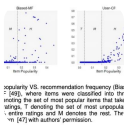
在另一个问题中，针对推荐系统里的各种坑儿，做过一个回答了，当时的角度是基于推荐系统的偏差bias问题的角度进行了回答。传送门如下：

**在你做推荐系统的过程中都遇到过什么坑?**
56 赞同 · 0 评论　回答



同时，经过一段时间的学习，结合各位知乎大佬对该问题的理解和回答，我做了下总结，主要从以下角度：

壹号坑：线上线下不一致问题
贰号坑：评估指标里问题
叁号坑：推荐系统健康度问题
群坑乱舞：踩坑合集

## 壹号坑：线上线下不一致问题

### 1. 特征不一致

这种在离线拼接样本和特征的Pipeline中比较常见。一般离线特征都是按照天处理的，考虑各种数据pipeline的流程，处理时间一般都会有延迟，离线特征处理完之后导到线上之后，用于线上模型预估时请求使用。那这种情况产生的原因是什么呢？在离线，我们使用T-n到T-1的数据训练模型，用T天的数据进行测评，拿到了很好的离线指标，比如AUC为0.82。但是在线服务的模型，并不是这样的理想情况，一个模型每天重新迭代训练，需要新一天（T-1天）的日志，日志从数据队列Q传输到大数据平台，进行日志的处理，新一天各种特征的计算，组织训练样本，进行模型训练，之后还要把模型从大数据平台更新到在线服务器，整个流程走下来几个小时过去了。那么在新模型上线前，在线服务的是T-2的模型，相当于在离线用T-2的模型去测评T天的样本，效果会大打折扣。因而线上一整天的平均测评指标，是低于离线测评指标的。

举个例子，例如12月15日这天，线上预估请求用的特征是12月14号的特征数据。到了12月16日，特征Pipeline开始处理数据，到了凌晨5点（有时候ETL作业集群有问题可能会到中午12点），离线特征处理完了导到线上。那么在12月16日0点-2月16日5点，这段时间线上请求的特征使用的是老的特征数据，也就是12月14日的特征数据。12月16日5点-12月16日24点，线上特征使用的是12月15日的数据。而在离线样本生成过程中，到了12月17日0点，如果是按天拼接的，那么12月16号这天的所有样本，都会使用12月15日的特征。

这样，12月16日0点--2月16日5点的样本，在离线样本拼接的阶段，使用的是12月15日的特征数据，而在线上请求特征的时候使用的还是12月14日的特征。特征Pipeline流程处理越长，这种不一致会越大。

那么问题来了，如果换成实时数据进行实时特征加工是不是就解决这个问题了？

实时特征在线使用的时候，经过客户端埋点的上报（这些先不考虑埋点系统的各种坑），流式计算Q处理日志数据进入在线数据源或特征库，需要经过一段时间。也就是说，如果你刚刚点击了某个“豪车、豪宅”视频，紧接着下滑翻页，系统是拿不到“豪车、豪宅”这个行为的。如果离线模型训练中有用到了带有“豪车、豪宅”的特征，由于近期行为的影响非常大，那么离在线的不一致会非常严重。

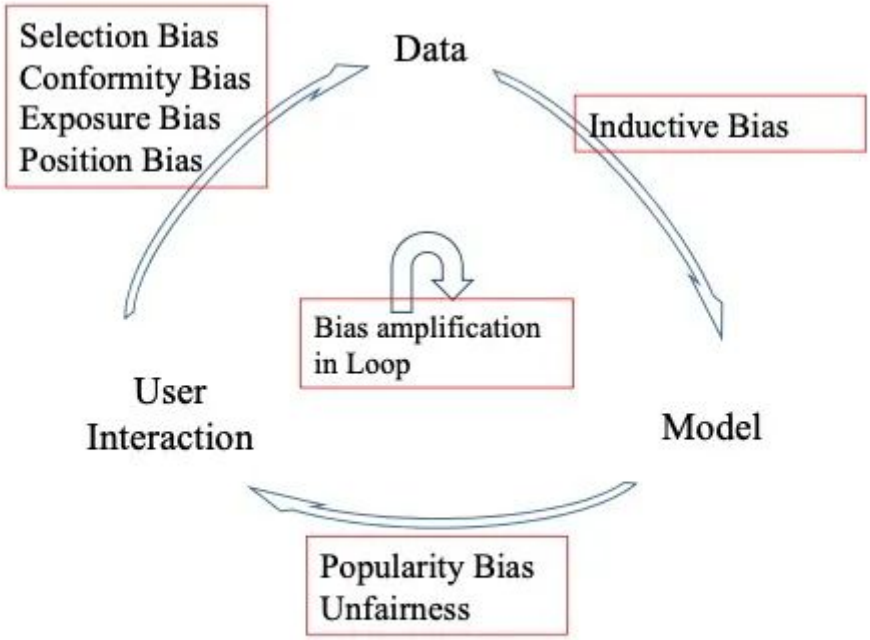



Fig. 2. Feedback loop in recommendation, where biases occur in different stages.

**2. 数据分布不一致**

如果仔细排查，既不存在数据泄漏，也没有出现不一致的问题，离线auc明明就是涨了很多，线上就是下降，而且是离线涨的越多，线上下降越多，还有一种可能就是数据的不一致，也就是数据的“冰山效应”——离线训练用的是有偏的冰山上的数据，而在线上预估的时候，需要预测的是整个冰山的数据，包括大量冰面以下的数据！

这种情况其实在推荐系统里非常常见，但是往往非常的隐蔽，一时半会很难发现。我们看下面这张图。左边是我们的Baseline，绿色的表示正样本，红色表示负样本，灰色部分表示线上由于推荐系统的“偏见”（预估分数较低），导致根本没有展现过的数据。

关于推进系统的偏差问题，之前的《推荐系统Bias大全》一文已经总结了推荐系统中所有Bias情况，有兴趣的可以跳转看一下。

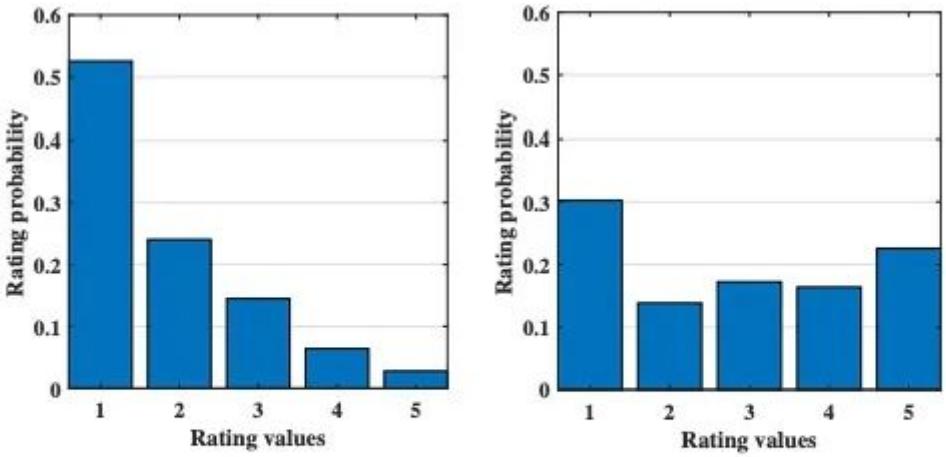
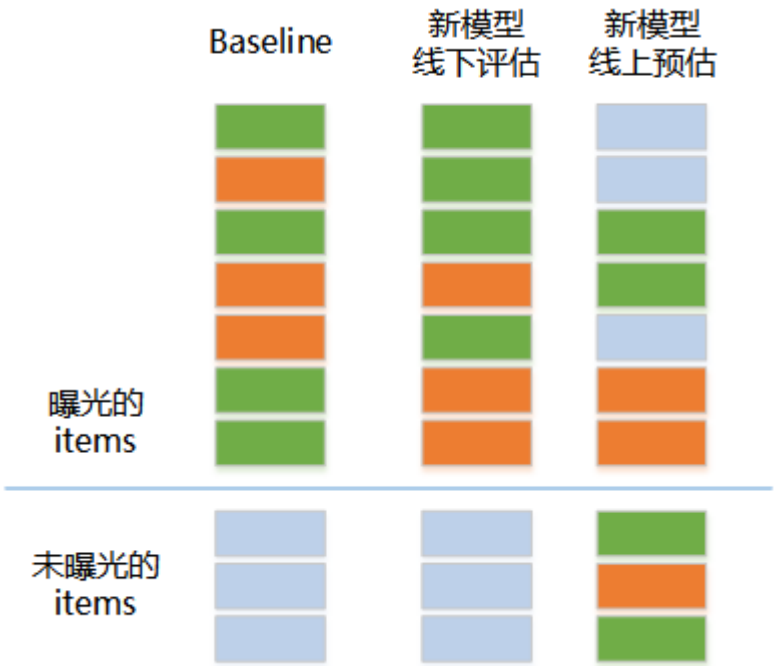



Fig. 3. Distribution of rating values for randomly selected items and user-selected items. The data is from [20] with permission.

离线阶段，我们通过各种优化，新模型的离线评估表现更好了，例如图中第二列，可以发现第4个绿色的正样本和第7个绿色的正样本排到了第3和第6的位置，离线的auc指标涨了。

到了真正线上的预估也就是第三列，发现对于这部分离线见过的样本，模型的预估序并未改变。但是新模型给了灰色没有见过的数据更高的预估分数，这部分数据一旦表现不好，很可能造成我们前面说的情况，离线（第二列）评估指标明明涨了不少，在线（第三列）评估指标CTR却下降。




这种情况也不是必现的，在LR以特征工程为主要迭代的时代很少见。主要的原因是模型的前后迭代差异并不大。新模型对比老模型最主要的特点是新加入了一部分特征，往往模型的打分差异并不大，从图中第二列到第三列，原来那些冰山下的数据也就是旧模型预估分数偏低的部分，在新模型中能够脱颖而出拿到很高的预估分数的概率并不高。

而在模型有较大变化的时候，例如lr->树模型，lr->深度模型，不同网络结构的深度模型变化，这种情况容易出现，原因就是新旧模型的变化较大，预估分数变化也较大。

举一个简单的例子，假设我们的baseline是热门模型，样本都是老的热门模型生产出的热门样本，这个时候我们用简单的lr模型去拟合，到了真正的线上预估的时候，对于大量之前没见过的非热门的数据，模型自然很难预估好。没有足够好的样本，模型也很难学到足够有用的信息。

说另一个很有意思的现象，之前在某个组的时候，两个team优化同一个场景，大家用的回流样本都是一样的，但是特征和模型都是自己独立优化和迭代。有意思的是，如果一个team的优化取得了比较明显的提升之后，另一个team哪怕什么都不做，过一段时间效果也会慢慢涨上来。

对于这种情况，最根本的手段就是解决数据的有偏问题。尤其是新模型，一开始相当于都是在拟合老模型产生的样本，刚上线效果如果比较差，经过一段时间迭代，影响的样本分布慢慢趋近于新模型，也能收敛，但效率较低。这里给下两个在我们这还比较有效的经验：

**对无偏数据进行上采样**
这里的无偏是相对的，可以是随机/探索流量产生的样本，也可以是新模型产生的样本。大概意思，就是尽可能利用这些对新模型有利的样本。
**线上线下模型融合**
比较trick的方法，没有太多方法论，但是确实能work。
新模型预估分数PCTRnew 和老模型预估分数PCTRold 直接在线上做线性融合，刚上线的时候a选取比较小，随着慢慢迭代，a慢慢放大。

## 贰号坑：评估指标里问题

在《推荐系统采样评估指标及线上线下一致性问题》一文中，主要阐述了该部分的观点：

在评估推荐算法的效果时,能不采样就不采样!
除了AUC, Precision@K, Recall@K, Average Precision, NDCG都是不一致的,采样计算得到的结果和真实结果可能差很大!
现在随机采样计算得到的评估指标的分数具有高偏差，低方差的问题，很多情况和真实情况不符合，结论可能也都错了!
如果一定要进行采样计算评估指标的值， 建议采用文中提出的纠正的方案，虽然可能会有较大的方差，但是偏差大大降低，更加接近真实情况；

举个例子，比如在信息流推荐中，低俗内容和标题党往往会在短期内对CTR指标有较好的提升，但是这些内容对整个生态在长期来看是有害的，如何处理这部分内容是值得思考的问题。又比如在电商推荐中，如何处理重复推荐也是一直都存在的问题。

推荐系统太难了。难到工程师和产品都还没清楚自己要的是什么。“推荐”这个问题本身都不是well-defined的。按照道理来讲，推荐系统要做的事情其实是“推荐用户希望看到的东西”，但是“用户希望看到的东西”落实到指标上，可就让人头大了。

以内容推荐为例。你说究竟要得到什么呢?

高CTR？那么擦边球的软色情以及热门文章就会被选出来
高Staytime？那么视频+文章feed流就成为为视频feed流和超长文章feed流
高read/U？那么短文章就会被选出来
这些指标相互依赖，此消彼长，目前主流是沿用计算广告的老路，按照CTR作为最广泛使用的评价指标来优化，这个指标的劣根性是显而易见的，然而至今并没有很好地指标来指导系统。

今日头条的做法是，优化CTR同时关注其他指标的变动；也有的从CTR开始，优化到瓶颈后进行Staytime的优化等等...
Medium的做法是，优化一个f(CTR, staytime,...)的多指标加权的综合指标，但是据我所知，这个加权的系数，还是一个magic number，是人拍脑门定的。

大家都在探索， 也并没有一个定论，究竟推荐系统该优化一些什么。

相信很多人刚入行的时候对单纯优化CTR都是有疑惑的，日子久了，也就都麻木了。

## 叁号坑：推荐系统健康度问题

推荐系统应该是一个良性循环的系统。这也就导致了E&E, exploration & exploitation问题的出现，简单说，就是保证精准推荐的同时，进行兴趣探索。

一说大家都明白了，这不就是所有推荐系统做的最差的地方吗？我看了一个东西，就使劲出一个东西，App明明很多东西，我却越用越窄。

这个问题更加玄学，更加让人无奈。

EE要不要做？肯定要做，你不能让用户只能看到一类新闻，这样久了他的feed 流只会越来越小，自己也觉得没劲，所以一定要做兴趣探索。

但是做，就势必牺牲指标，探索的过程是艰难的，大部分时间用户体验上也是负向的。那么，

牺牲多少CTR来保EE才算是合适的?
EE的ROI什么时候算是>1的?
怎么样确定EE的效果?
EE要E到什么程度?

其实大家也都没有想清楚，多避而不谈。

## 肆号坑：工程里的一些坑

### 模型工程

如何优化计算框架和算法，支持千亿特征规模的问题
如何优化召回算法和排序算法不一致性带来的信息损失
如何把多样性控制、打散、疲劳控制等机制策略融入到模型训练中去的问题
如何优化FTRL来更好地刻画最新样本的问题。
还有很多，像CF怎么优化的问题。至今对阿里ATA上那篇swing印象深刻。系统工程：这就更多，没有强大的工程系统支持的算法都是实验室的玩具。

### 系统工程

实时样本流中日志如何对齐的问题
如何保证样本流稳定性和拼接正确性
调研样本如何获取动态特征的问题：服务端落快照和离线挖出实时特征
基于fealib保证线上线下特征抽取的一致性问题。
在线预估服务怎么优化特征抽取的性能。如何支持超大规模模型的分布式存储，主流的模型通常在100G以上规模了。
内容系统进行如何实时内容理解，如何实时构建索引，以及高维索引等相关问题。

## 群坑乱舞：踩坑合集

看过了推荐系统在各种情况下的那些坑儿，最后，放一下 吴海波@知乎 的总结。

i2i/simirank等相似计算算法中的哈利波特问题，相似性计算在推荐系统的召回起到非常重要的作用，而热门物品和用户天然有优势。因此，处理方法基本上都是凭经验，会出现各种magic number的参数。
svd/svd++等算法，在各种比赛中大放异彩，但是据我所知，各大互联网公司中基本没有用它的。
i2i及它的各种变种真是太好用了，大部分公司的业务量，从ROI来看，其实没有必要再做什么优化了。
推荐的召回策略多优于少，但系统的计算rt是个问题，没有好的系统架构，也没有什么优化空间。
i2i等类似计算算法，实际业务中容易跑挂，比如用spark就容易oom，很容易发生倾斜，处理起来又有很多trick和dirty job。
推荐系统不只有召回，召回后的ranking起到非常大的作用，这个和广告的点击率预估有点像。
非常多的业务没有办法向头条/facebook之类的有稳定的用户留存指标，而留存是推荐系统非常重要目标，因此需要找各种折中的指标去知道abtest。
训练样本穿越/泄露问题，即在训练的特征中包含了测试集的样本的信息，这个是初期非常容易犯的错误，在后期有时也会隐秘的发生。
大部分推荐系统的训练数据需要在离线条件下去拼接，其中用户的行为状态和时间有关，拼接非常容易出问题，因为系统当时的状态会有各种状况（延时、数据不一致）等，导致训练和实际在线的数据不一致，建议把训练需要的日志在实际请求中直接拼接好打印落盘。
最后，算法的作用有点像前辈们讲的：短期被人高估，长期被人低估。按目前国内业务方对算法有了解的特别少，算法对生态的长短期影响，只能靠算法负责人去判断，因此，就算你只是个打工的，请有一颗做老板的心。

## 参考资料

zhihu.com/question/3221
zhihu.com/question/2824
zhihu.com/question/3221
推荐系统bias大全
是不是你的模型又线下线上不一致啦?