BUILDING A CLASSIFICATION MODEL WITH LOGISTIC REGRESSION ON REAL TIME BIG DATA USING R, APACHE HADOOP, RHADOOP & APACHE FLUME

*** Arunendra Mishra**

ABSTRACT

This paper encompass of building a classification model with logistic regression on R using open source RHadoop with robust & resilient Apache Hadoop using real time data handling capabilities of Apache Flume. We have integrated Hadoop with Flume to handle real time / streaming big data & used RHadoop to integrate R with HDFS. Then, we used R to build a classification model for log management. The objective of elastic classification model is to classifying logs into relevant & irrelevant. Reason for using streaming data is reduce lag. Time is the most important factor in our world of decision making. As it is said that if we take any correct decision but at inappropriate time; its ultimately INCORRECT.

Apache Hadoop is an open source Java framework for processing and querying vast amounts of data on large clusters of commodity hardware. Hadoop enables scalable, cost-effective, flexible, fault-tolerant solutions. Key components are:

1. Hadoop CORE / COMMON - HDFS, MapReduce, Yarn 2.0
2. Hadoop Essential - PIG, Hive, HBase, Zookeeper, Sqoop, Mahout
3. Hadoop Incubator - Flume, Ambari, Chuckwa etc

Apache Flume is a distributed, reliable, and available system for efficiently collecting, aggregating and moving large amounts of log data from many different sources to a centralized data store.

EVENT : An Event is the fundamental unit of data transported by Flume from its point of origination to its final destination. Event is a

* Consultant - Analytics Management Consulting, KPMG

byte array payload accompanied by optional headers.

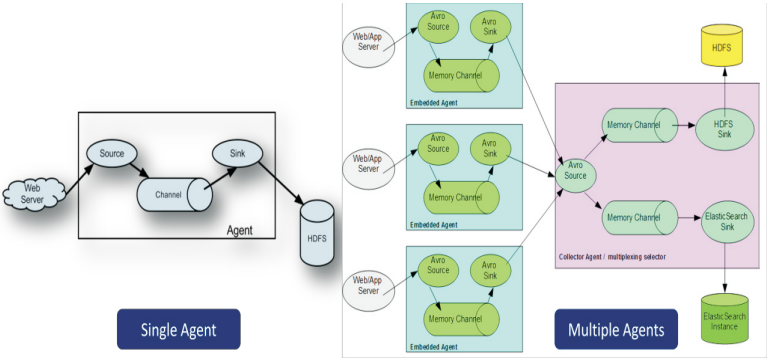
AGENT : A container for hosting Sources, Channels, Sinks and other components that enable the transportation of events from one place to another.

CLIENT : An entity that generates events and sends them to one or more Agents.

SOURCE *(INCOMING EVENTS)* : An active component that receives events from a specialized location or mechanism and places it on one or Channels.

CHANNEL *(EVENT QUEUE)* : A passive component that buffers the incoming events until they are drained by Sinks.

SINK *(OUTGOING EVENTS)* : An active component that removes events from a Channel and transmits them to their next hop



destination.

Apache Flume – Typical Aggregation Flow

R is a scripting language for statistical data manipulation and analysis. It provides a wide variety of statistical and graphical techniques (linear and nonlinear modeling, statistical tests, time series analysis, classification, clustering, ...). It was inspired by, and is mostly compatible with, the statistical language S developed by AT&T.

RHadoop is an open source collection of three R packages created by Revolution Analytics that allow users to manage and analyze data with Hadoop from an R environment.

RHadoop consists of the following packages:

- rmr2 - functions providing Hadoop MapReduce functionality in R
- rhdfs - functions providing file management of the HDFS from within R
- rhbase - functions providing database management for the HBase distributed database from within R

Section 1 - Writing Streaming Data Into HDFS: Hadoop Fully distributed

Prerequisites - Hadoop, R & Flume. We have deployed Hadoop fully distributed mode. Use *JPS* command to check the health of Hadoop setup. Also, we have already deployed Flume. Following is the configuration of Flume conf file. If Hadoop is deployed in pseudo or standalone mode, we need to do few changes in the suggested configuration file.

 # Describe definitions

a1.sources = r1
a1.sinks = hdfs-cluster1-sink
a1.channels = c1

 # Describe/configure the source

a1.sources.r1.type = syslogtcp *(#specifying type of configuration)*
a1.sources.r1.bind = hdfsproxy0 *(#specifying source machine)*
a1.sources.r1.port = 5140 *(#specifying port could be any number)*

 # Use a channel which buffers events in memory

a1.channels.c1.type = memory
a1.channels.c1.capacity = 1000
a1.channels.c1.transactionCapacity = 100

#Bind the source and sink to the channel

a1.sources.r1.channels = c1
a1.sinks.hdfs-cluster1-sink.channel = c1
a1.sinks.hdfs-cluster1-sink.type = hdfs *(# specifying type of sink i.e. here writing data into hdfs)*
a1.sinks.hdfs-cluster1-sink.hdfs.path = hdfs://nn1.bida.loc/user/ubuntu/flume_data
(#above line specifies path of sink machine. mention complete hdfs path)

a1.sinks.hdfs-cluster1-sink.hdfs.filePrefix = log- *(# specifying prefix of file name)*
a1.sinks.hdfs-cluster1-sink.hdfs.fileType = DataStream *(# specifying file type)*
a1.sinks.hdfs-cluster1-sink.hdfs.writeFormat = Text *(# specifying data format)*

Section 2 - Accessing data from HDFS

(# specifying environment variables)
Sys.setenv("HADOOP_CMD"="/home/arunendra/hadoop/hadoop-2.5.0/sbin")
Sys.setenv("HADOOP_STREAMING"="/home/arunendra/hadoop/hadoop-2.5.0/")
Sys.setenv("HADOOP_CONF" = "/home/arunendra/hadoop/hadoop-2.5.0/etc/hadoop")
Sys.getenv("HADOOP_CMD") *(# checking that path is correct & updated)*
(# Installing required R packages)
install.packages(c("rJava", "Rcpp", "RJSONIO", "bitops","digest", "functional", "stringr", "plyr", "reshape2"))
(# specifying path of libraries already downloaded from CRAN- here I am using my paths)
install.packages("/media/arunendra/RHadoop/rhdfs_1.0.8.tar.gz", repos = NULL, type = "source")
install.packages("/media/arunendra/RHadoop/rhbase_1.2.1.tar.gz", repos = NULL, type = "source")
install.packages("/media/arunendra/RHadoop/rmr2_3.2.0.tar.gz", repos = NULL, type = "source")
install.packages("/media/arunendra/RHadoop/rJava_0.9-6.tar.gz", repos = NULL, type = "source")
(# loading libraries)
library(rhdfs) library(rhbase) library(rmr2)
(# Reading & writing a job on data)
hdfs.data<-file.path(hdfs.root, 'data')
hdfs.out<-file.path(hdfs.root, 'out')
else, we can write files in logD using this command *hdfs.logD <-*

file.path(hdfs.root, 'logD'). Now, we can access hdfs from R console. Here we build a data table "logD" consists of all logs to build a classification model.

Section 3 - Building and applying a logistic regression model

We have introduced a variable " rgroup" to segregate Training & Test sets. The Training dataset is used to build the model, relavant variables are "rgroup", "response", "setlog" & "pred".

```
logTrain <- subset(logD,logD$rgroup>=10)
logTest <- subset(logD,logD$rgroup<10)
logVars <- setdiff(colnames(logD),list('rgroup','log'))
logFormula <- as.formula(paste('log=="log"',
paste(logVars,collapse=' + '),sep=' ~ '))
logModel <-
glm(logFormula,family=binomial(link='logit'),data=logTrain)
logTrain$pred <- predict(logModel,newdata=logTrain,
type='response')
logTest$pred <- predict(logModel,newdata=logTest, type='response')
print(with(logTest,table(y=log,glmPred=pred>0.5)))
sample <- logTest[c(7,35,224,327),c('log','pred')]
```

Now, we have build the model. To evaluate this we need to use following tools.

1. CONFUSIONMATRIX <- table(truth=logTest$log,prediction=logTest$pred>0.5)
2. Useful measures - Accuracy, Precision, Recall, Sensitivity, Specificity

References:

Books

- An Introduction to R by The R Core Team:http://cran.r-project.org/doc/manuals/Rintro.html
- The R Book by Michael J. Crawley
- R Cookbook by Paul Teetor
- ggplot2: Elegant Graphics for Data Analysis by Hadley Wickham
- Julian J. Faraway - Practical Regression and Anova using R
- Nina Zumel and John Mount - Practical Data Science with R

- An Introduction to Statistical Learning, with Applications in R by James, Witten, Hastie and Tibshirani (Springer, 2013)
- Trevor Hastie and Robert Tibshirani - Stanford StatisticalLearning

Websites

- http://www.rproject.org/
- http://flume.apache.org/
- http://hadoop.apache.org/
- Rbloggers: http://www.rbloggers.com/
- Impetus : Running Map-Reduce jobs in Hadoop with R (http://blogs.impetus.com)
- https://www.udemy.com/blog - R, Hadoop, and How They Work Together
- http://archive.cloudera.com/cdh/3/flume/User Guide
- RHadoop and MapR - Revolution analytics & MapR Technologies
- https://github.com/