Rapid Deployment of Automatic Scoring Models to Hadoop Production Systems

System Overview, Case Study, and Recommendations
About me:

- Studied Statistics in the Hebrew University in Jerusalem, with a focus on non-numerical data and graphical models.
- Consultant for bio-tech, gambling and IT companies.
- Since 2014 data mining analyst at InnoGames GmbH in Hamburg, Germany

About InnoGames:

- Started as a hobby-project in 2003 by our managing directors; The first game “Tribal Wars” is still growing.
- F2P business model: moved from browser to cross platform developer & publisher.
- +350 employees from 30 nations.
- 135M registered player.
Motivation

Data Scientists are data consumers, but have a hard time becoming data suppliers. I want to help data scientists become efficient suppliers of data.
Production Cycle

Red tape, bugs, QA

2X Work, Bugs, Compatibility

Rollout

HDFS

Data Extraction

Science

Adapt
Production Cycle

Less red tape

Rollout

HDFS

Science

Data Extraction
The basic building blocks of RHadoop.
- Allows building MapReduce functions in R

HDFS file management from within R (hdfs terminal)
- Read, write, copy, rename, etc., etc.

A higher level function toolkit.
- This is the package the Data Scientist will most likely want to use when analyzing data en masse.

Hive functionality in R.
- Great replacement for ODBC driver issues
- Also contains HDFS functions
- Doesn’t work on Windows
Setup

- Data Scientist
- Hadoop
- R Server

- Development Environment
- Data
- Runs Product
get_score(get_data, model_data, make_output, ...)

get_score(get_data, model_data, make_output,...)
get_score(get_data, model_data, make_output, ...)

Process
> require('RHive')
> require('mixtools')
>
> Sys.setenv("HADOOP_CMD" = '/usr/bin/hadoop')
> Sys.setenv("HADOOP_STREAMING" = '/usr/lib/hadoop-mapreduce/hadoop-streaming.jar')
>
> require('plyrmr')
>
> rhive.init(hiveHome = '/usr/lib/hive',
+ hadoopHome = '/usr/lib/hadoop',
+ hadoopConf = '/etc/hadoop/conf/')
>
> rhive.connect(host = '<ip_address>',
+ port = 10000,
+ hiveServer2 = T,
+ defaultFS = 'hdfs_namenode_url',
+ user = 'hdfs',
+ password = '')
get_data_mvnormal <- function(table.name){
  query <- paste('select * from', table.name)
  return(rhive.query(query = query))
}

model_data_mvtnorm_em <- function(data){
  res <- mvnormalmixEM(data)
  score <- apply(res$posterior, 1, which.max)
  return(cbind(data, score))
}

make_output_matrix_to_hive <- function(score.df){
  rhive.write.table(score.df, 'results', rowName = F)
}
```r
get_score <- function(get_data, model_data, make_output, ...)
{
  extra_input <- list(...)

  Get_Data <- match.fun(get_data)
  Model_Data <- match.fun(model_data)
  Make_Output <- match.fun(make_output)

  Get_Data(extra_input$table_name) %>%
    Model_Data

  Make_Output(results)

  return(0)
}
```
```r
> get_data_mvnormal <- function(table.name){
+   query <- paste('select * from', table.name)
+   return(rhive.query(query = query))
+ }
```

```r
> model_data_mvtnorm_em <- function(data){
+   res <- mvnormalmixEM(data)
+   score <- apply(res$posterior, 1, which.max)
+   return(cbind(data, score))
+ }
```

```r
> make_output_matrix_to_hive <- function(score.df){
+   rhive.write.table(score.df, 'results', rowName = F)
+ }
```
get_data_mvnormal <- function(tbl_name_1, tbl_name_2){
  query_1 <- paste('select * from', tbl_name_1)
  query_2 <- paste('select * from', tbl_name_1)
  res_1 <- rhive.query(query = query_1)
  res_2 <- rhive.query(query = query_2)
  p_res <- combine_and_process_data(res_1, res_2)
  return(p_res)
}

model_data_mvtnorm_em <- function(data){
  res <- mvnormalmixEM(data)
  score <- apply(res$posterior, 1, which.max)
  return(cbind(data, score))
}

make_output_matrix_to_hive <- function(score.df){
  rhive.write.table(score.df, 'results', rowName = F)
}

(Greatly) Increasing Complexity

main.R

Get_score

get_data
All users with a playtime of less than 58 minutes in the fourth week are more likely to churn than to remain active (score above 0.5)
The red-er the more likely to churn.

Bigger Bubble = More Importance.