机器学习平台开发手册

算法平台部的机器学习平台旨在为广大开发者（包括但不限于数据分析师、数据科学家、算法工程师）提供统一、易用、高效、完善的机器学习相关功能和服务。该手册用于解释、描述该机器学习平台的使用、开发以及代码贡献等步骤和方法。该文档包含两个部分，即：

1. 使用者开发手册：详尽描述平台的使用、开发步骤和注意事项；
2. DeveloperAPITutorial：描述平台代码贡献者该如何补充、完善平台功能；
3. 使用者开发手册
4. 运行环境

客户机：10.77.16.104

用户：weibo\_bigdata\_ds

用户密码：d123w123

准备运行环境：source $ENV\_DIR/env.sh

1. 开发步骤
2. 准备运行脚本pipeline.sh，内容如下

(参考/data0/work\_space/wulei3/weispark/pipelineLR/pipeline.sh)：

#!/bin/bash

/data0/work\_space/service/spark-2.0.0-bin-hadoop2.4/bin/spark-submit \

--jars /usr/local/hadoop-2.4.0/share/hadoop/common/lib/hadoop-lzo-cdh4-0.4.15-gplextras.jar \

--master yarn \

--deploy-mode client \

--num-executors 10 \

--executor-cores 2 \

--executor-memory 4g \

--class ml.Runner weipark.jar \

pipeline.xml

1. 开发配置文件pipeline.xml

配置文件pipeline.xml为标准XML文件格式，XML标签结构如下所示：

(参考/data0/work\_space/wulei3/weispark/pipelineLR/pipeline.xml)

<configuration>

<spark>

<property name=”Spark相关属性名”>属性值</property>

</spark>

<pipeline>

<stagename=“对本次Pipeline作业命名”>

<processid=“处理任务编号1”>任务名1</process>

<processid=“处理任务编号2”>任务名2</process>

</stage>

</pipeline>

<processes>

<processname=“任务名1”>

<objectname>调用类名1</objectname>

<args>

<argname=“参数名”>参数值</arg>

<argname=“参数名”>参数值</arg>

</args>

</process>

<processname=“任务名2”>

<objectname>调用类名2</objectname>

<args>

<argname=“参数名”>参数值</arg>

<argname=“参数名”>参数值</arg>

</args>

</process>

</processes>

</configuration>

1. 不同模型pipeline.xml开发详解
2. Logistic Regression

首先，将/data0/work\_space/wulei3/weispark/pipelineLR目录下所有内容Copy到您的工作目录；

然后，根据下面对pipeline.xml的内容描述，适当修改pipeline.xml中对应参数，来满足您的处理需求。

Logistic Regression的开发文件pipeline.xml，包含如下几个步骤/阶段：

* 1.生成data.conf文件（createDataConf）；
* 2.原始数据检查（rawDataCheck）；
* 3.生成feature.conf文件（genFeatureConf）；
* 4.训练样本生成、存储（genSampleData）；
* 5.LR模型训练、评估、存储（logisticRegression）；

后面的步骤/阶段依赖于前面步骤/阶段的成功执行，尤其是第一个步骤“1.生成data.conf文件（createDataConf）”在执行成功后，还需要手动调整data.conf文件，因此在您第一次在本平台进行机器学习开发时，强烈建议您逐一运行以上各个步骤/阶段，在每个步骤/阶段成功执行后，观察输出结果进而判断是否达到您期望的效果，然后再编辑开发文件pipeline.xml，继续执行后面的步骤/阶段。

下面，我们将逐一对上面的5各步骤进行详细的讲解，从而让您了解想要达到自己的开发目标，应该修改哪些可配置参数。

1.生成data.conf文件（createDataConf）：

<process name="createDataConf">

<objectname>etl.createDataConf</objectname>

<args>

<arg name="table">feed\_log\_mainfeed\_merge</arg>

</args>

</process>

其中，objectname标签值固定，为etl.createDataConf；name为table的属性，需要您填写准备创建data.conf的Hive表名，如例子中的feed\_log\_mainfeed\_merge。该步骤/阶段的输出为，当前目录下的“Hive表名.data.conf”文件。data.conf文件生成后，需要**人为手动地修改data.conf文件**中的内容。

1. 在行首添加”@flag:列序号“，其中列序号表示的是，Hive表中的哪一列用于当做label，在主feed流训练的例子中，具体值为“@flag:0”；
2. 用#号注释掉不准备用于模型训练的特征列，这里特征的取舍，由业务人员/开发人员决定，可以根据业务经验进行取舍，也可以通过步骤/阶段2.原始数据检查（rawDataCheck）来丢弃掉数据有问题的特征；
3. 根据格式“# format: index@name<feature-conf-name>@map-type@operator@args”为每一个特征列添加map-type、operator、args。这里map-type的可选值为bool、origin、enum，其中bool表示布尔值，如是否包含图片等特征列，origin为特征值在0-1直接的特征列，如历史互动率，enum为连续值，如关注数、赞数、评论数等等；如果map-type为enum，可以指定operator，来对连续数据进行离散化，如log10，log2等，在我们的主Feed流训练中，log10是最常见的；最后的args用于人为指定enum特征列的最大值，业务人员/开发人员根据经验来判断，某个特征列的最大值为多少。

2.原始数据检查（rawDataCheck）：

<process name="rawDataCheck">

<objectname>etl.rawDataCheck</objectname>

<args>

<arg name="table">feed\_log\_mainfeed\_merge</arg>

<arg name="dataConf"></arg>

<arg name="where">dt=20161010</arg>

</args>

</process>

其中，objectname标签值固定，为etl.rawDataCheck；name为table的属性，需要您填写准备进行数据检查的Hive表名，name为dataConf的属性，需要您指定data.conf文件，如若不指定，则默认为‘Hive表名.data.conf’，name为where的属性，需要您指定查询条件，如例子中的“dt=20161010”。该步骤/阶段的输出为，当前目录下的“features\_checking.csv”文件，您可以用Microsoft Excel将该csv打开，对每一个特征的取值范围、中位数、非零比例等度量进行查看，从而决定某些特征的原始数据是否存在问题。

3.生成feature.conf文件（genFeatureConf）：

<process name="genFeatureFile">

<objectname>etl.featureConfFile</objectname>

<args>

<arg name="table">feed\_log\_mainfeed\_merge</arg>

<arg name="dataConf"></arg>

<arg name="where">dt=20161011</arg>

</args>

</process>

其中，objectname标签值固定，为etl.featureConfFile；name为table的属性，需要您填写准备生成feature.conf的Hive表名，name为dataConf的属性，需要您指定data.conf文件，如若不指定，则默认为‘Hive表名.data.conf’，name为where的属性，需要您指定查询条件，如例子中的“dt=20161010”。该步骤/阶段的输出为，当前目录下的“Hive表名.feature.conf”文件，该文件将用于后面的步骤/阶段4.训练样本生成、存储（genSampleData）以及线上的实时预测中。

4.训练样本生成、存储（genSampleData）：

<process name="genSampleData">

<objectname>etl.genSampleHDFS</objectname>

<args>

<arg name="raw-data">/user/weibo\_bigdata\_ds/wulei3/feed\_log\_mainfeed\_merge/dt=20160721</arg>

<arg name="dataConf">feed\_log\_mainfeed\_merge.data.conf</arg>

<arg name="featureConf">feed\_log\_mainfeed\_merge.feature.conf</arg>

<arg name="sample-data">/user/weibo\_bigdata\_ds/wulei3/feed\_log\_mainfeed\_merge.sample.libsvm</arg>

</args>

</process>

其中，objectname标签值固定，为etl.genSampleHDFS；name为raw-data的属性，需要您填写准备生成libsvm样本的原始Hive表文件路径，且该路径为HDFS上的存储路径，切记。name为dataConf的属性，需要您指定data.conf文件，name为featureConf的属性，需要您指定上一步骤/阶段生成的\*.feature.conf文件。注意：data.conf和feature.conf都是本地文件；name为sample-data的属性，需要您填写生成的libsvm样本文件的HDFS存储路径。该步骤/阶段的输出为，由“sample-data”属性指定的HDFS目录下的libsvm格式样本文件，可以通过hdfs dfs –tail命令来查看libsvm样本文件中内容是否符合期望。

5.LR模型训练、评估、存储（logisticRegression）：

<process name="logisticRegression">

<objectname>algorithms.logisticRegression</objectname>

<args>

<arg name="input\_path">/user/weibo\_bigdata\_ds/wulei3/ex.libsvm4</arg>

<arg name="model\_path">logisticRegression.model.20161024</arg>

<arg name="regParam">0.1,0.01</arg>

</args>

</process>

其中，objectname标签值固定，为etl.logisticRegression；name为input\_path的属性，需要您填写训练数据libsvm样本的HDFS存储路径。name为model\_path的属性，需要您指定用于存储模型文件的本地路径，name为regParam的属性，用于指定Logistic Regression模型的超参数，用‘,’逗号隔开。该步骤/阶段的输出为，由“model\_path”属性指定的本地模型文件，可以通过cat或vi命令来查看模型文件格式是否符合期望。

1. NaïveBayes
2. Gradient BoostingDecisionTrees
3. \*\*\*
4. DeveloperAPITutorial