Data Analysis in R

Yuwu Chen

HPC @ LSU

Some materials are borrowed from the EXST 7142/7152 data mining courses by Dr. Bin Li at Statistics Dept.

Outline

- Data acquisition and inspection
- Preprocess the dataset
- Data analysis

Steps for Data Analysis in R

- Get the data
- Read and inspect the data
- Preprocess the data
- Analyze the data
- Generate the report

How does R work

- R works best if you have a dedicated folder for each separate project - the working folder. Put all data files in the working folder (or in subfolders).

```
> getwd()  #Show current working folder
[1] "/home/ychen64"
> dir.create("data") #Create a new folder
> getwd()
[1] "/home/ychen64"
> setwd("data")
> getwd()
[1] "/home/ychen64/data“
> list.files()   # List files in current folder
```

- Work on the project - your objects can be automatically saved in the .RData file
- To quit use `q()` or `CTRL + D` or just kill the window. R will ask “Save workspace image?”. You can choose:
  - No: leave R without saving your results in R;
  - Yes: save your results in .RData in your working directory;
  - Cancel: not quitting R.

Case Study: Forbes Fortune List

- The forbes dataset consists of 2000 rows (observations) describing companies' rank, name, country, category, sales, profits, assets and market value.

http://www.hpc.lsu.edu/training/weekly-materials/Downloads/Forbes2000.csv.zip

Getting Data

- Downloading files from internet
  - Manually download the file to the working directory
  - or with R function `download.file()`

```
> download.file("http://www.hpc.lsu.edu/training/weekly-materials/Downloads/Forbes2000.csv.zip", "Forbes2000.csv.zip")
> unzip("Forbes2000.csv.zip","Forbes2000.csv")
```

Steps for Data Analysis in R

- Get the data
- Read and inspect the data
- Preprocess the data
- Analyze the data
- Generate the report

Reading and Writing Data

- R understands many different data formats and has lots of ways of reading/writing them (csv, xml, excel, sql, json etc.)

`read.table` `read.csv`	`write.table` `write.csv`	for reading/writing tabular data
`readLines`	`writeLines`	for reading/writing lines of a text file
`source`	`dump`	for reading/writing in R code files
`dget`	`dput`	for reading/writing in R code files
`load`	`save`	for reading in/saving workspaces

Reading Data with `read.table` (1)

```
# List of arguments of the read.table() function
> str(read.table)
function (file, header = FALSE, sep = "", quote = "\"'", dec = ".", row.names,
col.names, as.is = !stringsAsFactors, na.strings = "NA", colClasses = NA, nrows = -1,
skip = 0, check.names = TRUE, fill = !blank.lines.skip, strip.white = FALSE,
blank.lines.skip = TRUE, comment.char = "#", allowEscapes = FALSE, flush = FALSE,
stringsAsFactors = default.stringsAsFactors(), fileEncoding = "", encoding = "unknown",
text, skipNul = FALSE)
```

Reading Data with `read.table` (2)

- `file` - the name of a file, or a connection
- `header` - logical indicating if the file has a header line
- `sep` - a string indicating how the columns are separated
- `na.strings` - a character vector of strings which are to be interpreted as `NA` values
- `nrows` - the number of rows in the dataset
- `comment.char` - a character string indicating the comment character
- `skip` - the number of lines to skip from the beginning
- `stringsAsFactors` - should character variables be coded as factors?

Reading Data with `read.table` (3)

- The function will
  - Skip lines that begin with #
  - Figure out how many rows there are (and how much memory needs to be allocated)
  - Figure out what type of variable is in each column of the table
- Telling R all these things directly makes R run faster and more efficiently.
- `read.csv()` is identical to `read.table()` except that the default separator is a comma.

```
> forbes <- read.csv("Forbes2000.csv",header=T,stringsAsFactors =
FALSE,na.strings ="NA",sep=",")
```

Reading EXCEL spreadsheets

- The simplest method is to save each worksheet separately as a csv file and use `read.csv()` on each.
- The XLConnect library can open both .xls and .xlsx files. It is Java-based, so it is cross platform. But it may be very slow for loading large datasets.

```
>library(XLConnect)
wb <- loadWorkbook("Forbes2000.xls")
setMissingValue(wb, value = c("NA"))
forbes <- readWorksheet(wb, sheet=1, header=TRUE)
```

- There are at least two other ways: read.xlsx from library(xlsx) (slow for large datasets) and read.xls from library(gdata) (require PERL installed).

```
>library(xlsx)
>forbes <- read.xlsx("Forbes2000.xls", 1)
```

- Note: the libraries above requires both Java Dev Kit and rJava library. The later is not available for R version installed on QB2 and SuperMic.

Quiz

- **After importing the raw data, the R data object used for carrying the data is a:**

a) vector
b) matrix
c) array
d) list
e) data frame

Inspecting Data (1)

- `class()`: it is a data frame
- `dim()`: dimension of the data
- `head()`: print on screen the first few lines of data, may use `n` as argurement
- `tail()`: print the last few lines of data

```
> head(forbes)
  rank                name        country             category  sales profits
1    1           Citigroup  United States              Banking  94.71   17.85
2    2    General Electric  United States        Conglomerates 134.19   15.59
3    3 American Intl Group  United States            Insurance  76.66    6.46
4    4          ExxonMobil  United States Oil & gas operations 222.88   20.96
5    5                  BP United Kingdom Oil & gas operations 232.57   10.27
6    6     Bank of America  United States              Banking  49.01   10.81
   assets marketvalue
1 1264.03      255.30
2  626.93      328.54
3  647.66      194.87
4  166.99      277.02
5  177.57      173.54
6  736.45      117.55
```

Inspecting Data (2)

- Displays the structure of the “forbes” dataframe.

```
> str(forbes)
'data.frame':   2000 obs. of  8 variables:
 $ rank       : num  1 2 3 4 5 6 7 8 9 10 ...
 $ name       : chr  "Citigroup" "General Electric" "American Intl Group" "ExxonMobil" ...
 $ country    : chr  "United States" "United States" "United States" "United States" ...
 $ category   : chr  "Banking" "Conglomerates" "Insurance" "Oil & gas operations" ...
 $ sales      : num  94.7 134.2 76.7 222.9 232.6 ...
 $ profits    : num  17.85 15.59 6.46 20.96 10.27 ...
 $ assets     : num  1264 627 648 167 178 ...
 $ marketvalue: num  255 329 195 277 174 ...
```

Inspecting Data (3)

- Statistical summary of the "Forbes" dataframe.

```
> summary(forbes)
      rank              name             country            category        
 Min.   :   1.0   Length:2000        Length:2000        Length:2000       
 1st Qu.: 500.8   Class :character   Class :character   Class :character  
 Median :1000.5   Mode  :character   Mode  :character   Mode  :character  
 Mean   :1000.5                                                           
 3rd Qu.:1500.2                                                           
 Max.   :2000.0                                                           

     sales             profits              assets            marketvalue    
 Min.   :  0.010   Min.   :-25.8300   Min.   :   0.270   Min.   :  0.02  
 1st Qu.:  2.018   1st Qu.:  0.0800   1st Qu.:   4.025   1st Qu.:  2.72  
 Median :  4.365   Median :  0.2000   Median :   9.345   Median :  5.15  
 Mean   :  9.697   Mean   :  0.3811   Mean   :  34.042   Mean   : 11.88  
 3rd Qu.:  9.547   3rd Qu.:  0.4400   3rd Qu.:  22.793   3rd Qu.: 10.60  
 Max.   :256.330   Max.   : 20.9600   Max.   :1264.030   Max.   :328.54  
                   NA's   :5                                             
```

- Note: there are missing values in the profits.

Summary - get, read and inspect the data

- Get
  - put all data into a dedicated folder
- Read
  - `read.csv`
  - `read.table`
- Inspect
  - Querying Object Attributes

Steps for Data Analysis in R

- Get the data
- Read and inspect the data
- Preprocess the data (missing and dubious values, discard columns not needed etc.)
- Analyze the data
- Generate the report

Preprocessing - Missing Values

- Missing values are denoted in `R` by `NA` or `NaN` for undefined mathematical operations.
  - `is.na()` is used to test objects if they are NA
  - Which one is NA? `which(is.na(x))`

```
> which(is.na(forbes$profits))
```

  - How many NAs? `table(is.na(x))`

```
> table(is.na(forbes$profits))
```

  - list of observations with missing values on profits `x(is.na(x),)`

```
> forbes[is.na(forbes$profits),]
```

- Make sure when reading data `R` can recognize the missing values. E.g. `setMissingValue(wb, value = c("NA"))` when using `XLConnect`
- Many R functions also have a logical "`na.rm`" option
  - `na.rm=TRUE` means the NA values should be discarded

```
> mean(forbes$profits,na.rm=T)
```

- **Note: Not all missing values are marked with "NA" in raw data!**

Preprocessing - Missing Values

- The simplest way to deal with the missing values is to remove them.
  - If a row (observation) has a missing value, remove the row with `na.omit()`. e.g.

```
> forbes <- na.omit(forbes)
> dim(forbes)
```

  - If a column (variable) has a high percentage of the missing value, remove the whole column or just don't use it for the analysis

Preprocessing - Missing Values

- Alternatively, the missing values can be replaced by basic statistics e.g.
  - replace by mean

```
for(i in 1:nrow(forbes)){
  if(is.na(forbes$profits[i])==TRUE){
  forbes$profits[i] <- mean(forbes$profits, na.rm = TRUE)
  }
}
```

  - or use advanced statistical techniques. List of popular R Packages:

    MICE

    Amelia (named after Amelia Earhart)

    missForest (non parametric imputation method)

    Hmisc

    mi

Preprocessing - Subsetting Data

- At most occasions we do not need all of the raw data
- There are a number of methods of extracting a subset of R objects
- Subsetting data can be done either by row or by column

Preprocessing - Subsetting Data

- Subsetting by row: use conditions

```
# Find all companies with negative profit

>forbes[forbes$profits < 0,c("name","sales","profits","assets")]
                           name sales profits  assets
350           Allianz Worldwide 96.88   -1.23  851.24
354                    Vodafone 47.99  -15.51  256.28
364             Deutsche Telekom 56.40  -25.83  132.01
```

Preprocessing - Subsetting Data

- Subsetting by row: use conditions

```
# Find three companies with largest sale vol.

> companies <- forbes$name
> companies <- forbes[,"name"] #same as above
> order_sales <- order(forbes$sales, decreasing=T)
> companies[order_sales[1:3]]
[1] "Wal-Mart Stores" "BP"              "ExxonMobil“

> head(sort(forbes$sales,decreasing=T),n=3)
[1] 256.33 232.57 222.88
```

Preprocessing - Subsetting Data

- Subsetting by row: use the `subset()` function

```
# Find the business category to which most of the
Bermuda island companies belong.
```

```
>Bermudacomp <- subset(forbes, country == "Bermuda")
>table(Bermudacomp[,"category"]) #frequency table of categories

             Banking        Capital goods       Conglomerates
                   1                    1                   2
Food drink & tobacco         Food markets           Insurance
                   1                    1                  10
               Media Oil & gas operations  Software & services
                   1                    2                   1
```

Preprocessing - Subsetting Data

- Subsetting by column

```
# Create another data frame with only numeric
variables

> forbes2 <- data.frame(sales=forbes$sale,profits=forbes$profits,
           assets=forbes$assets, mvalue=forbes$marketvalue)
> str(forbes2)

# Or simply use indexing
> forbes3 <- forbes[,c(5:8)]
> str(forbes3)
```

Preprocessing – Factors

- factors are variables in R which take on a limited number of different values; such variables are often referred to as categorical variables

```
# Convert characters to (unordered) factors
```

```
> forbes$country<-factor(forbes$country)
> str(forbes)
'data.frame':   2000 obs. of  8 variables:
 $ rank       : int  1 2 3 4 5 6 7 8 9 10 ...
 $ name       : chr  "Citigroup" "General Electric" "American Intl Group" "ExxonMobil" ...
 $ country    : Factor w/ 61 levels "Africa","Australia",..: 60 60 60 60 56 60 56 28 60 60 ...
...
...
```

CENTER FOR COMPUTATION & TECHNOLOGY

Preprocessing – Factors

- Small classes could be merged into a larger class. Why?
  - For better model performance. E.g. Classification and Regression Trees tend to split using the variables with many categories.
  - Actual needs
- Some categories have just a few subjects

```
> table(forbes$country)

                   Africa                 Australia 
                        2                        37 
  Australia/ United Kingdom                 Austria 
                        2                         8 
                  Bahamas                   Belgium 
                        1                         9 
...
...
```

Preprocessing – Factors

- Merge small classes into a larger classes

```
>forbes$country[(forbes$country=="Bahamas")|(forbes$country=="Ber
muda")|(forbes$country=="Brazil")|(forbes$country=="Cayman
Islands")|(forbes$country=="Chile")|(forbes$country=="Panama/
United Kingdom")|(forbes$country=="Peru")]<-"Venezuela"
```

Preprocessing – Factors

- Merge small classes into a larger classes

```
> forbes$country[(forbes$country=="Austria")|(forbes$country=="Belgium")|(forbes$country=="Czech
Republic")|(forbes$country=="Denmark")|(forbes$country=="Finland")|(forbes$country=="France")|(forbes$country=="German
y")|(forbes$country=="Greece")|(forbes$country=="Hungary")|(forbes$country=="Ireland")|(forbes$country=="Italy")|(forb
es$country=="Luxembourg")|(forbes$country=="Netherlands")|(forbes$country=="Norway")|(forbes$country=="Poland")|(forbe
s$country=="Portugal")|(forbes$country=="Russia")|(forbes$country=="Spain")|(forbes$country=="Sweden")|(forbes$country
=="Switzerland")|(forbes$country=="Turkey")|(forbes$country=="France/ United Kingdom")|(forbes$country=="United
Kingdom/ Netherlands")|(forbes$country=="Netherlands/ United Kingdom")]<-"United Kingdom"

> forbes$country[(forbes$country=="China")|(forbes$country=="Hong
Kong/China")|(forbes$country=="Indonesia")|(forbes$country=="Japan")|(forbes$country=="Kong/China")|(forbes$country=="
Korea")|(forbes$country=="Malaysia")|(forbes$country=="Philippines")|(forbes$country=="Singapore")|(forbes$country=="S
outh Korea")|(forbes$country=="Taiwan")]<-"Thailand"

>forbes$country[(forbes$country=="Africa")|(forbes$country=="Australia")|(forbes$country=="India")|(forbes$country=="A
ustralia/ United
Kingdom")|(forbes$country=="Islands")|(forbes$country=="Israel")|(forbes$country=="Jordan")|(forbes$country=="Liberia"
)|(forbes$country=="Mexico")|(forbes$country=="New Zealand")|(forbes$country=="Pakistan")|(forbes$country=="South
Africa")|(forbes$country=="United Kingdom/ Australia")]<-"United Kingdom/ South Africa"
```

Preprocessing – Factors

- Drop those levels with zero counts

```
> forbes$country<-droplevels(forbes$country)
> table(forbes$country)

                  Canada                Thailand
                      56                     499
          United Kingdom United Kingdom/ South Africa
                     531                     115
           United States               Venezuela
                     751                      48
```

- Rename each class

```
> levels(forbes$country)<-c("Canada","East/Southeast Asia","Europe","Other","United
States","Latin America")
> levels(forbes$country)
[1] "Canada"              "East/Southeast Asia" "Europe"
[4] "Other"               "United States"       "Latin America"
```

Export the Dataset (Optional)

- Save forbes to Forbes2000_clean.csv

```
> write.csv(forbes,"Forbes2000_clean.csv",row.names=FALSE)
```

Homework 1

1. Import dataset forbes, save it as forbes
2. Run the following commands:

```
head(forbes)
str(forbes)
summary(forbes)
```

3. Remove the observations with missing values
4. Find all German companies with negative profit
5. Find the 50 companies in the Forbes dataset with the highest profit
6. Find the average value of sales for the companies in each country (Hint: use tapply function)
7. Find the number of companies in each country with profits above 5 billion US dollars
8. Arbitrarily merge the classes of category to three classes: industry, services and finance

Steps for Data Analysis in R

- Get the data
- Read and inspect the data
- Preprocess the data
- Analyze the data
- Generate the report

Two common questions

- Which statistical model should I use for my data analysis?
- How to choose the right R packages for my data analysis?

Which statistical model should I use for my data analysis?

- This is not a Statistics Class…
- If you need to learn more about the data mining and data analysis from statisticians:
  - EXST7142 - Statistical Data Mining
    http://statweb.lsu.edu/faculty/li/teach/exst7142/
  - EXST7152 - Advanced Topics in Statistical Modeling
    http://statweb.lsu.edu/faculty/li/teach/exst7152/

How to choose the right R packages for my data analysis?

- The most popular packages are most frequently mentioned
- CRAN task views

  https://cran.r-project.org/web/views/
- RDocumentation
  - a website, an R package and an API

  https://www.rdocumentation.org

Import the Clean Dataset (Optional)

- Subsetting by column

```
# Create a data frame with the clean data

> forbes <- read.csv("Forbes2000_clean.csv",header=T,stringsAsFactors = T,na.strings
="NA",sep=",")
```

Extract Variables

- Subsetting by column

```
# Create another data frame with only numeric
variables + country
```

```
> forbes2 <- forbes[,c(3, 5:8)]
> str(forbes2)
```

Training Set and Test Set

- Dataset could be randomly split into two parts: training set and test set.
- The model is fitted on the training set and predicted on the test set. Why?

Bias Variance Tradeoff

- Two competing forces govern the choice of learning method, i.e. **bias** and **variance.**
- Bias refers to the error that is introduced by modeling a real life problem (which is usually extremely complicated) by a much simpler problem.
  - For example, linear regression assumes that there is a linear relationship between Y and X, which is unlikely in real life.
  - In general, the more flexible/complex a method is, the less bias it will have
- Variance refers to how much your estimate for f would change by if you had a different (test) dataset.
  - Generally, the more flexible/complex a method, the more variance it will have.

Bias Variance Tradeoff

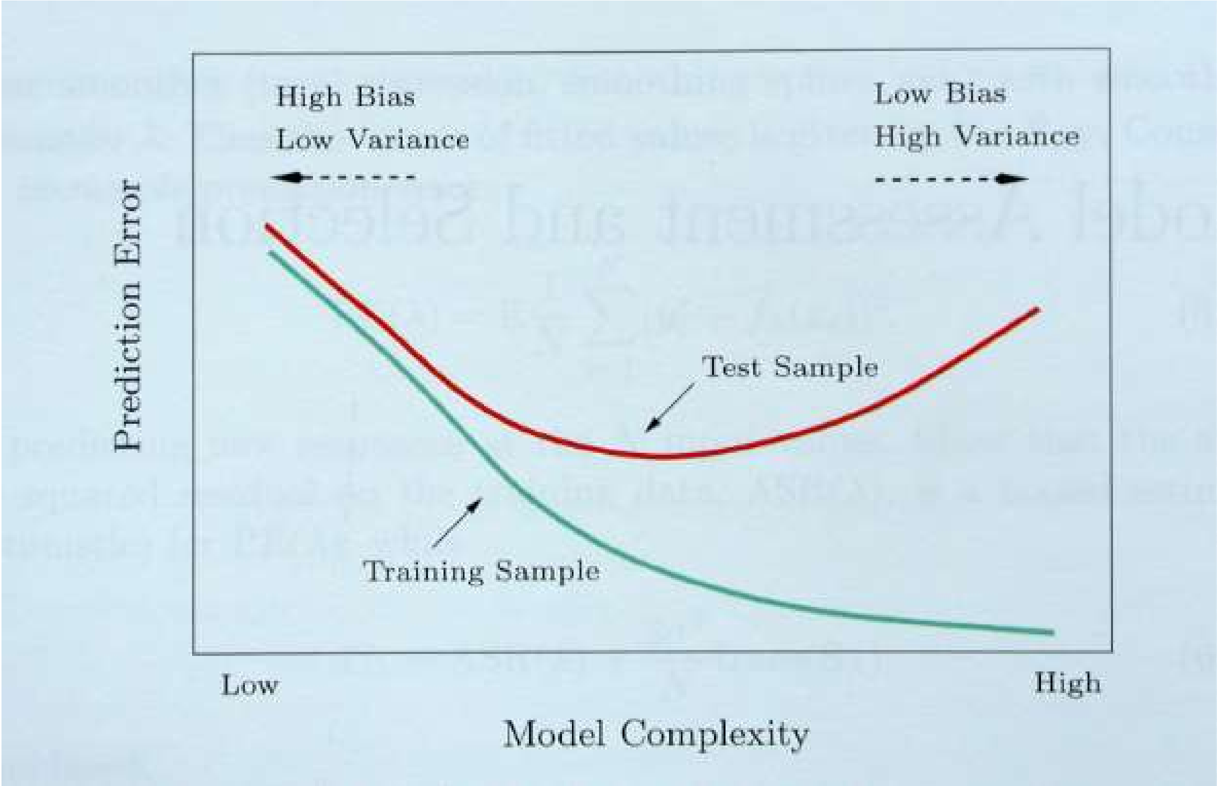


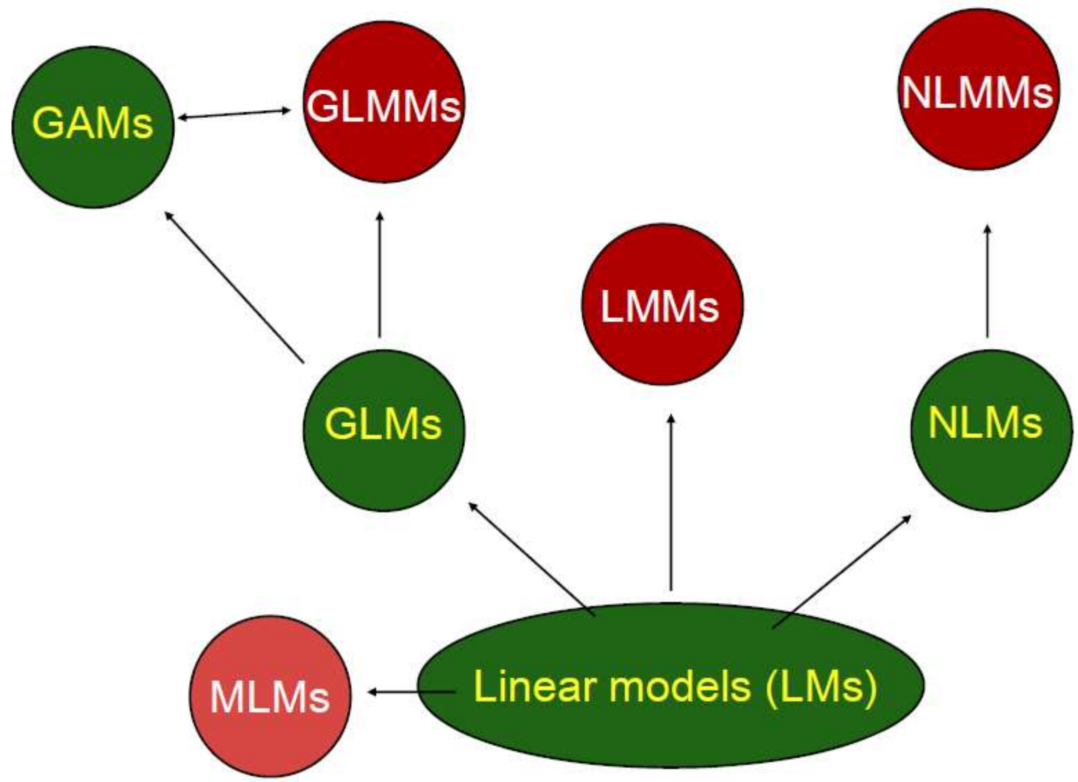
Figure from EOSL 2001

Training Set and Test Set

- Dataset could be randomly split into two parts: training set and test set.

```
> set.seed(1) #set random seed reproducible
> indx <- sample(1:1995,size=1995,replace=F)
> forbes.train <- forbes2[indx[1:1600],]
> forbes.test <- forbes2[indx[1601:1995],]
```

Roadmap of Generalizations of Linear Models



Explanation of Acronyms

Models	Acronym	R function
Linear Models	LM	lm, aov
MultivariateLMs	MLM	manova
Generalized LMs	GLM	glm
Linear Mixed Models	LMM	lme, aov
Non-linear Models	NLM	nls
Non-linear Mixed Models	NLMM	nlme
Generalized LMMs	GLMM	glmmPQL
Generalized Additive Models	GAM	gam

Symbol Meanings in Model Formulae

Symbol	Example	Meaning
+	+X	Include this variable in the model
-	-X	Exclude this variable in the model
:	X:Z	Include the interaction between X and Z
*	X*Z	Include X and Z and the interactions
|	X|Z	Conditioning: include X given Z
^	(A+B+C)^3	Include A, B and C and all the interactions up to three way
/	/(X*Z)	As is: include a new variable consisting of these variables multiplied

CENTER FOR COMPUTATION & TECHNOLOGY

Model Formulae

General form: response ~ $term_1$ + $term_2$

Example	Meaning
y ~ x	Simple regression
y ~ -1 + x	LM through the origin
y ~ x + x^2	Quadratic regression
y ~ x1 + x2 + x3	Multiple regression
y ~ .	All variables included
y ~ . - x1	All variables except X1
y ~ A + B + A : B	Add interaction
y ~ A * B	Same above
y ~ (A+B)^2	Same above

A Multiple Linear Regression Example

marketvalue ~ profits + sales + assets + country

```
> lm <- lm(marketvalue ~ ., data = forbes.train)
> summary(lm)
Call:
lm(formula = marketvalue ~ ., data = forbes.train)
Residuals:
    Min      1Q  Median      3Q     Max
-82.532  -4.842  -1.719   1.516 225.259
Coefficients:
                            Estimate Std. Error t value Pr(>|t|)
(Intercept)                 1.941600   2.568998   0.756    0.450
countryEast/Southeast Asia -2.191134   2.700858  -0.811    0.417
countryEurope               0.617738   2.699779   0.229    0.819
countryLatin America        0.175543   3.913749   0.045    0.964
countryOther                0.612666   3.089536   0.198    0.843
countryUnited States        3.639061   2.654924   1.371    0.171
sales                       0.626963   0.030984  20.235   <2e-16 ***
profits                     3.726989   0.257696  14.463   <2e-16 ***
assets                      0.050135   0.004834  10.371   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 16.99 on 1591 degrees of freedom
Multiple R-squared:  0.4899,    Adjusted R-squared:  0.4873
F-statistic:   191 on 8 and 1591 DF,  p-value: < 2.2e-16
```

A Multiple Linear Regression Example

- R has created a n-1 variables each with two levels. These n-1 new variables contain the same information as the single variable. This recoding creates a table called contrast matrix.

```
> contrasts(forbes.train$country)
                    East/Southeast Asia Europe Latin America Other United States
Canada                                0      0             0     0             0
East/Southeast Asia                   1      0             0     0             0
Europe                                0      1             0     0             0
Latin America                         0      0             1     0             0
Other                                 0      0             0     1             0
United States                         0      0             0     0             1
```

- The decision to code dummy variables is arbitrary, and has no effect on the regression computation, but does alter the interpretation of the coefficients.

A Stepwise Regression Example

- The function `regsubsets()` in the `leaps` library allow us to do the stepwise regression

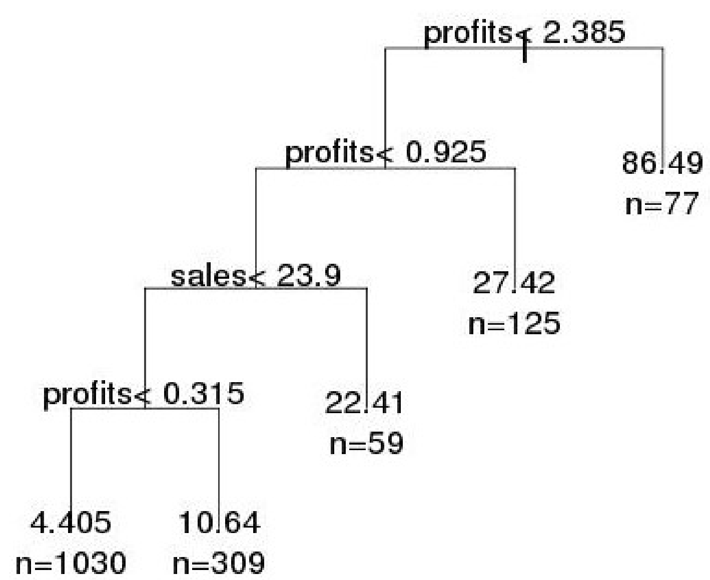
```
> library(leaps)
> bwd <- regsubsets(marketvalue ~ ., data = forbes.train,nvmax =3,method ="backward")
> summary(bwd)
Subset selection object
Call: regsubsets.formula(marketvalue ~ ., data = forbes.train, nvmax = 3, 
    method = "backward")
8 Variables  (and intercept)
                           Forced in Forced out
countryEast/Southeast Asia     FALSE      FALSE
...
1 subsets of each size up to 3
Selection Algorithm: backward
         countryEast/Southeast Asia countryEurope countryLatin America
1  ( 1 ) " "                        " "           " "                 
2  ( 1 ) " "                        " "           " "                 
3  ( 1 ) " "                        " "           " "                 
         countryOther countryUnited States sales profits assets
1  ( 1 ) " "          " "                  "*"   " "     " "   
2  ( 1 ) " "          " "                  "*"   "*"     " "   
3  ( 1 ) " "          " "                  "*"   "*"     "*"   
```

An asterisk indicates that a given variable is included in the corresponding model.

A Regression Tree Example

- The function `rpart()` in the `rpart` library allow us to grow a regression tree

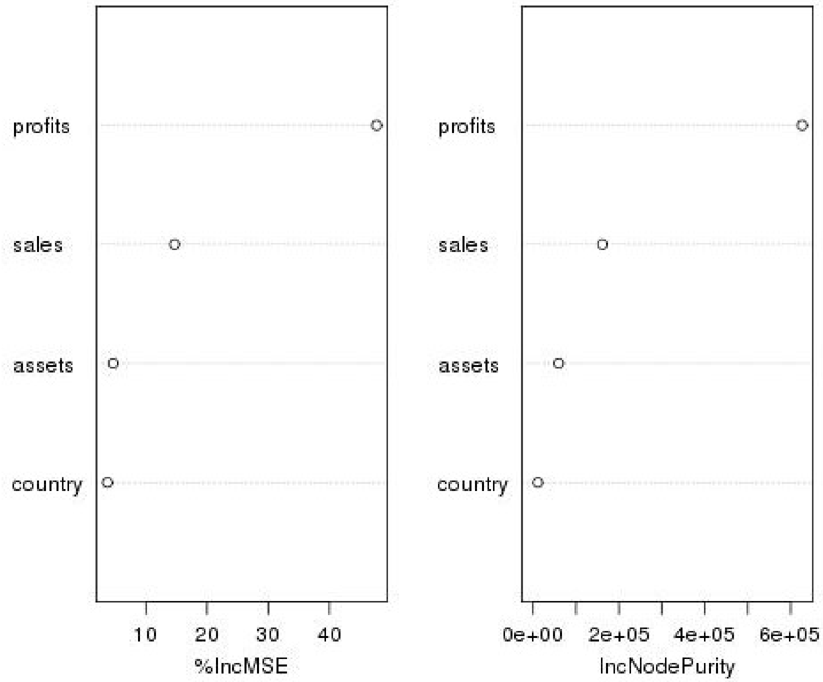
```
> library (rpart)
> rpart <- rpart(marketvalue ~ ., data = forbes.train,control = rpart.control(xval =
10, minbucket = 50))
> jpeg('rplot1%03d.jpg')
> par(mfrow=c(1,1),xpd=NA,cex=1.5)
> plot(rpart,uniform=T)
> text(rpart,use.n=T)
> dev.off()
```



A Bagging Tree Example

- The function `randomForest()` in the `randomForest` library allow us to grow a regression tree

```
> library (randomForest)
> bag <- randomForest(marketvalue ~ ., data = forbes.train, importance =TRUE)
> jpeg('rf%03d.jpg')
> importance(bag)
          %IncMSE IncNodePurity
country  8.060405      33769.61
sales   17.627031     200418.63
profits 32.844743     371824.72
assets  11.890230     159419.77
> varImpPlot(bag)
> dev.off()
```



The Predictive Results in Terms of the MAD and RMSE Values

$$RMSE = \sqrt{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2/N} \qquad MAD = \frac{1}{N} \times \sum_{i=1}^{N} |y_i - \hat{y}_i|$$

Model	Package	RMSE	MAD
MLR		14.41041	6.436288
Backward	leaps	14.41041	6.436288
Pruned tree	rpart	17.85625	5.899107
Bagging tree	randomForest	11.69301	4.944942

Other Common Regression Models and Packages in R

Model	Package
MLR	
Stepwise	leaps, MASS
Ridge, Lasso, Elesticnet	glmnet
Neural network	nnet, neuralnet
SVM-linear kernel	kernlab
single tree	rpart
MARS	earth
Generalized additive	gam
Boost tree	gbm
Bagging tree	randomForest

Train models with Resampling Methods

- Train method in this training session: The `train()` function in the `caret` package
  - Can train hundreds of models with resampling methods
  - Easy to manipulate, well documented.
  - Will automatically parallelize when multiple cpu cores are registered

Train models with Resampling Methods

Model	Resampling method	Tuning parameter
MLR	bootstrapping	intercept
Backward Selection	cross-validation	#Randomly Selected Predictors
Ridge	cross-validation	λ
Lasso	cross-validation	λ
Elesticnet	cross-validation	α and λ
SVM-linear kernel	cross-validation	cost
Pruned tree	bootstrapping	cp
MARS	bootstrapping	#prune and degree
Boost tree	repeat cross-validation	#.trees, shrinkage interaction.depth,
Bagging (RF)	cross-validation	#Randomly Selected Predictors

Parallel Computing in R

- Motivation: Save computation time.
  - A for loop can be very slow if there are a large number of computations that need to be carried out.
  - Almost all computers now have multicore processors.
  - As long as these computations do not need to communicate (resampling methods are excellent examples), they can be spread across multiple cores and executed in parallel.

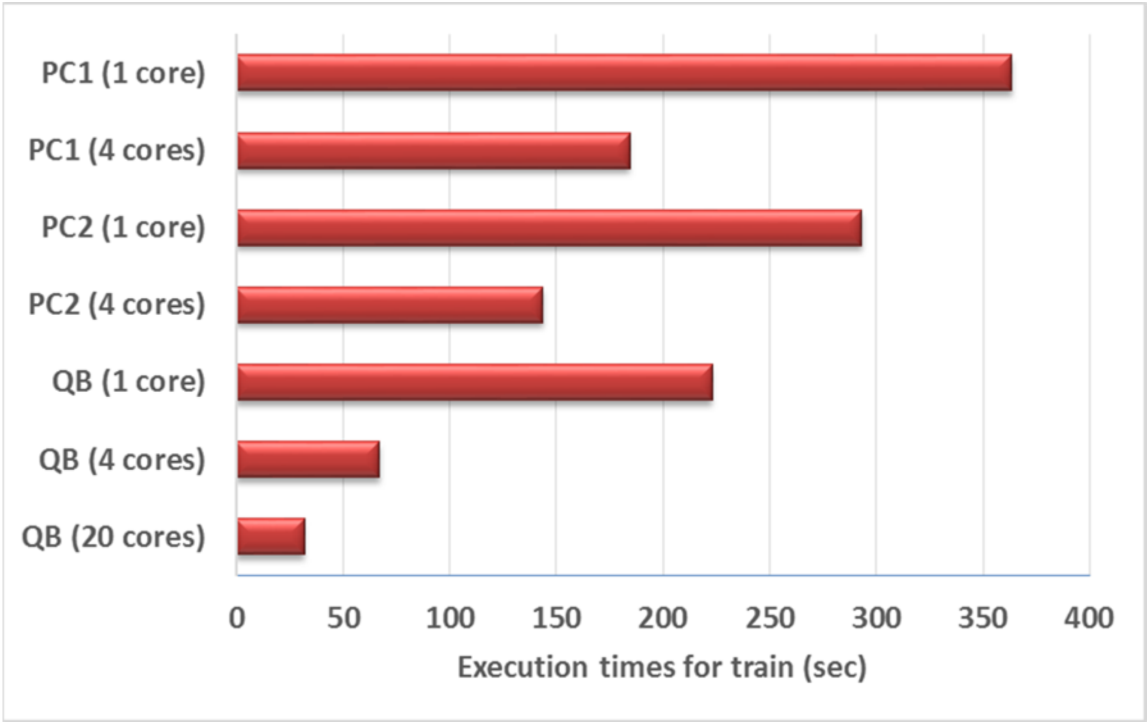
Parallel Computing in R

- The parallel package

```
#In the R, load library(doParallel)
> library(doParallel)
# Find out how many cores are available
> detectCores()
[1] 16
# Create cluster with desired number of cores
> cl <- makeCluster(16)
# Register cluster
> registerDoParallel(cl)
# Find out how many cores are being used
> getDoParWorkers()
[1] 16
```

Clusters are Better for Resource-demanding Jobs

- Training random forest model
- Resampling method: 10-fold cross-validation



Training Bagging Trees to (Random Forest)

```
> bagtrain <- train(marketvalue ~ ., data = forbes.train,method = "rf",tuneGrid =
NULL,tuneLength = 3)
> bagtrain
Random Forest
1600 samples
   4 predictors
No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 1600, 1600, 1600, 1600, 1600, 1600, ...
Resampling results across tuning parameters:
s = abs(bag.y - bag.yhat)
bag.mad = (sum(bag.abs))/395
bag.mad
jpeg('rf2%03d.jpg')
imp  mtry  RMSE      Rsquared   MAE
  2     13.55779  0.6860085  5.619290
  5     13.33941  0.6846835  5.157681
  8     13.92276  0.6640880  5.374219
RMSE was used to select the optimal model using the smallest value.
The final value used for the model was mtry = 5.
```

Training Improvement

	RMSE		MAD	
	untrained	trained	untrained	trained
MLR	14.41041	14.41041	6.436288	6.436288
Backward	14.41041	14.36738	6.436288	6.352504
Pruned tree	17.85625	12.91093	5.899107	5.321366
BaggingTree	11.69301	10.30676	4.944942	4.488556

Put Everything Together

- Run R commands in batch mode with `Rscript`

```
[ychen64@mike001 R]$ cat forbes.R
# Check if the data directory exists; if not, create it.
if (!file.exists("data")) {
        dir.create("data")
}

# Check if the data file has been downloaded; if not, download it.
if (!file.exists("Forbes2000.csv")) {
        download.file("http://www.hpc.lsu.edu/training/weekly-
materials/Downloads/Forbes2000.csv.zip", "Forbes2000.csv.zip")
}
…

[ychen64@make001 R]$ Rscript forbes.R
```

Steps for Data Analysis in R

- Get the data
- Read and inspect the data
- Preprocess the data
- Analyze the data
- Generate the report

Report Generation with R Markdown

- R markdown
  - Allows one to generate dynamic report by weaving R code and human readable texts together
- The `knitr` and `rmarkdown` packages can convert them into documents of various formats
- Help make your research reproducible

Take-home message

- Get the data
- Read and inspect the data
- Preprocess the data
  - missing values, discard rows, columns not needed etc.
- Analyze the data
  - choose the right model and R package
  - common R functions and syntax for regressions
  - model training basics with the `caret` package
  - parallel computing in R
- Generate the report

Not Covered

- Unsupervised models
  - Cluster analysis
  - Principal Component Analysis
- Deep learning in R

More R Tutorials – Data Visualization in R

- This training provided an introduction to the R graphics in detail
- An overview on how to create and save graphs in R, then focus on the ggplot2 package.
- http://www.hpc.lsu.edu/training/archive/tutorials.php

More R Tutorials – Parallel Computing with R

- This training focused on how to use the "parallel" package in R and a few related packages to parallelize and enhance the performance of R programs
- http://www.hpc.lsu.edu/training/archive/tutorials.php

Next HPC Tutorial – Introduction to Singularity

- This training will introduce creating and running Containers on HPC with Singularity
- Date: March 27th, 2019

Getting Help

- User Guides
  - LSU HPC: http://www.hpc.lsu.edu/docs/guides.php#hpc
  - LONI:http://www.hpc.lsu.edu/docs/guides.php#loni
- Documentation: http://www.hpc.lsu.edu/docs
- Contact us
  - Email ticket system: sys-help@loni.org
  - Telephone Help Desk: 225-578-0900

Questions?

Homework 2

1. Use the lm() function to perform a multiple linear regression with profits as the response and all other numeric variables as the predictors. Use the summary() function to print the results.
2. Comment on the output. For instance: Is there a relationship between the predictors and the response?
3. Which predictors appear to have a statistically significant relationship to the response?
4. What does the coefficient for the sales variable suggest?
5. Use the * and : symbols to fit linear regression models with interaction effects.
   Do any interactions appear to be statistically significant?