

BEYOND A SINGLE INSTANCE

USING THE PUBLIC CLOUD TO SCALE YOUR CAPABILITIES

Created by [Eoin Brazil](#) / [@eoinbrazil](#)

<http://github.com/braz/DublinR-BeyondASingleInstance>

RECAP - JOURNEY TO HERE

Two previous ML talks to Dublin R User Group and ended with an example of running RStudio in EC2. This talk picks up that journey and looks to the issues of scaling beyond the single instance.

- Github: <https://github.com/braz/DublinR-ML-treesandforests>
- Slides: <https://speakerdeck.com/braz/introduction-to-machine-learning-with-r>
- Github: <https://github.com/braz/DublinR-ML-machine>
- Slides: <https://speakerdeck.com/braz/machine-learning-of-machines-with-r>

TALK OUTLINE

- Scaling Landscape
- Cluster Landscape
- Examples

SCALING IN R

Variety of Packages/Libraries

- High Level
- Low Level
- Tailored for specific tasks

ON A SINGLE MACHINE

multicore

foreach

parallel (core)

ON MULTIPLE MACHINES

Rmpi

nws

snow or snowfall

SPRINT

SYNCHRONOUS VERSUS ASYNCHRONOUS

Continuous communication to 'Master' R process or a job scheduler with batched jobs managing the resources.

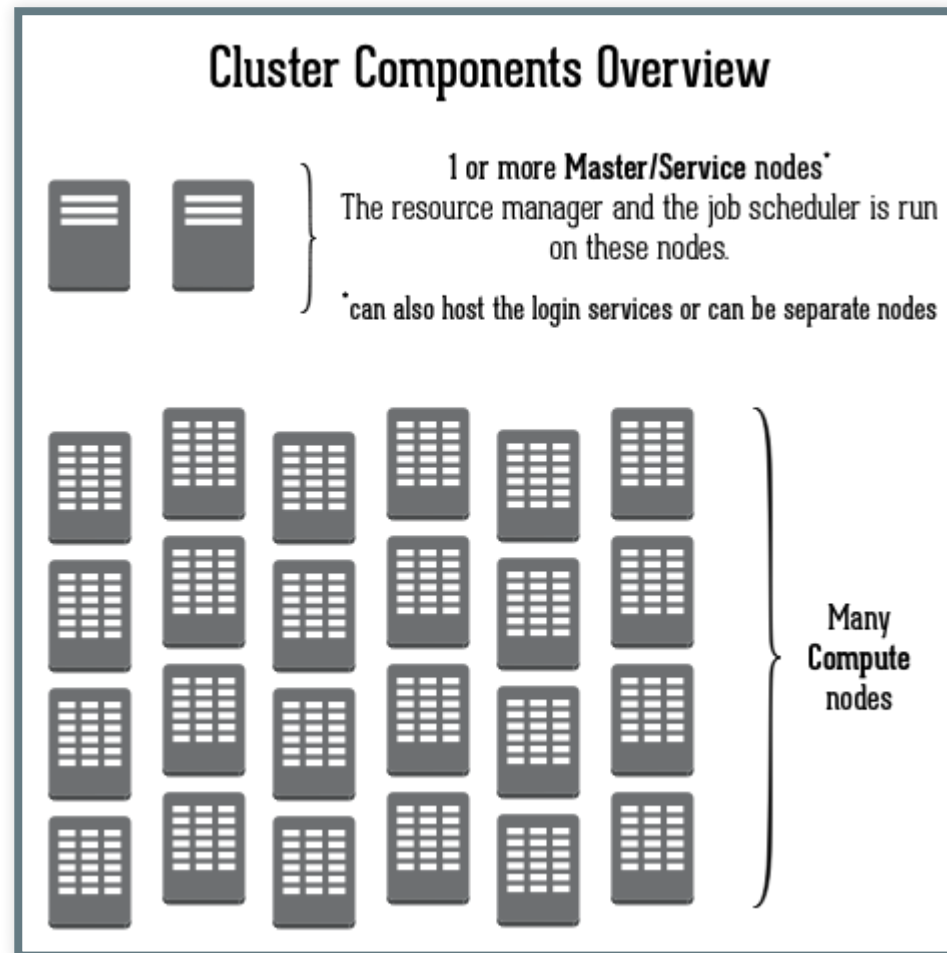
Aim to optimally exploit available computational resources.

ENSURE YOU CONFIGURE YOUR INSTANCE

- ext4 or xfs - mount with noatime
- I/O Scheduler – use deadline or noop
- Use EBS-optimized instances
- Enable enhanced networking (SR-IOV) - install ixgbevf module & set sriovNetSupport

```
sudo echo performance | sudo tee /sys/devices/system/cpu/cpuX/cpufreq/scaling_governor > /dev/null
sudo echo 120 | sudo tee /proc/sys/net/ipv4/tcp_keepalive_time > /dev/null
sudo echo never | sudo tee /sys/kernel/mm/transparent_hugepage/enabled > /dev/null
sudo echo never | sudo tee /sys/kernel/mm/transparent_hugepage/defrag > /dev/null
sudo echo 0 | sudo tee /sys/kernel/mm/transparent_hugepage/khugepaged/defrag > /dev/null
sudo echo 0 | sudo tee /proc/sys/vm/zone_reclaim_mode > /dev/null
sudo echo tsc | sudo tee /sys/devices/system/clocksource/clocksource0/current_clocksource > /dev/null
```


A TYPICAL CLUSTER



WHAT DOES A WORKLOAD MANAGER DO ?

Allocates access to resources for period.

Manages starting, stopping, monitoring work on set of allocated nodes.

Manages queue/s of work against available resources.

WORKLOAD MANAGERS / SCHEDULERS

- Terascale Open-source Resource and QUEue Manager (Torque)
- Load Sharing Facility (LSF)
- Simple Linux Utility for Resource Management (SLURM)
- Maui
- Portable Batch System (PBS)
- OpenLava
- Sun Grid Engine (SGE)
- Many others....

BATCHJOBS - MAP/REDUCE R PACKAGE

- Interactive on local machine
- Parallel/multi-core on local machine
- Distributed on SSH cluster
- Distributed/queued on cluster

BATCHJOBS - CRAN & GITHUB

To install, do:

- `install.packages("BatchJobs")`

Documentation (e.g. Technical Report):

- <https://github.com/tudo-r/BatchJobs>

EXAMPLES

- k-Means
- randomForest
- glmnet
- Random x-y
- US Census
- cDNA microarray
- Credit scoring

GEN-DATA.R

```
nrow <- 100000
sd <- 0.5
real.centers <- list( x=c(-1.3, -1.1, -0.7, -0.4, -0.1, +0.3, -0.5, +0.7
                        y=c(-1.0, +0.5, +1.0, -0.3, +0.1, +0.5, +0.2, -1.3
data <- matrix(nrow=0, ncol=2)
colnames(data) <- c("x", "y")
for (i in seq(1, 10)) {
  x0 <- rnorm(nrow, mean=real.centers$x[[i]], sd=sd)
  y0 <- rnorm(nrow, mean=real.centers$y[[i]], sd=sd)
  data <- rbind( data, cbind(x0,y0) )
}
write.csv(data, file='dataset.csv', row.names=FALSE)
```

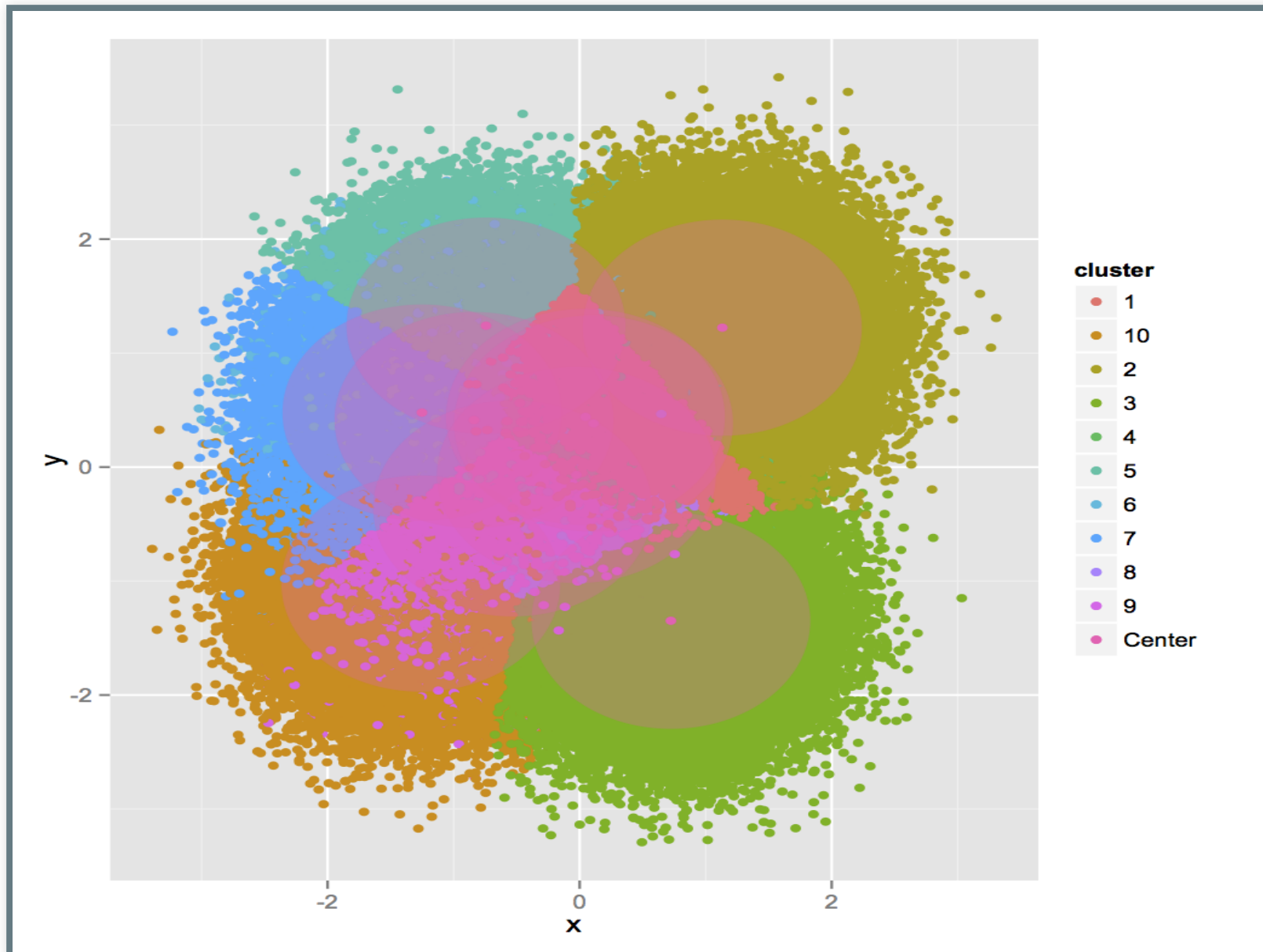
Code Source: <https://github.com/glennklockwood/paraR/blob/master/kmeans/gen-data.R>

SERIAL.R

```
library(ggplot2)
#load the generated dataset
data <- read.csv('dataset.csv')
# run k-means and classify the data into clusters
result <- kmeans(data, centers=10, nstart=100)
# print the cluster centers based on the k-means run
print(result$centers)
data$cluster = factor(result$cluster)
centers = as.data.frame(result$centers)
plot = ggplot(data=data, aes(x=x, y=y, color=cluster )) + geom_point() +
print(plot)
```

Code Source: <https://github.com/glennklockwood/paraR/blob/master/kmeans/serial.R>

Visualisation of the k-means clustering



ADULT DATASET

It is an extract from the 1994 US census database of 32,563 rows. It covers a range of demographic information for a set of citizens including education, race, gender, marital status and can be used for a variety of purposes including building models to predict key measures like income.

CLASSIFIER PERFORMANCE

| Classifier | Nodes | Observations | Folds | Iterations | Time (secs) | Result |
|--------------|-------|---------------|-------|------------|----------------|--------------------------------------|
| randomForest | 1 | 32561 (adult) | 3 | 1 | 608 | 0.01 (aggr) 0.01 (mean) 0.00 (sd) |
| randomForest | 3 | 32561 (adult) | 3 | 1 | 250 | 0.01 (aggr) 0.01 (mean) 0.00 (sd) |

ADULTRF.R

```
library("mlr")
setwd("~/examples")
lrn = makeLearner("classif.randomForest")
adult = read.table("data/adult.data",
                  sep=",",header=F,col.names=c("age", "type_employer", "education_num", "marital", "occupation", "relationship", "capital_gain", "capital_loss", "hr_per_week", "country"),
                  fill=F,strip.white=T
)
adult.task = makeClassifTask(data = adult, target = "education")
rdesc = makeResampleDesc("CV", iters = 3)
system.time(res <- resample(lrn, adult.task, rdesc))
res
```

Code Source: <http://www.teraproc.com/teraproc-blog/seeing-the-forest-and-the-trees/>

ADULTRFPPARALLEL.R

```
library("parallelMap")
library("BatchJobs")
library("mlr")

setwd("~/examples")
conf = BatchJobs::getBatchJobsConf()
conf$cluster.functions = makeClusterFunctionsOpenLava("../batch.tmpl")
storagedir = getwd()
parallelStartBatchJobs(storagedir = storagedir)

lrn = makeLearner("classif.randomForest")
adult = read.table("data/adult.data",
                  sep=",",header=F,col.names=c("age", "type_employer", "education_num", "marital", "occupation", "relationship", "capital_gain", "capital_loss", "hr_per_week", "country"),
                  fill=FALSE,strip.white=T
)
adult_task = makeClassifTask(data = adult, target = "education")
```

Code Source: <http://www.teraproc.com/teraproc-blog/seeing-the-forest-and-the-trees/>

HIIRAGI2013 DATASET

A set of microarray expression profiles of single cells from mouse embryos at stages E3.25, E3.5 and E4.5. Explore a binary classification of transcriptome (cDNA) samples from single cells based on their microarray expression data and group these into two groups.

CLASSIFIER PERFORMANCE

| Classifier | Nodes | Observations | Folds | Iterations | Time (secs) | Result |
|--------------|-------|-------------------------------------------|-------|------------|-------------|-----------------------------------|
| glmnet | 1 | 45101 features, 101 samples (Hiiragi2013) | 10 | 12 | 22 | 0.01 (aggr) 0.01 (mean) 0.04 (sd) |
| glmnet.tuned | 1 | 45101 features, 101 samples (Hiiragi2013) | 10 | 12 | 1161 | 0.01 (aggr) 0.01 (mean) 0.04 (sd) |
| glmnet.tuned | 3 | 45101 features, 101 samples (Hiiragi2013) | 10 | 12 | 168 | 0.01 (aggr) 0.01 (mean) 0.03 (sd) |

GLMNET CONFUSION MATRIX

| glmnet | response - E3.25 | other |
|---------------|------------------|-------|
| truth - E3.25 | 53 | 0 |
| other | 1 | 47 |
| glmnet tuned | response - E3.25 | other |
| truth - E3.25 | 53 | 0 |
| other | 1 | 47 |

CDNA_MICROARRAY_SERIAL_EX1.R

```
library("knitr")
library("Biobase")
library("Hiiragi2013")
library("glmnet")
library("mlr")
data( "x", package = "Hiiragi2013" )

rowV <- data.frame( v = rowVars(exprs(x)) )
selectionThreshold <- 10^(-0.5)
selectedFeatures <- ( rowV$v > selectionThreshold )
embryoSingleCells <- data.frame( t(exprs(x)[selectedFeatures, ]), check.m
embryoSingleCells$tg <- factor( ifelse( x$Embryonic.day == "E3.25", "E3.2
with( embryoSingleCells, table( tg ) )

task <- makeClassifTask( id = "Hiiragi", data = embryoSingleCells, target
lrn = makeLearner( "classif.glmnet", predict.type = "prob" )
rdesc <- makeResampleDesc( method = "CV", stratify = TRUE, iters = 12 )
```

Code Source: <https://bioconductor.org/help/course-materials/2015/CSAMA2015/lab/classification.html>

CDNA_MICROARRAY_OPENLAVA-EX2.R

```
!!! Add same libraries as cDNA_microarray_ex1.R !!!  
library("BatchJobs")  
library("parallelMap")  
library("parallel")  
  
setwd("~/examples")  
  
conf = BatchJobs:::getBatchJobsConf()  
conf$cluster.functions = makeClusterFunctionsOpenLava("../batch.tmpl")  
storagedir = getwd()  
parallelStartBatchJobs(storagedir = storagedir)  
  
!!! Add same code as cDNA_microarray_ex1.R !!!  
  
parallelStop()
```

GERMAN CREDIT DATASET

This consists of 1000 rows, each row has information on the credit status of an individual. It provides both qualitative and quantitative, such as loan purpose, sex, loan duration, and installment rate as percentage of their disposable income.

CLASSIFIER PERFORMANCE

| Classifier | Nodes | Observations | Folds | Iterations | Time (secs) | Result |
|--------------|-------|----------------------------|-------|------------|----------------|-----------------------------------|
| glmnet | 1 | 800 (German Credit Scores) | 10 | 5 | 6 | 0.26 (aggr) 0.26 (mean) 0.04 (sd) |
| glmnet.tuned | 3 | 800 (German Credit Scores) | 10 | 5 | 53 | 0.26 (aggr) 0.26 (mean) 0.04 (sd) |

GERMANCREDIT-SERIAL.R

```
library(kernlab)
library(caret)
library(mlr)
setwd("~/examples")

data(GermanCredit)
GermanCredit <- GermanCredit[, -nearZeroVar(GermanCredit)]
GermanCredit$CheckingAccountStatus.lt.0 <- NULL
GermanCredit$SavingsAccountBonds.lt.100 <- NULL
GermanCredit$EmploymentDuration.lt.1 <- NULL
GermanCredit$EmploymentDuration.Unemployed <- NULL
GermanCredit$Personal.Male.Married.Widowed <- NULL
GermanCredit$Property.Unknown <- NULL
GermanCredit$Housing.ForFree <- NULL
set.seed(100)
inTrain <- createDataPartition(GermanCredit$Class, p = .8)[[1]]
GermanCreditTrain <- GermanCredit[inTrain, ]
GermanCreditTest <- GermanCredit[-inTrain, ]
```

Code Source: <http://jaehyeon-kim.github.io/r/2015/01/24/Benchmark-Example-in-MLR-Part-I/>

GERMANCREDIT-EX1.R

```
!!! Add same libraries as germancredit-serial.R !!!  
library("BatchJobs")  
library(parallel)  
  
setwd("~/examples")  
  
conf = BatchJobs::getBatchJobsConf()  
conf$cluster.functions = makeClusterFunctionsOpenLava("../batch.tmpl")  
storagedir = getwd()  
parallelStartBatchJobs(storagedir = storagedir)  
  
!!! Add same code as germancredit-serial.R !!!  
  
parallelStop()
```

SUMMARY

Advantages to using the cloud for scaling your machine models which can help reduce the time to create the models or explore larger problem spaces than possible by running many parallel similar models. Disadvantages include it may be unnecessary, need to think about parallelisation, consider the communication costs, and adds to the setup overhead.

- [Trees And Forests](#)
- [Machine Learning Machines](#)
- [Beyond A Single Instance](#)