怎么理解基于机器学习“四大支柱”划分的学习排序方法

原创 Thinkgamer 搜索与推荐Wiki 2020-03-20

收录于话题
#推荐相关笔记

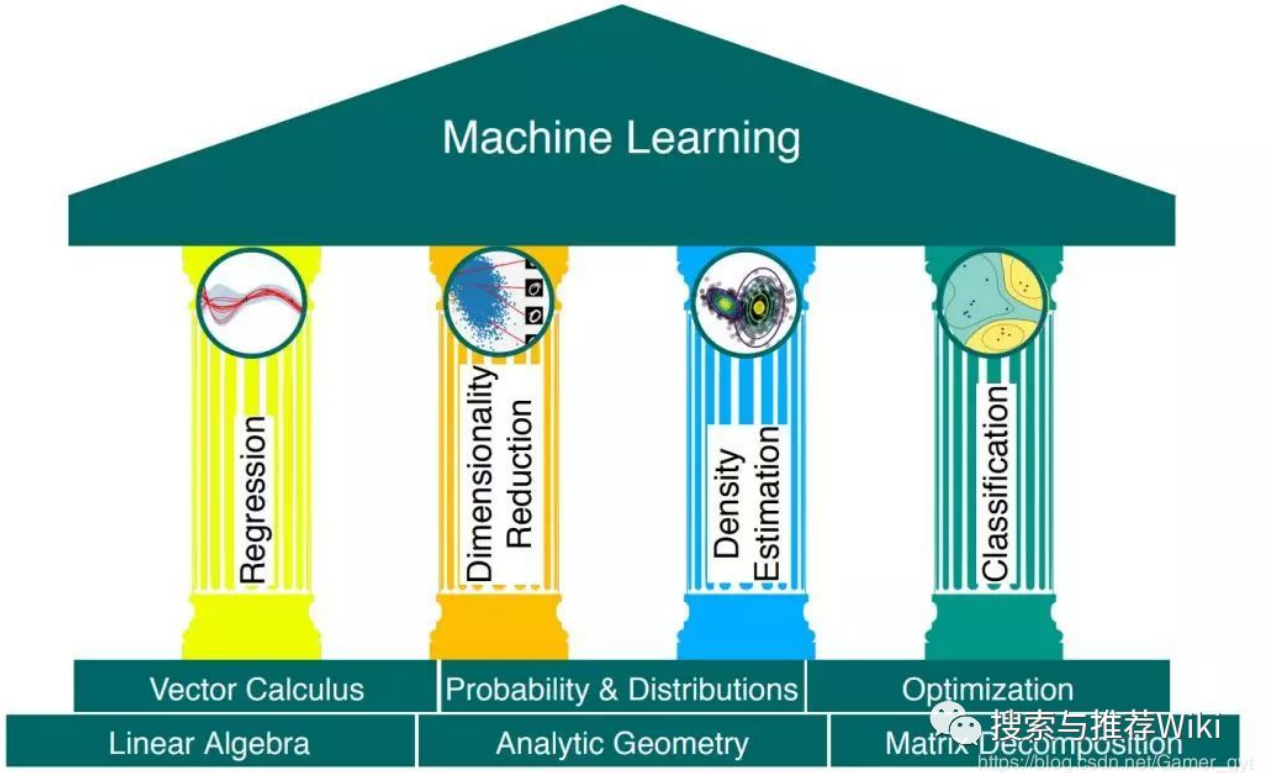
32个

Learning to rank（LTR，L2R）也叫排序学习，泛指机器学习中任何用户排序的技术，是指一类监督学习（Supervised Learning）排序算法。LTR被应用在很多领域，比如信息检索（Information Retrieval）、推荐系统（Recommend System）、搜索引擎（Search Engine）。

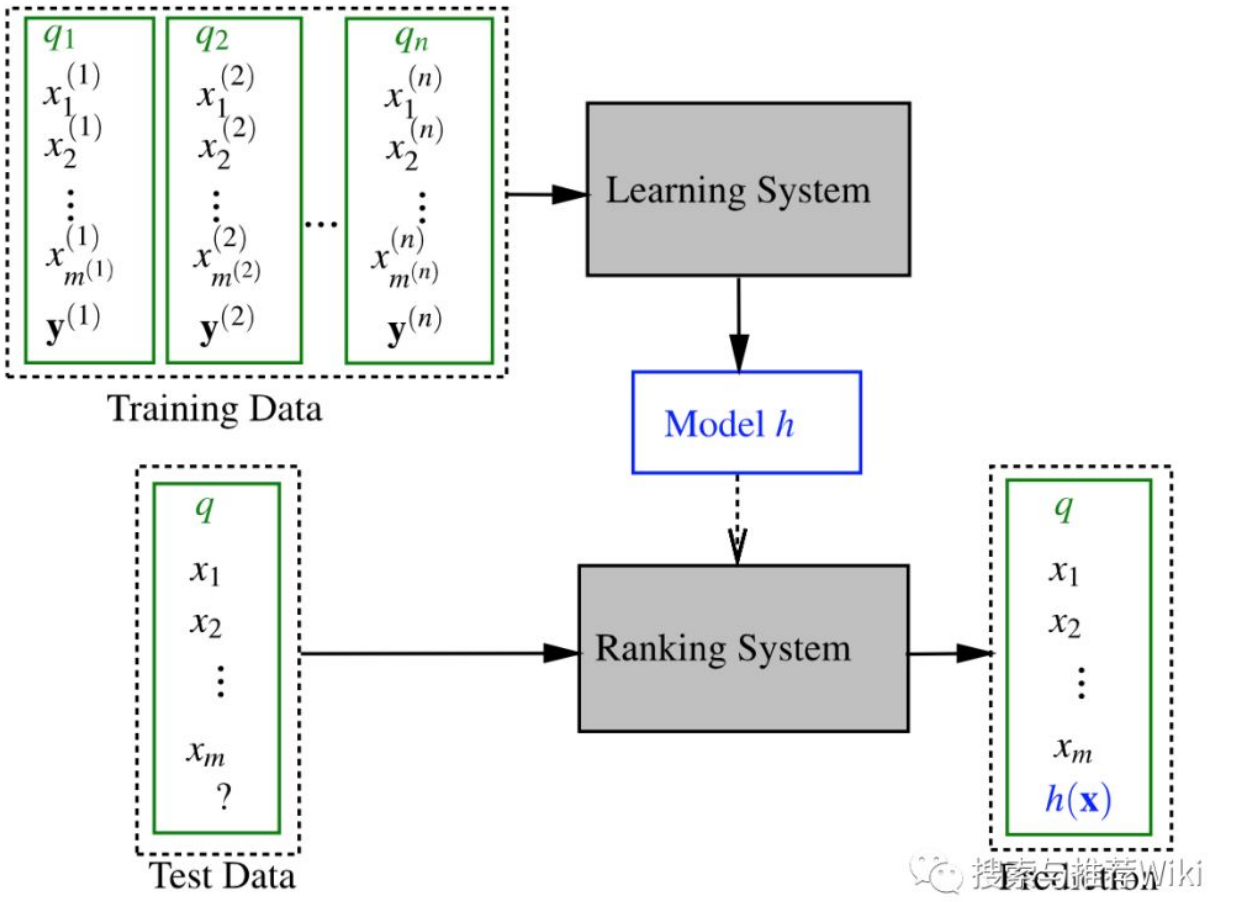
LTR框架

一般来讲，根据机器学习的“四大支柱”，LTR分为三类方法：Pointwise Approach、Pairwise Approach、Listwise Approach。不同的方法通过不同的方式去训练模型，他们定义不同的输入、不同的输出、不同的假设、不同的损失函数。

什么是机器学习的四大支柱？看下图



LTR框架对应的则是排序模型，一个通用的LTR框架如下：



LTR框架

通常来说，一个训练集由n个query组成$q_i(i=1,2,3...,n)$，每一个query都会有一系列与之相关的documents（通常每个documents都有向量表示，向量内容可以是有意义的特征，也可以是embedding得出的向量）$X^i=\{x_j^{(i)}\}_{y=1}^{m^{(i)}}$，$m^{(i)}$表示与$q_i$相关的documents个数，$y$代表documents的正确标签（可能是0或者1，也能是相关度）。上图中对应的流程是：准备训练数据->特征提取->训练模型-数据预测->效果评估

Pointwise Approach

解释

Pointwise是“逐样本”的训练方法，即仅考虑单个样本的得分$h(x_i)$与样本的真实得分$y^{(i)}$的关系。

Pointwise 将问题转化为多分类或回归问题。如果归结为多分类问题，对于某个 Query，对文档与此 Query 的相关程度打标签，标签分为有限的类别，这样就将问题转为多分类问题；如果归结为回归问题，对于某个 Query，则对文档与此 Query 的相关程度计算相关度 Score，这样就将问题归结为回归问题。

例如，对于一组样本数据，用户一段时间内在几个商品类别下的浏览，收藏，分享，购买次数，然后label为喜好等级或是否喜欢，如下表所示（数据伪造只为了说明问题）：

用户	类别	浏览	收藏	分享	购买	喜好等级	是否喜欢
userA	数码产品	43	8	4	2	高	是
userB	生活用户	24	4	2	1	中	否
userC	办公用品	12	1	4	0	低	否
userD	衣服	25	5	1	1	中	是

如上表所示，用户在每个类别下的行为特征有四个，喜好等级分为：高、中、低三档。然后我们可以采用机器学习中的任意一种多分类方法计算用户对物品类别的喜好等级，当然我们也可以采用二分类的方法计算用户是否喜欢这个类别，这和推荐系统中基于CTR来做物品排序和pointwise中的二分类思路是一致的。

缺点

- Pointwise完全从单文档的分类角度计算，没有考虑文档之间的相对顺序。而且它假设相关度是查询无关的，只要$(query，\ y_i)$的相关度相同，那么他们就被划分到同一个级别中，属于同一类。比如在CTR场景中构建的训练集中，所有正样本之间的相关性，Pointwise是不会考虑，同样所有的负样本之间的相关性Pointwise也是不会考虑的。
- 损失函数中没有捕获预测排序中的位置信息，因此，损失函数可能无意的过多强调那些不重要的结果，即那些排序在后面对用户体验影响小的结果。

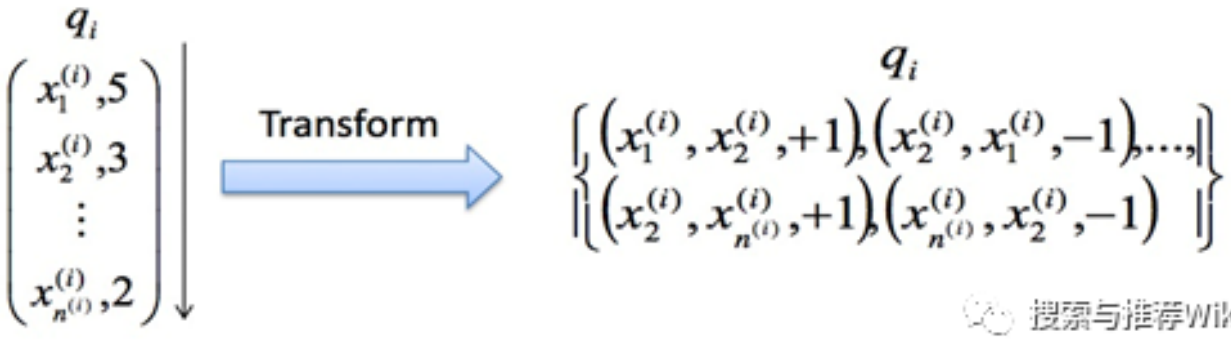
Pairwise Approach

解释

相对Pointwise而言，Pairwise更在乎的是文档之间的顺序，他主要将排序问题归结为二元分类问题，这时候相应的机器学习算法就比较多了，比如Boost、SVM、神经网络等。对于同一query的相关文档集中，对任何两个不同label的文档，都可以得到一个训练实例(y_i, y_j)，如果$y_i > y_j$则赋值+1，反之-1，于是我们就得到了二元分类器训练所需的训练样本了。

Pairwise通常用在搜索系统中，系统接收到用户查询后，返回相关文档列表，所以问题的关键是确定文档之间的先后顺序关系。Pointwise方法完全从单个文档的分类得分角度计算，没有考虑文档之间的顺序关系。文档对方法将排序问题转化为多个pair的排序问题，比较不同文章的先后顺序。

在Pairwise算法中，每个输入数据为一对具有偏序关系（preference relation）的文档，通过对这些数据对的有监督学习来获得一个排序模型，其学习目标是使得结果列表中的错误的偏序对越少越好。



目前公认最为经典的三个Pairwise算法是：基于SVM的Ranking SVM算法、基于神经网络的RankNet算法和基于Boosting的RankBoost算法。

缺点

- 样本数据大多数为有序类别，转化成 pairwise preference 后必定会损失掉一些更细粒度的相关度标注信息。
- 转化成 pairwise preference 后，样本数量将会增加很多倍，在进行模型训练时也会需要更大的计算资源。
- Pairwise 类方法相对 Pointwise 类方法对噪声标注更敏感，即一个错误标注会引起多个pair标注错误。
- Pairwise 类方法仅考虑了pair 的相对位置，同样也没有考虑样本在预测排序中的位置信息，同时也没有考虑同一个label对应的数据的内部依赖性。

Listwise Approach

解释

Pariwise和Pointwise忽视了一个事实就是答案选择就是从一系列候选句子中的预测问题，相对于 Pointwise 和 Pairwise 方法来说，它不再将排序问题转化为一个分类问题或者回归问题，而是直接针对评价指标对文档的排序结果进行优化，在listwise中单一训练样本是：query和它的所有候选回答句子。

ListWise 中常用的优化指标有： MAP 、 NDCG 。常用的ListWise方法有： LambdaRank 、AdaRank、SoftRank、LambdaMART。

优缺点

- 优点：相较 Pointwise、Pairwise 对 ranking 的 model 更自然，解决了 ranking 应该基于 query 和 position 问题。
- 缺点：一些 ranking 算法需要基于排列来计算 loss，从而使得训练复杂度较高。

Pointwise VS Pairwise

在推荐系统领域，Pointwise的应用是比Pairwise更广的，因为效果更好，但是在搜索系统领域Pairwsie表现比Pointwise更加出色。

在搜索系统中，搜索是带 query 的、有意识的被动推荐，对于搜索而言，相关性是及其重要的事情。query 限制了你召回商品相关性，比如 “华为手机”，召回回来一批相似性极高的手机，基于用户的主观诉求也决定了他将高度关注商品之间的细微差别，比如价格、颜色、配置等，因此这些商品才有必要比个高下。

在推荐系统中，推荐是发散的、无意识的主动推荐，相比搜索而言，准确性不再是第一要务（想象下因为你点过一些手机给你出一整屏手机的感觉），多样性是一个必要的指标，这导致了推荐结果极其发散。用户对推荐结果多样性的诉求使得他不关注两个商品之间的比较，对于算法而言不再关注商品之间两两的比较，我只要每个都预测准了，反正最后也要打散的。而且多样性也导致了推荐场景没有像搜索一样适合做 Pairwise 的样本。

因为虽然用的都是学习排序算法，但是在不同的场景中，算法的选择和使用还是有区别的，要因地制宜！

这篇文章分享就到这里，如果你有不懂的可以在留言区留言，当然如果你觉得不错，可以分享给更多人！