# 3.6 离线排序模型训练

3.6.1 离线排序模型-CTR预估



### 3.6.2 排序模型---下面来看一下排序模型有哪些

* 宽模型 + 特征⼯程
  + LR +特征工程
  + LR/MLR(mixed logistic regression) + 非ID类特征(⼈⼯离散/GBDT/FM(Factorization Machine))
  + mlr介绍https://blog.csdn.net/dpengwang/article/details/105760862
  + FM介绍 <https://zhuanlan.zhihu.com/p/50426292>
* 宽模型 + 深模型
  + wide&deep,DeepFM
* 深度模型
  + DNN + 特征embedding

为什么深度模型在推荐中用的不多呢，因为推荐的特征不够多，数据量不够大

**user\_profile\_hbase是存储在hbase中，映射过来的**

### 3.6.3 特征处理原则

### 3.6.5 spark LR 进行预估

* + 1、需要通过spark读取HIVE外部表，需要新的sparksession配置
    - 增加HBASE配置
  + 2、读取用户点击行为表，与用户画像和文章画像，构造训练样本
    - 用户画像关键词权重：权重值排序TOPK，这里取10个
    - 文章频道号：channel*id, ID类型通常要做one*hot编码，变成25维度(25个频道)---25维度的one-hot
      * **这里由于我们的历史点击日志测试时候是只s有18号频道，所以不进行转换**
    - 文章向量：articlevector
    - 文章权重特征：取出TOPK， 10
    - 总共：10 + 1+ 100 + 10= 121

用户偏好10个，1是频道，100是文章向量，10个article\_profile的特征

* + **对于其中toDF存在一些列没办法确定类型，手动指定DataFrame列的类型**
  + columns = ['article*id', 'user*id', 'channel*id', 'articlevector', 'user*weights', 'article\_weights', 'clicked']
  + VectorAssembler().setInputCols(columns[2:6]).setOutputCol('features').transform(train\_vector)
  + 3、LR模型进行训练
  + 4、LR模型预测、结果评估

### 3.6.6 模型评估-Accuracy与AUC

* + roc = BinaryClassificationMetrics(score\_label) roc.areaUnderROC
  + AUC结果：
  + from sklearn.metrics import roc*auc*score, accuracy\_score
  + 准确率：accuracy\_score(arr[:, 0], arr[:, 1].round())
  + 预测结果有两个概率：
    - [不点击的概率，点击概率]

# 3.8 离线ctr特征中心更新

* 存储形式
  + **存储到数据库HBASE中**

### 3.8.2 用户特征中心更新

* + 目的：计算用户特征更新到HBASE
  + 获取特征进行用户画像权重过滤
    - **获取构造用户的25个频道的特征**

#每个用户都有25个频道特征，每个频道均有10个weights

* + 特征批量存储
    - batch
    - with ctr\_feature\_user.batch(transaction=True) as b:  
       for i in range(len(res)):  
       # user\_id res[i][0]  
       # feature res[i][1]  
       for j in range(25):  
       b.put(str(res[i][0]).encode(), {  
       'channel:{}'.format(j+1).encode(): str(res[i][1][j]).encode()  
       })

### 3.8.2 文章特征中心更新

* **关键词权重**
* **文章的频道**
* **文章向量结果**

**保存到特征数据库中**

hbase

#### 用户和文章特征中心的定时更新

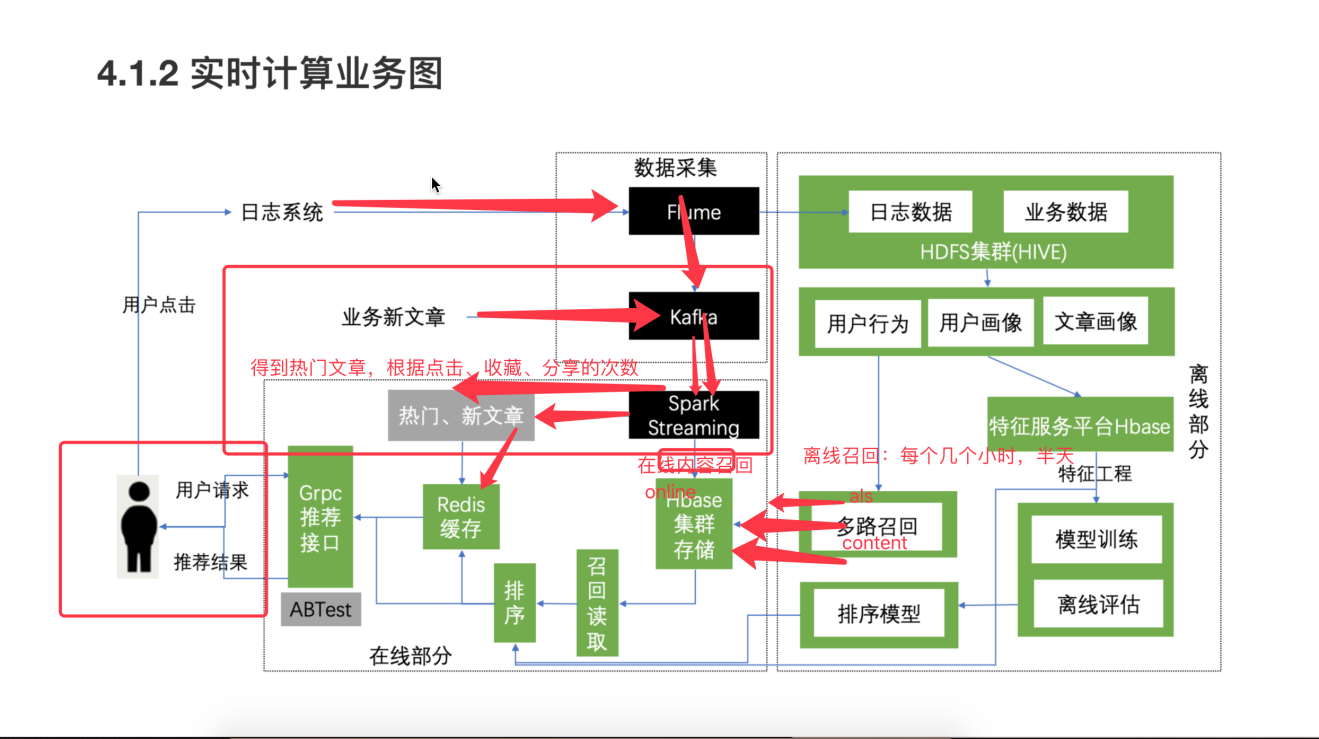
* 添加update
* 添加main.py

Ctr\_lr 主要做的是点击率预估，计算ROC，准确率，计算了用户的特征，文章的特征，存储到hbase

# 4.1 实时计算业务介绍

### 4.1.1 实时计算业务需求

* 解决用户冷启动问题 ----一开始我们没有用户的数据，如何去推荐，这叫冷启动
* 实时计算能够根 据用户的点击实时反馈，快速跟踪用户的喜好



我们把新文章，热门文章推荐放到redis，推荐给新用户，到时候结合排序后的结果，推荐给用户

同时将每个时刻用户产生的点击行为收集到KAFKA当中，等待spark streaming程序去消费，实现在线内容的召回，放到online中，就可以推荐给用户，这就是在线内容召回（因为离线召回需要等几个小时呢）

# 4.2 实时日志分析

### 4.2.1 Flume收集日志到Kafka

* 目的：收集本地实时日志行为数据，到kafka

#### 1、开启zookeeper以及kafka测试

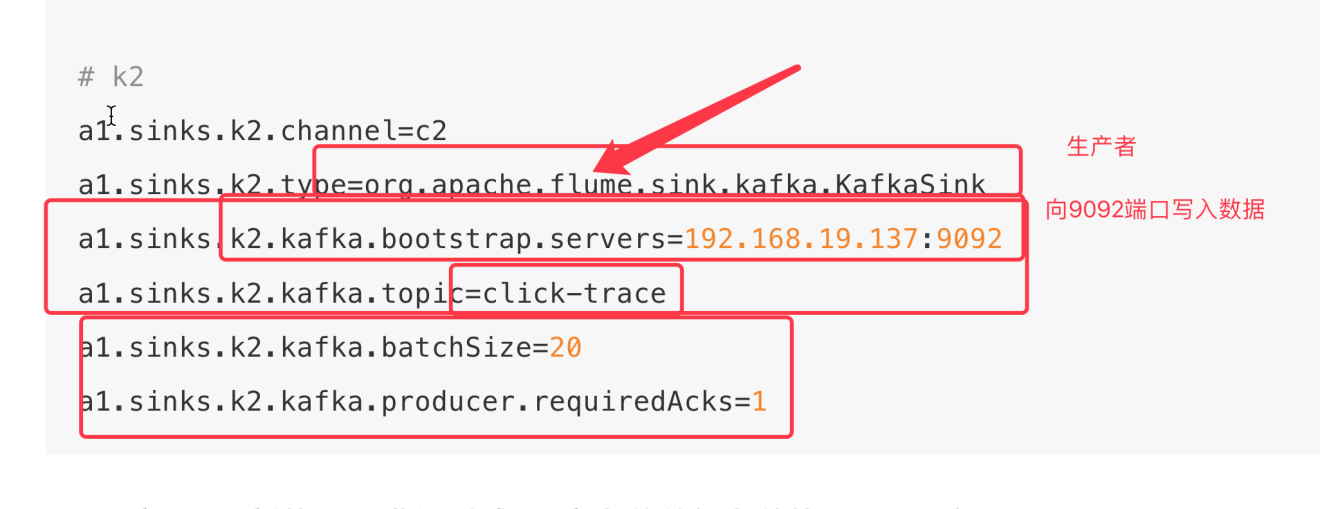
* + zookeeper:默认使用2181（给大家的镜像已经改好了），**记得不要与hBase当中zookeeper端口冲突(22181)，从而实现互不干扰**
  + 1、开启Kafka服务器
  + 2、开启生产者，消费者进行测试，注意两个topic要一致

#### 2、创建flume配置文件，开启flume

vim /root/bigdata/flume/conf/collect\_click.conf

现在的日志配置需要和2.2节 用户行为收集到hive进行对比来理解

* + 添加到kafka的配置
  + 原来的s1,c1都一样
  + 现在添加了k2，c2



* + 重新开启flume读取新配置
  + start collect-click

#### 3、开启kafka进行日志写入测试

* + Kafka的启动配置也做成一个脚本(可以加入zookeeper一起)，supervisor管理

就是都要执行，才能启动kafka

* + **#!/usr/bin/env bash**
  + */root/bigdata/kafka/bin/zookeeper-server-start.sh* -daemon /root/bigdata/kafka/config/zookeeper.properties
  + */root/bigdata/kafka/bin/kafka-server-start.sh* /root/bigdata/kafka/config/server.properties

以上两条都要执行,才能启动kafka

Kafka启动后是卡主的，如果报错就会结束

Kafka启动后有两个进程，一个是kafka，一个是QuorumPeerMain

* + */root/bigdata/kafka/bin/kafka-topics.sh* --zookeeper 192.168.19.137:2181 --create --replication-factor 1 --topic click-trace --partitions 1

The configuration 'requiredAcks' was supplied but isn't a known config.

#### 4、脚本添加以及supervisor管理

这里处理时，参考2.2中的2.3.3 Supervisor进程管理

配置好以后，启动方法是 supervisord -c /etc/supervisord.conf

使用方法是supervisorctl

往日志中写内容没用，flume的问题

# 4.3.1 实时召回实现

* 实时内容召回结果获取存储
* 用户有新行为—>flume—>kafka—>实时spark streaming去读取—>获取这个新行为文章相似的文章—>这些相似文章存储到用户这个召回集结果 'cb\_recall' 'online'
* 目的：对用户日志进行处理，实时达到求出相似文章，放入用户召回集合中
* 1、配置spark streaming信息
  + spark streaming配置
  + kafka配置
  + happybase配置
* 2、读取点击行为日志数据，获取相似文章列表

这会我们没有用jupyter，那么远程运行需要加入下面的这个依赖

把上面的环境变量直接贴入运行--》编辑配置中的---》Environment variables:即可

JAVA\_HOME=/root/bigdata/jdk

SPARK\_HOME=/root/bigdata/spark

HADOOP\_HOME=/root/bigdata/hadoop

PYSPARK\_PYTHON=/miniconda2/envs/py365/bin/python

PYSPARK\_DRIVER\_PYTHON=/miniconda2/envs/py365/bin/python

PYSPARK\_SUBMIT\_ARGS=--packages org.apache.spark:spark-streaming-kafka-0-8\_2.11:2.2.2 pyspark-shell

**原来这里多加了单引号造成的**

放到运行---》编辑配置---》environment variables

get "article\_similar",117180

喂数据

{"actionTime":"2019-04-10 21:04:39","readTime":"","channelId":0,"param":{"action": "click", "userId": "2", "articleId": "117180", "algorithmCombine": "C2"}}

* 3、过滤历史文章集合
* 4、存入召回结果以及历史记录结果

对于A文章，我们实时内容召回需要取，热门召回内容也要取

