**Thinking1 在实际工作中，FM和MF哪个应用的更多，为什么？**

FM（Factorization Machine）应用更多：

* 泛化能力强，解决大规模稀疏数据下的特征组合问题，不仅是UserID, ItemID特征
* 因为矩阵分解MF（Matrix Factorization）只是FM在计算二阶特征组合系数时候的特例，即特征只有User ID 和Item ID的FM模型，只适用于评分预测，进行简单的特征计算，无法利用其他的特征。而FM引入了更多辅助信息（Side information）作为特征
* 计算复杂度，可以在线性时间对样本做出预测，通过公式变换将计算复杂度降到O(k\*n)

**Thinking2 FFM与FM有哪些区别？**

FM（Factorization Machines，因子分解机），在数据非常稀疏的情况下，依然能估计出可靠的参数进行预测。与传统的简单线性模型不同的是，因子分解机考虑了特征间的交叉，对所有嵌套变量交互进行建模（类似于SVM中的核函数），因此在推荐系统和计算广告领域关注的点击率CTR（click-through rate）和转化率CVR（conversion rate）两项指标上有着良好的表现。此外，FM的模型还具有可以用线性时间来计算，以及能够与许多先进的协同过滤方法相融合等优点。

FFM 算法，全称是 Field-aware Factorization Machines，是 FM（Factorization Machines）的改进版。通过引入 field 的概念，FFM 把相同性质的特征归于同一个 field。在 FFM 中，每一维特征 ，针对其它特征的每一种 field ，都会学习一个隐向量 。因此，隐向量不仅与特征相关，也与 field 相关。这也是 FFM 中 “Field-aware” 的由来。

* FFM 由于引入了 Field，使得每两组特征交叉的隐向量都是独立的，可以取得更好的组合效果， FM 可以看做只有一个 Field 的 FFM。
* FFM隐向量的长度为 k，二次参数有 nfk 个，多于FM模型的 nk 个。
* 由于隐向量与field相关，**FFM二次项并不能够化简，计算复杂度是** **。FFM的k值一般远小于FM的k值。**明显在计算速度上**FFM**比FM模型慢得多。

**Thinking3 DeepFM相比于FM解决了哪些问题，原理是怎样的？**

**DeepFM=FM+DNN，在低阶和高阶特征组合上更接近真实世界，因此效果也更好。很多特征，在高阶情况下，人很难理解，但是机器可以发现规律（使用DNN模型）。**

整个模型大体分为两部分：FM（左边）和DNN（右边）。模型大概的流程如下：

1. 首先利用FM进行embedding得到Dense Embeddings的输出。
2. 将Dense Embeddings的结果作为左边FM模块和右边DNN模块的输入。通过一定方式组合后，模型左边FM模块的输出完全模拟出了FM的效果，而右边的DNN模块则学到了比FM模块更加高阶的特征交叉。
3. 最后将DNN和FM的结果组合后激活输出。

**Thinking4 Surprise工具中的baseline算法原理是怎样的？BaselineOnly和KNNBaseline有什么区别？**

Baseline算法的思想就是设立基线，并引入用户的偏差以及item的偏差



μ为所有用户对电影评分的均值

bui：待求的基线模型中用户u给物品i打分的预估值

bu：user偏差（如果用户比较苛刻，打分都相对偏低， 则bu<0；反之，bu>0）；

bi为item偏差，反映商品受欢迎程度

（ALS 求得Bi和Bu，也就是评分矩阵）

KNNBaseline is a basic collaborative filtering algorithm taking into account a baseline rating. Needs to consider parameter k(int) – The (max) number of neighbors to take into account for aggregation. Default is 40.

**Thinking5 基于邻域的协同过滤都有哪些算法，请简述原理。**

knns.KNNBasic 基本的协同过滤算法

knns.KNNWithMeans 协同过滤算法的变种，考虑每个用户的平均评分

knns.KNNWithZScore 协同过滤算法的变种，考虑每个用户评分的归一化操作

knns.KNNBaseline 协同过滤算法的变种，考虑每个用户评分的基线