

lightGBM用于排序(Learning to Rank)

作者: 小天猫

时间: 2019/12/10 10:45

标签: 机器学习 深度学习 python 人工智能 pytorch



戳这里，加关注喔~

向AI转型的程序员都关注了这个号👉👉👉

机器学习AI算法工程 公众号: datayx

Learning to Rank 简介

去年实习时, 因为项目需要, 接触了一下Learning to Rank(以下简称L2R), 感觉很有意思, 也有很大的应用价值。L2R将机器学习的技术很好的应用到了排序中, 并提出了一些新的理论和算法, 不仅有效地解决了排序的问题, 其中一些算法(比如LambdaRank)的思想非常新颖, 可以在其他领域中进行借鉴。鉴于排序在许多领域中的核心地位, L2R可以被广泛的应用在信息(文档)检索, 协同过滤等领域。

本文将对L2R做一个比较深入的介绍, 主要参考了刘铁岩、李航等人的几篇相关文献, 我们将围绕以下几点来介绍L2R: 现有的排序模型, 为什么需要使用机器学习的方法来进行排序, L2R特征的选取, L2R训练数据的获取, L2R训练和测试, L2R算法分类和简介, L2R效果评价等。

1. 现有的排序模型

排序(Ranking)一直是信息检索的核心研究问题, 有大量的成熟的方法, 主要可以分为以下两类: 相关度排序模型和重要性排序模型。

1.1 相关度排序模型(Relevance Ranking Model)

相关度排序模型根据查询和文档之间的相似度来对文档进行排序。常用的模型包括：布尔模型(Boolean Model)，向量空间模型(Vector Space Model)，隐语义分析(Latent Semantic Analysis)，BM25，LMIR模型等等。

1.2 重要性排序模型(Importance Ranking Model)

重要性排序模型不考虑查询，而仅仅根据网页(亦即文档)之间的图结构来判断文档的权威程度，典型的权威网站包括Google，Yahoo!等。常用的模型包括PageRank，HITS，HillTop，TrustRank等等。

2. 为什么需要使用机器学习的方法来进行排序

对于传统的排序模型，单个模型往往只能考虑某一个方面(相关度或者重要性)，所以只是用单个模型达不到要求。搜索引擎通常会组合多种排序模型来进行排序，但是，如何组合多个排序模型来形成一个新的排序模型，以及如何调节这些参数，都是一个很大的问题。

使用机器学习的方法，我们可以把各个现有排序模型的输出作为特征，然后训练一个新的模型，并自动学得这个新的模型的参数，从而很方便的可以组合多个现有的排序模型来生成新的排序模型。

3. L2R的特征选取

与文本分类不同，L2R考虑的是给定查询的文档集合的排序。所以，L2R用到的特征不仅仅包含文档d本身的一些特征(比如是否是Spam)等，也包括文档d和给定查询q之间的相关度，以及文档在整个网络上的重要性(比如PageRank值等)，亦即我们可以使用相关性排序模型和重要性排序模型的输出来作为L2R的特征。

- 1). 传统排序模型的输出, 既包括相关性排序模型的输出 $f(q,d)$, 也包括重要性排序模型的输出。
- 2). 文档本身的一些特征, 比如是否是Spam等。

4. L2R训练数据的获取

L2R的训练数据可以有三种形式: 对于每个查询, 各个文档的绝对相关值(非常相关, 比较相关, 不相关, 等等); 对于每个查询, 两两文档之间的相对相关值(文档1比文档2相关, 文档4比文档3相关, 等等); 对于每个查询, 所有文档的按相关度排序的列表(文档1>文档2>文档3)。这三种形式的训练数据之间可以相互转换。

训练数据的获取有两种主要方法: 人工标注[3]和从日志文件中挖掘[4]。

人工标注: 首先从搜索引擎的搜索记录中随机抽取一些查询, 将这些查询提交给多个不同的搜索引擎, 然后选取各个搜索引擎返回结果的前K个, 最后由专业人员来对这些文档按照和查询的相关度进行标注。

从日志中挖掘: 搜索引擎都有大量的日志记录用户的行为, 我们可以从中提取出L2R的训练数据。Joachims提出了一种很有意思的方法[4]: 给定一个查询, 搜索引擎返回的结果列表为 L , 用户点击的文档的集合为 C , 如果一个文档 d_i 被点击过, 另外一个文档 d_j 没有被点击过, 并且 d_j 在结果列表中排在 d_i 之前, 则 $d_i > d_j$ 就是一条训练记录。亦即训练数据为: $\{d_i > d_j | d_i \in C, d_j \in L - C, p(d_j) < p(d_i)\}$, 其中 $p(d)$ 表示文档 d 在查询结果列表中的位置, 越小表示越靠前。

5. L2R模型训练

L2R是一个有监督学习过程。

对与每个给定的查询-文档对(query document pair), 抽取相应的特征(既包括查询和文档之间的各种相关度, 也包括文档本身的特征以及重要性等), 另外通过或者人工标注或者从日志中挖掘的方法来得到给定查询下文档集合的真实序列。然后我们使用L2R的各种算法来学到一个排序模型, 使其输出的文档序列和真实序列尽可能相似。

6. L2R算法分类和简介

L2R算法主要包括三类别：PointWise, PairWise, ListWise。

1). PointWise L2R

PointWise方法只考虑给定查询下，单个文档的绝对相关度，而不考虑其他文档和给定查询的相关度。亦即给定查询 q 的一个真实文档序列，我们只需要考虑单个文档 d_i 和该查询的相关程度 c_i ，亦即输入数据应该是如下的形式：

$$\begin{pmatrix} q_i \\ x_1^{(i)}, 5 \\ x_2^{(i)}, 3 \\ \vdots \\ x_{M^{(i)}}^{(i)}, 2 \end{pmatrix} \xrightarrow{\text{Transform}} \begin{pmatrix} q_i \\ (x_1^{(i)}, c_4), (x_2^{(i)}, c_3), \dots, (x_{M^{(i)}}^{(i)}, c_1) \end{pmatrix}$$

$$c_1 < c_2 < c_3 < c_4$$

<https://blog.csdn.net/14326>

微信号: datayx

Pointwise方法主要包括以下算法：Pranking (NIPS 2002), OAP-BPM (EMCL 2003), Ranking with Large Margin Principles (NIPS 2002), Constraint Ordinal Regression (ICML 2005)。

Pointwise方法仅仅使用传统的分类，回归或者Ordinal Regression方法来对给定查询下单个文档的相关度进行建模。这种方法没有考虑到排序的一些特征，比如文档之间的排序结果针对的是给定查询下的文档集合，而Pointwise方法仅仅考虑单个文档的绝对相关度；另外，在排序中，排在最前的几个文档对排序效果的影响非常重要，Pointwise没有考虑这方面的影响。

2). Pairwise L2R

Pairwise方法考虑给定查询下，两个文档之间的相对相关度。亦即给定查询 q 的一个真实文档序列，我们只需要考虑任意两个相关度不同的文档之间的相对相关度： $d_i > d_j$ ，或者 $d_i < d_j$ 。

$$\begin{pmatrix} q_i \\ x_1^{(i)}, 5 \\ x_2^{(i)}, 3 \\ \vdots \\ x_{n^{(i)}}^{(i)}, 2 \end{pmatrix} \xrightarrow{\text{Transform}} \begin{pmatrix} q_i \\ \left\{ \begin{aligned} &(x_1^{(i)}, x_2^{(i)} + 1), (x_2^{(i)}, x_1^{(i)} - 1), \dots \\ &(x_2^{(i)}, x_{n^{(i)}}^{(i)} + 1), (x_{n^{(i)}}^{(i)}, x_2^{(i)} - 1) \end{aligned} \right\} \end{pmatrix}$$

<https://blog.csdn.net/14326>

微信号: datayx

Pairwise方法主要包括以下几种算法：

Learning to Retrieve Information (SCC 1995),

Learning to Order Things (NIPS 1998),

Ranking SVM (ICANN 1999),

RankBoost (JMLR 2003),

LDM (SIGIR 2005),

RankNet (ICML 2005),

Frank (SIGIR 2007),

MHR(SIGIR 2007),

Round Robin Ranking (ECML 2003),

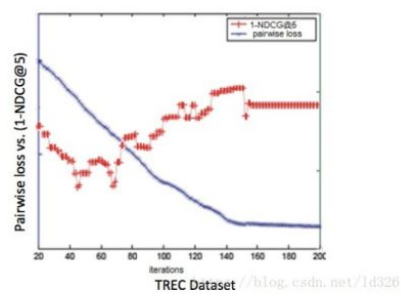
GBRank (SIGIR 2007),

QBRank (NIPS 2007),

MPRank (ICML 2007),

IRSVM (SIGIR 2006) 。

相比于Pointwise方法，Pairwise方法通过考虑两两文档之间的相对相关度来进行排序，有一定的进步。但是，Pairwise使用的这种基于两两文档之间相对相关度的损失函数，和真正衡量排序效果的一些指标之间，可能存在很大的不同，有时甚至是负相关，如下图所示(pairwise的损失函数和NDCG之呈现出负相关性)：

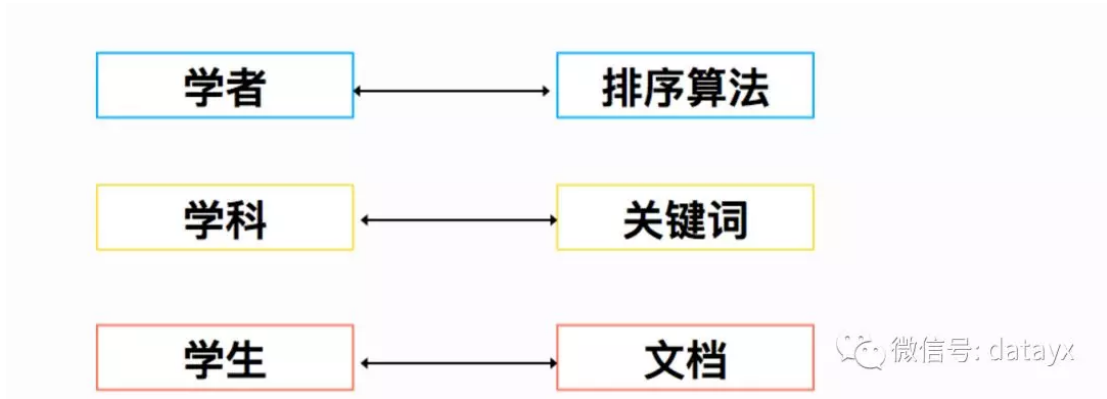


微信号: datayx

另外，有的Pairwise方法没有考虑到排序结果前几名对整个排序的重要性，也没有考虑不同查询对应的文档集合的大小对查询结果的影响(但是有的Pairwise方法对这些进行了改进，比如IR SVM就是对Ranking SVM针对以上缺点进行改进得到的算法)。

列表训练算法

仍然以关键词搜索文章为例，排序学习算法的目标是为给定的关键词对文章列表进行排序。做为类比，假定有一个学者想要对各科学生排名进行预测。各种角色的对应关系如下：



首先我们要告诉学者每个学生的各种属性，这就像我们要告诉排序算法文档特征。对于目标值，我们却有三种方式来告诉学者：

- 对于每个学科，我们可以告诉学者每个学生的成绩。比较每个学生的成绩，学者当然可以算出每个学生的最终排名。这种训练方法被称为Pointwise。对于Pointwise算法，如果最终预测目标是一个实数值，就是一个回归问题。如果目标是概率预测，这就是一个分类问题，例如CTR预估。
- 对于每个学科，我们可以告诉学者任意两个学生的相互排名。根据学生之间排名的情况，学者也可以算出每个学生的最终排名。这种训练方法被称为Pairwise。Pairwise算法的目标是减少逆序的数量，所以是个二分类问题。
- 对于每个学科，我们可以直接告诉学者所有学生的整体排名。这种训练方法被称为Listwise。Listwise算法的目标往往是直接优化nDCG、ERR等评价指标。

这三种方法表面看上去有点像玩文字游戏，但是背后却是工程师和科学家们不断探索的结果。最直观的方案是Pointwise算法，例如对于广告CTR预估，在训练阶段需要标注某个文档的点击概率，这相对来说容易。Pairwise算法一个重要分支是Lambda系列，包括LambdaRank、LambdaMart等，它的核心思想是：很多时候我们很难直接计算损失函数的值，但却很容易计算损失函数梯度(Gradient)。这意味着我们很难计算整个列表的nDCG和ERR等指标，但却很容易知道某个文档应该排的更靠前还是靠后。Listwise算法往往效果最好，但是如何为每个请求对所有文档进行标注是一个巨大的挑战。

3). Listwise L2R

与Pointwise和Pairwise方法不同，Listwise方法直接考虑给定查询下的文档集合的整体序列，直接优化模型输出的文档序列，使得其尽可能接近真实文档序列。

Listwise算法主要包括以下几种算法：

LambdaRank (NIPS 2006),

AdaRank (SIGIR 2007),

SVM-MAP (SIGIR 2007),

SoftRank (LR4IR 2007),

GPRank (LR4IR 2007),

CCA (SIGIR 2007),

RankCosine (IP&M 2007), ListNet (ICML 2007),

ListMLE (ICML 2008) 。

相比于Pointwise和Pairwise方法，Listwise方法直接优化给定查询下，整个文档集合的序列，所以比较好的解决了克服了以上算法的缺陷。Listwise方法中的LambdaMART(是对RankNet和LambdaRank的改进)在Yahoo Learning to Rank Challenge表现出最好的性能。

7. L2R效果评价

L2R是用机器学习的方法来进行排序，所以评价L2R效果的指标就是评价排序的指标，主要包括以下几种：

1) WTA(Winners take all) 对于给定的查询 q ，如果模型返回的结果列表中，第一个文档是相关的，则 $WTA(q)=1$ ，否则为0.

2) MRR(Mean Reciprocal Rank) 对于给定查询 q ，如果第一个相关的文档的位置是 $R(q)$ ，则 $MRR(q)=1/R(q)$ 。

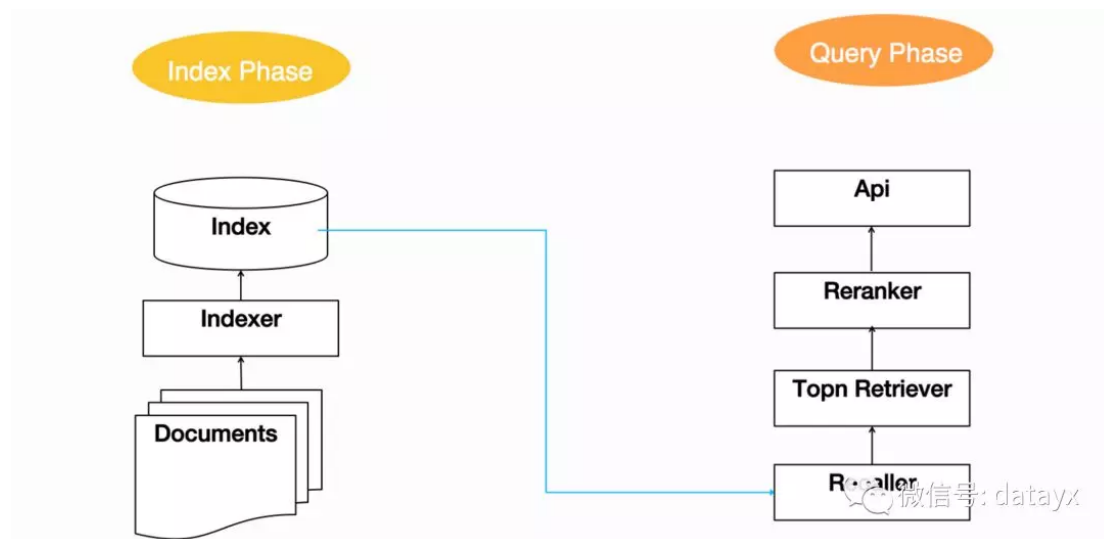
3) MAP(Mean Average Precision) 对于每个真实相关的文档 d ，考虑其在模型排序结果中的位置 $P(d)$ ，统计该位置之前的文档集合的分类准确率，取所有这些准确率的平均值。

4) NDCG(Normalized Discounted Cumulative Gain) 是一种综合考虑模型排序结果和真实序列之间的关系的一种指标，也是最常用的衡量排序结果的指标，详见Wikipedia。

5) RC(Rank Correlation) 使用相关度来衡量排序结果和真实序列之间的相似度，常用的指标是Kendall's Tau。

在线排序架构

典型的信息检索包含两个阶段：索引阶段和查询阶段。这两个阶段的流程以及相互关系可以用下图来表示：



索引阶段的工作是由索引器（Indexer）读取文档（Documents）构建索引（Index）。

查询阶段读取索引做为召回，然后交给Topn Retriever进行粗排，在粗排后的结果里面将前n个文档传给Reranker进行精排。这样一个召回、粗排、精排的架构最初是由Google提出来的，也被称为“Two-Phase Scheme”。

索引部分属于离线阶段，这里重点讲述在线排序阶段，即查询阶段。

三大挑战

在线排序架构主要面临三方面的挑战：特征、模型和召回。

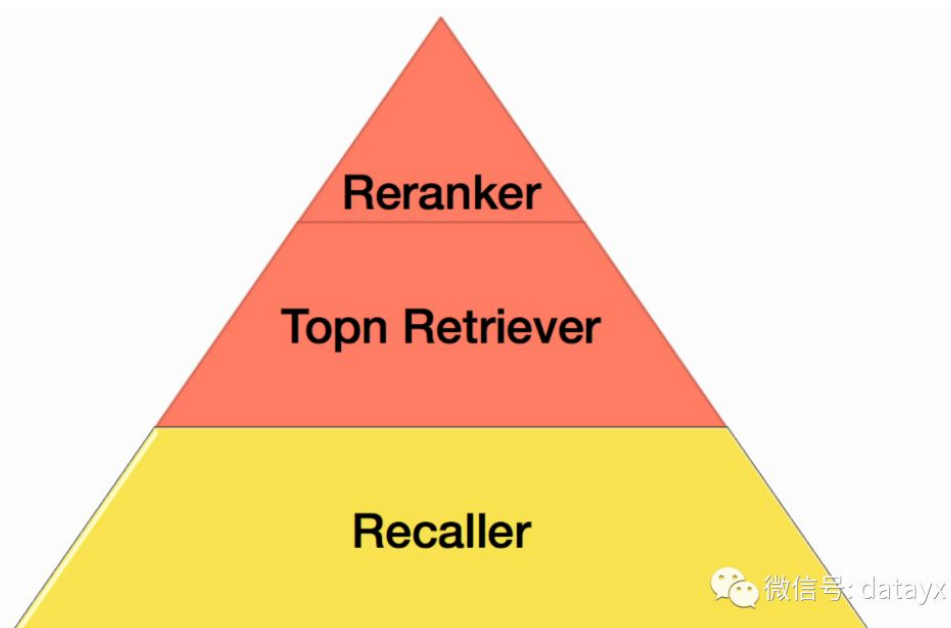
- 特征挑战包括特征添加、特征算子、特征归一化、特征离散化、特征获取、特征服务治理等。
- 模型挑战包括基础模型完备性、级联模型、复合目标、A/B实验支持、模型热加载等。
- 召回挑战包括关键词召回、LBS召回、推荐召回、粗排召回等。

三大挑战内部包含了非常多更细粒度的挑战，孤立地解决每个挑战显然不是好思路。在线排序作为一个被广泛使用的架构值得采用领域模型进行统一解决。Domain-driven design(DDD)的三个原则分别是：领域聚焦、边界清晰、持续集成。

基于以上分析，我们构建了三个在线排序领域模型：召回治理、特征服务治理和在线排序分层模型。

召回治理

经典的Two-Phase Scheme架构如下图所示，查询阶段应该包含：召回、粗排和精排。但从领域架构设计的角度来讲，粗排对于精排而言也是一种召回。和基于传统的文本搜索不同，美团点评这样的O2O公司需要考虑地理位置和距离等因素，所以基于LBS的召回也是一种召回。与搜索不同，推荐召回往往基于协同过滤来完成的，例如User-Based CF和Item-Based CF。

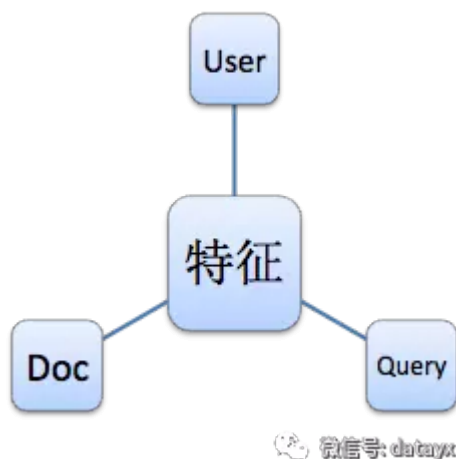


综上所述，召回总体而言分成四大类：

- 关键词召回，我们采用Elasticsearch解决方案。
- 距离召回，我们采用K-D tree的解决方案。
- 粗排召回。
- 推荐类召回。

特征服务治理

传统的视角认为特征服务应该分为用户特征（User）、查询特征（Query）和文档特征（Doc），如下图：



这是比较纯粹的业务视角，并不满足DDD的领域架构设计思路。由于特征数量巨大，我们没有办法为每个特征设计一套方案，但是我们可以把特征进行归类，为几类特征分别设计解决方案。每类技术方案需要统一考虑性能、可用性、存储等因素。从领域视角，特征服务包含四大类：

- 列表类特征。一次请求要求返回实体列表特征，即多个实体，每个实体多个特征。这种特征建议采用内存列表服务，一次性返回所有请求实体的所有特征。避免多次请求，从而导致数量暴增，造成系统雪崩。

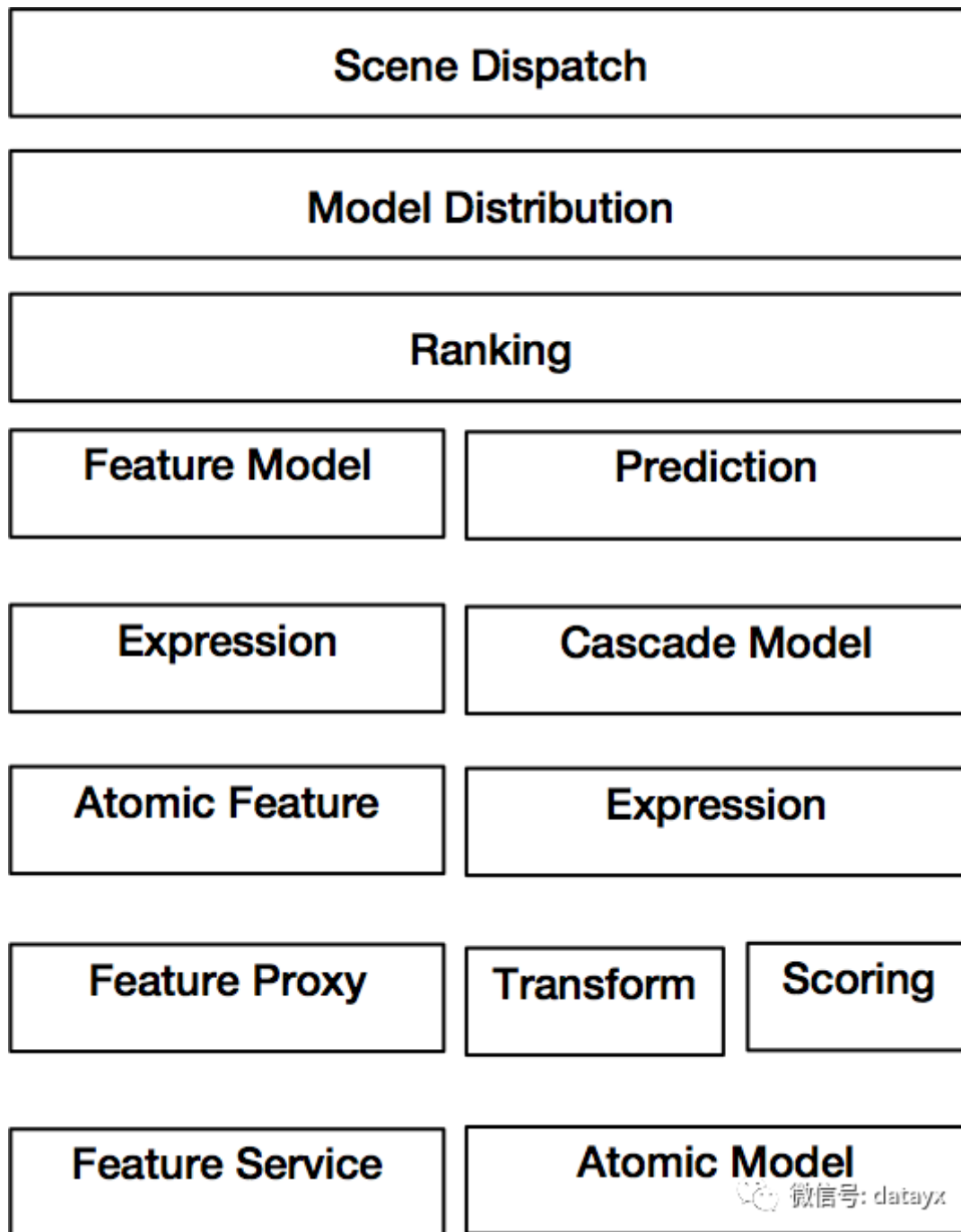
- 实体特征。一次请求返回单实体的多个特征。建议采用Redis、Tair等支持多级的Key-Value服务中间件提供服务。
- 上下文特征。包括召回静态分、城市、Query特征等。这些特征直接放在请求内存里面。
- 相似性特征。有些特征是通过计算个体和列表、列表和列表的相似度而得到的。建议提供单独的内存计算服务，避免这类特征的计算影响在线排序性能。本质上这是一种计算转移的设计。

在线排序分层模型

如下图所示，典型的排序流程包含六个步骤：场景分发（Scene Dispatch）、流量分配（Traffic Distribution）、召回（Recall）、特征抽取（Feature Retrieval）、预测（Prediction）、排序（Ranking）等等。



按照DDD的设计原则，我们设计了如下在线排序分层模型，包括：场景分发（Scene Dispatch）、模型分发（Model Distribution）、排序（Ranking）、特征管道（Feature Pipeline）、预测管道（Prediction Pipeline）。我们将逐一进行介绍。



LTR(learning to rank)经常用于搜索排序中，开源工具中比较有名的是微软的ranklib，但是这个好像是单机版的，也有好长时间没有更新了。所以打算想利用lightgbm进行排序，但网上关于lightgbm用于排序的代码很少，关于回归和分类的倒是一堆。这里我将贴上python版的lightgbm用于排序的代码，里面将包括训练、获取叶结点、ndcg评估、预测以及特征重要度等处理代码，有需要的朋友可以参考一下或进行修改。

其实在使用时，本人也对比了ranklib中的lamdamart和lightgbm，令人映像最深刻的是lightgbm的训练速度非常快，快的起飞。可能lamdamart训练需要几个小时，而lightgbm只需要几分钟，但是后面的ndcg测试都差不多，不像论文中所说的lightgbm精度高一点。lightgbm的训练速度快，我想可能最大的原因可能是：

- a.节点分裂用到了直方图，而不是预排序方法；
- b.基于梯度的单边采样，即行采样；
- c.互斥特征绑定，即列采样；
- d.其于leaf-wise决策树生长策略；
- e.类别特征的支持等

代码

第一部分代码块是主代码，后面三个代码块是用到的加载数据和ndcg。运行主代码使用命令如训练模型使用：python lgb.py -train等。

本文相关代码 项目获取方式：

关注微信公众号 datayx 然后回复 **排序** 即可获取。

AI项目体验地址 <https://loveai.tech>

利用lightgbm做learning to rank 排序，主要包括：

- 数据格式处理
- 模型训练
- 预测
- ndcg评估
- 特征重要度

- 样本的叶结点输出

(要求pip install lightgbm)

```
1 import os
2 import lightgbm as lgb
3 from sklearn import datasets as ds
4 import pandas as pd
5
6 import numpy as np
7 from datetime import datetime
8 import sys
9 from sklearn.preprocessing import OneHotEncoder
10
11 def split_data_from_keyword(data_read, data_group, data_feats):
12     '''
13     利用pandas
14     转为lightgbm需要的格式进行保存
15     :param data_read:
16     :param data_save:
17     :return:
18     '''
19     with open(data_group, 'w', encoding='utf-8') as group_path:
20         with open(data_feats, 'w', encoding='utf-8') as feats_path:
21             dataframe = pd.read_csv(data_read,
22                                     sep=' ',
23                                     header=None,
24                                     encoding="utf-8",
25                                     engine='python')
26             current_keyword = ''
27             current_data = []
28             group_size = 0
29             for _, row in dataframe.iterrows():
30                 feats_line = [str(row[0])]
31                 for i in range(2, len(dataframe.columns) - 1):
32                     feats_line.append(str(row[i]))
33                 if current_keyword == '':
```

 微信号: datayx

```

34         current_keyword = row[1]
35         if row[1] == current_keyword:
36             current_data.append(feats_line)
37             group_size += 1
38         else:
39             for line in current_data:
40                 feats_path.write(' '.join(line))
41                 feats_path.write('\n')
42             group_path.write(str(group_size) + '\n')
43
44             group_size = 1
45             current_data = []
46             current_keyword = row[1]
47             current_data.append(feats_line)
48
49         for line in current_data:
50             feats_path.write(' '.join(line))
51             feats_path.write('\n')
52         group_path.write(str(group_size) + '\n')
53
54 def save_data(group_data, output_feature, output_group):
55     '''
56     group与features分别进行保存
57     :param group_data:
58     :param output_feature:
59     :param output_group:
60     :return:
61     '''
62     if len(group_data) == 0:
63         return
64     output_group.write(str(len(group_data)) + '\n')
65     for data in group_data:
66         # 只包含非零特征


```

 微信号: datayx

```

67     # feats = [p for p in data[2:] if float(p.split(":")[1]) != 0.0]
68     feats = [p for p in data[2:]]
69     output_feature.write(data[0] + ' ' + ' '.join(feats) + '\n') # data[0] => level ;
data[2:] => feats
70
71 def process_data_format(test_path, test_feats, test_group):
72     '''
73     转为lightgbm需要的格式进行保存
74     '''
75     with open(test_path, 'r', encoding='utf-8') as fi:
76         with open(test_feats, 'w', encoding='utf-8') as output_feature:
77             with open(test_group, 'w', encoding='utf-8') as output_group:
78                 group_data = []
79                 group = ''
80                 for line in fi:
81                     if not line:
82                         break
83                     if '#' in line:
84                         line = line[:line.index('#')]
85                     splits = line.strip().split()
86                     if splits[1] != group: # qid => splits[1]
87                         save_data(group_data, output_feature, output_group)
88                         group_data = []
89                         group = splits[1]
90                     group_data.append(splits)
91                 save_data(group_data, output_feature, output_group)
92

```

 微信号: datayx

参考

深入浅出排序学习：写给程序员的算法系统开发实践

<https://tech.meituan.com/2018/12/20/head-in-l2r.html>

阅读过本文的人还看了以下文章：

【全套视频课】最全的目标检测算法系列讲解，通俗易懂！

《美团机器学习实践》_美团算法团队.pdf

《深度学习入门：基于Python的理论与实现》高清中文PDF+源码

python就业班学习视频，从入门到实战项目

2019最新《PyTorch自然语言处理》英、中文版PDF+源码

《21个项目玩转深度学习：基于TensorFlow的实践详解》完整版PDF+附书代码

《深度学习之pytorch》pdf+附书源码

PyTorch深度学习快速实战入门《pytorch-handbook》

【下载】豆瓣评分8.1,《机器学习实战:基于Scikit-Learn和TensorFlow》

《Python数据分析与挖掘实战》PDF+完整源码

汽车行业完整知识图谱项目实战视频(全23课)

李沐大神开源《动手学深度学习》，加州伯克利深度学习（2019春）教材

笔记、代码清晰易懂！李航《统计学习方法》最新资源全套！

《神经网络与深度学习》最新2018版中英PDF+源码

将机器学习模型部署为REST API

FashionAI服装属性标签图像识别Top1-5方案分享

重要开源！CNN-RNN-CTC 实现手写汉字识别

yolo3 检测出图像中的不规则汉字

同样是机器学习算法工程师，你的面试为什么过不了？

前海征信大数据算法：风险概率预测

【Keras】完整实现‘交通标志’分类、‘票据’分类两个项目，让你掌握深度学习图像分类

VGG16迁移学习，实现医学图像识别分类工程项目

特征工程(一)

特征工程(二) :文本数据的展开、过滤和分块

特征工程(三):特征缩放,从词袋到 TF-IDF

特征工程(四): 类别特征

特征工程(五): PCA 降维

特征工程(六): 非线性特征提取和模型堆叠

特征工程(七): 图像特征提取和深度学习

如何利用全新的决策树集成级联结构gcForest做特征工程并打分?

Machine Learning Yearning 中文翻译稿

蚂蚁金服2018秋招-算法工程师（共四面）通过

全球AI挑战-场景分类的比赛源码(多模型融合)

斯坦福CS230官方指南：CNN、RNN及使用技巧速查（打印收藏）

python+flask搭建CNN在线识别手写中文网站

中科院Kaggle全球文本匹配竞赛华人第1名团队-深度学习与特征工程

不断更新资源

深度学习、机器学习、数据分析、python

搜索公众号添加： datayx



长按图片，识别二维码，点关注

机器学习算法资源社群

不断上传电子版PDF资料

技术问题求解

QQ群号： 333972581



长按图片，识别二维码

海淘美妆



微信号: datayx

本文分享自微信公众号 - 机器学习AI算法工程 (datayx)。

如有侵权，请联系 support@oschina.cn 删除。

本文参与“OSC源创计划”，欢迎正在阅读的你也加入，一起分享。

本文地址：<https://my.oschina.net/u/4585416/blog/4398683>

原文地址：