

# 智能向导系统配置文档

版本/修改状态 A/0

## 目录

智能向导系统配置文档.....	1
零、配置要求 操作系统 Linux ubuntu 18.04 版本.....	1
一、安装 Java 1.8 版本 .....	1
1.1 安装环境.....	2
1.2 安装步骤.....	2
二、安装 Mysql 5.7 .....	2
2.1 ubuntu 更换软件源.....	2
2.2 ubuntu 安装 mysql5.7 .....	2
三、配置 hadoop 2.9.2 文件.....	3
四、配置 spark 2.4.5 文件.....	3
五、配置 zookeeper 3.4.14 文件.....	3
六、配置 hbase 的配置文件.....	3
七、配置 Hive 3.1.2 文件 .....	4
八 、配置 kylin 2.6.6 .....	4
8.1 kylin2.6.6 环境配置.....	4
8.2 使用 kylin 建数据 cube.....	5
8.2 可能存在的问题.....	7
九、配置 anaconda 2.5.0.0.....	8
9.1 配置 py3.6.....	8
十、可视化推荐 部署 .....	8
十一、后端 jar 包 部署 .....	9
十二、前端部署.....	11
十三、列推荐配置: .....	12
十四、虚拟化.....	13
十五、参考配置: .....	14

**零、配置要求** 操作系统 Linux ubuntu 18.04 版本

## 一、安装 Java 1.8 版本

参考: <https://www.cnblogs.com/xuliangxing/p/7066913.html>

## 1.1 安装环境

操作系统：Red Hat Enterprise Linux 6 64 位(版本号 6.6)

JDK 版本：1.8

工具：Xshell5、Xftp5

## 1.2 安装步骤

### 1.2.1 下载安装包

下载 Linux 环境下的 jdk1.8，请去（官网）中下载 jdk 的安装文件；

### 1.2.2 解压安装包

解压命令进行解压

```
tar -zxvf jdk-8u131-linux-x64.tar.gz
```

### 1.2.3 修改环境变量

```
vim /etc/profile
```

保存并退出(按：wq!)

`source /etc/profile` 让 profile 文件立即生效

### 1.2.3 测试是否安装成功

- ①、使用 `javac` 命令，不会出现 `command not found` 错误
- ②、使用 `java -version`，出现版本为 `java version "1.8.0_131"`
- ③、`echo $PATH`，看看自己刚刚设置的的环境变量配置是否都正确

## 二、安装 Mysql 5.7

### 2.1 ubuntu 更换软件源

参考：[https://blog.csdn.net/baidu\\_36602427/article/details/86551862](https://blog.csdn.net/baidu_36602427/article/details/86551862)

这里选用阿里云源

### 2.2 ubuntu 安装 mysql5.7

参考：<https://blog.csdn.net/u013064585/article/details/104538415>

先启动 mysql

```
service mysql start
```

查看密码，修改密码

### 三、配置 hadoop 2.9.2 文件

- 1.在 `hadoop/hdfs` 文件夹下分别创建 `name data tmp` 文件夹
2. Hadoop 的 `jobhistory` 服务器端口为 `10020` `web` 端口为 `19888`
- 3.将配置文件中 `yarn-site.xml` 文件的 `name` 为 `yarn.nodemanager.aux-services.spark_shuffle.class` 的 `property` 的 `value` 改为 `org.apache.spark.network.yarn.YarnShuffleService`
4. Hadoop 的 `web` 端口为 `50070`
5. Hadoop 格式化  
`hadoop namenode -format`

### 四、配置 spark 2.4.5 文件

1. Spark 的 `web` 端口为 `8080`
- 2.在 `spark` 文件夹下创建 `/data/work` 文件夹和 `/data/log` 文件夹

### 五、配置 zookeeper 3.4.14 文件

- 1.在 `zookeeper` 文件夹下创建 `zookeeper/data` 文件夹  
并在 `data` 文件夹下创建 `log` 文件夹
- 2.zookeeper 的用户端口定义为 `2181`

### 六、配置 hbase 的配置文件

1. 配置文件 `hbase-env.sh` 中的 `HBASE_MANAGES_ZK` 的值改为 `false`  
即 `HBASE_MANAGES_ZK=false`
2. hbase 在 `hdfs` 中的路径为 `/hbase`

即将 **hbase-site.xml** 文件中的 `hbase.rootdir` 的值改为 `hdfs://master:9000/hbase`

## 七、配置 Hive 3.1.2 文件

1. hive 相关配置可参考 **hive\_config** 文件夹中内容

Hive 与 mysql 连接账户定义为 `username: hiveuser` 密码为 **123**

需要自行在 mysql 中创建并赋予权限。

```
create user 'hiveuser' identified by '123';  
grant all privileges on *.* to 'hiveuser'@'%' with grant option;
```

**hive-site.xml** 中修改: `hive.metastore.schema.verification` 设置成 **false**  
修改 `datanucleus.schema.autoCreateAll` 设置为 **true**

**mysql-connector-java-5.1.48.jar** 放至 `/usr/local/hive/lib` 目录下

在 **mysql** 中创建 **hivemetastore** 数据库

```
CREATE DATABASE hivemetastore
```

初始化 Schema

```
schematool -dbType mysql -initSchema
```

## 八、配置 kylin 2.6.6

### 8.1 kylin2.6.6 环境配置

**etc/profile** 设置环境变量为

```
export KYLIN_HOME=/usr/local/apache-kylin-2.6.6-bin-hadoop3
```

参考: <http://kylin.apache.org/cn/docs/install/index.html>

确认 Linux 安装了 **netstat**

```
apt-get install net-tools
```

检查 kylin 环境:

```
$KYLIN_HOME/bin/check-env.sh
```

```
/usr/local/apache-kylin-2.6.6-bin-hadoop3/bin/check-env.sh
```

启动 kylin

```
/usr/local/apache-kylin-2.6.6-bin-hadoop3/bin/kylin.sh start
```

Web UI is at <http://master.default.svc.cluster.local:7070/kylin>

Eg: <http://39.106.205.230:7070/kylin/login>

<http://47.95.237.68:7070/kylin/login>

kylin 的

用户名: **ADMIN**

密码: **KYLIN**

将项目部署在云服务器上时更改后端代码中 kylin 的 url

Application.properties 文件

```
smartinteraction.ip = 8.141.58.53
```

KylinExecutor

```
private String url = "jdbc:kylin:// master:7070/Daslab";
```

## 8.2 使用 kylin 建数据 cube

建 cube 分为创建和构建两步

1. 创建 cube: [http://kylin.apache.org/cn/docs/tutorial/create\\_cube.html](http://kylin.apache.org/cn/docs/tutorial/create_cube.html)

注意事项:

(1) 新建 project

(2) 加载数据 (一张宽表)

需要注意数据表名称, 比如是否跟已有的表重名, 是否需要 reload 或删掉重新导入

(3) 新建 model

注意命名

针对 Bigbench 数据: dimension 选取时属性中若另外存在同名带 1 的属性, 则选中带 1 的属性, 然后把其他属性都选上; measure 选取时将剩下的都选上

(4) 新建 cube

第三步中选择 bulk add measures, 选中 SUM MAX MIN RAW 下的全部属性

Bulk Add Measures

SUM MAX MIN RAW PERCENTILE

<input checked="" type="checkbox"/>	Name	Parameters	Return Type
<input checked="" type="checkbox"/>	SUM_PRICE	WEBSALES_HOME_MYSHOP.PRICE	double
<input checked="" type="checkbox"/>	SUM_QUANTITY	WEBSALES_HOME_MYSHOP.QUANTITY	bigint
<input checked="" type="checkbox"/>	SUM_DISCOUNT	WEBSALES_HOME_MYSHOP.DISCOUNT	double
<input checked="" type="checkbox"/>	SUM_AGE	WEBSALES_HOME_MYSHOP.AGE	bigint

OK Cancel

第五步中将日期类型的属性放在前面，并修改为 date 类型

## Rowkeys 📘

**Important:** Dimension's position on HBase rowkey is critical for performance. You can drag and drop to adjust the sequence. In short, put filtering dimension before non-filtering dimension, and put high cardinality dimension before low cardinality dimension.

ID	Column	Encoding	Length	Shard By
1	WEBSALES_HOME_MYSHOP.SOLDDATE	date	C...	false
2	WEBSALES_HOME_MYSHOP.ITEMNAME	dict	C...	false
3	WEBSALES_HOME_MYSHOP.PRICE1	dict	C...	false
4	WEBSALES_HOME_MYSHOP.QUANTIT...	dict	C...	false
5	WEBSALES_HOME_MYSHOP.DISCOU...	dict	C...	false

cube engine 选择 Spark

## Cube Engine 📘

Engine Type : Spark

2. 构建 cube: [http://kylin.apache.org/cn/docs/tutorial/cube\\_build\\_job.html](http://kylin.apache.org/cn/docs/tutorial/cube_build_job.html)

3. 判断是否构建成功

(1) cube 状态显示为 READY

Cubes								Search by name
Name	Status	Cube Size	Source Records	Last Build Time	Owner	Create Time	Actions	Admins
bigbench_cube	READY	30.31 MB	26,969	2021-06-15 12:31:36 UTC	ADMIN	2021-06-15 08:36:23 UTC	Action	Action

(2) 可以在 insights 中执行查询

New QuerySaved QueriesQuery History

1

```
select count(*) from bigbench_100g.websales_home_mysshop;
```

Tips: Ctrl+Shift+Space or Alt+Space(Windows), Command+Option+Space(Mac) to list keywords in query box.

Project: Daslab

☒ LIMIT50000Submit

Results

1 ✓ ✕ |

Status: All

Query String

Start Time: 2021-06-18 02:00:48 UTCDuration: 0.33sRerunSave

Status: SuccessProject: DaslabCubes: CUBE[name=bigbench\_cube]

Results (1)

VisualizationExport

EXPR\$0
26969

## 8.2 可能存在的问题

1.问题: jps 发现 Nodemanager 没有起来 缺少 `spark-xxx-yarn-shuffle.jar`

解决方案:

<https://utf7.github.io/2020/09/16/nodemanager-startup-failed/#%E8%A7%A3%E5%86%B3%E5%8A%9E%E6%B3%95>

将`${SPARK_HOME}/yarn/spark-3.0.1-yarn-shuffle.jar` 拷贝到 `${HADOOP_HOME}/share/hadoop/yarn/lib/`

启动 Nodemanager: `${HADOOP_HOME}/bin/yarn --daemon start nodemanager`

2. 10020 连接问题

报错: jps 发现没有 JobHistoryServer 进程

解决方案: 启动 JobHistoryServer

3. spark 资源问题

yarn.site 配置中 根据需求提高最小需求

由 1G 提升 4G

参考 <https://blog.csdn.net/xiaoshunzi111/article/details/51221139>

将 `yarn.scheduler.minimum-allocation-mb` 参数修改为 4096

遇到的问题:

java.io.FileNotFoundException: /usr/local/spark/jars/derbyLocale\_cs.jar (No such file or directory)

参考 <https://blog.csdn.net/liuxiao723846/article/details/80657558>

一些 jar 包需要拷过来

根据报错查原因找包 补充一些 jar 包：

报错信息：

```
at org.apache.tomcat.util.scan.JarFileUrlJar.<init>(JarFileUrlJar.java:65)
```

```
2021-06-02 06:57:29,517 WARN [localhost-startStop-1] scan.StandardJarScanner:146 : Failed to scan  
[file:/usr/local/spark/jars/derbyLocale_cs.jar] from classloader hierarchy
```

java.io.FileNotFoundException: /usr/local/spark/jars/derbyLocale\_cs.jar (No such file or directory)

## 九、配置 anaconda 2.5.0.0

1.安装 Anaconda，最新版即可。

注意安装时全部选 yes（一般默认为 yes，但有的默认为 no）

2.创建新的 conda 环境：

```
conda create -n py27 anaconda python=2.7
```

3.激活新环境：

```
conda activate py27
```

4.安装需要的包：

```
conda install mysql-python=1.2.5
```

```
conda install impyla=0.15.0
```

```
pip install uniout==0.3.7
```

```
pip install thrift-sasl==0.4.2
```

```
pip install kylinpy==2.7.0
```

### 9.1 配置 py3.6

执行命令：

```
conda create -n py3.6 anaconda python=3.6
```

```
conda activate py3.6
```

## 十、可视化推荐 部署

1.启动命令( 版本为 python3 )

```
[ bigbench_10t_sample 数据库名    websales_home_myshop_10000 表名    9 维度    solddate price  
discount category age gender province nationality    维度    int date double double varchar int varchar  
varchar varchar 数据类型    quantity    (感兴趣的维度,这个是由之前列推荐感兴趣的维度) "and quantity1  
between '12' and '22'and solddate between '2020-05-25' and '2020-07-13'and price1 between '769' and  
'5978'and discount1 between '1' and '8'and age1 between '15' and '39'" 查询筛选范围 ]
```

Eg:

```
python/deepeye_kylin/partial_order_kylin_where.py bigbench_10t_sample.websales_home_myshop_10000 9
```



quantity solddate price discount category age gender province nationality int date double double varchar int  
varchar varchar varchar **quantity** "and quantity1 between '12' and '22'and solddate between '2020-05-25'  
and '2020-07-13'and price1 between '769' and '5978'and discount1 between '1' and '8'and age1 between  
'15' and '39'"

## 2. 数据导入

脚本文件：将 create.sh websales\_home\_myshop.txt 文件放入根目录

输入命令： create.sh 表名 数据库名 列名 数据类型

# 十一、后端 jar 包 部署

## 1. 下载代码

```
git clone -b sql-result
```

```
https://e.coding.net/kaimary/Smart_interaction_Project.git
```

2. 在 smartinteract-core 下面修改全局变量 ip 地址为 master 或自己的 ip 地址，修改数据库名,修改表名

注意 anaconda 版本和本机版本的对应

```
smartinteraction.py3env.path = /root/anaconda3/envs/py3.6/bin/python3
```

C:\Users\13780\Desktop\conf\back\_end\Smart\_interaction\_Project\smartinteract\_spark\smartinteract-core\src\main\java\com\daslab\smartinteract\kylin

KylinExecutor.java 文件中

```
public class KylinExecutor {  
  
    private Connection conn;  
    private Driver driver;  
    private String user = "ADMIN";  
    private String pwd = "KYLIN";  
    private String url = "jdbc:kylin://master:7070/Daslab";  
    private JSONArray Schema;
```

```
jdbc:kylin://master:7070/bigbench_10t
```

**bigbench\_10t** 是 kylin 里 project 名字

修改为对应的 project 名

3. 将 SparkExecutor.scala 中

```
.config("spark.sql.warehouse.dir", "hdfs://master:9000/hive/warehouse/
```

改为对应路径的位置 如：

```
.config("spark.sql.warehouse.dir", "hdfs://master:9000/hive/warehouse/bigbench_100g.db")
```

4. 将 PCA\_RandomForest.Py

```
args = ["path", "10.141.212.155", 10010, "", "", "bigbench_1t_sample",  
"websales_home_myshop_10000"]
```

修改为对应的 ip 数据集 数据库

```
args = ["path", "10.141.212.155", "10010", "", "", "bigbench_100g", "websales_home_myshop"]  
kylin_util = KylinUtil('master', '7070', 'ADMIN', 'KYLIN', 'bigbench_100g')
```

5.将: HiveConfig.java

```
conn = DriverManager.getConnection("jdbc:hive2://" + MASTER +  
":10010/bigbench_10t_sample;auth=none");
```

改为:

```
conn = DriverManager.getConnection("jdbc:hive2://" + MASTER +  
":10010/bigbench_100g;auth=none");
```

对应的数据库名: 例如 bigbench\_100g

若仍有报错原代码可 修改为

```
conn =  
DriverManager.getConnection("jdbc:hive2://localhost:10010/bigbench_100g;auth=none");
```

6. mvn 打 jar 包 mvn clean package

打包 smartinteract-core

目标文件夹在

\smartinteract\_spark\smartinteract-core\target

7. anaconda 切换 环境为 py2.7

```
conda activate py2.7
```

8. 上传 jar 包 mv XXX.jar XXX-6.2 jar

9. 输入运行命令:

```
nohup /usr/local/spark/bin/spark-submit --master spark://localhost:7077  
--class com.daslab.smartinteract.SpringBootApplication --driver-memory 8g --  
executor-memory 8g --total-executor-cores 16 smartinteract-core-0.0.1-  
SNAPSHOT.jar > /home/scidb/nohup.out &
```

10.将 ipconfiguration.txt 放在 root 下 其内容为主机地址如:

例如: 10.176.24.40

101.201.237.69

11. jps 查看 sparksubmit 进程是否存在

后端部署

看日志

```
vim /usr/local/apache-kylin-2.6.6-bin-hadoop3/logs/kylin.log
```

确认报错

在 mysql 中创建一个用户

```
insert into users(uid,category,password,username,created_at,updated_at)  
values('1','Home & Kitchen','123','user1','2020-09-28 09:22:19','2020-  
09-28 09:22:19');
```

在 mysql 的 创建 rangetable 表 :

```
CREATE TABLE `rangetable` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `tablename` varchar(100) CHARACTER SET utf8 COLLATE utf8_general_ci NOT
NULL,
  `data1` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data2` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data3` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data4` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data5` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data6` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data7` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data8` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data9` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  `data10` text CHARACTER SET utf8 COLLATE utf8_general_ci NULL,
  PRIMARY KEY (`id`) USING BTREE
) ENGINE = InnoDB AUTO_INCREMENT = 6 CHARACTER SET = utf8 COLLATE =
utf8_general_ci ROW_FORMAT = Dynamic;
SET FOREIGN_KEY_CHECKS = 1;
```

建表语句

## 十二、前端部署

### 1. Apache2 安装

Ubuntu 中 Apache2 安装、配置、卸载

```
apt-get update
```

```
apt-get install apache2
```

2. apache2 默认的几个配置文件：

(1) `/etc/apache2/apache2.conf` 是主要配置文件(这个文件的末尾可以看到，include 了其它所有的配置文件)。

(2) `/etc/apache2/ports.conf` 始终包含在主配置文件中。它用于确定传入连接的侦听端口，默认为 **80**，我们一般都会重新配置新的端口。

```
# If you just change the port or add more ports here, you will likely
also
```

```
# have to change the VirtualHost statement in
```

```
# /etc/apache2/sites-enabled/000-default.conf
```

```
# Listen 80
```

```
<IfModule ssl_module>
```

```
    Listen 443
```

```
</IfModule>
```

(3) 其它配置文件在 `/etc/apache2/sites-enabled`，`/etc/apache2/conf-enabled`，`/etc/apache2/mods-enabled` 目录下。

apache2 的默认 web 目录: `/var/www/html`。(在`/etc/apache2/sites-enabled/000-default.conf` 里可以看到这个 `DocumentRoot /var/www/html` 配置)  
apache2 的默认用户是 `www-data`, 定义在 `/etc/apache2/envvars` 文件中。  
设置默认主页的配置文件`/etc/apache2/mods-enabled/dir.conf`

3. 前端所有的 ip 地址 换成自己的 ip 地址

Eg : [39.105.134.200:31240](http://39.105.134.200:31240)

前端所有的表名

`websales_home_myshop` 也一样替换

(3) apache2 的几个简单命令: 启动、停止、重启、状态

```
/etc/init.d/apache2 [ start | stop | restart | status ]
```

```
service apache2 [ start | stop | restart | status ]
```

---

## 十三、列推荐配置:

- 1、确定 python 版本为 2.7.x, 并且装好所有的包。
- 2、修改 `env.py` 中的 `host` 为 `127.0.0.1`。
- 3、打开`__init__.py`, 检查 `mysql` 和 `hive` 的 `host` 和 `port` 是否正确。检查 `mysql` 和 `hive` 的用户名、密码是否正确。

- 4、在 `mysql` 中创建数据库 `userdb`

```
create database userdb;
```

- 5、在 `mysql` 中创建表 `column_recommend`。

```
CREATE TABLE `column_recommend` (  
  `ID` int(11) NOT NULL AUTO_INCREMENT,  
  `table_name` varchar(4500) DEFAULT NULL,  
  `recommend_col` varchar(4500) DEFAULT NULL,  
  `recommend_range` varchar(4500) DEFAULT NULL,  
  `time_stamp` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`ID`)  
) ENGINE=InnoDB AUTO_INCREMENT=53 DEFAULT CHARSET=latin1;
```

- 6、确定 python 版本为 2.7.x, 并且装好所有的包。

- 7、运行`__init__.py` 脚本即可。

```
python __init__.py databaseName tableName columnName
```

Eg: `python __init__.py bigbench_100g websales_home_myshop 123`

```
String[] args = new String[]{" /root/anaconda3/envs/py2.7/bin/python2.7",  
SCRIPT_DIR + "single-row-chart/__init__.py", dbname, tablename, columnname};  
//  
/root/anaconda3/envs/py2.7/bin/python2.7
```

修改为 自己 linux anaconda 环境的 `python2.7` 的 绝对路径

## 十四、虚拟化

1. 为防止数据不一致 Kill 掉 master 和 slave 上的所有进程

Jps 查看所有 java 进程

2. 查看开放的端口号 : netstat -ptlen

在阿里云服务器上查看开放端口命令

```
netstat -ptlen | grep
```

```
service mysql stop
```

3. 在阿里云 集群信息 菜单的 节点管理 中选择 节点



4. 在每个节点中选择更多 , 选择 VNC 远程连接

输入 VNC 密码

```
docker ps | grep master
```

 查询 Container id 记下这个 id

```
d009cfc66769 692b0f4e2da? "/bin/bash -c 'while'" 2 weeks a
go Up 2 weeks k8s_master-master-6559688498-fs.jh2_default_a18d825f-cd8f-11eb-baf9-00163e106f
a1_0
```

```
docker commit [OPTIONS] CONTAINER [REPOSITORY[:TAG]]
```

OPTIONS 说明:

- -a :提交的镜像作者;
- -c :使用 Dockerfile 指令来创建镜像;
- -m :提交时的说明文字;
- -p :在 commit 时, 将容器暂停。

```
Eg: docker commit -m "version7.1" XXXXXX daslab/vm3:v3
```

完成后生成一个序列号, 证明已经完成

Docker image 查看 该镜像

5. 打包上传 至 Dockerhub

首先 master 和各个 slave 修改容器名为用户名, 防止上传出错

```
Eg: docker tag daslab/vm1:v3 daslab1/vm1:v3
```

```
nohub docker push daslab1/vm1:v3 &
```

## 十五、参考配置：

- 1.配置文档中参考的配置文件见附件 conf 文件夹 /conf/各个配置文件
- 2.脚本文件见/conf/脚本文件
- 3.启动命令参考/conf/run\_master\_new.txt