Revolution Confidential

Extending R for Mining Big Data

Derek McCrae Norton
Senior Sales Engineer

Revolution Confidential
Who am I?

- 10+ Years in Predictive Analytics and Consulting
- Educated as a Statistician
- Founder / Director of Atlanta R User Group
- Presented at 4 useR! Conferences
  - Automated Modeling
  - Building R Presence in a SAS environment
  - 3 Panels on UseR Groups
- 1.5 Year at Revolution
Overview

- Data Mining
- Data Mining With R
- Revolution R Enterprise
- Mining Big Data
- Examples
Data Mining

What is it?
**Definition**¹

- The process of discovering interesting and useful patterns and relationships in **large volumes of data**.

- Combines tools from
  - Statistics
  - Artificial intelligence
  - Database management

---

¹ Encyclopedia Britanica (http://goo.gl/mlfO6)
Data Mining with R

All is well with the world!
How to do it?

- **Web**
  - Rdatamining.com
  - Zhao - R and data mining.pdf
  - Revolution Analytics - Introduction to R for Data Mining
  - CRAN Machine Learning Task View

- **Books**
  - Data Mining with R: Learning with Case Studies
  - Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery
What’s next?

- R provides me with the tools I need to...
- discover interesting and useful patterns and relationships in large volumes of data.
- What about large volumes of...?
  - Use another language.
  - Open source projects address to varying degrees of success
    - ff, bigmemory, biglm, etc.
  - Revolution R Enterprise

I don’t want to use another language.
I don’t want varying degrees of success.
… but that costs money.
Revolution R Enterprise

100% R & more™
Revolution R Enterprise:

- Performance Enhancements
- Greater Productivity & Ease of Use
- Tackle “Big Data”
- IT-Friendly Enterprise Deployment
- On-Call Experts

Performance
Productivity
Big Data Analysis
Training & Consulting
Technical Support
Enterprise Deployment
Open Source
Revolution R Enterprise has Open-Source R Engine at the core

3,700 community packages and growing exponentially

- Multi-Threaded Math Libraries
- Cluster support
- Web Services API
- Big Data Analysis
- Parallel Tools
- Technical Support
- Community Packages
- Build Assurance
- IDE / Developer GUI
- R Engine Language Libraries
- Multi-Threaded Math Libraries
- Cluster support
- Web Services API
- Big Data Analysis
- Parallel Tools
- Technical Support
- Community Packages
- Build Assurance
- IDE / Developer GUI
RevoScaleR brings the power of Big Data to R

- Parallel External Memory Algorithms that are distributed among available compute resources (cores & computers) independent of platform
- Distributed Statistical Algorithms
- Communications Framework
- Data Source API
- R Language Interface
- Abstracted layer for providing communication between compute nodes in a cluster (MPI, MapReduce, In-Database)
- API for integrating external data sources (files, databases, HDFS) that provides optimized reading of rows and columns in blocks
- Familiar, high-productivity programming paradigm for R users
Revolution R Enterprise

Why should I care?
How do I get it?

- It is free to academia

- It is free for Kaggle competitors.

- It is a great product, and affordable if neither of the above fit you.
Mining Big Data

A Framework
My Goals

- A good friend told me something that has always stuck…
  - I try to be as lazy as possible
- What he meant was don’t do more work than you ever have to.
  - If someone wrote something that is perfectly good and works for you – Use it.
  - If it almost works for you – Tweak it.
  - If nothing works – Develop it.
My Toolbox

- R
  - A great language.
  - Lots of statistical, machine learning, and data mining functionality.

- RevoScaleR
  - A collection of pre-parallelized algorithms to operate on big data.
  - A framework that let’s me create my own algorithms while leveraging R.
Example

More to come...

blog.revolutionanalytics.com
Naïve Bayes

- A simple probabilistic classifier based on applying Bayes' theorem with strong (naive) independence assumptions.
- A more descriptive term for the underlying probability model would be "independent feature model".

\[
\text{classify}(f_1, \ldots, f_n) = \arg\max_c p(C = c) \prod_{i=1}^{n} p(F_i = f_i | C = c).
\]
How to do it?

- **R – e1071**
  - `naiveBayes(formula, data, laplace = 0, ..., subset, na.action = na.pass)`

- **Really just calculating:**
  - Proportions for categorical variables (with possible Laplace correction)
  - Probabilities based on a normal distribution for numeric variables
Using RevoScaleR

- **Proportions**
  - `rxCrossTabs` - create contingency tables from cross-classifying factors using a formula interface.

- **Normal Probabilities**
  - `rxSummary` - produce univariate summaries
    - Means
    - Standard Deviations

\[
P(x = v | c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} e^{-\frac{(v - \mu_c)^2}{2\sigma_c^2}}
\]
Bring it all together

- Use existing e1071 code and replace calculation of proportions with big data versions.
  - Results are not big data!
  - Existing methods work on object!
I will show some code in a minute… Calm down.

QUESTIONS?
est <- function(vars) {
  catSum <- numSum <- NULL
  if (!is.null(vars[["categorical"]])) {
    catFun <- function(x) {
      form <- as.formula(paste("~", paste(Yname, x, sep = ":")))
      tab <- rxCrossTabs(form, data, returnXtabs = TRUE)
      class(tab) <- "table"
      attr(tab, "call") <- NULL
      (tab + laplace)/(rowSums(tab) + laplace * catLength[x])
    }
    catSum <- lapply(vars[["categorical"]], catFun)
  }
  if (!is.null(vars[["numeric"]])) {
    form <- as.formula(paste("~", paste(vars[["numeric"]], Yname, sep = ":", collapse = "+")))
    numVars <- rxSummary(form, data)$categorical
    numFun <- function(x) {
      ret <- as.matrix(x[, c("Means", "StdDev")])
      myNames <- vector("list", 2)
      myNames[[1]] <- x[, 2]
      dimnames(ret) <- myNames
      return(ret)
    }
    numSum <- lapply(numVars, numFun)
  }
  ret <- c(catSum, numSum)
}
rxNaiveBayes <- function (formula, data, laplace = 0, ...) {
  call <- match.call()
  vars <- all.vars(formula)
  Yname <- vars[1]
  x <- vars[-1]
  varInfo <- rxGetVarInfo(data)
  if (x == ".") {
    x <- names(varInfo)
    x <- x[!x %in% Yname]
  }
  origOrder <- x
  catVars <- sapply(varInfo, "[", "varType") == c("factor")][x]
  catVars <- catVars[order(catVars, decreasing = TRUE)]
  x <- names(catVars)
  catLength <- sapply(varInfo[names(which(catVars))], function(x) length(x$levels))
  sumVars <- list(categorical = x[catVars], numeric = x[!catVars])
  form <- as.formula(paste("~", Yname))
  apriori <- rxCrossTabs(form, data, returnType = TRUE)
  class(apriori) <- "table"
  attr(apriori, "call") <- NULL
  tables <- est(sumVars)
  names(tables) <- x
  for (i in 1:length(tables)) names(dimnames(tables[[i]])) <- c("Y", x[i])
  names(dimnames(apriori)) <- "Y"
  structure(list(apriori = apriori, tables = tables, levels = varInfo[[Yname]][["levels"]],
                 call = call), class = c("rxNaiveBayes", "naiveBayes"))
}