同时喜欢物品i和物品j的用户数，这个公式分母部分惩罚了物品j的权重，因此可以减轻热门物品会和很多物品相似的可能性。但是在实际中热门的j仍然会获得比较大的相似度，这个后面可以考虑在分母上加大对热门物品的惩罚和权重值的调整。 从上面看到，在协同过滤中两个物品产生相似度是因为它们共同被很多用户喜欢，也就是说每个用户都可以通过它们的历史兴趣列表给物品贡献相似度。

但是这个公式过于粗糙，由于每个用户的兴趣列表都会对物品的相似度产生贡献，但是活跃用户的兴趣显然没有非活跃用户的兴趣那么集中，应该使活跃用户对物品的相似度贡献地域不活跃用户的才合理，因此可以引进IUF，即用户活跃度对数的倒数的参数，对上面余弦相似度公式进行改进，相当于对活跃用户做了一种软性惩罚：

$$w_{ij} = \frac{\sum_{u \in N(i) \cap N(j)} \frac{1}{\log 1 + |N(u)|}}{\sqrt{|N(i)||N(j)|}}$$

物品相似度的归一化

先说归一化的好处：不仅能增加推荐的准确度，还可以提高推荐的覆盖率和多样性

如果我们已经得到了物品的相似度矩阵w，那么可以使用下面公式得到归一化之后的相似度矩阵w'：

$$w'_{ij} = \frac{w_{ij}}{\max_j w_{ij}}$$

一般来说，物品总是属于很多不同的类，每一类中的物品联系比较紧密。假设有两种电影-纪录片和动画片，那么计算出来的一般是纪录片和纪录片的相似度或者动画片和动画片的相似度大于纪录片和动画片的相似度。但是纪录片之家的相似度和动画片之间的相似度却不一定相同。假设物品分两类–A和B，A类物品之间相似度0.5，B类物品之间相似度0.6，A类和B类之间相似度0.2。如果一个用户喜欢了5个A类物品和5个B类物品，推荐结果就都是B类物品，因为B类物品之间相似度大。但是如果做了归一化之后，A类物品之间相似度变成了1，B类物品之间相似度也变成了1，那这种情况下推荐结果中A类和B类物品的数量也应该大致相等。

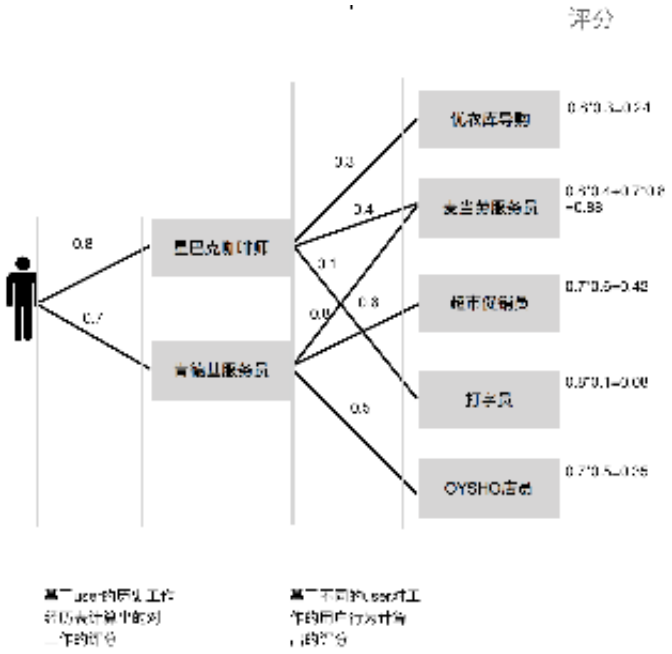
还有一点，对于热门的类下面的物品，她们之间相似度一般比较大，如果不进行归一化就会推荐比较热门的类里面的物品，因此推荐覆盖率就比较低。

计算用户对每个岗位的兴趣

在得到物品之间的相似度后，我们可以通过以下公式计算某一个用户u对一个物品j的兴趣：

$$p_{uj} = \sum_{i \in N(u) \cap S(j,K)} w_{ji} r_{ui}$$

这里N(u)是用户喜欢的物品的集合，S(j,K)是和物品j最相似的K个物品的集合，wji是物品j和i的相似度，rui是用户u对物品i的兴趣。（对于隐反馈数据集，如果用户u对物品i有过行为，即可令rui=1。）该公式的含义是，和用户历史上感兴趣的物品越相似的物品，越有可能在用户的推荐列表中获得比较高的排名。



上图是基于岗位推荐的一个例子，某用户之前报名了星巴克和肯德基的兼职，然后这两个兼职对应的给出三个推荐兼职，根据公式计算该用户对每个兼职的感兴趣程度。比如推荐该用户麦当劳服务员这个兼职，是因为麦当劳服务员和肯德基服务员相似度高达0.8，和星巴克咖啡师相似度也有0.4，那么该用户对麦当劳服务员的兴趣度就是0.8*0.4+0.7*0.8=0.88。

评测指标

一般通过准确率（precision）和召回率（recall）来度量

令R(u)是根据用户在训练集上的行为给用户做出的推荐列表，而T(u)是用户在测试集上的行为列表，因埋点数据暂时不够支持，故我们采用列表页的点击报名比和曝光点击比衡量推荐效果。