干货 | 机器学习算法线上部署方法

2017-01-05 潘鹏举 携程技术中心

作者简介

潘鹏举，携程酒店研发 BI 经理，负责酒店服务相关的业务建模工作，主要研究方向是用机器学习实现业务流程自动化、系统智能化、效率最优化，专注于算法实践和应用。

我们经常会碰到一个问题：用了复杂的GBDT或者xgboost大大提升了模型效果，可是在上线的时候又犯难了，工程师说这个模型太复杂了，我没法上线，满足不了工程的要求，你帮我转换成LR吧，直接套用一个公式就好了，速度飞速，肯定满足工程要求。这个时候你又屁颠屁颠用回了LR，重新训练了一下模型，心里默骂千百遍：工程能力真弱。

这些疑问，我们以前碰到过，通过不断的摸索，试验出了不同的复杂机器学习的上线方法，来满足不同场景的需求。在这里把实践经验整理分享，希望对大家有所帮助。（我们的实践经验更多是倾向于业务模型的上线流程，广告和推荐级别的部署请自行绕道）。

首先在训练模型的工具上，一般三个模型训练工具，Spark、R、Python。这三种工具各有千秋，以后有时间，我写一下三种工具的使用心得。针对不同的模型使用场景，为了满足不同的线上应用的要求，会用不同的上线方法：

一、总结来说，大体分这三种场景，请大家对号入座，酌情使用。

1. **如果是实时的、小数据量的预测应用，则采用的SOA调用Rserve或者python-httpserve来进行应用；这种应用方式有个缺点是需要启用服务来进行预测，也就是需要跨环境，从Java跨到R或者Python环境。**对于性能，基本上我们用Rserver方式，针对一次1000条或者更少请求的预测，可以控制95%的结果在100ms内返回结果，100ms可以满足工程上的实践要求。更大的数据量，比如10000/次，100000/次的预测，我们目前评估下来满足不了100ms的要求，建议分批进行调用或者采用多线程请求的方式来实现。

2. **如果是实时、大数据量的预测应用，则会采用SOA，训练好的模型转换成PMML（关于如何转换，我在下面会详细描述），然后把模型封装成一个类，用Java调用这个类来预测。**用这种方式的好处是SOA不依赖于任何环境，任何计算和开销都是在Java内部里面消耗掉了，所以这种工程级别应用速度很快、很稳定。用此种方法也是要提供两个东西，模型文件和预测主类；

3. **如果是Offline（离线）预测的，D+1天的预测，则可以不用考虑第1、2中方式，可以简单的使用Rscript x.R或者python x.py的方式来进行预测。**使用这种方式需要一个调度工具，如果公司没有统一的调度工具，你用shell的crontab做定时调用就可以了。

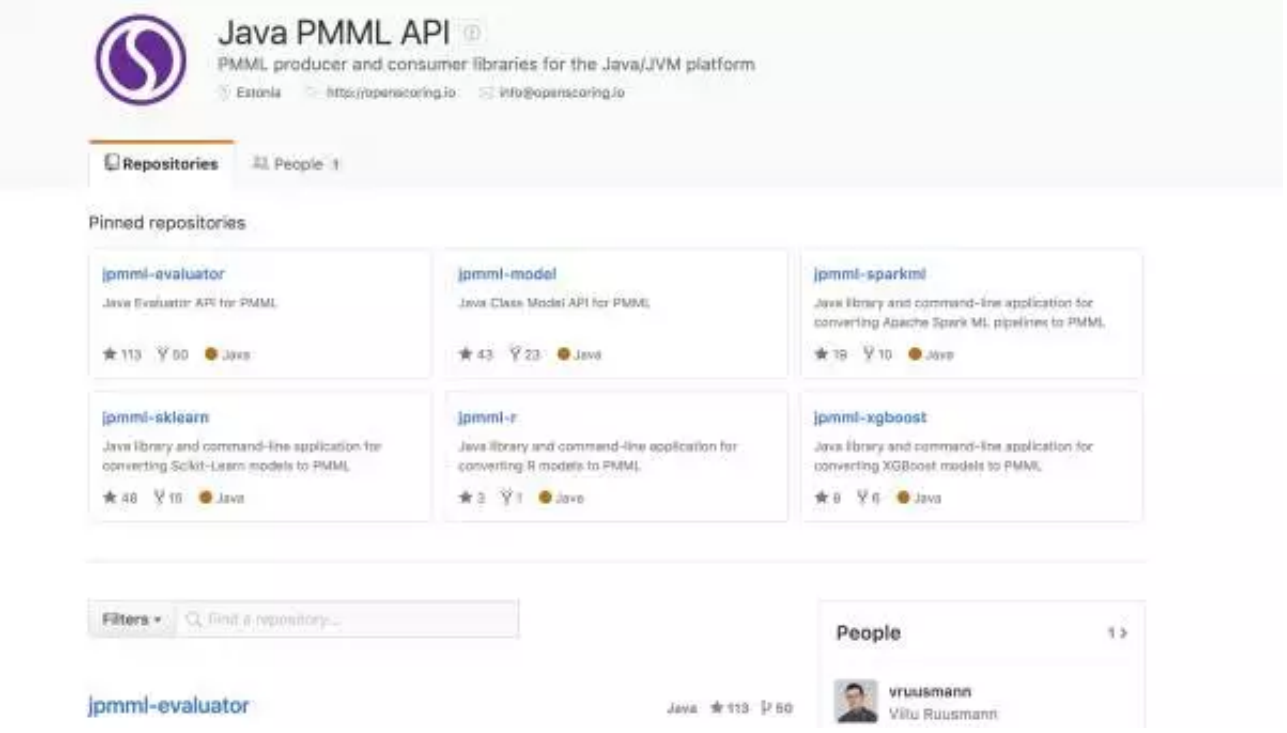
以上三种做法，都会用SOA里面进行数据处理和变换，只有部分变换会在提供的Function或者类进行处理，一般性都建议在SOA里面处理好，否则性能会变慢。

大概场景罗列完毕，简要介绍一下各不同工具的线上应用的实现方式；

二、如何转换PMML，并封装PMML

大部分模型都可以用PMML的方式实现，PMML的使用方法调用范例见：

- jpmml的说明文档：GitHub - jpmml/jpmml-evaluator: Java Evaluator API for PMML（https://github.com/jpmml/jpmml-evaluator）
- Java调用PMML的范例（https://github.com/pjpan/PPJUtils/tree/master/java/pmml），此案例是我们的工程师写的范例，大家可以根据此案例进行修改即可；
- Jpmml支持的转换语言，主流的机器学习语言都支持了，深度学习类除外；
- 从下图可以看到，它支持R、python和spark、xgboost等模型的转换，用起来非常方便；



三、各个算法工具的工程实践：

- **python模型上线：目前使用了模型转换成PMML上线方法**；

  - python-sklearn里面的模型都支持，也支持xgboost，并且PCA，归一化可以封装成preprocess转换成PMML，所以调用起来很方便。
  - 特别需要注意的是：缺失值的处理会影响到预测结果，大家可以可以看一下
  - 用PMML方式预测，模型预测一条记录速度是1ms，可以用这个预测来预估一下根据你的数据量，整体的速度有多少；

- **R模型上线-这块我们用的多，可以用R model转换PMML的方式来实现。**

这里我介绍另一种的上线方式：Rserve。

具体实现方式是：用SOA调用Rserve的方式去实现，我们会在服务器上部署好R环境和安装好Rserve，然后用JAVA写好SOA接口，调用Rserve来进行预测；

- java调用Rserve方式见网页链接：Rserve - Binary R server（http://www.rforge.net/Rserve/example.html）
- centos的Rserve搭建方法见：centos -Rserve的搭建（https://github.com/pjpan/DataScience/blob/master/R/RServe%E7%9A%84%E6%90%AD%E5%BB%BA.md），这里详细描述了Rserve的搭建方式；

Rserve方式可以批量预测，跟PMML的单个预测方式相比，在少数据量的时候，PMML速度更快，但是如果是1000一次一批的效率上看，Rserve的方式会更快；

用Rserve上线的文件只需要提供两个：
- 模型结果文件（XX.Rdata）
- 预测函数（Pred.R）；

Rserve_1启动把模型结果(XX.Rdata)常驻内存。预测需要的输入Feature都在Java里定义好不同的变量，然后你用Java访问Rserve_1，调用Pred.R进行预测，获取返回的List应用在线上。最后把相关的输入输出存成log进行数据核对。

```
Pred.R <- function(x1,x2,x3){
data <- cbind(x1,x2,x3)
# feature engineering
score <- predict(modelname, data, type = 'prob')
return(list(score))
}
```

- **Spark模型上线-好处是脱离了环境，速度快；**

Spark模型的上线就相对简单一些，我们用scala训练好模型（一般性都用xgboost训练模型）然后写一个Java Class，直接在JAVA中先获取数据，数据处理，把处理好的数据存成一个数组，然后调用模型Class进行预测。模型文件也会提前load在内存里面，存在一个进程里面，然后我们去调用这个进程来进行预测。所以速度蛮快的。

- Spark模型上线，放在spark集群，不脱离spark环境，方便，需要自己打jar包；
- 我们这里目前还没有尝试过，有一篇博客写到了如果把spark模型导出PMML,然后提交到spark集群上来调用，大家可以参考一下：Spark加载PMML进行预测（http://blog.csdn.net/fansy1990/article/details/53293024）

四、只用Linux的Shell来调度模型的实现方法-简单粗暴；

因为有些算法工程师想快速迭代，把模型模拟线上线看一下效果，所以针对离线预测的模型形式，还有一种最简单粗暴的方法，这种方法开发快速方便，具体做法如下：

1. 写一下R的预测脚本，比如predict.R，是你的主预测的模型；
2. 然后用shell封装成xx.sh，比如predict.sh，shell里面调用模型，存储数据；

predict.sh的写法如下：

```
# 数据导出
data_filename = xxx
file_date = xxx
result = xxx
updatedt = xxx
cd path
hive -e "USE tmp_xxxdb;SELECT * FROM db.table1;" > ${data_filname};
# R脚本预测
Rscript path/predict.R   $file_date
if [ $? -ne 0 ]
then
echo "Running RScript Failure"
fi
# R预测的结果导入Hive表
list1="use tmp_htlbidb;
load data local inpath 'path/$result'
overwrite into table table2 partition(dt='${updatedt}');"
```

```
hive -e "$list1"
```

3. 最后用Crontab来进行调度，很简单，如何设置crontab，度娘一下就好了：

```
>crontab -e
------------------------
### 每天5点进行预测模型；
0 5 * * * sh predict.sh
```

五、说完了部署上线，说一下模型数据流转的注意事项：

1. **区分offline和realtime数据**，不管哪种数据，我们根据key和不同的更新频次，把数据放在redis里面去，设置不同的key和不同的过期时间；
2. 大部分**redis数据都会存放两个批次的数据**，用来预防无法取到最新的数据，则用上一批次的数据来进行填充；
3. 针对offline数据，用调度工具**做好依赖**，每天跑数据，并生成信号文件让redis来进行读取；
4. 针对realtime数据，我们区分两种类型，一种是历史+实时，比如最近30天的累计订单量，则我们会做两步，第一部分是D+1之前的数据，存成A表，今天产生的实时数据，存储B表，A和B表表结构相同，时效性不同；我们分别把A表和B表的数据放在Redis上去，然后在SOA里面对这两部分数据实时进行计算；
5. **模型的输入输出数据进行埋点，进行数据跟踪**，一是用来校验数据，二来是用来监控API接口的稳定性，一般性我们会用ES来进行log的查看和性能方面的监控。
6. **任何接口都需要有容灾机制，**如果接口超时，前端需要进行容灾，立即放弃接口调用数据，返回一个默认安全的数值，这点对于工程上非常重要。

以上就是我们在模型部署的经验分享，欢迎大家一起来探讨相关工程上的最佳实践。

延伸阅读：

- 什么才是好的工程师文化
- 如何用纯CSS方式实现内容区域的更多展示效果
- 深度剖析服务发现组件Netflix Eureka
- 携程上万坐席呼叫中心异地双活架构及系统设计
- 携程如何把大数据用于实时风控

啊，你看到这里啦~

携程技术中心目前开设了架构/移动/大数据/前端/运维5个微信群，方便关注同一领域的小伙伴们交流。我们也会在群里及时同步携程技术中心主办的相关话题线上和线下活动。

如想进群交流，请加携程技术中心小助手个人微信号ctrip_tech，标注相关领域（如大数据），之后小助手会拉你入群哦。

喜欢我们的会点赞，爱我们的会分享~