矩阵分解

基本思想：用户都有自己的偏好，同时每个物品也有自己的属性恰好就是用户的偏好信息，我们人为用户对物品的高评分体现的是物品中所包含的偏好信息恰好就是用户喜好的信息。所有我们只要得到**用户-潜在因子矩阵Q**和**物品-潜在因子矩阵P**就可以计算出用户对物品的评分信息。

$$\hat{R} = QP^T$$

1.SVD

SVD的想法是根据已有的评分情况，分析出评分者对各个物品因子的喜好程度，以及各个物品对这些因子的包含程度，最后反过来根据分析结果预测评分。通过SVD的方式挖掘影响评分的因子，通过这些因子可以挖掘更多有意义的关联关系。

SVD的数学定义：将给定评分矩阵R分解成三个矩阵的成绩，其中U、V成为左、右奇异向量，$\sum$对角线上的值称为奇异值。

$$R_{n*m} = U_{n*n} * \Sigma_{n*m} * V^T_{m*m}$$

因为前1%的奇异值的和就占了全部奇异值的和的99%以上。所以，可以截断为：

$$R_{n*m} = U_{n*k} * \Sigma_{k*k} * V^T_{k*m}$$

SVD分解要求矩阵是稠密的，也就是说矩阵的所有位置不能有空白。有空白时我们的是没法直接去SVD分解的。但是如果这个矩阵是稠密的，那不就是说我们都已经找到所有用户物品的评分了嘛，那还要SVD干嘛呢？ 的确，这是一个问题，传统SVD采用的方法是对评分矩阵中的缺失值进行简单的补全，比如用全局平均值或者用用户物品平均值补全，得到补全后的矩阵。接着可以用SVD分解。使用上述简单的矩阵分解方式虽然可以解决缺省值的情况，**但是由于推荐系统中的用户量和物品量是特别大的，所以直接使用原始SVD的矩阵分解是比较困难的。**

2.FunkSVD

由于在SVD矩阵分解中，将矩阵分解为三个矩阵效率比较低，那么我们可以通过矩阵分解为两个矩阵来降低执行的消耗； 即将U、V矩阵进行转换得到用户因子矩阵Q和物品因子矩阵，那么最终的评分预测为rui也进行对应的转换：

$$Q_{n*k} = U_{n*k} * (\Sigma_{k*k})^{1/2}, \quad P_{m*k} = ((\Sigma_{k*k})^{1/2} \times V^T_{m*k})^T$$

$$R_{n*m} = Q_{n*k} * P^T_{m*k}$$

$$\hat{r}_{ui} = q_u p_i^T = \sum_{k=1}^{K} q_{uk} p_{ik}$$

用户因子矩阵Q和物品因子矩阵P的计算可以利用线性回归的思想，通过随机梯度下降的方式进行学习，迭代式的更新相关参数即可，使用SVD矩阵因子分解推荐算法对于评分稀疏矩阵也可以进行正常处理，**对于没有评分的不用计算误差值，直接令误差值为0。**

$$\hat{r}_{ui} = q_u p_i^T \qquad e_{ui} = r_{ui} - \hat{r}_{ui}$$

$$min_{p^*,q^*} \frac{1}{2} \sum_{u,i} (r_{ui} - q_u p_i^T)^2$$

$$p_i^{k+1} = p_i^k + \alpha e_{ui} q_u^k$$

$$q_i^{k+1} = q_i^k + \alpha e_{ui} p_u^k$$

3.BiasSVD

同普通的协同过滤算法一样，我们可以更改一下FunkSVD预测值公式，可以认为最终的预测值是在基准评分/偏置项基础上的一个变化，从而我们可以得到下列预测公式：

$$\hat{r}_{ui} = \mu + b_u + b_i + q_u p_i^T$$
$$e_{ui} = r_{ui} - \hat{r}_{ui}$$

同样的我们可以得到一个加入正则化项后的目标函数：

$$\min_{q_u, p_i, b_u, b_i} \frac{1}{2} \left(\sum_{u,i} \left(r_{u,i} - \hat{r}_{u,i} \right)^2 + \lambda \left(b_u^2 + b_i^2 + \|q_u\|^2 + \|p_i\|^2 \right) \right)$$

4.SVD++

在SVD中，预测用户u对商品i的评价时，只考虑了用户特征向量和商品特征向量，以及评分相对于平均分的偏置向量。在SVD++中，更进一步地考虑了用户对其所有有过评分行为的商品的隐世反馈。预测函数$\hat{r}_{ui}$的变形如下形式：

$$\hat{r}_{ui} = \mu + b_i + b_u + (q_u + |I|^{-\frac{1}{2}} \sum_{j \in I_u} y_j) p_i^T$$