

Data Analytics, R, Scaling, and Comet Examples

Paul Rodriguez SDSC



Outline

I. Data Analytics

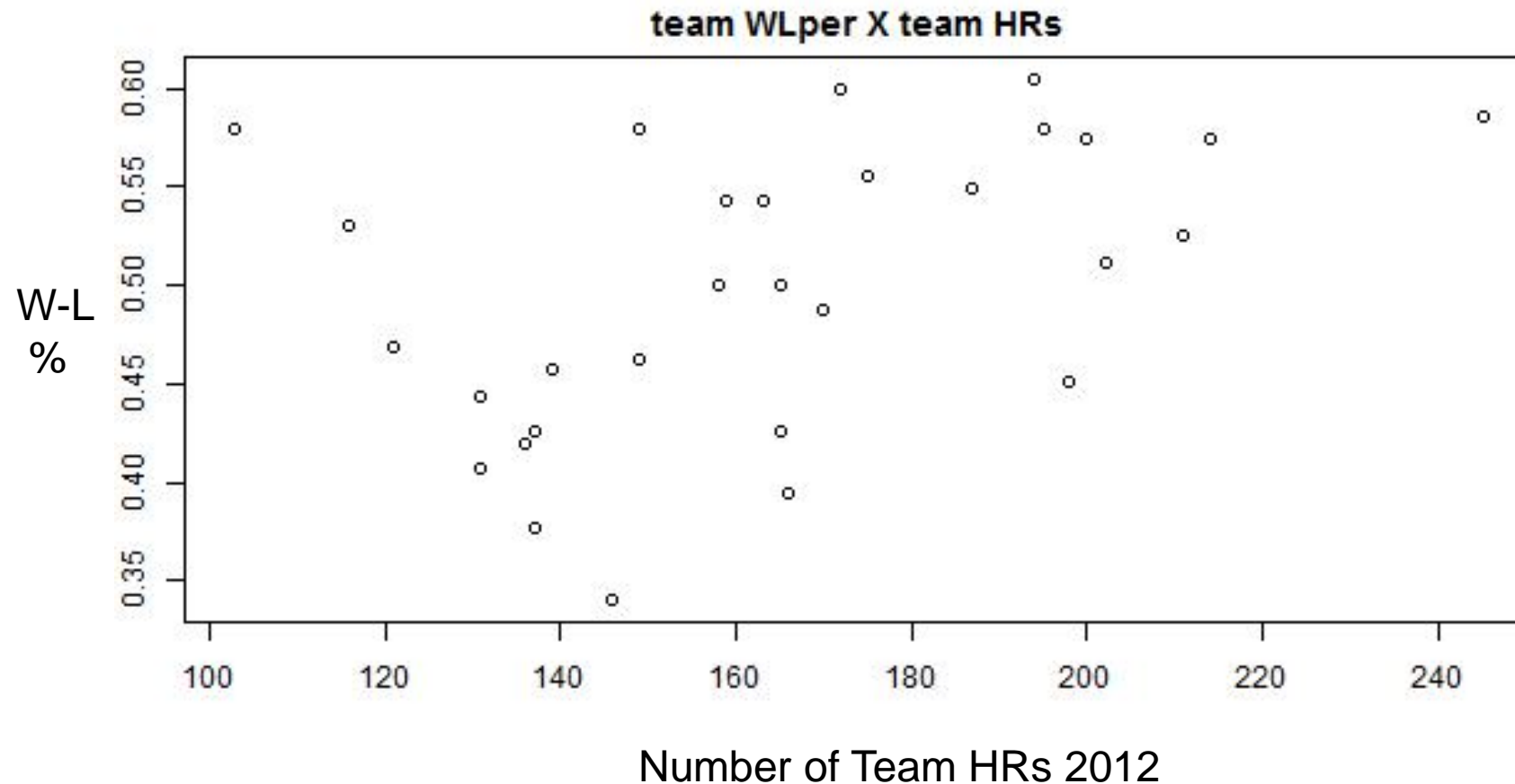
Machine Learning in a Nutshell

Data Mining also

II. R, Scaling in R, Parallel R

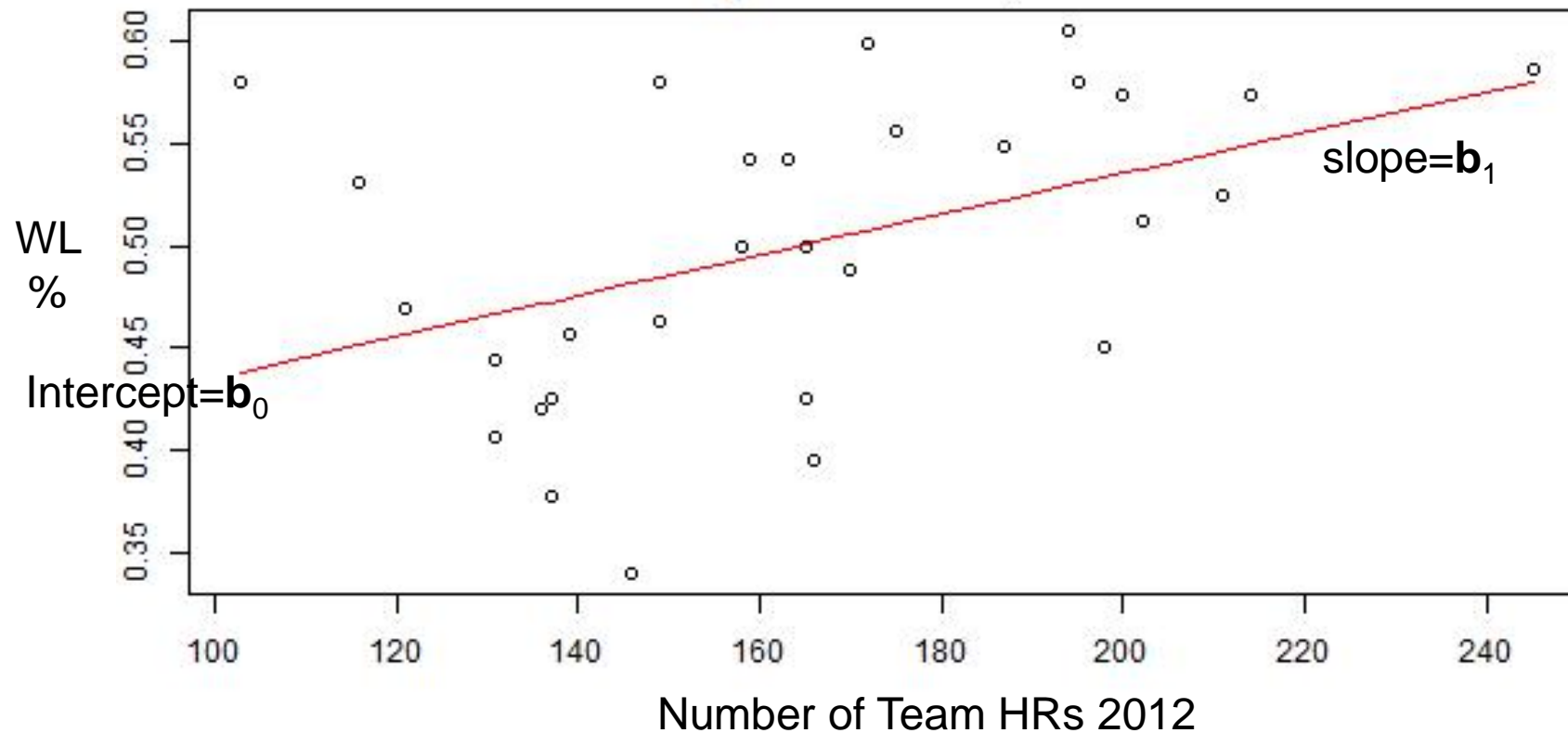
III. Analytics Cases in Comet

A data example: Home Runs and W-L



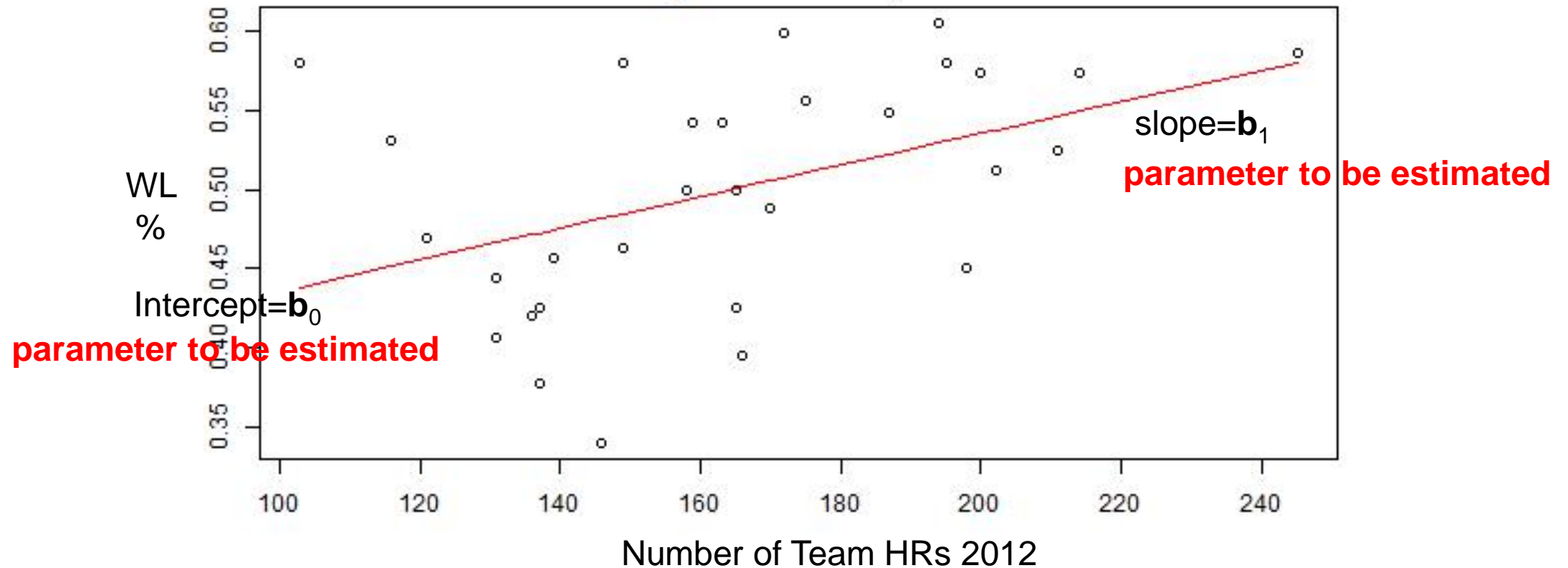
Recall Linear Regression is Fitting a Line

the Model: $y_i = f(x, b) = b_0 * 1 + b_1 * x_i$ (Won-loss percent as function of team home runs)



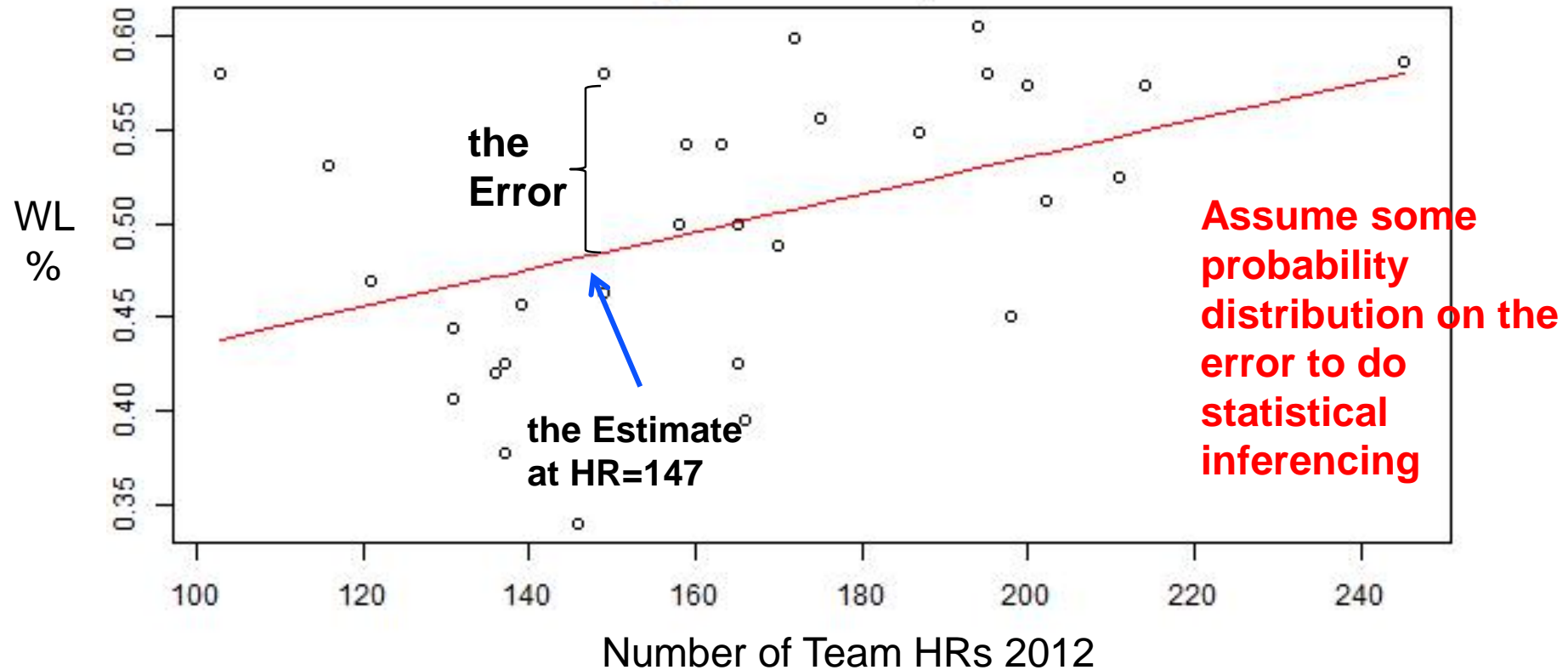
Recall Linear Regression is Fitting a Line

the Model: $y_i = f(x, b) = b_0 * 1 + b_1 * x_i$ (Won-loss percent as function of team home runs)



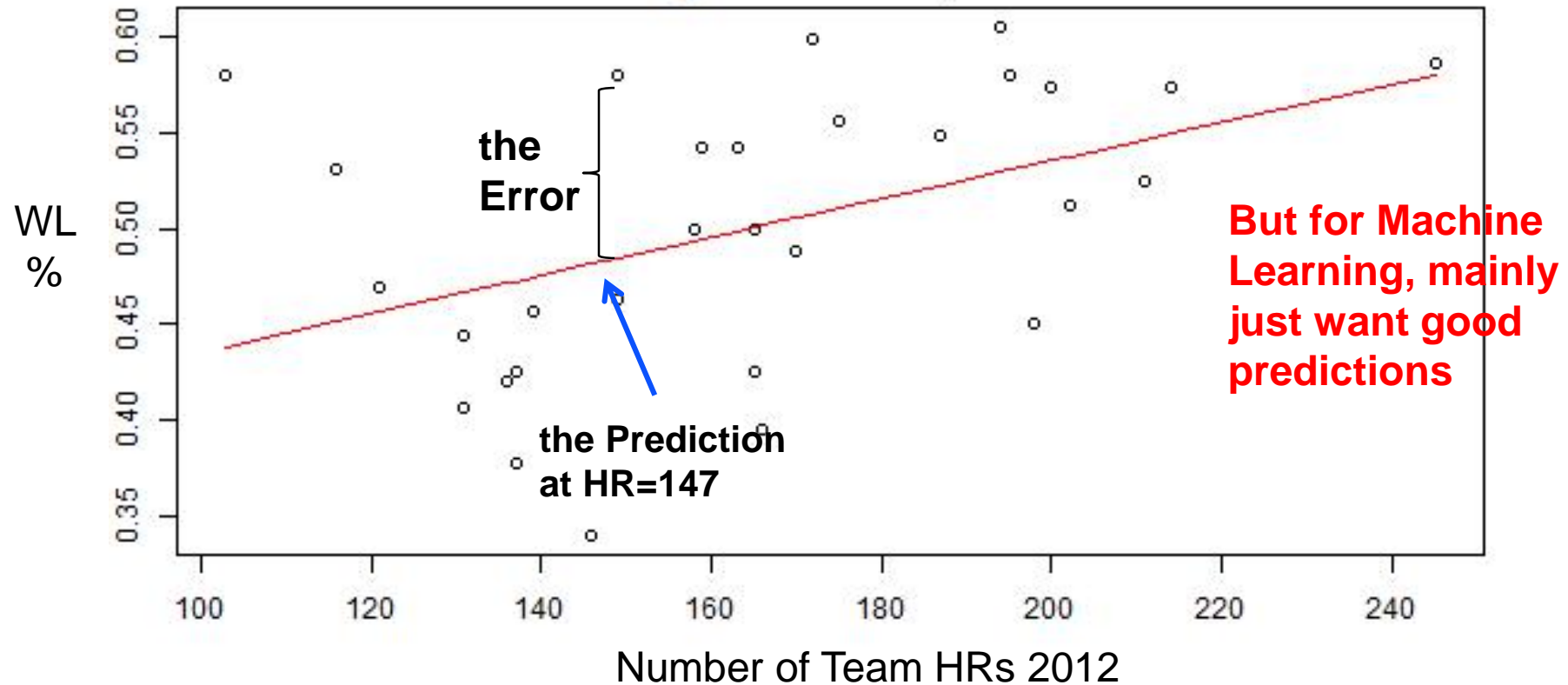
Recall Linear Regression is Fitting a Line

the Model: $y_i = f(x, b) = b_0 * 1 + b_1 * x_i + \text{Gaussian error}$



Recall Linear Regression is Fitting a Line

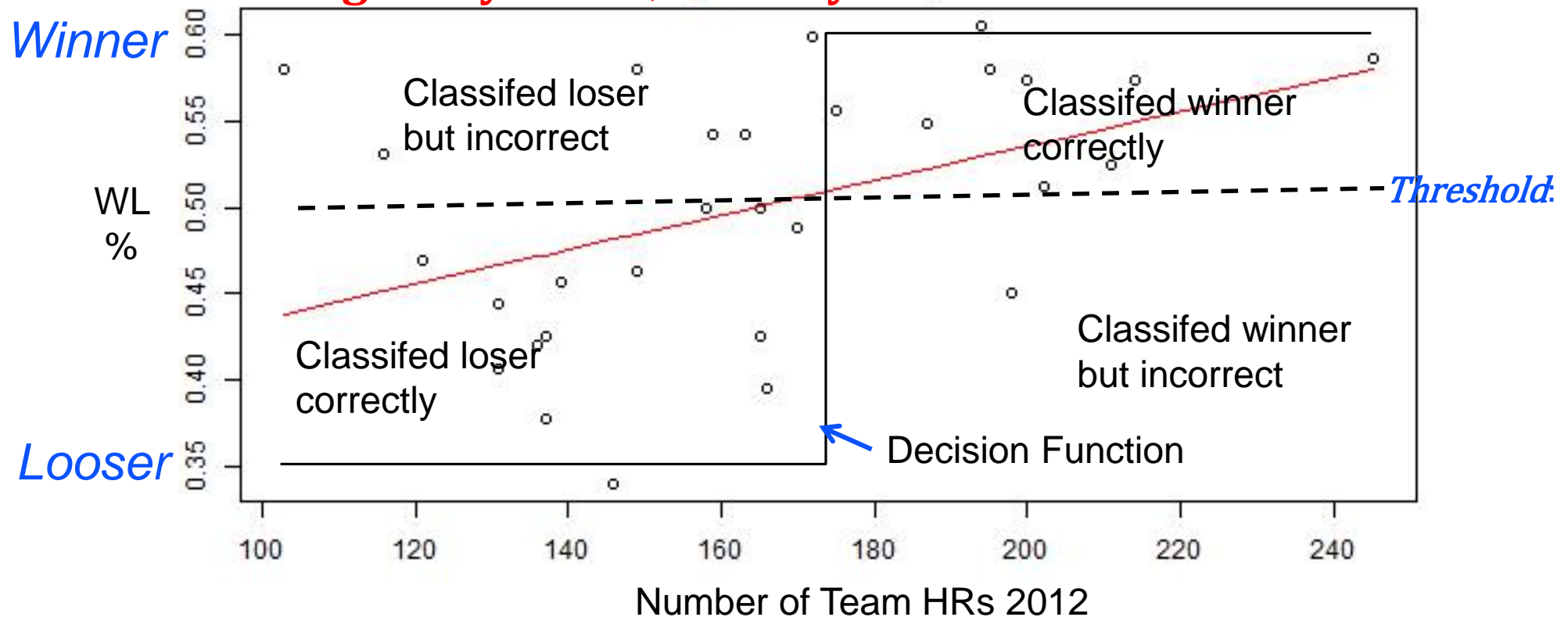
the Model: $y_i = f(x, b) = b_0 * 1 + b_1 * x_i$



Linear Regression with a Decision Threshold

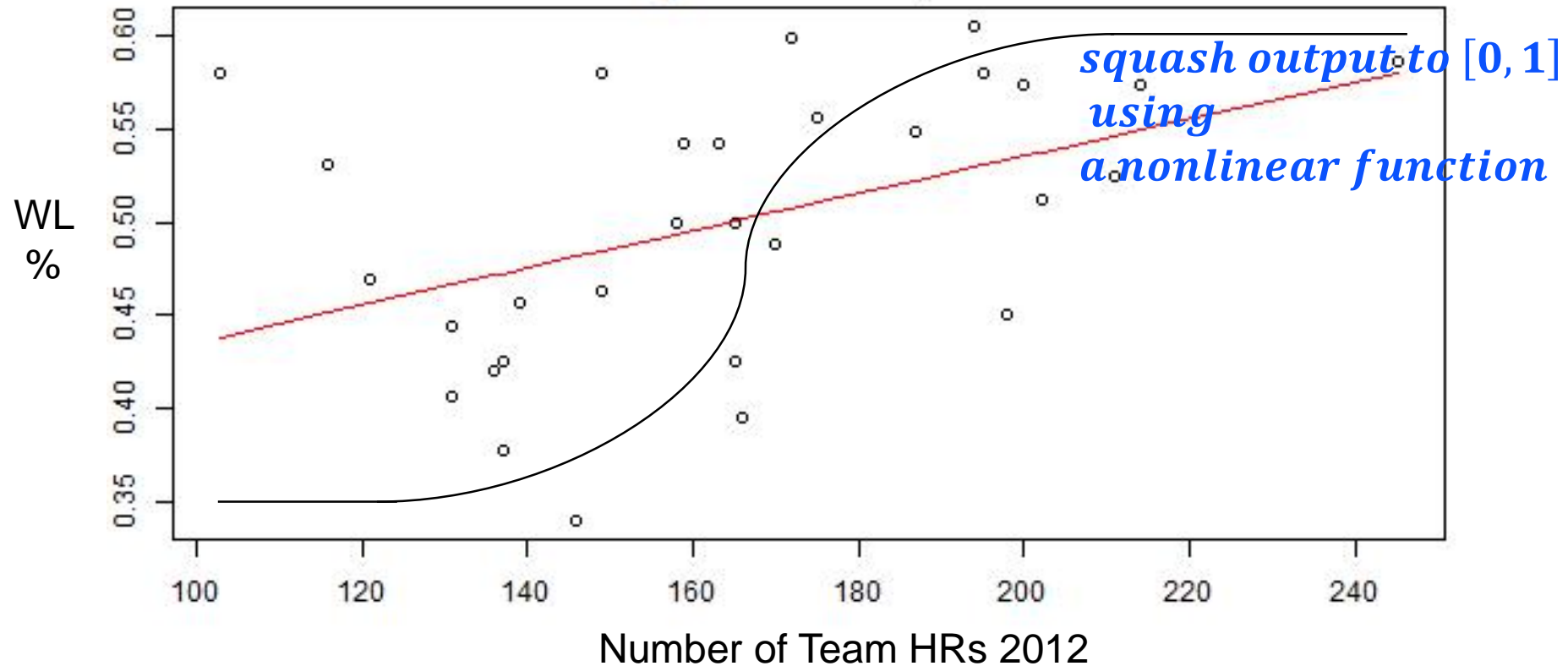
the Model: $y_i = f(x, b) = b_0 * 1 + b_1 * x_i$

*For Classification choose a decision function:
e. g. For $y > 0.5$, Classify as Winner*



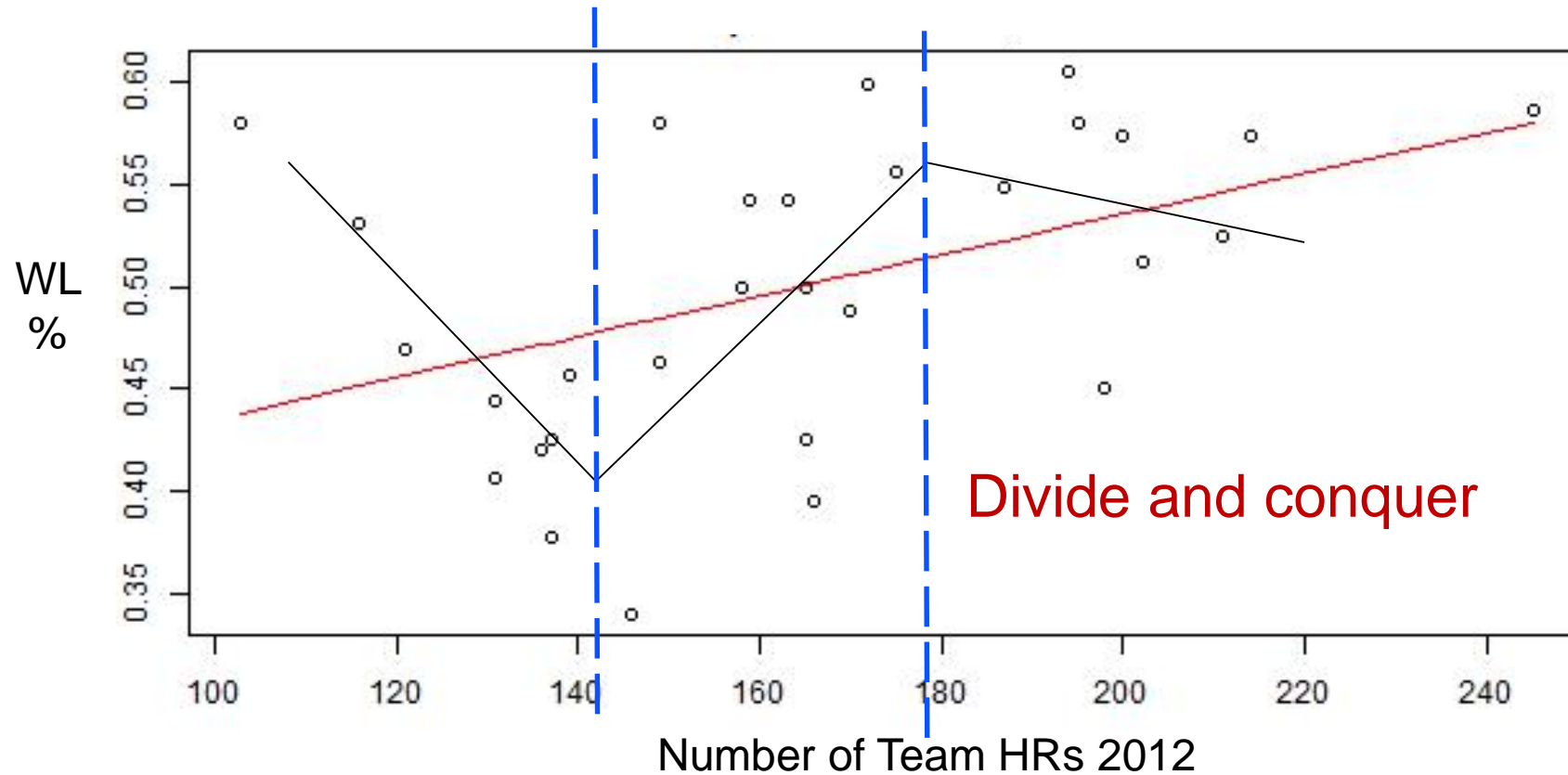
Regression with Logistic Function gives Probability

the Model: $P(y_i = \text{winner} | x_i) = \text{squash}(b_0 * 1 + b_1 * x_i)$



Piecewise Regression

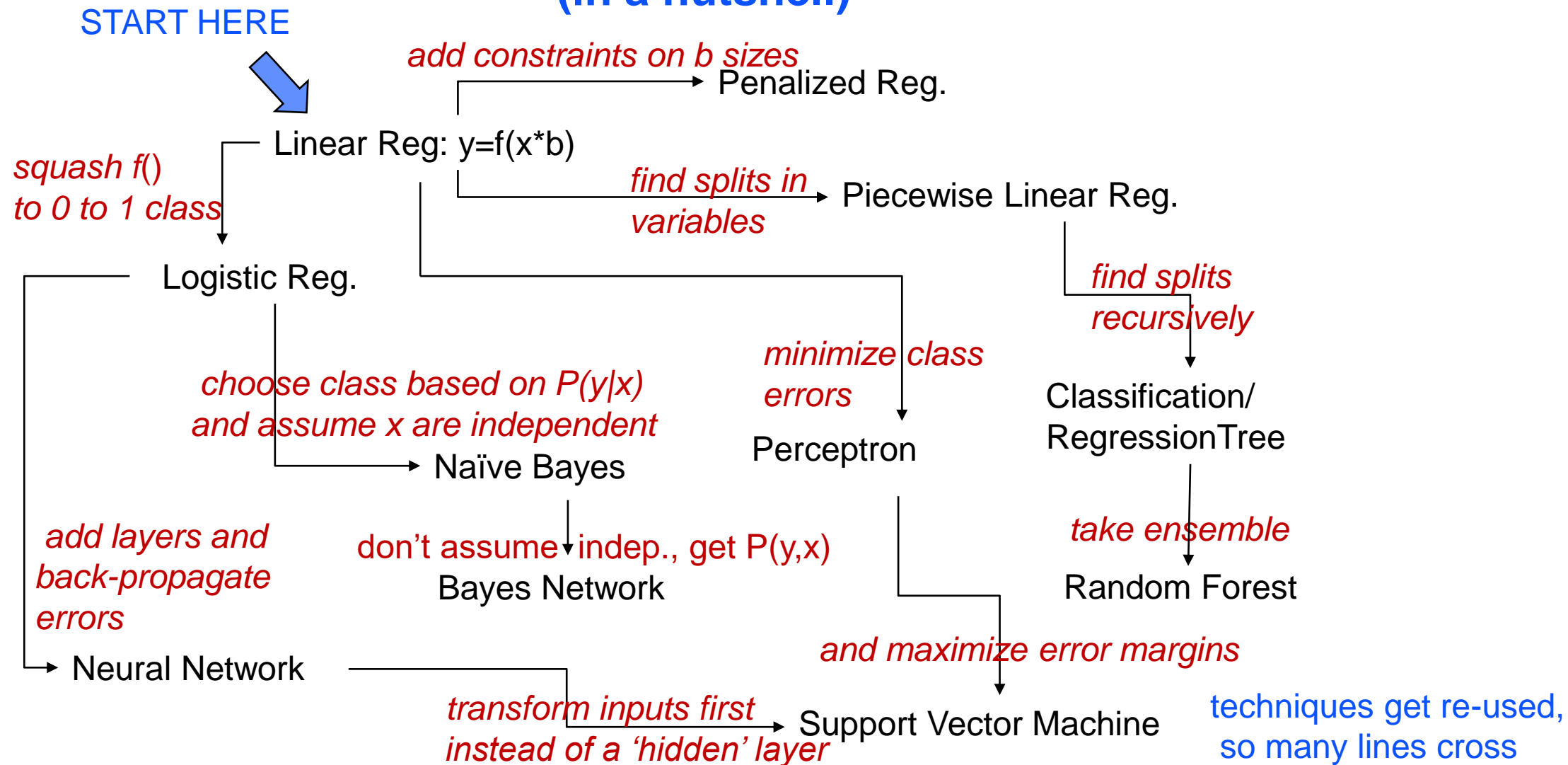
the Model: $y_i = f(x, b) = b_0 * 1 + b_1 * x_i$ for each x segment



Machine Learning Models Are Just Different Functions and Optimizations

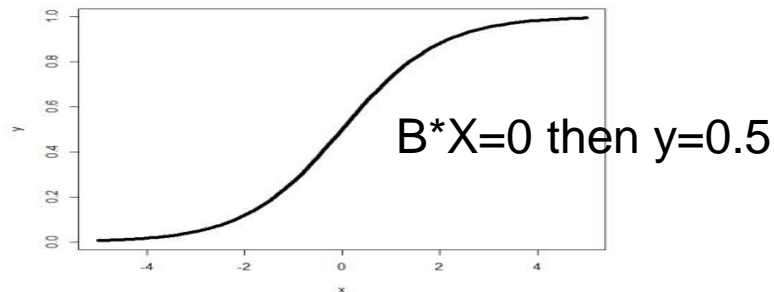
- **What kinds of functions to use**
 - E.g. Linear vs NonLinear
 - E.g. divide input into pieces
- **What to Optimize**
 - Minimize Prediction Error
 - Minimize Classification Errors
 - Maximize Probabilities
- **How to Optimize**
 - Solve directly, take derivatives, or search solutions
 - Use constraints and heuristics

A Map of Machine Learning Models (in a nutshell)



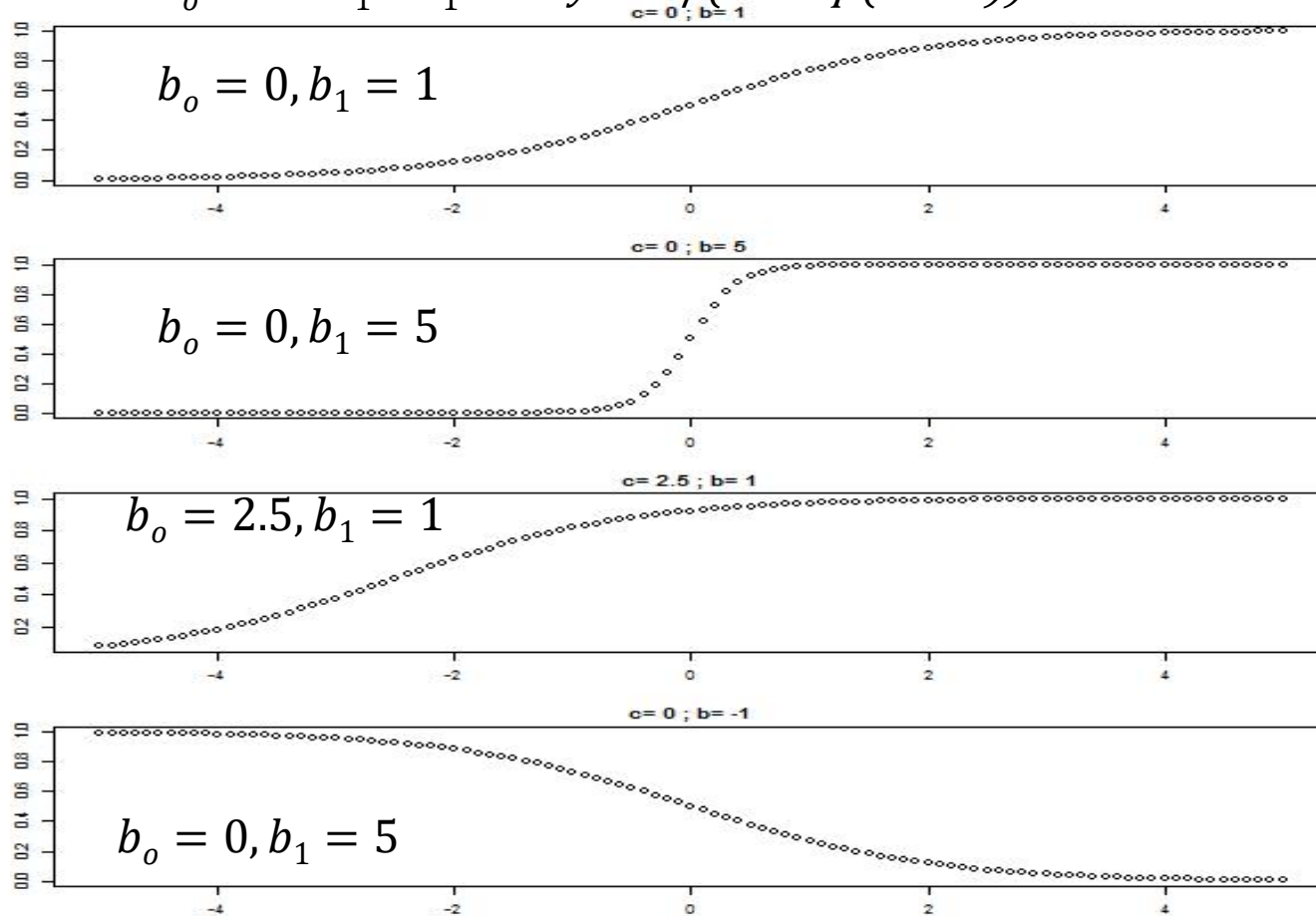
Linear to Logistic to Neural Network model

- $y_1 = b_0 * 1 + b_1 * x_1 + b_2 * x_2 \dots = \mathbf{B} * \mathbf{X}$
- Squash $b_0 * 1 + b_1 * x_1$ to 0,1 range using Logistic Function:



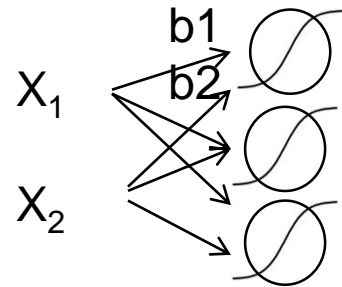
Logistic function w/various B

for $x * b = b_o * 1 + b_1 * x_1$ and $y = 1 / (1 + \exp(-x * b))$

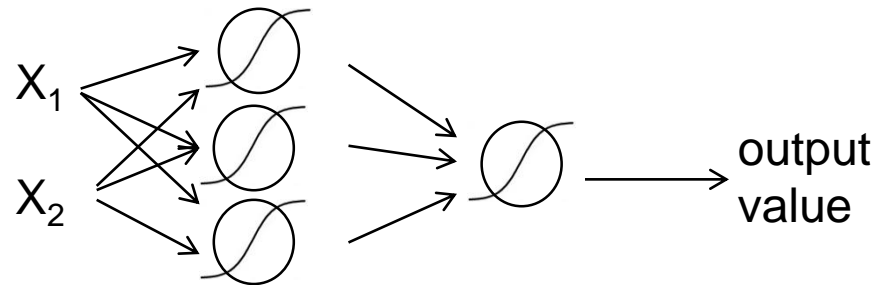


Logistic Regression to Neural Networks

- Use several squash functions (hidden layer)



- Take final combination (output layer)

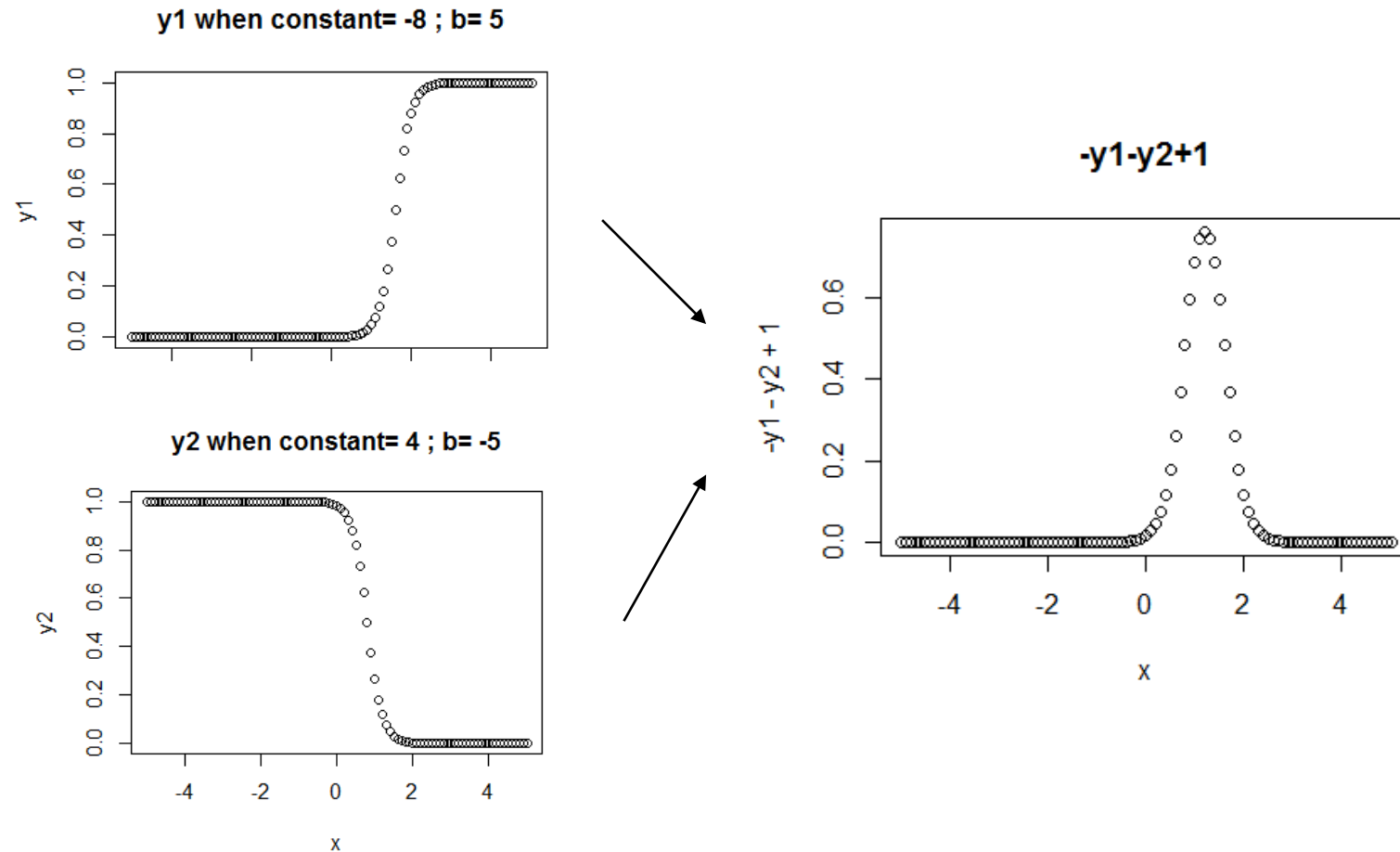


- More powerful but more complex
many parameters, many options, needs more training

Logistic function combination

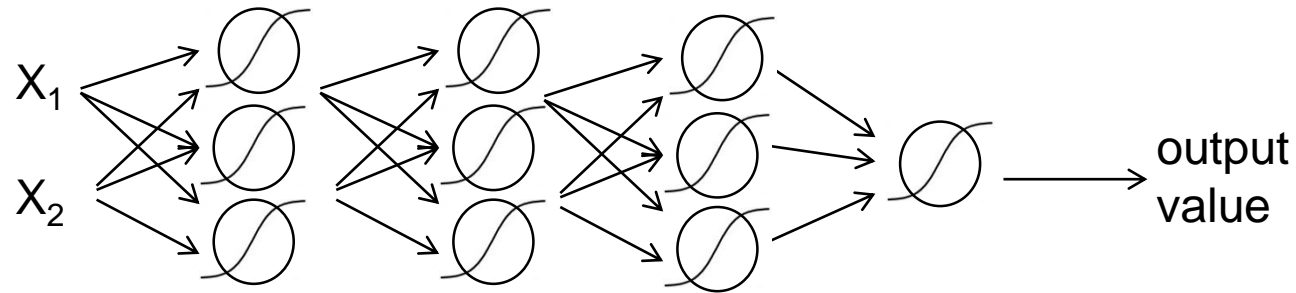
A new 'layer' of logistic functions enables highly nonlinear functions.

They can be considered new features.



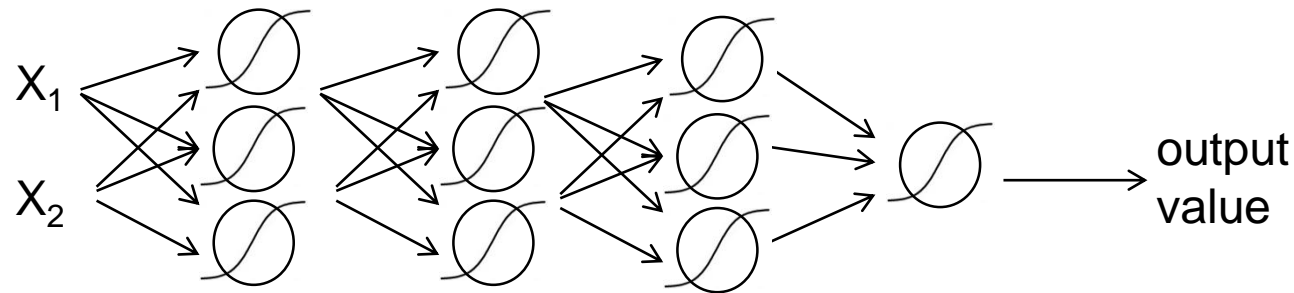
Logistic Regression to Neural Networks

- More hidden layers => More varied features, or 'Deep' Learning



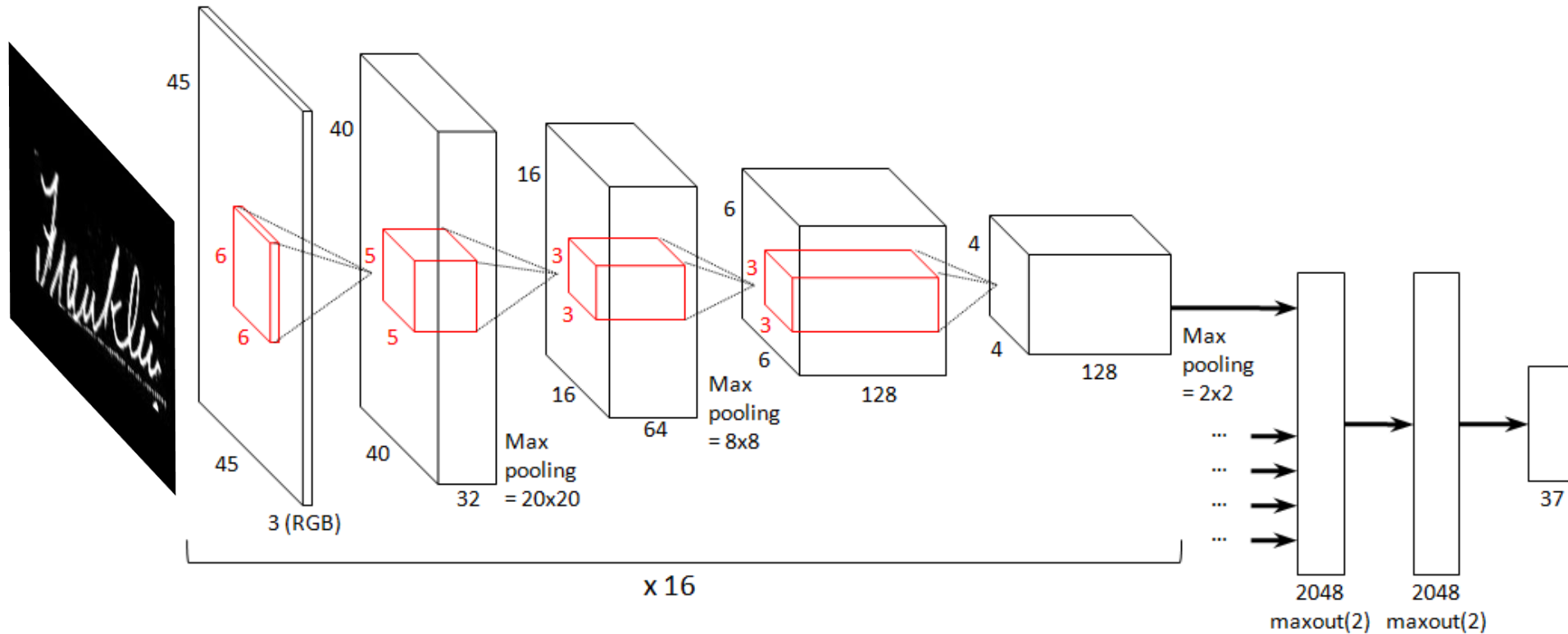
Logistic Regression to Neural Networks

- More hidden layers => More varied features, or 'Deep' Learning



Many more parameters, and error signal at final output layer gets drowned out for lower layers- but penalizing weight sizes, network ensembles, and more data help!

Organize connections into cells, add layers (deepen), add special pooling operations at some layers - you get a convolution network – good for image processing



Machine Learning In practice

- **Complexity:** more parameters => more complex,
*Sometimes you need heuristics for optimization procedures
(ie training)*
Often good to start simple – it helps with interpreting results
- **Tradeoff,** complexity usually => more potential to overfit
(and more sensitivity to data)

validation with test data can help

Data Mining refers to Modelling Workflow

1. Gathering and 'Wrangling' Data

2. Exploratory Data

- Review Variables
- Clustering
 - e.g. Kmeans finds K groups with high inter-group, low intra-group variance
- Dimension Reduction/Factor Analysis
 - e.g. Principal Components find good projections that 'line up' with data variance directions

3. Data Preparation

- Selecting, transforming, scaling variables

4. Build and Evaluate Model

Data Mining also:

- **Data Mining is often used generally to encompass related analysis, such as:**
 - Text/Document Analytics
 - e.g. word clouds and topic modelling
 - Association Learning
 - e.g. what consumer shopping choices are associated
 - Network Analysis
 - e.g. which friends have more influence in a social network

Pause

R, Scaling R, Parallel R

- **A Glimpse of R**
- **R strengths**
- **R and Scaling**
- **Parallel options for R**

The What and Why of R

- **A statistical computing environment**
 - Full set of Statistical/Mathematical functions
 - Programming Language for complete data manipulation
- **Free, Open Source**
- **Extended with user written *packages***
- **Widely used in academic and increasingly in industry**

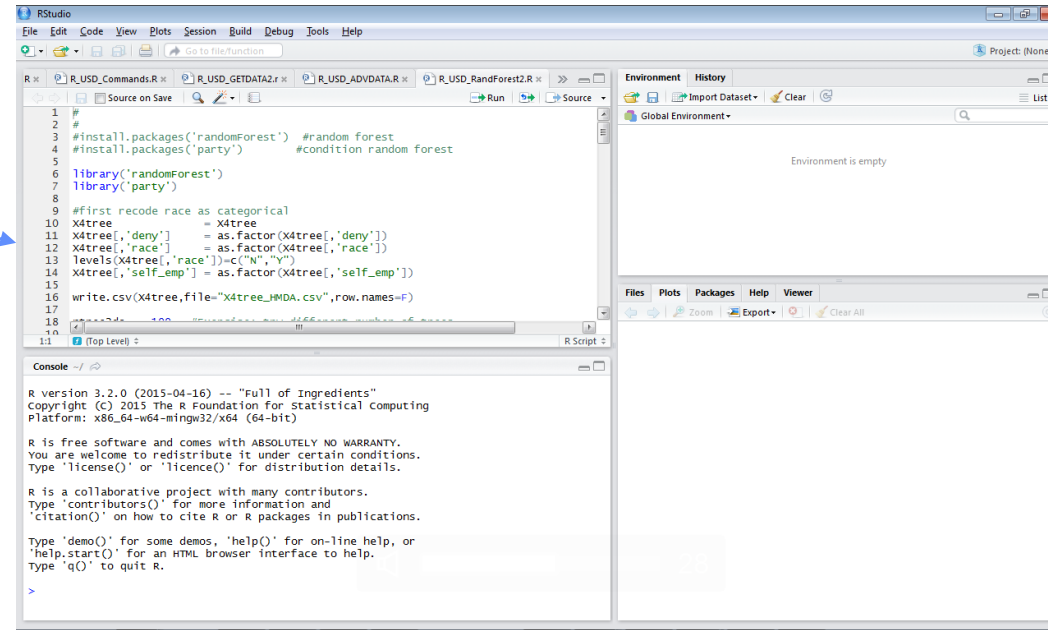
A typical R development workflow

- R studio: An Integrated development environment for R on your local machine – good for development

Menu tab →

Edit window to
Build scripts →

R console →



Environment
Information on
variables and
command
history →

Plots, help docs,
package lists →

R commands in brief

- A typical R code workflow:

#READ DATA (housing mortgage cases)

```
X = read.csv('hmda_aer.csv', header=T, stringsAsFactors=T)
```

#SUBSET DATA

```
indices_2keep = which(X[, 's13'] %in% c(3,4,5))
```

```
X = X[unique(indices_2keep),]
```

#CREATE/TRANSFORM VARIABLES

```
pi_rat = as.numeric(X[, 's46']/100) #debt2income ratio
```

```
race = as.numeric(X[, 's13'] %in% c(3,4)) #make race values 1-4 into values 0 or 1
```

```
deny = as.numeric(X[, 's7']==3) #make deny values into 0 or 1,  
1 only for deny='3'
```

#RUN MODEL and SHOW RESULTS


```
lm_result = lm(deny~race+pi_rat) #lm is 'linear model'
```

```
summary(lm_result)
```

R strengths

- **Sampling/bootstrap methods,**
- **Data Wrangling,**
- **Particular Statistical procedures that you won't find implemented anywhere else, e.g.**
 - Multiple Imputation methods,
 - Instrument Variable (2 stage) Regression
 - Matching subjects for pairwise analysis

R website – everything you want to know...



CRAN
[Mirrors](#)
[What's new?](#)
[Task Views](#)
[Search](#)

About R
[R Homepage](#)
[The R Journal](#)

Software
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Other](#)

Documentation
[Manuals](#)
[FAQs](#)
[Contributed](#)

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

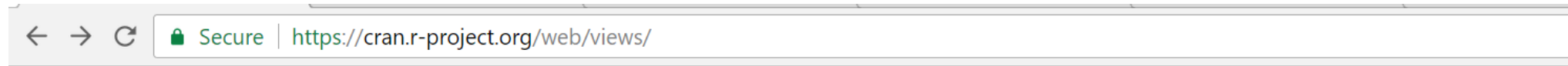
R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (Friday 2017-04-21, You Stupid Darkness) [R-3.4.0.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

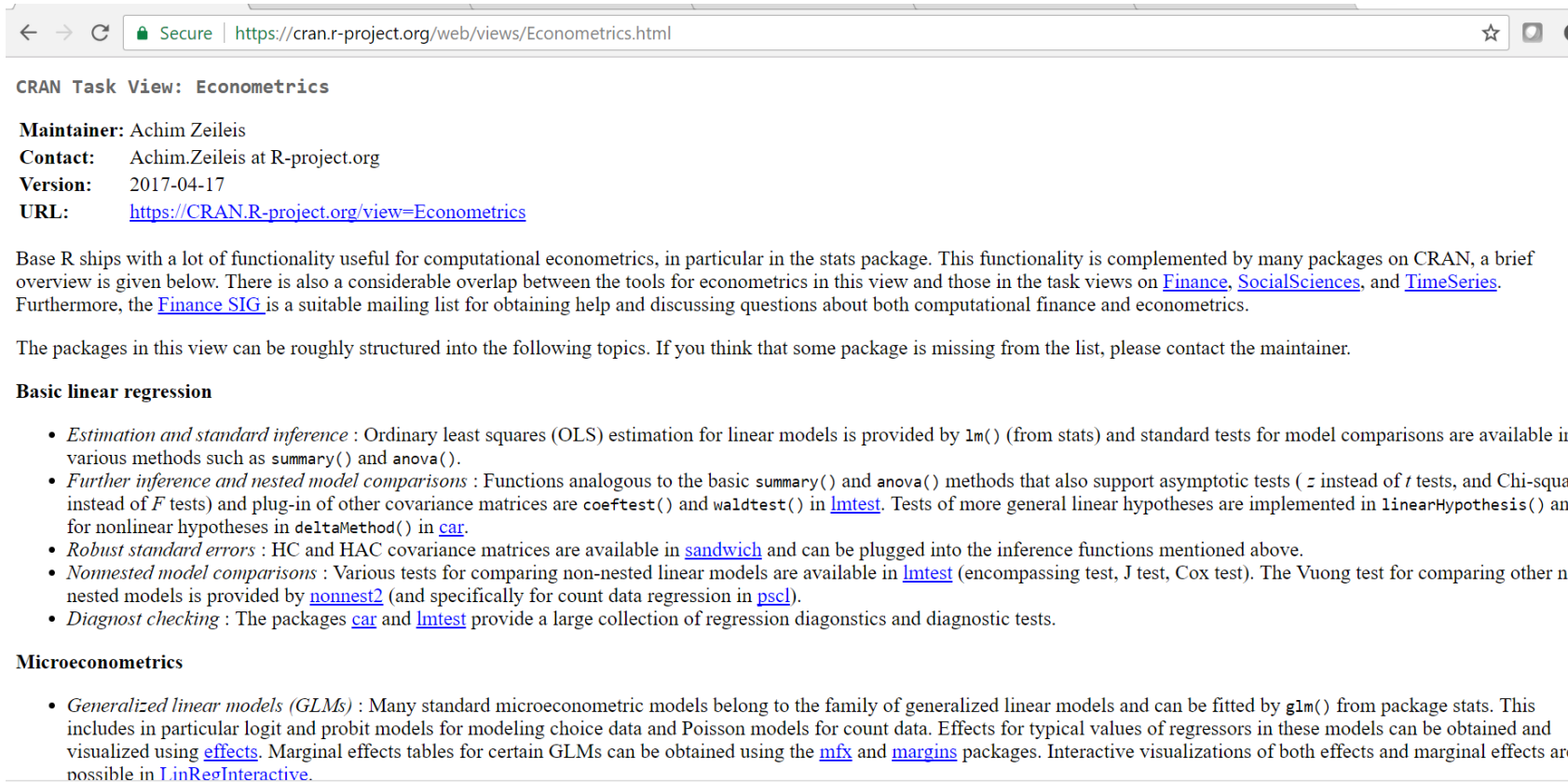
e.g. CRAN -> Task Views



CRAN Task Views

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis

e.g. CRAN -> Task Views -> Econometrics



The screenshot shows a web browser window with the address bar displaying "https://cran.r-project.org/web/views/Econometrics.html". The page title is "CRAN Task View: Econometrics". The maintainer is Achim Zeileis, with contact information "Achim.Zeileis at R-project.org", version "2017-04-17", and URL "https://CRAN.R-project.org/view=Econometrics". The text describes the base R functionality for computational econometrics, mentioning the stats package and various CRAN packages like Finance, SocialSciences, and TimeSeries. It also mentions the Finance SIG mailing list. The packages are structured into topics: Basic linear regression and Microeconometrics. The Basic linear regression section lists five topics: Estimation and standard inference, Further inference and nested model comparisons, Robust standard errors, Nonnested model comparisons, and Diagnost checking. The Microeconometrics section lists one topic: Generalized linear models (GLMs).

CRAN Task View: Econometrics

Maintainer: Achim Zeileis
Contact: Achim.Zeileis at R-project.org
Version: 2017-04-17
URL: <https://CRAN.R-project.org/view=Econometrics>

Base R ships with a lot of functionality useful for computational econometrics, in particular in the stats package. This functionality is complemented by many packages on CRAN, a brief overview is given below. There is also a considerable overlap between the tools for econometrics in this view and those in the task views on [Finance](#), [SocialSciences](#), and [TimeSeries](#). Furthermore, the [Finance SIG](#) is a suitable mailing list for obtaining help and discussing questions about both computational finance and econometrics.

The packages in this view can be roughly structured into the following topics. If you think that some package is missing from the list, please contact the maintainer.

Basic linear regression

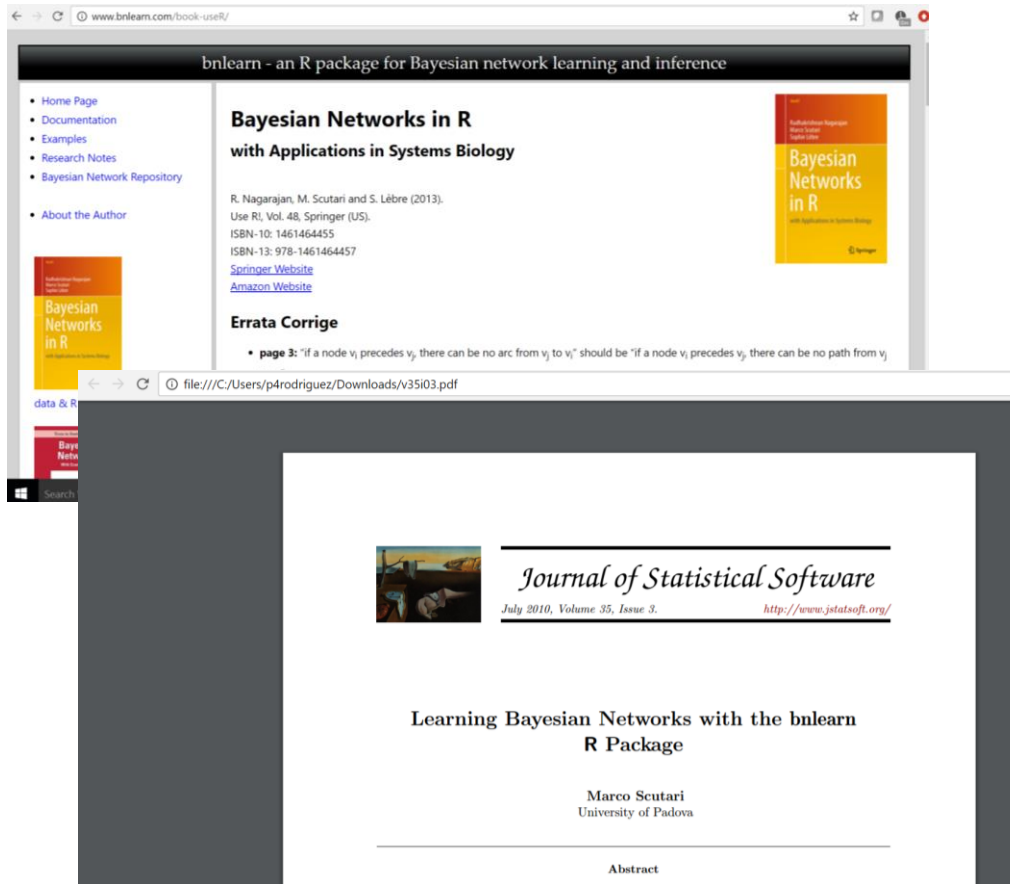
- *Estimation and standard inference* : Ordinary least squares (OLS) estimation for linear models is provided by `lm()` (from stats) and standard tests for model comparisons are available in various methods such as `summary()` and `anova()`.
- *Further inference and nested model comparisons* : Functions analogous to the basic `summary()` and `anova()` methods that also support asymptotic tests (z instead of t tests, and Chi-square instead of F tests) and plug-in of other covariance matrices are `coefTest()` and `waldtest()` in [lmtest](#). Tests of more general linear hypotheses are implemented in `linearHypothesis()` and for nonlinear hypotheses in `deltaMethod()` in [car](#).
- *Robust standard errors* : HC and HAC covariance matrices are available in [sandwich](#) and can be plugged into the inference functions mentioned above.
- *Nonnested model comparisons* : Various tests for comparing non-nested linear models are available in [lmtest](#) (encompassing test, J test, Cox test). The Vuong test for comparing other non-nested models is provided by [nonnest2](#) (and specifically for count data regression in [pscl](#)).
- *Diagnost checking* : The packages [car](#) and [lmtest](#) provide a large collection of regression diagnostics and diagnostic tests.

Microeconometrics

- *Generalized linear models (GLMs)* : Many standard microeconomic models belong to the family of generalized linear models and can be fitted by `glm()` from package stats. This includes in particular logit and probit models for modeling choice data and Poisson models for count data. Effects for typical values of regressors in these models can be obtained and visualized using [effects](#). Marginal effects tables for certain GLMs can be obtained using the [mfx](#) and [margins](#) packages. Interactive visualizations of both effects and marginal effects are possible in [LinRegInteractive](#).

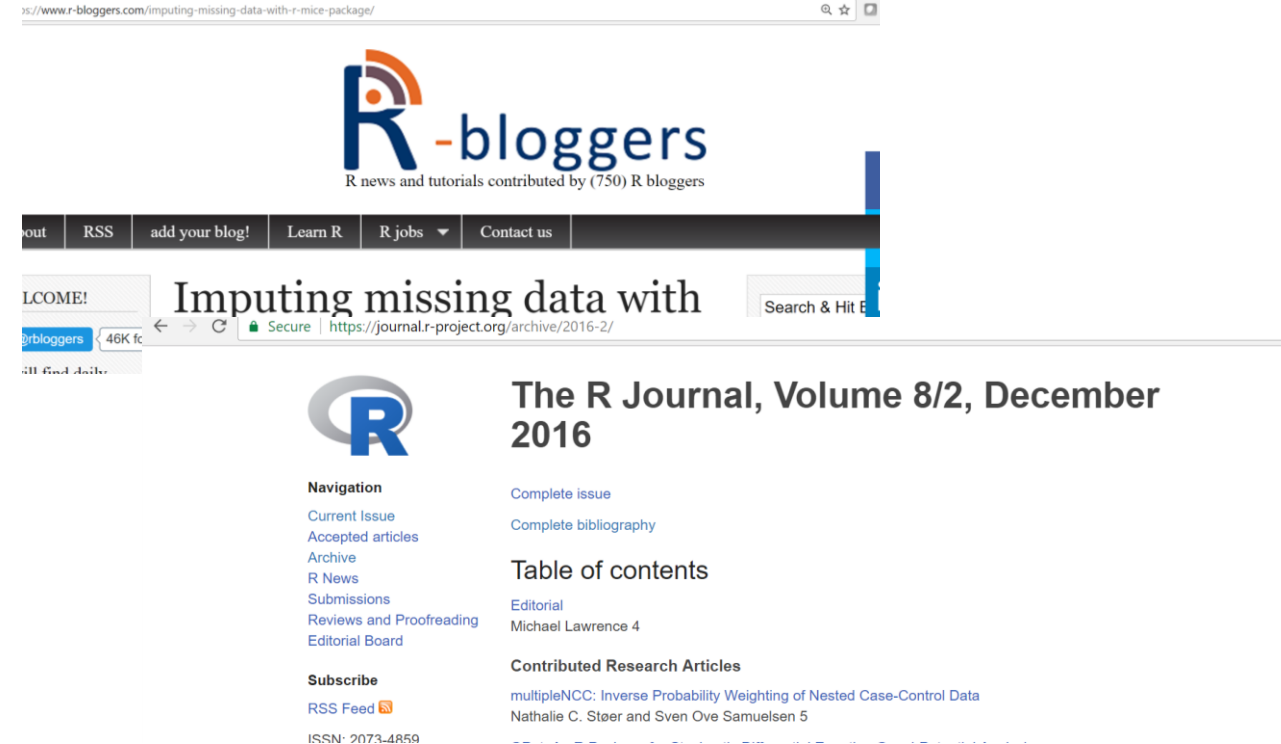
But how do you know which package to choose?

Other places to find guidance



The top part of the image shows a web browser displaying the **bnlearn** website. The page title is "bnlearn - an R package for Bayesian network learning and inference". It features a sidebar with links like "Home Page", "Documentation", "Examples", "Research Notes", and "Bayesian Network Repository". The main content area is titled "Bayesian Networks in R with Applications in Systems Biology" by R. Nagarajan, M. Scutari and S. Lèbre (2013). It includes the book cover, ISBN information, and links to the Springer and Amazon websites. Below this is an "Errata Corrigé" section.

The bottom part of the image shows a PDF document titled "Journal of Statistical Software". The article is "Learning Bayesian Networks with the bnlearn R Package" by Marco Scutari, University of Padova. The journal issue is "July 2010, Volume 35, Issue 3". The URL <http://www.jstatsoft.org/> is provided.



The top part of the image shows the **R-bloggers** website. The header features the "R-bloggers" logo and the tagline "R news and tutorials contributed by (750) R bloggers". A navigation bar includes links for "about", "RSS", "add your blog!", "Learn R", "R jobs", and "Contact us". The main content area is titled "Imputing missing data with" and shows a search bar with the text "46K f".

The bottom part of the image shows the website for "The R Journal, Volume 8/2, December 2016". It features the R logo and a navigation menu with links for "Current Issue", "Accepted articles", "Archive", "R News", "Submissions", "Reviews and Proofreading", and "Editorial Board". There is also a "Subscribe" section with an "RSS Feed" link. The main content area lists "Contributed Research Articles" including "multipleNCC: Inverse Probability Weighting of Nested Case-Control Data" by Nathalie C. Steer and Sven Ove Samuelsen.

Consider: R and missing data packages

- **Several packages, such as ‘mice’ , ‘amelia’**
- **Produces multiple data sets**
- **Iterates over missing data estimates and linear model estimates**
 - Mice uses Gibbs sampling (slower)
 - Amelia uses Expectation Maximization (faster)
- **Beware of correlation in variables**
 - Matrices not invertible (affects Amelia)

R and missing data

- **Sample R code using Amelia:**

Data: UN conflict data in pairs of countries

300K rows ~ 1 hour on Gordon compute node (not run on the user's PC)

1K-100K entries missing per col for about 20 of 50 cols

```
# run the imputation
library('amelia')
a.out <- amelia(data, ts = "year", cs = "dyadid",
               idvars = c("dyadidyr", "cntryera", "statea", "stateb"),
               intercs=FALSE, p2s = 2, m=10, parallel = "multicore")
```

time variable → `ts = "year"`

crosssection → `cs = "dyadid"`

ignore 'id' variables → `idvars = c("dyadidyr", "cntryera", "statea", "stateb")`

interactions → `intercs=FALSE`

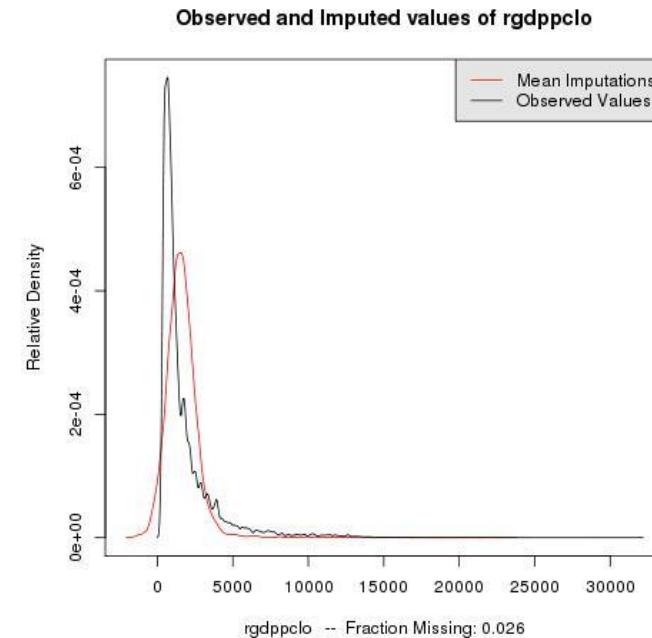
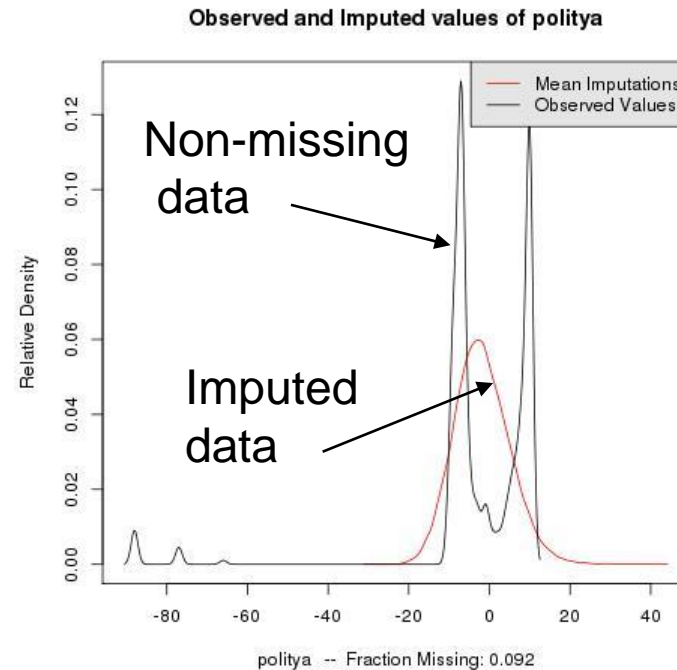
parallel options → `p2s = 2, m=10, parallel = "multicore"`

R and missing data

#Perform QA on missing data by comparing density of imputed & original data

```
compare.density(a.out, var="politya")
```

```
compare.density(a.out, var='rgdpcontg')
```



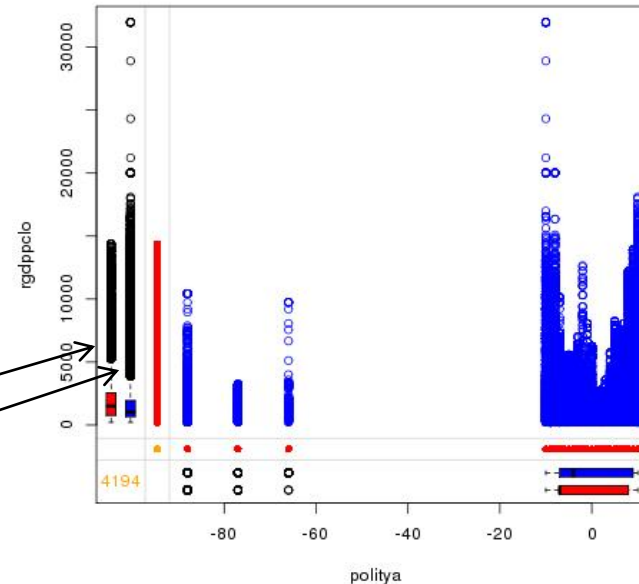
R and missing data

Useful library for printing margin plots, to compare histograms for single variables
and histograms conditional on missing/non-missing data

```
library('VIM')
```

```
marginplot(gart2use[,c('politya', 'rgdppclo')],  
           col=c('blue','red','orange'))
```

*dist of rgdppclo when
politya missing
politya present*

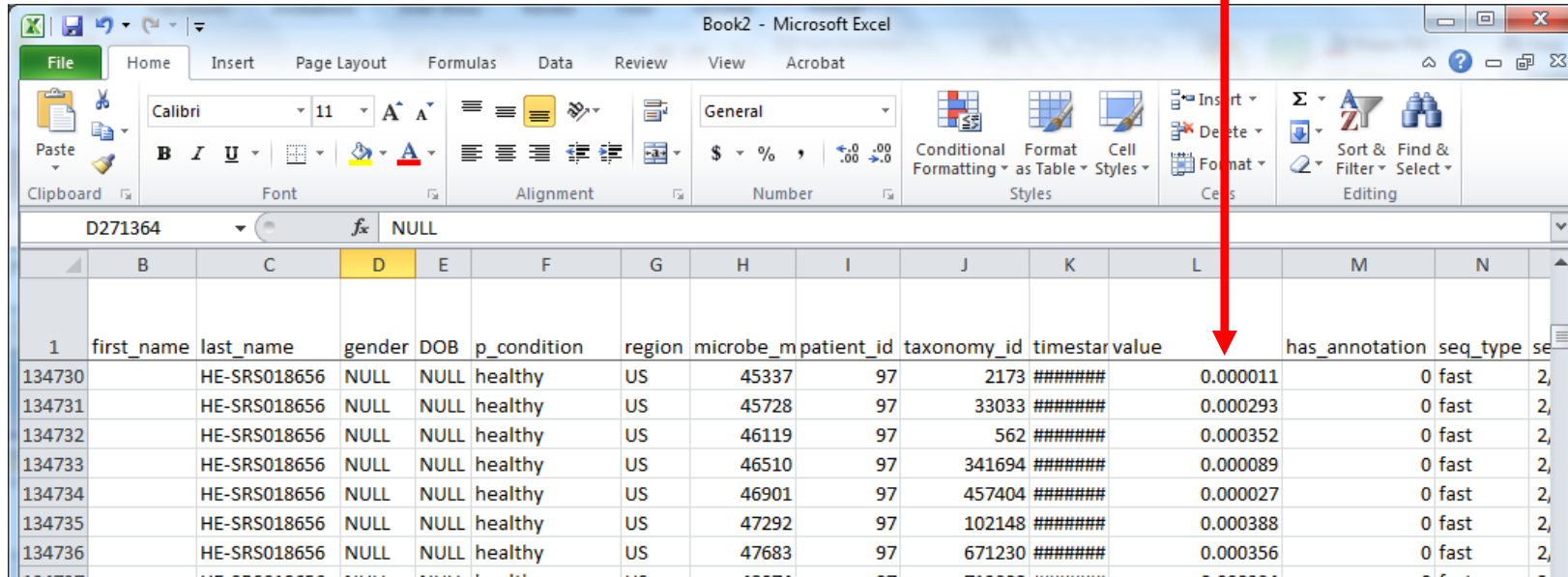


A Data Wrangling example

Issue:

~3000 rows for each subject (1 row for each measurement).

But need 1 row per subject.



The screenshot shows a Microsoft Excel spreadsheet titled 'Book2 - Microsoft Excel'. The ribbon includes 'File', 'Home', 'Insert', 'Page Layout', 'Formulas', 'Data', 'Review', 'View', and 'Acrobat'. The 'Home' ribbon is active, showing options for Clipboard, Font, Alignment, Number, Styles, Cells, and Editing. The formula bar shows 'D271364' and 'NULL'. The data table has columns B through N. A red arrow points to the 'value' column (column L).

	B	C	D	E	F	G	H	I	J	K	L	M	N
1	first_name	last_name	gender	DOB	p_condition	region	microbe_mpatient_id	taxonomy_id	time	star	value	has_annotation	seq_type
134730		HE-SRS018656	NULL	NULL	healthy	US	45337	97	2173	#####	0.000011	0	fast
134731		HE-SRS018656	NULL	NULL	healthy	US	45728	97	33033	#####	0.000293	0	fast
134732		HE-SRS018656	NULL	NULL	healthy	US	46119	97	562	#####	0.000352	0	fast
134733		HE-SRS018656	NULL	NULL	healthy	US	46510	97	341694	#####	0.000089	0	fast
134734		HE-SRS018656	NULL	NULL	healthy	US	46901	97	457404	#####	0.000027	0	fast
134735		HE-SRS018656	NULL	NULL	healthy	US	47292	97	102148	#####	0.000388	0	fast
134736		HE-SRS018656	NULL	NULL	healthy	US	47683	97	671230	#####	0.000356	0	fast

Data Wrangling example

Reshape command to get a 'wide' format with repeated measurements in separate columns of the same record

```
Xa = read.csv('patmeasjoinall_0219.csv', header=T, sep='\t')
```

```
Xatest = reshape(Xa, direction = "wide",  
                 idvar=c("patient_id", "gender", "DOB", "p_condition", "region"),  
                 v.names="value", timevar="taxonomy_id", sep=".")
```

```
write.csv(Xatest, file="Xa_reshaped");
```

Row keys

column keys

measurement values are now columns in one row

	patient_id	gender	DOB	p_condition	region	timestamp	value.217	value.3303	value.562	value.3416	value.457	value.102	value.671	value.102
1	2	NA	NA	healthy	EU	1/1/2012	0.00137	2.80E-05	0.008496	2.20E-05	0	6.60E-05	1.90E-05	5.00E-05
2	3	NA	NA	healthy	EU	1/1/2012	0	1.70E-05	2.90E-05	1.50E-05	1.10E-05	5.00E-05	2.80E-05	3.00E-05
3	4	NA	NA	healthy	EU	1/1/2012	0.00527	1.30E-05	0.009828	2.60E-05	0	4.10E-05	1.70E-05	2.00E-05
4	5	NA	NA	ulcerou	EU	1/1/2012	0.0154	2.30E-05	0.005054	1.30E-05	0	0.00041	0	5.00E-05
5	6	NA	NA	ulcerou	EU	1/1/2012	0.00038	1.20E-05	6.70E-05	1.80E-05	1.90E-05	5.00E-05	1.80E-05	2.00E-05
6	7	NA	NA	ulcerou	EU	1/1/2012	0	0	0.000138	0	0	0	0	0
7	36	NA	NA	crohn's	EU	1/1/2012	0	0	0.011707	1.70E-05	0	5.50E-05	0	0
8	37	NA	NA	crohn's	EU	1/1/2012	0.00237	1.80E-05	0.000158	1.70E-05	0	1.00E-05	2.30E-05	4.00E-05

P.R. SDSC UCSD

Complex Social Science Gateway – a tool for cross-cultural analysis in R

Select dataset,
Select variables,
Submit analysis

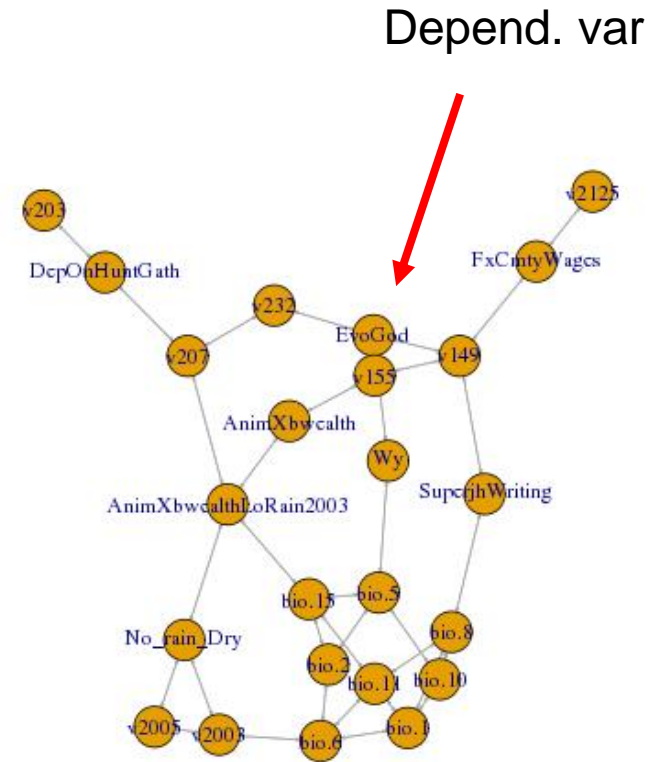
<http://socscicompute.ss.uci.edu/>

The screenshot shows the Galaxy / CoSci web interface at socscicompute.ss.uci.edu. The interface is divided into several panels:

- Tools Panel (Left):** Contains a search bar and a list of tools categorized under "COSSCI TOOLS" and "GALAXY TOOLS". Under "COSSCI TOOLS", the tool "DEF01d Dow Eff" is selected, indicated by a blue arrow from the text "Select dataset, Select variables, Submit analysis".
- Main Panel (Center):** Displays the configuration for the "DEF01d (version DEF01d)" tool. It includes fields for:
 - Dataset:** Set to "SCCS".
 - Dummy variables:** A text input field.
 - Dependent variables:** Set to "v2007".
 - Independent variables in restricted model:** Set to "AnimXbwealth,bio.5,FxCmtyWages,No_rain_Dry,v53,lati".
 - Independent variables in UNrestricted model:** Set to "v206,v208,v2125,v61,v2008,v2009,v2010,v2003,v855".
 - Exogenous variables:** Set to "v149v2006,v149,v2006".
 - Additional variables to consider:** A text input field.
 - Variables:** A section for defining variable names, with "Variable 1" set to "dx\$FxCmtyWages".
- History Panel (Right):** Shows a list of previous analyses. The top entry is "174: DEF01d Map" with a size of 8.6 MB. Below it, a code block shows the R script generated for the analysis, including dataset loading and variable selection.

R Analysis options

- **Two-stage least squares to handle spatially correlated errors (OLS, logit, multinomial logit)**
- **Bootstrap sampling of Bayesian network (package bnlearn) to confirm OLS effects, or suggest other moderating/mediating effects**



Scaling, practically

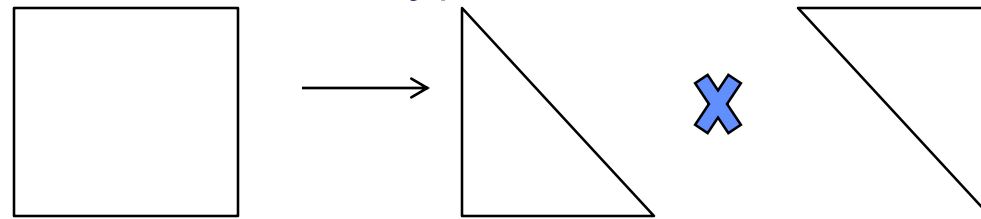
- **Scaling (with or without more data):**
 - more complex analysis (ie optimizations)
 - more sampling (ie more trees in Random Forest)
- **Sometimes easy to parallelize (like with sampling),**
- **Sometimes too much communication between parts (matrix inversion)**

Scaling In a nutshell

- R takes advantage of math libraries for vector operations
- R packages provide multicore, multinode (snow), or map/reduce (RHadoop) options
- However, model implementations not necessarily built to use parallel backends
 - Some models more amenable to parallel versions

Consider Regression Computations

- **Linear Model:** $Y = X * B$
where Y =outcomes , X =data matrix
- **Algebraically, we could:**
 - take “inverse” of $X * Y = B$ (time consuming)
 - Or take gradient descent (for likelihoods and generalized models)
- **Or, better:**
 - QR decomposition of X into triangular matrices (easier to solve but more memory)



Consider Regression models in R

- **Related Models and Functions :**

lm() #Linear Model

glm() #Generalized Linear Model
 (logistic regression, etc)

aov() #Analysis of Variance
 (returns ANOVA table of F-scores)

All these work on system of equations

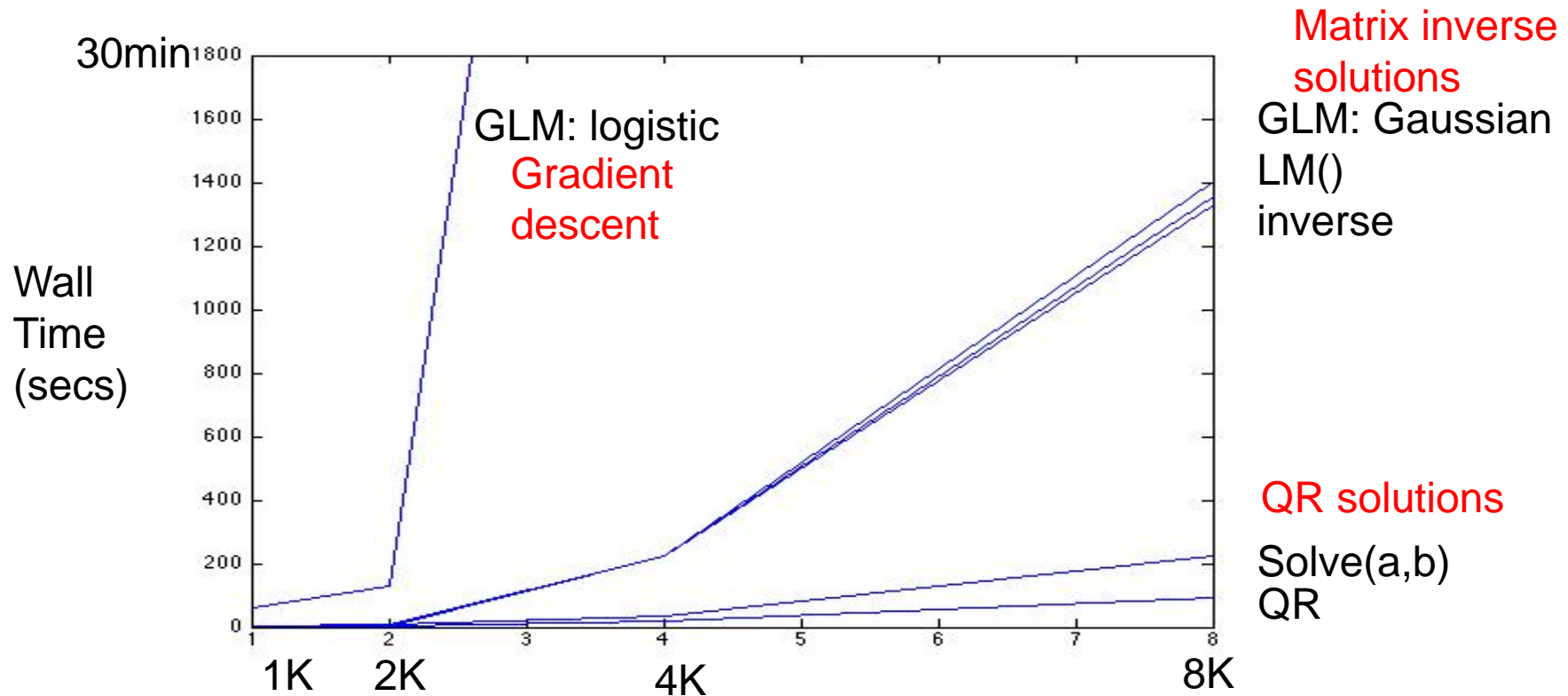
Solving Linear Systems

Performance with R, 1 compute node

R:

```
glm(Y~X,family=gaussian) #gaussn regrssn (like lm)
```

```
glm(Y~X,family=binomial) # logistic regrssn (Y=0 or 1)
```



Data Matrix Size (i.e. square, rowsXcol)

R multicore

- **Intel Math Kernel Libraries provides fast operations for vector operations**
- **Uses threads across cpu cores to pass data & commands**

R multicore

- Run loop iterations on separate cores

```
install.packages(doMC)
library(doMC)
registerDoMC(cores=24)
getDoParWorkers()

results = foreach(i=1:24,.combine=rbind) %dopar%
{ ... your code here
  return( a variable or object )
}
```

allocate workers ←

%dopar% puts loops across cores, (loops are independent) %do% runs it serially ↙

↗ *returned items 'combined' into list by default*

↖ *specify to combine results into array with row bind*

R multinode: parallel backend

- Run loop iterations on separate nodes

```
install.packages('doSNOW')
library('doSNOW')
...
cl <- makeCluster( mpi.universe.size()-1, type='MPI' )
clusterExport(cl,c('data'))
registerDoSNOW(cl)

results = foreach(i=1:47,.combine=rbind) %dopar%
{ ... your code here

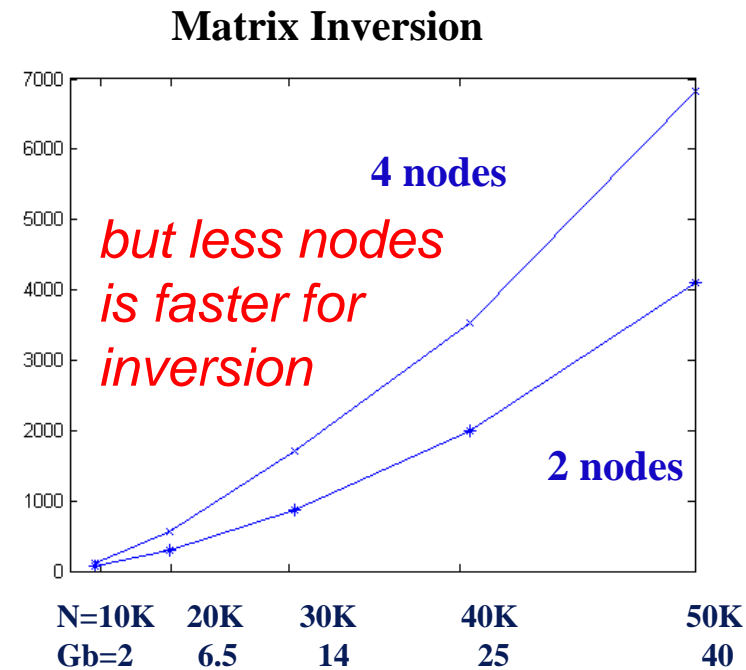
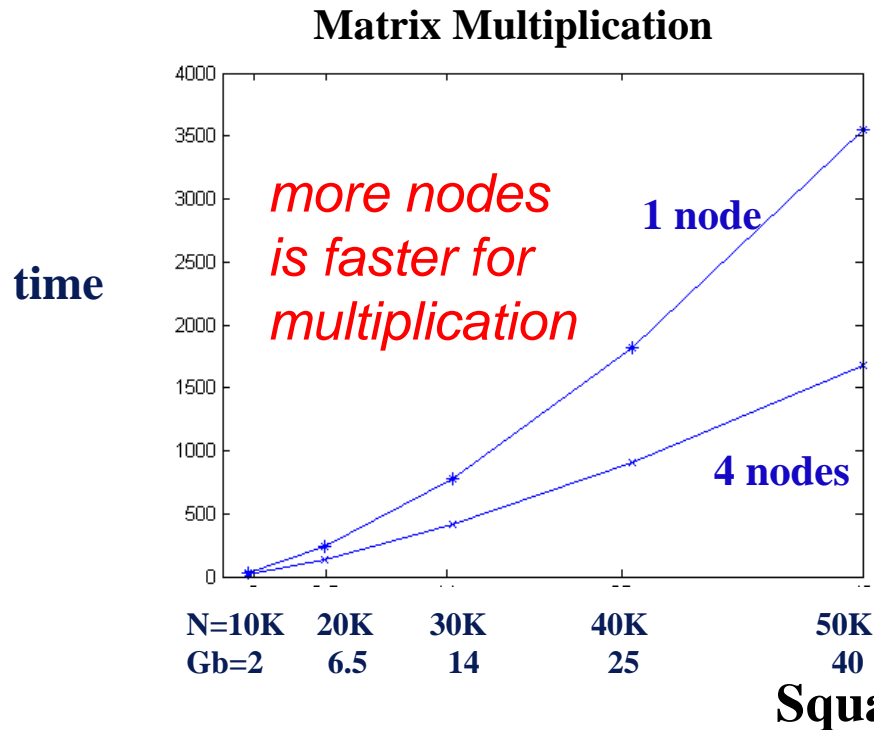
      return( a variable or object )
})
stopCluster(cl)
mpi.exit()
```

allocate cluster as
parallel backend

%dopar% puts loops
across cores and
nodes

Multiple Compute Nodes not always help

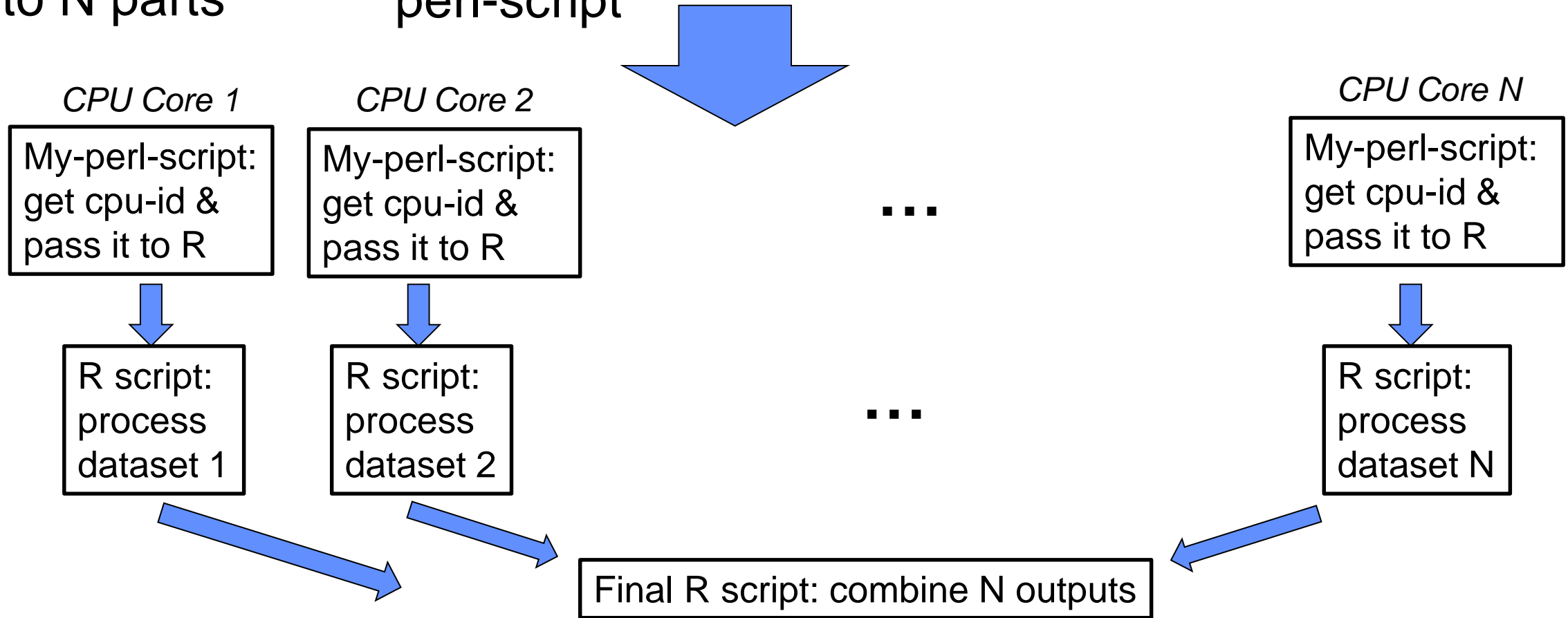
(tested on Gordon)



Another option for (embarrassingly) Parallel R

1. Split up data into N parts

2. Slurm Batch Script: `mpirun -N processors My-perl-script`



Serial Packing with large Random Forest (ensemble of decision trees) job

- Option 1: Run separate trees on separate cores

```
install.packages(doMC)
registerDoMC(cores=15)
getDoParWorkers()
library("randomForest");
results = foreach(i=1:15,.combine=rbind) %dopar%
  {RF1 <-randomForest(formula,data=X,na.action=na.omit,
    importance=TRUE,
    ntree=100000,
    do.trace=1,
    nodesize=1)
  classRF1$importance
  })
```

allocate workers ←

%dopar% puts loops across cores, (loops are independent) %do% runs it serially ←

returned items 'combined' into list ↗ ↘

Sampling on large data could be huge ←

Serial Packing with large Random Forest job

- **Randforest takes bootstrap sampling of data and creates an ensemble of decision trees**

SO Option 2:

**split sampling to make it embarrassingly parallel
run R script on separate CPU cores
average results**

Serial Packing with large Random Forest job

- A GWAS (*genome wide analysis*) study (~20K subjects, ~80K SNPs):
 1. Take 1000 samples on subjects and SNPs
 2. Run 50 trees (with resampling again) on each sample => 50K total trees
 3. Run 1 instance of R on each core: 64 cores < 16hours (on Gordon)
- Runs better than using R multicore with foreach b/c less total memory across nodes

Installing your own R Packages

- In R, `install.packages('package-name')` to get specific functionality you want
 - See <https://cran.r-project.org/>
- **But on Comet**
 - Need to specify URL when prompted and say 'Y' to personal library;
 - If compiling is required you might get an error, call user support

Other R parallelization packages:

- **Rspark** - R interface to Spark
- **pdbR** - higher level over R-MPI, distributed matrix support and other
- **Rgputools** – GPU support
- **R openMP** , better data mgt than dopar, parallel (mclapply)

R, Python, or Matlab?

- **My very rough classification:**

Python: free, more ‘computer scientist like’

Matlab: \$, more ‘engineer like’

R: free, more ‘statistician/experimentalist like’

All are mature environments and capable, preferences depend on specific needs

Spark ML for bigger data:

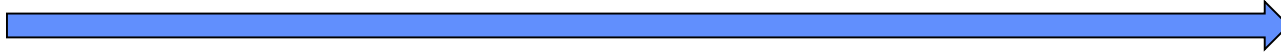
- **Spark MLlib –**
 - Many standard Machine Learning models that are easiest to parallelize
 - Matrix Factorization
 - Naïve Bayes
 - Linear/NonLinear Regression Models with gradient descent optimization
 - Kmeans
 - Some support for large matrix operations

Other Examples on Comet

Image Analysis of Rural Photography

Computer vision and text analytics with 171,021 depression era photographs from Library of Congress.

Feature extraction to database to interface to visual data mining



Title: "Barber and shop"
Location: Omaha, Nebraska
Photographer: John Vachon.
Date Created: 1938.

Image Gray Scale:

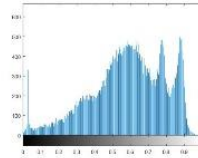


Image Content

OCR + RandomForest :
BARBER SHOP;
ENGLISH

FACE DETECTION: 1 face

Metadata

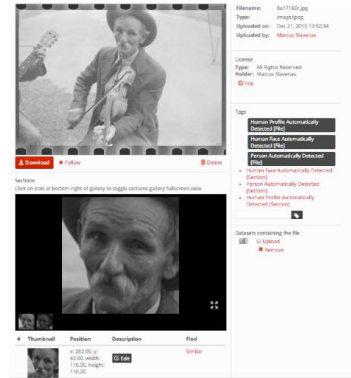
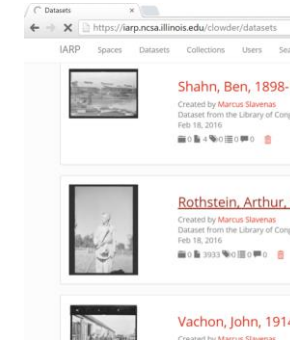
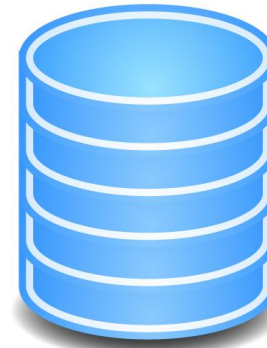
SEMANTICS:

<shop::business;structure;entity>

<barber::worker;person>

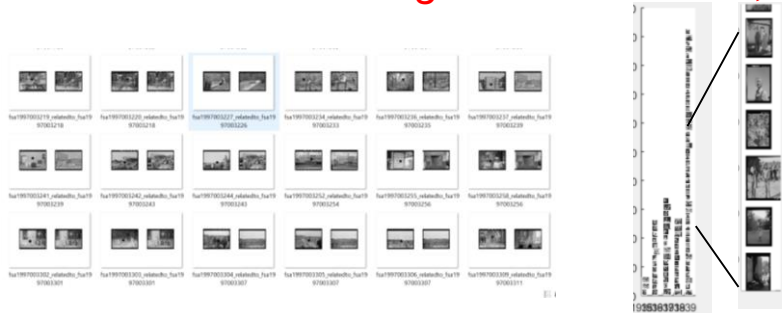
GEO : 41.2°N, 95.9°W

etc...



Data Mining on American life, visual rhetoric, and aesthetics.

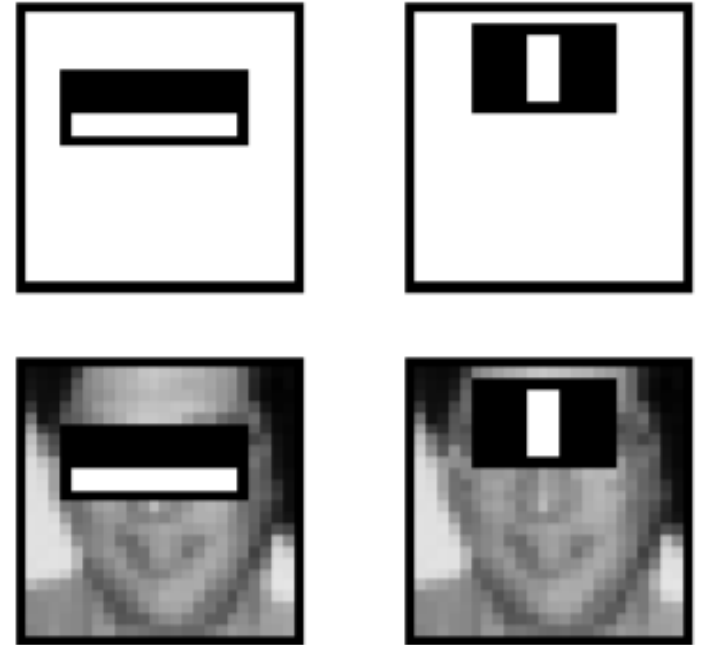
SQL with
visualizations



cotton festival barn
jersey ranch negro pipe
cooperative waiting
men sale farmers pickers
street well one cattle
truck wife factor
this people shop auction
state fair woman spinach
two white man window town
the man unloading
project place national day
old children tenant porch
sugarcane child rice group
saturday laborer construction

Face Detection detail:

- Using Python and OpenCV face package
- **Features are combinations of on/off pixels**
 - a dark patch above a light patch, as with eyebrows and eyes
- **Pre-trained classifier uses a large number (100's) of these features.**





Turned or obscured faces do not work well.



Frontal faces work well. False positives are often small face-like patterns.

Overall performance is about 95% accuracy for these digitized photos.
Performance is often higher for contemporary images.
171K 1kx1k images take about 12 hours on 1 Comet node.

Text Analytics processing of metadata:

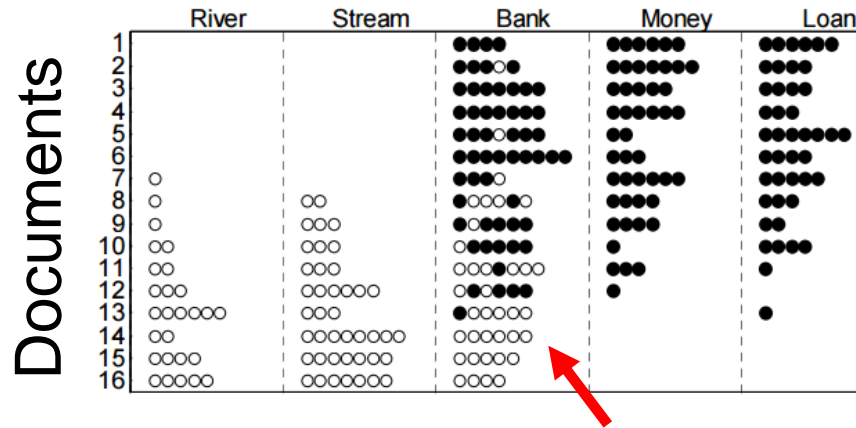
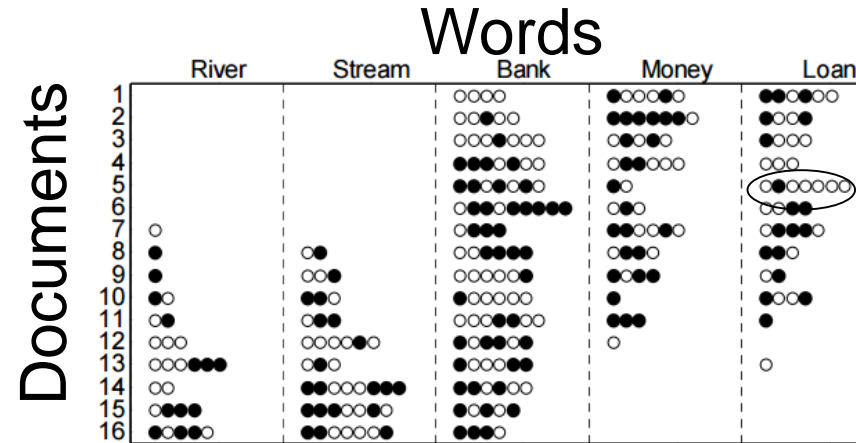
- **Parse, tag speech, search ontology**
 - using Stanford NLP tools, Wordnet ontology, Python NLP toolkit, 101K titles ~ 96 hours on 1 comet node
- **Several words identify ‘person’**
- **SQL: give me all pictures by Lange with possible ‘person’ and num_faces > 0**

Title:

”Destitute pea pickers in California. Mother of seven children.” By D. Lange, 1936, California, [metadata]

Topic Modelling with Latent Dirichlet Allocation

- **Example:**
Get docsXwords matrix
Start with random assignments
- After learning,
topics are well formed



LDA optimization

- **Model: List of K topics and V word probabilities:**

$P(\text{word}|\text{topic})$

$P(\text{topic}|\text{document})$

K, V given by user

- **Optimization:**

- Start with initial guess of word, topic assignments

- Loop until convergence:

Compute the expected frequency of word= w for each topic= t

Adjust parameters to maximize likelihood t given w

- **Example**

Input: articles from JSTOR's Data for Research website (dfr.jstor.org) with certain key words

Output: tree map showing top words



Topic Modelling with LDA on Comet

- **R LDA package:** wraps C programs for Gibbs sampling or EM
Slow for > 5000 journal articles,
Easy to use and interactively explore
- **Mallet: Gibbs sampling, option for multicore, java code**
Slow for > 50000 articles, easy to use from Unix command line
- **Spark LDA: EM**
Fastest, big datasets OK, harder to set up, but EM not as robust as Gibbs sampling
- **Asymptotic Distributed LDA : MPI based**
Faster and Robust, big datasets OK, but difficult to use

THE END