Structure of a Data Analysis

Part 1

Jeffrey Leek, Assistant Professor of Biostatistics
Johns Hopkins Bloomberg School of Public Health

Steps in a data analysis

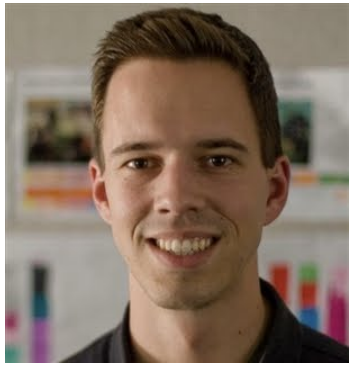
- Define the question
- Define the ideal data set
- Determine what data you can access
- Obtain the data
- Clean the data
- Exploratory data analysis
- Statistical prediction/modeling
- Interpret results
- Challenge results
- Synthesize/write up results
- Create reproducible code

Steps in a data analysis

- Define the question
- Define the ideal data set
- Determine what data you can access
- Obtain the data
- Clean the data
- Exploratory data analysis
- Statistical prediction/modeling
- Interpret results
- Challenge results
- Synthesize/write up results
- Create reproducible code

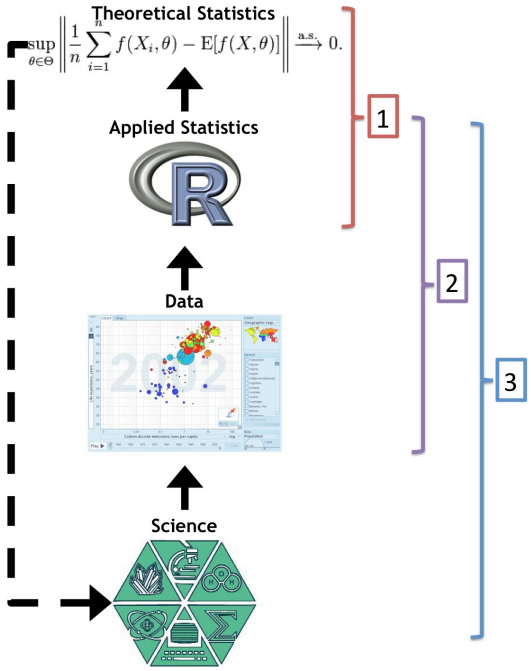
The key challenge in data analysis

> “Ask yourselves, what problem have you solved, ever, that was worth solving, where you knew knew all of the given information in advance? Where you didn’t have a surplus of information and have to filter it out, or you didn’t have insufficient information and have to go find some?”



Dan Myer, Mathematics Educator

Defining a question



1. Statistical methods development
2. Danger zone!!!
3. Proper data analysis

5/16

An example

Start with a general question

Can I automatically detect emails that are SPAM that are not?

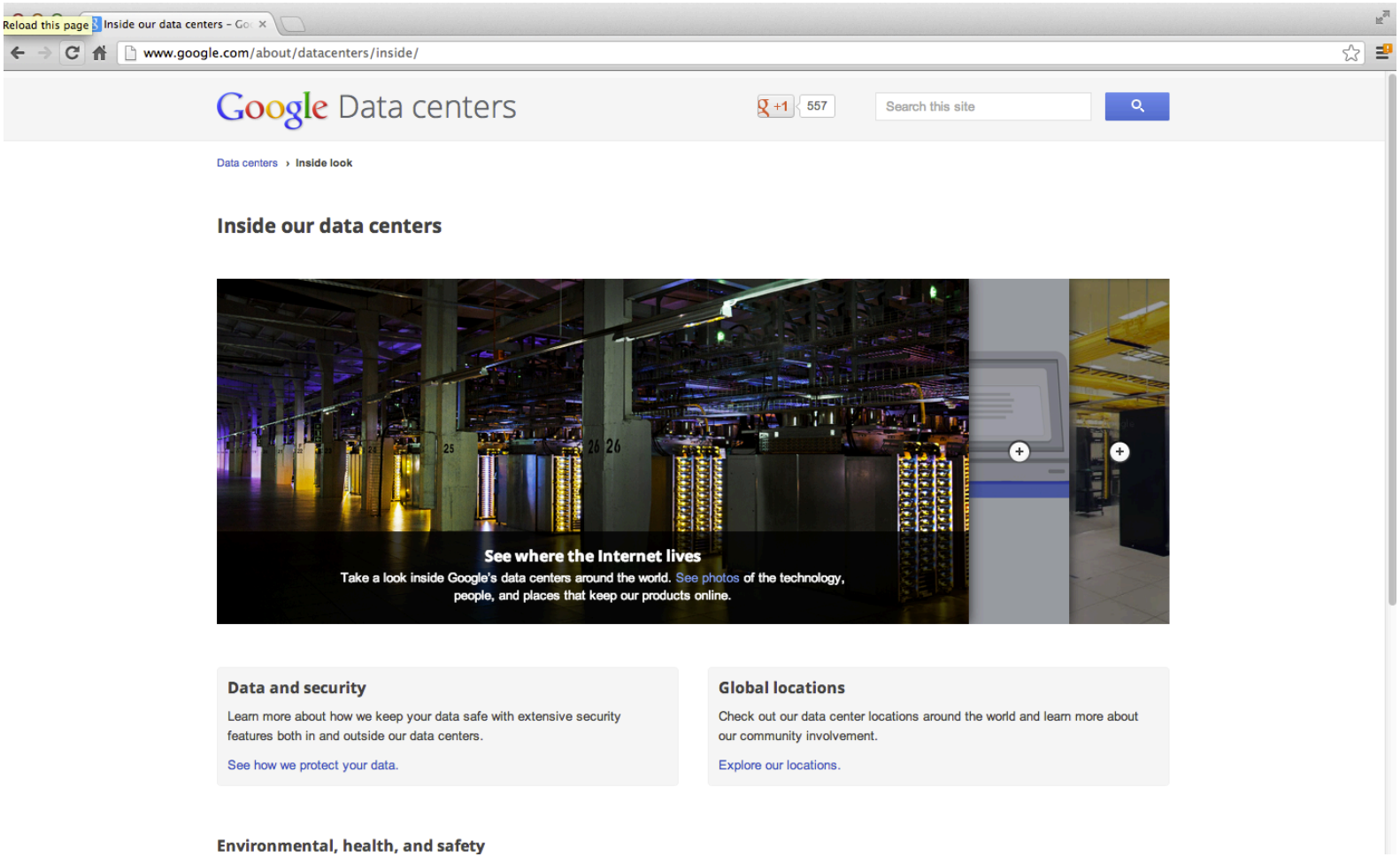
Make it concrete

Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?

Define the ideal data set

- The data set may depend on your goal
  - Descriptive - a whole population
  - Exploratory - a random sample with many variables measured
  - Inferential - the right population, randomly sampled
  - Predictive - a training and test data set from the same population
  - Causal - data from a randomized study
  - Mechanistic - data about all components of the system

Our example



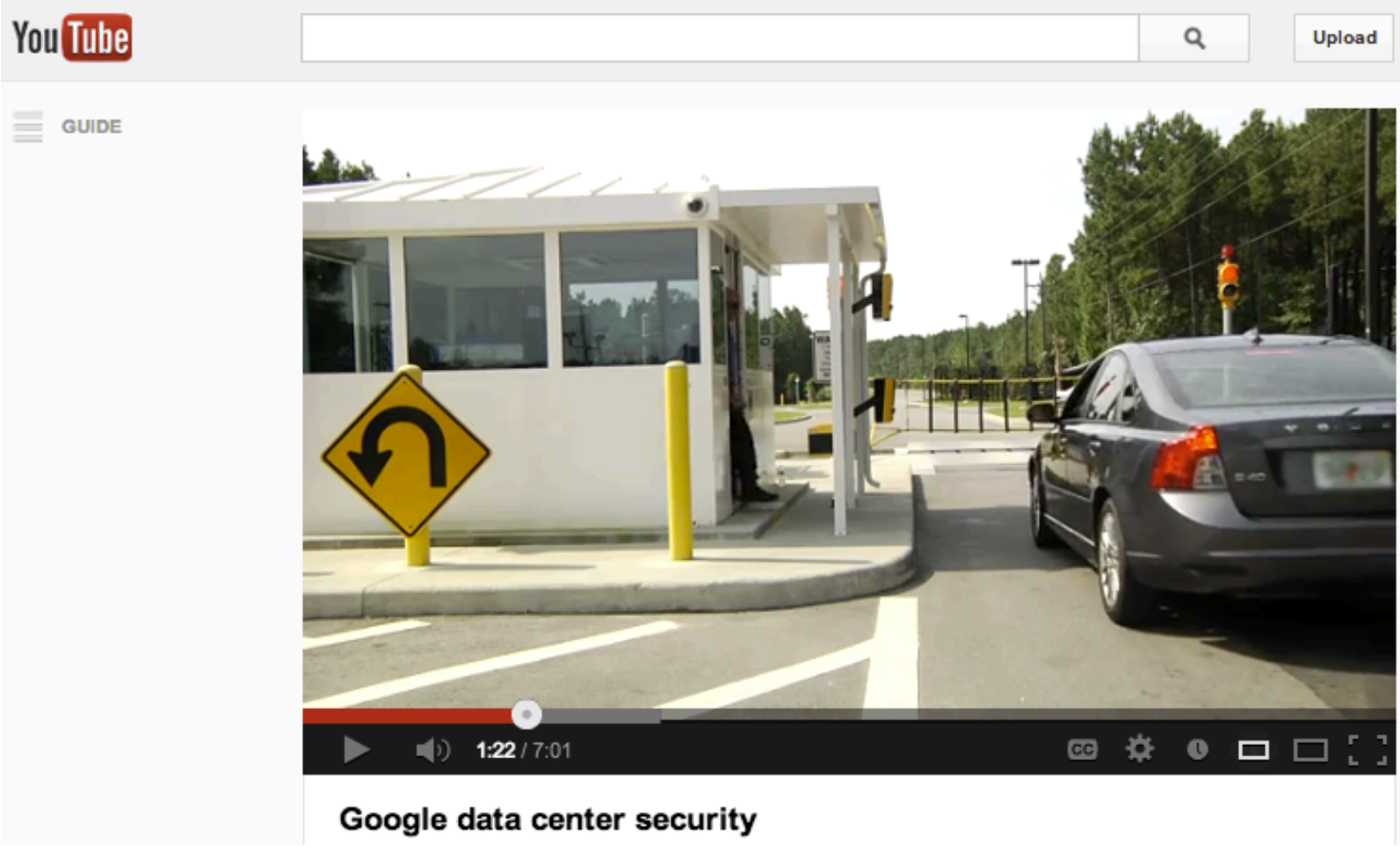
http://www.google.com/about/datacenters/inside/

8/16

Determine what data you can access

- Sometimes you can find data free on the web
- Other times you may need to buy the data
- Be sure to respect the terms of use
- If the data don't exist, you may need to generate it yourself

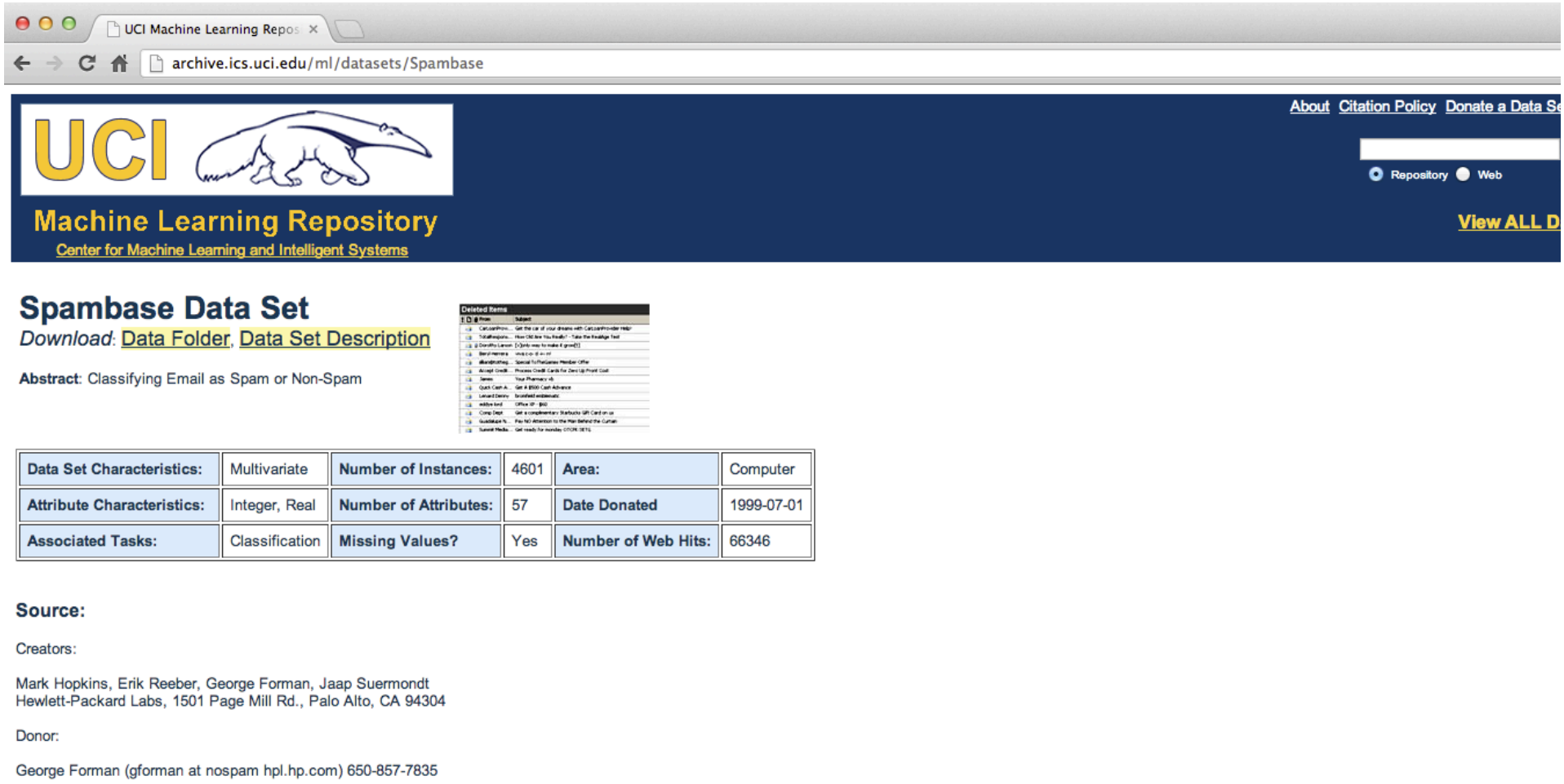
Back to our example



Google data center security

10/16

A possible solution



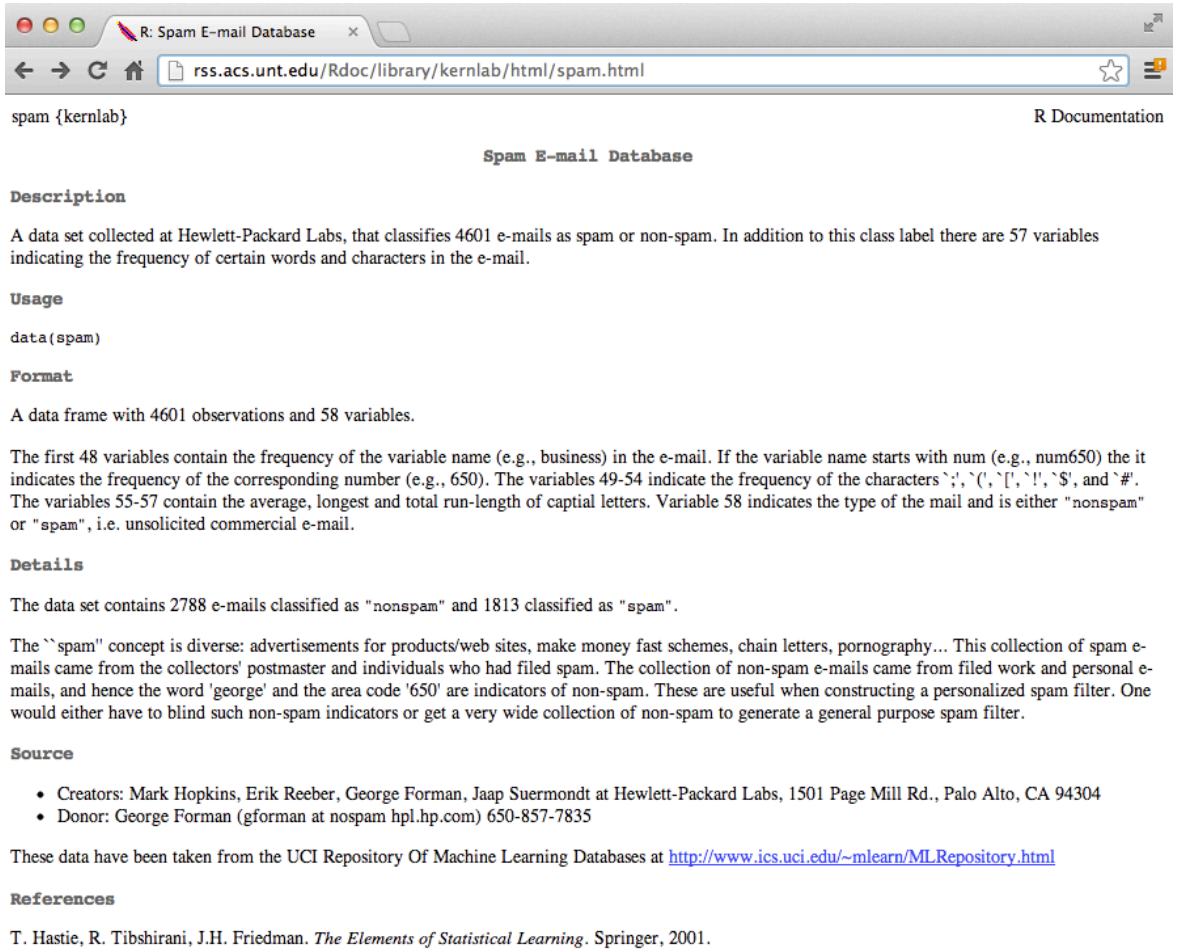
http://archive.ics.uci.edu/ml/datasets/Spambase

11/16

Obtain the data

- Try to obtain the raw data
- Be sure to reference the source
- Polite emails go a long way
- If you will load the data from an internet source, record the url and time accessed

Our data set



R: Spam E-mail Database

rss.acs.unt.edu/Rdoc/library/kernlab/html/spam.html

spam {kernlab} R Documentation

Spam E-mail Database

Description

A data set collected at Hewlett-Packard Labs, that classifies 4601 e-mails as spam or non-spam. In addition to this class label there are 57 variables indicating the frequency of certain words and characters in the e-mail.

Usage

```
data(spam)
```

Format

A data frame with 4601 observations and 58 variables.

The first 48 variables contain the frequency of the variable name (e.g., business) in the e-mail. If the variable name starts with num (e.g., num650) the it indicates the frequency of the corresponding number (e.g., 650). The variables 49-54 indicate the frequency of the characters `;', `(', `[', `!', `$', and `#'. The variables 55-57 contain the average, longest and total run-length of captial letters. Variable 58 indicates the type of the mail and is either `"nonspam"` or `"spam"`, i.e. unsolicited commercial e-mail.

Details

The data set contains 2788 e-mails classified as `"nonspam"` and 1813 classified as `"spam"`.

The ``spam'' concept is diverse: advertisements for products/web sites, make money fast schemes, chain letters, pornography... This collection of spam e-mails came from the collectors' postmaster and individuals who had filed spam. The collection of non-spam e-mails came from filed work and personal e-mails, and hence the word 'george' and the area code '650' are indicators of non-spam. These are useful when constructing a personalized spam filter. One would either have to blind such non-spam indicators or get a very wide collection of non-spam to generate a general purpose spam filter.

Source

- Creators: Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt at Hewlett-Packard Labs, 1501 Page Mill Rd., Palo Alto, CA 94304
- Donor: George Forman (gforman at nospam hpl.hp.com) 650-857-7835

These data have been taken from the UCI Repository Of Machine Learning Databases at http://www.ics.uci.edu/~mlearn/MLRepository.html

References

T. Hastie, R. Tibshirani, J.H. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

http://rss.acs.unt.edu/Rdoc/library/kernlab/html/spam.html

Clean the data

- Raw data often needs to be processed
- If it is pre-processed, make sure you understand how
- Understand the source of the data (census, sample, convenience sample, etc.)
- May need reformating, subsampling - record these steps
- **Determine if the data are good enough** - if not, quit or change data

Our cleaned data set

```
# If it isn't installed, install the kernlab package
library(kernlab)
data(spam)
dim(spam)
```

```
[1] 4601   58
```

http://rss.acs.unt.edu/Rdoc/library/kernlab/html/spam.html

Subsampling our data set

We need to generate a test and training set (prediction)

```
set.seed(3435)
trainIndicator = rbinom(4601,size=1,prob=0.5)
table(trainIndicator)
```

```
trainIndicator
   0    1
2314 2287
```

```
trainSpam = spam[trainIndicator==1,]
testSpam = spam[trainIndicator==0,]
dim(trainSpam)
```

```
[1] 2287   58
```

16/16

Structure of a Data Analysis

Part 2

Jeffrey Leek, Assistant Professor of Biostatistics
Johns Hopkins Bloomberg School of Public Health

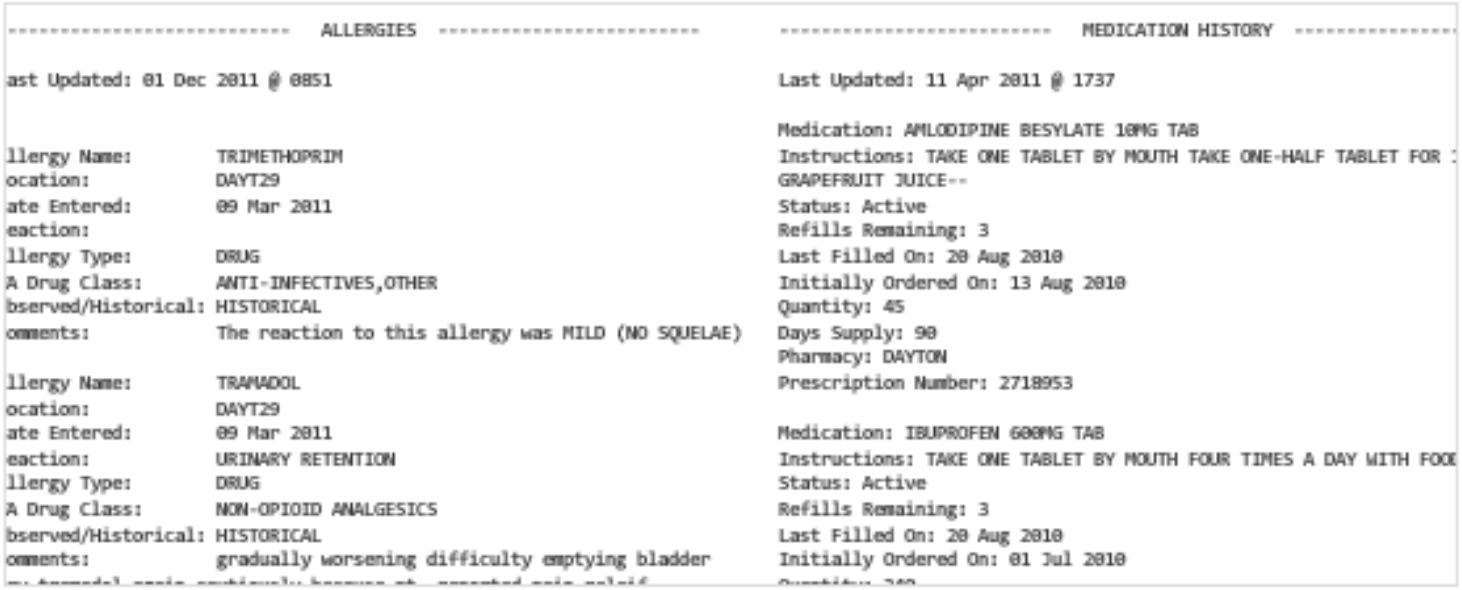
Organizing a data analysis

Jeffrey Leek, Assistant Professor of Biostatistics
Johns Hopkins Bloomberg School of Public Health

Data analysis files

- Data
  - Raw data
  - Processed data
- Figures
  - Exploratory figures
  - Final figures
- R code
  - Raw scripts
  - Final scripts
  - R Markdown files (optional)
- Text
  - Readme files
  - Text of analysis

Raw Data



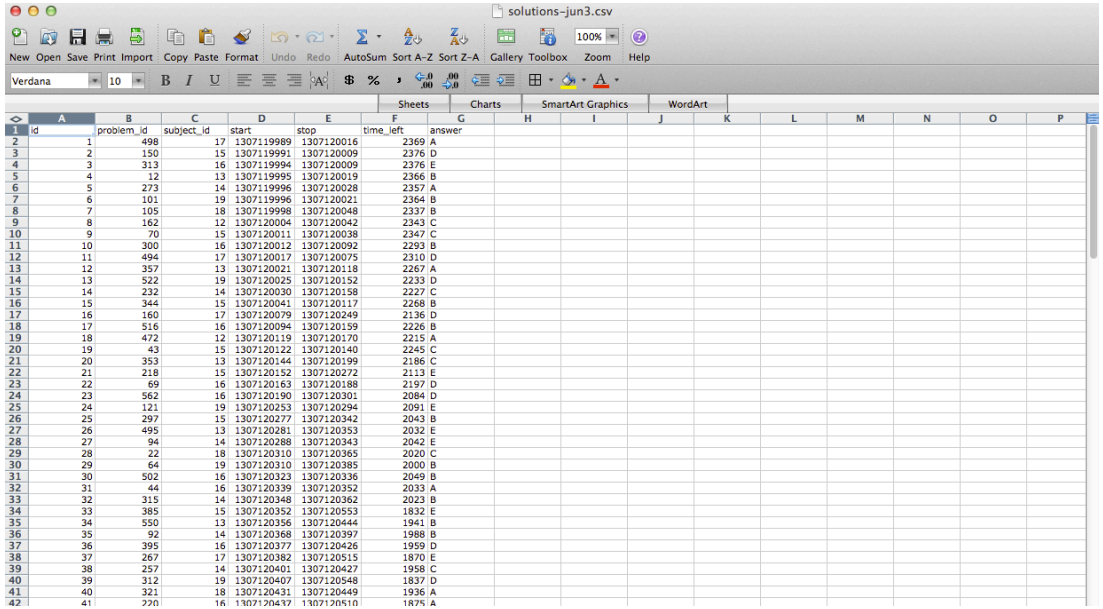
```
--------------------------  ALLERGIES  -------------------------      --------------------------  MEDICATION HISTORY  ----------------

ast Updated: 01 Dec 2011 @ 0851                                      Last Updated: 11 Apr 2011 @ 1737

                                                                     Medication: AMLODIPINE BESYLATE 10MG TAB
llergy Name:        TRIMETHOPRIM                                     Instructions: TAKE ONE TABLET BY MOUTH TAKE ONE-HALF TABLET FOR
ocation:            DAYT29                                           GRAPEFRUIT JUICE--
ate Entered:        09 Mar 2011                                      Status: Active
eaction:                                                             Refills Remaining: 3
llergy Type:        DRUG                                             Last Filled On: 20 Aug 2010
A Drug Class:       ANTI-INFECTIVES,OTHER                            Initially Ordered On: 13 Aug 2010
bserved/Historical: HISTORICAL                                       Quantity: 45
omments:            The reaction to this allergy was MILD (NO SQUELAE)  Days Supply: 90
                                                                     Pharmacy: DAYTON
llergy Name:        TRAMADOL                                         Prescription Number: 2718953
ocation:            DAYT29
ate Entered:        09 Mar 2011                                      Medication: IBUPROFEN 600MG TAB
eaction:            URINARY RETENTION                                Instructions: TAKE ONE TABLET BY MOUTH FOUR TIMES A DAY WITH FOOD
llergy Type:        DRUG                                             Status: Active
A Drug Class:       NON-OPIOID ANALGESICS                            Refills Remaining: 3
bserved/Historical: HISTORICAL                                       Last Filled On: 20 Aug 2010
omments:            gradually worsening difficulty emptying bladder  Initially Ordered On: 01 Jul 2010
```

- Should be stored in your analysis folder
- If accessed from the web, include url, description, and date accessed in README

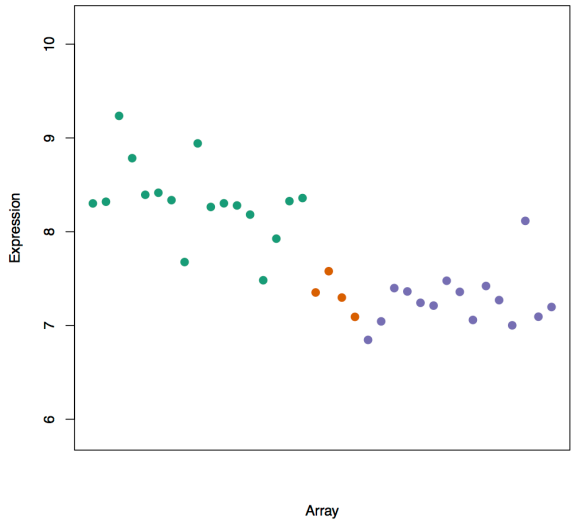
Processed data



	id	problem_id	subject_id	start	stop	time_left	answer
1	1	498	17	1307119989	1307120016	2369	A
2	2	150	15	1307119991	1307120009	2376	D
3	3	313	16	1307119994	1307120009	2376	E
4	4	12	13	1307119995	1307120019	2366	B
5	5	273	14	1307119996	1307120028	2357	A
6	6	101	19	1307119996	1307120021	2364	B
7	7	105	18	1307119998	1307120048	2337	B
8	8	162	12	1307120004	1307120042	2343	C
9	9	70	15	1307120011	1307120038	2347	C
10	10	300	16	1307120012	1307120092	2293	B
11	11	494	17	1307120017	1307120075	2310	D
12	12	357	13	1307120021	1307120118	2267	A
13	13	522	19	1307120025	1307120152	2233	D
14	14	232	14	1307120030	1307120158	2227	C
15	15	344	15	1307120041	1307120117	2268	B
16	16	160	17	1307120079	1307120249	2136	D
17	17	516	16	1307120094	1307120159	2226	B
18	18	472	12	1307120119	1307120170	2215	A
19	19	43	15	1307120122	1307120140	2245	C
20	20	353	13	1307120144	1307120199	2186	C
21	21	218	15	1307120152	1307120272	2113	E
22	22	69	16	1307120163	1307120188	2197	D
23	23	562	16	1307120190	1307120301	2084	D
24	24	121	19	1307120253	1307120294	2091	E
25	25	297	15	1307120277	1307120342	2043	B
26	26	495	13	1307120281	1307120353	2032	E
27	27	94	14	1307120288	1307120343	2042	E
28	28	22	18	1307120310	1307120365	2020	C
29	29	64	19	1307120310	1307120385	2000	B
30	30	502	16	1307120323	1307120336	2049	B
31	31	44	16	1307120339	1307120352	2033	A
32	32	315	14	1307120348	1307120362	2023	B
33	33	385	15	1307120352	1307120553	1832	E
34	34	550	13	1307120356	1307120444	1941	B
35	35	92	14	1307120368	1307120397	1988	B
36	36	395	16	1307120377	1307120426	1959	D
37	37	267	17	1307120382	1307120515	1870	E
38	38	257	14	1307120401	1307120427	1958	C
39	39	312	19	1307120407	1307120548	1837	D
40	40	321	18	1307120431	1307120449	1936	A
41	41	220	16	1307120437	1307120510	1875	A

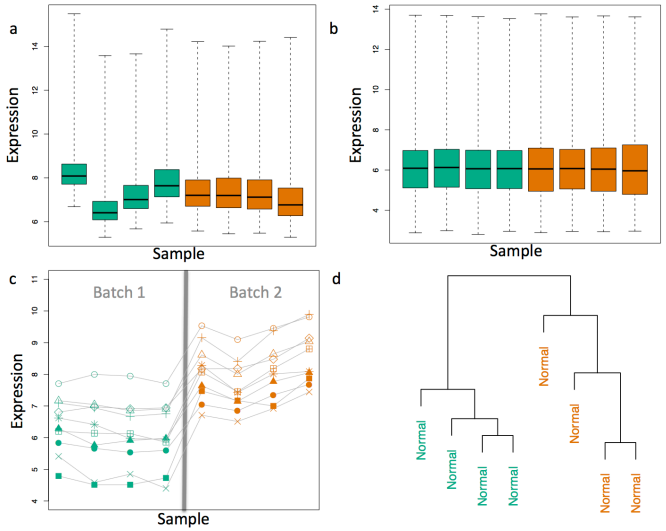
- Processed data should be named so it is easy to see which script generated the data.
- The processing script - processed data mapping should occur in the README
- Processed data should be tidy

Exploratory figures



- Figures made during the course of your analysis, not necessarily part of your final report.
- They do not need to be "pretty"

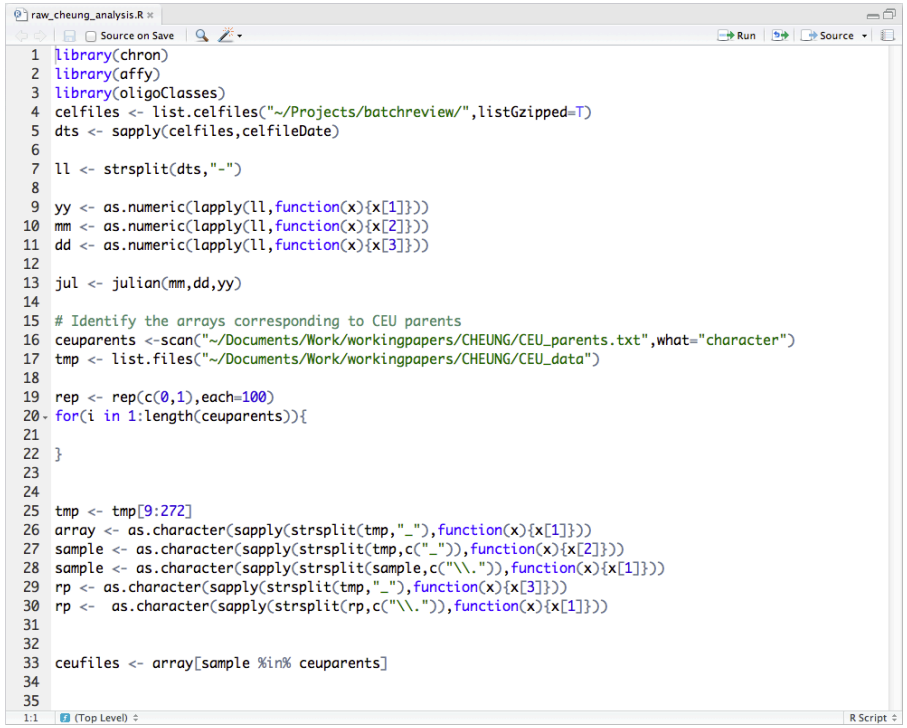
Final Figures



- Usually a small subset of the original figures
- Axes/colors set to make the figure clear
- Possibly multiple panels

6/12

Raw scripts



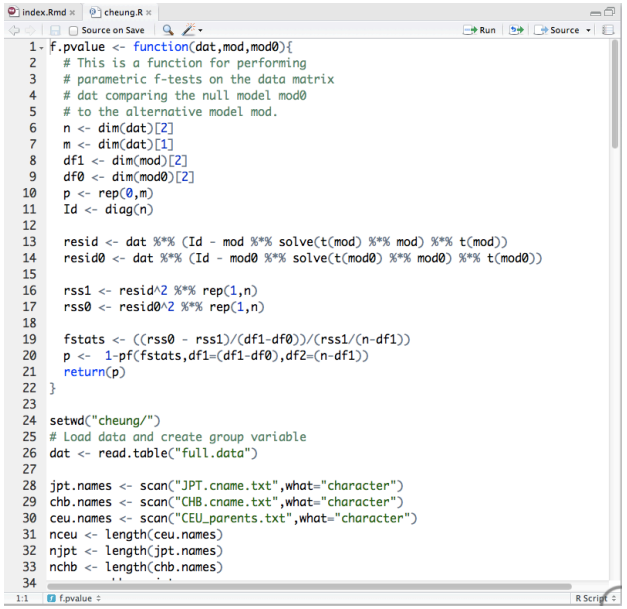
raw_cheung_analysis.R

```
1  library(chron)
2  library(affy)
3  library(oligoClasses)
4  celfiles <- list.celfiles("~/Projects/batchreview/",listGzipped=T)
5  dts <- sapply(celfiles,celfileDate)
6
7  ll <- strsplit(dts,"-")
8
9  yy <- as.numeric(lapply(ll,function(x){x[1]}))
10 mm <- as.numeric(lapply(ll,function(x){x[2]}))
11 dd <- as.numeric(lapply(ll,function(x){x[3]}))
12
13 jul <- julian(mm,dd,yy)
14
15 # Identify the arrays corresponding to CEU parents
16 ceuparents <-scan("~/Documents/Work/workingpapers/CHEUNG/CEU_parents.txt",what="character")
17 tmp <- list.files("~/Documents/Work/workingpapers/CHEUNG/CEU_data")
18
19 rep <- rep(c(0,1),each=100)
20 for(i in 1:length(ceuparents)){
21
22 }
23
24
25 tmp <- tmp[9:272]
26 array <- as.character(sapply(strsplit(tmp,"_"),function(x){x[1]}))
27 sample <- as.character(sapply(strsplit(tmp,c("_")),function(x){x[2]}))
28 sample <- as.character(sapply(strsplit(sample,c("\\.")),function(x){x[1]}))
29 rp <- as.character(sapply(strsplit(tmp,"_"),function(x){x[3]}))
30 rp <-  as.character(sapply(strsplit(rp,c("\\.")),function(x){x[1]}))
31
32
33 ceufiles <- array[sample %in% ceuparents]
34
35
```

1:1 (Top Level) R Script

- May be less commented (but comments help you!)
- May be multiple versions
- May include analyses that are later discarded

Final scripts



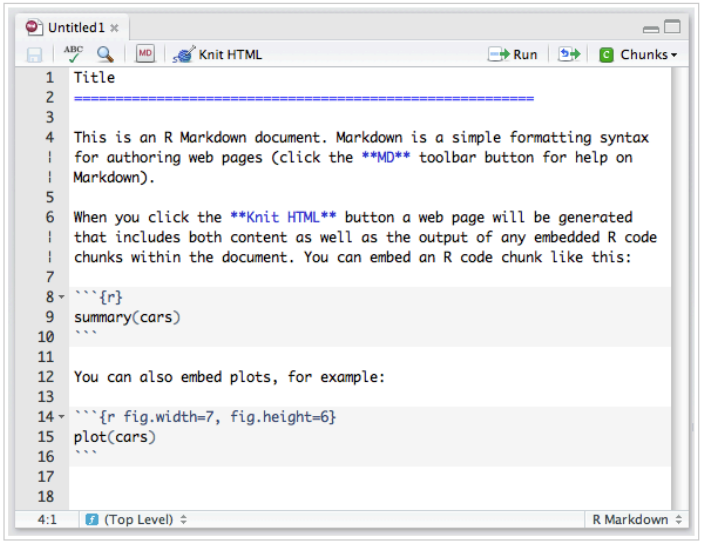
- Clearly commented
  - Small comments liberally - what, when, why, how
  - Bigger commented blocks for whole sections
- Include processing details
- Only analyses that appear in the final write-up

R markdown files

R Markdown Documents

To work with R Markdown (.Rmd) files in RStudio you first need to ensure that the knitr package (version 0.5 or later) in installed.

To create a new R Markdown file, go to **File | New |** and select **R Markdown**. A new file is create with a default template to get you oriented:



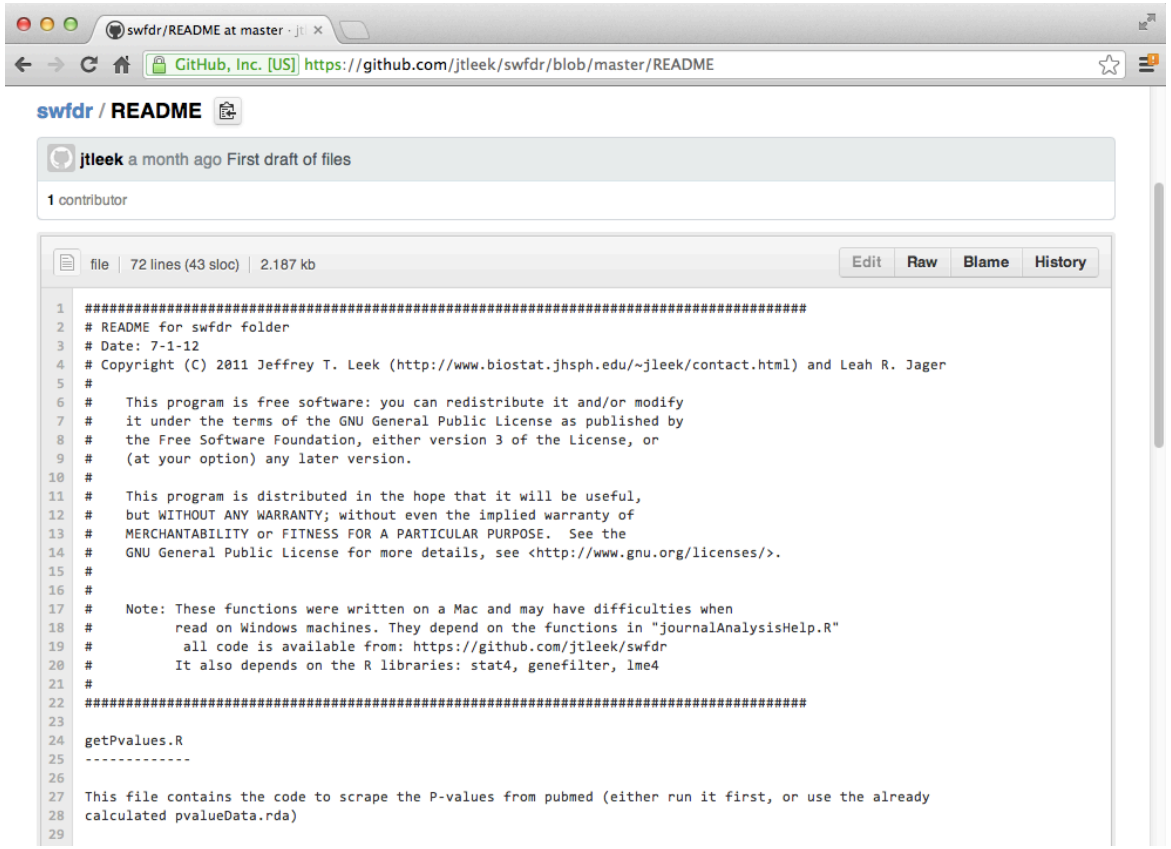
Note that the toolbar provides some useful tools for working with R Markdown:

- **Quick Reference** — Click the **MD** toolbar button to open a quick reference guide for Markdown.
- **Knit HTML** — Click to knit the current document to HTML, see the **Knitting to HTML** section below for more details.
- **Run** — Run the current line or selection of lines in the console. This allows running R code inside a code chunk similar to a normal R source file.
- **Chunks** — The chunks menu provides assistance with inserting, running, and chunk navigation. See the **Chunk Menu and Options** section below for more details.

- R markdown files can be used to generate reproducible reports
- Text and R code are integrated
- Very easy to create in Rstudio

9/12

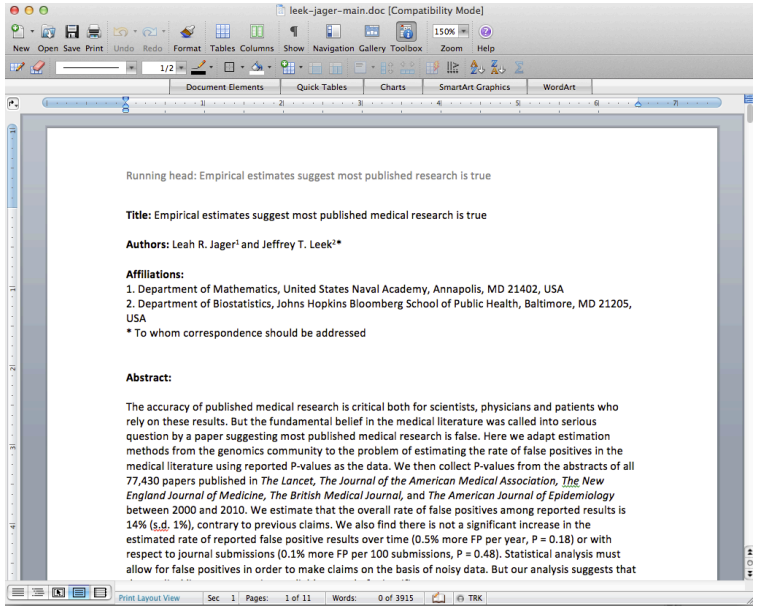
Readme files



- Not necessary if you use R markdown
- Should contain step-by-step instructions for analysis
- Here is an example https://github.com/jtleek/swfdr/blob/master/README

10/12

Text of the document



- It should include a title, introduction (motivation), methods (statistics you used), results (including measures of uncertainty), and conclusions (including potential problems)
- It should tell a story
- *It should not include every analysis you performed*
- References should be included for statistical methods

11/12

Further resources

- Information about a non-reproducible study that led to cancer patients being mistreated: The Duke Saga Starter Set
- Reproducible research and Biostatistics
- Managing a statistical analysis project guidelines and best practices
- Project template - a pre-organized set of files for data analysis

12/12

Steps in a data analysis

- Define the question
- Define the ideal data set
- Determine what data you can access
- Obtain the data
- Clean the data
- Exploratory data analysis
- Statistical prediction/modeling
- Interpret results
- Challenge results
- Synthesize/write up results
- Create reproducible code

Steps in a data analysis

- Define the question
- Define the ideal data set
- Determine what data you can access
- Obtain the data
- Clean the data
- Exploratory data analysis
- Statistical prediction/modeling
- Interpret results
- Challenge results
- Synthesize/write up results
- Create reproducible code

An example

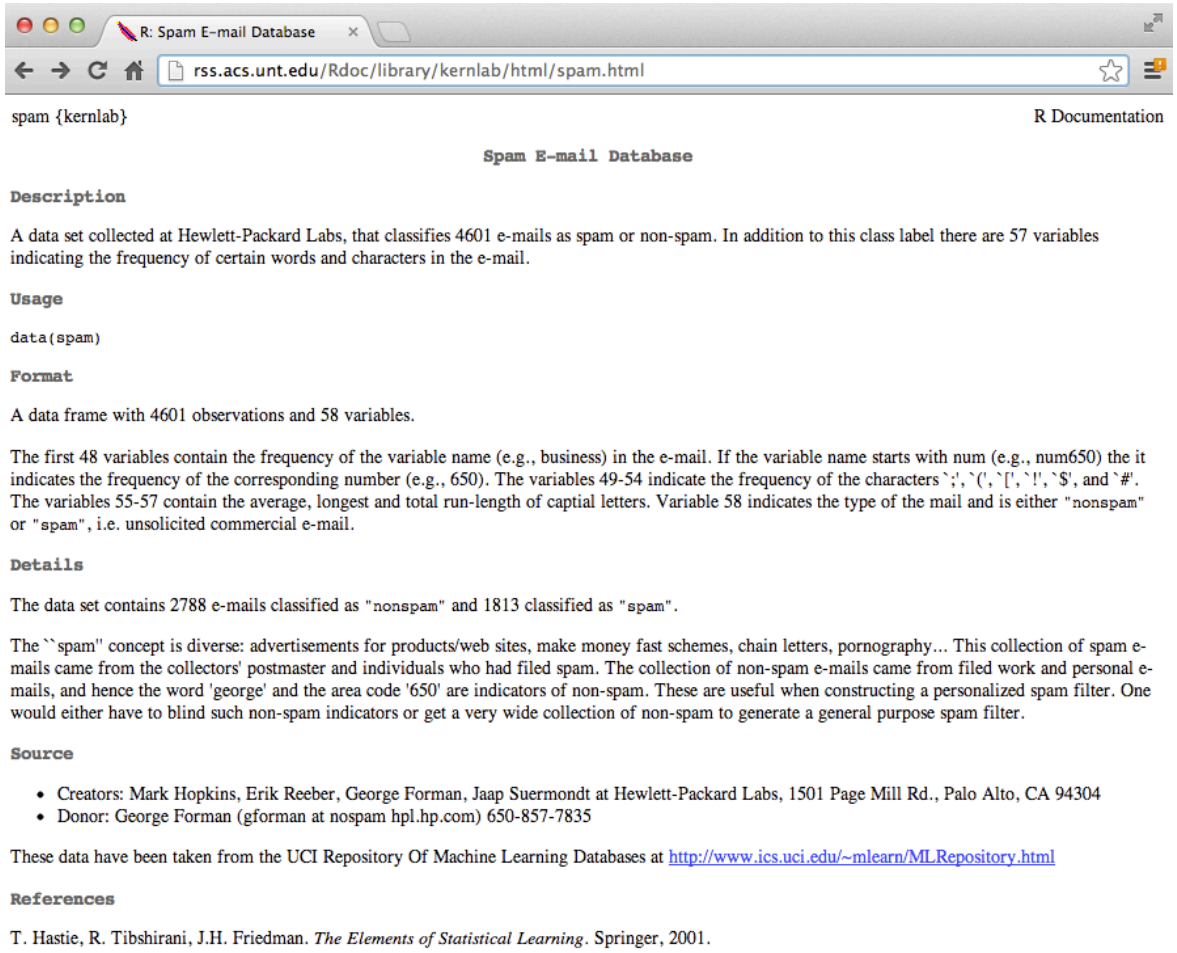
Start with a general question

Can I automatically detect emails that are SPAM that are not?

Make it concrete

Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?

Our data set



R: Spam E-mail Database

rss.acs.unt.edu/Rdoc/library/kernlab/html/spam.html

spam {kernlab} R Documentation

Spam E-mail Database

Description

A data set collected at Hewlett-Packard Labs, that classifies 4601 e-mails as spam or non-spam. In addition to this class label there are 57 variables indicating the frequency of certain words and characters in the e-mail.

Usage

```
data(spam)
```

Format

A data frame with 4601 observations and 58 variables.

The first 48 variables contain the frequency of the variable name (e.g., business) in the e-mail. If the variable name starts with num (e.g., num650) the it indicates the frequency of the corresponding number (e.g., 650). The variables 49-54 indicate the frequency of the characters `;', `(', `[', `!', `$', and `#'. The variables 55-57 contain the average, longest and total run-length of captial letters. Variable 58 indicates the type of the mail and is either `"nonspam"` or `"spam"`, i.e. unsolicited commercial e-mail.

Details

The data set contains 2788 e-mails classified as `"nonspam"` and 1813 classified as `"spam"`.

The ``spam'' concept is diverse: advertisements for products/web sites, make money fast schemes, chain letters, pornography... This collection of spam e-mails came from the collectors' postmaster and individuals who had filed spam. The collection of non-spam e-mails came from filed work and personal e-mails, and hence the word 'george' and the area code '650' are indicators of non-spam. These are useful when constructing a personalized spam filter. One would either have to blind such non-spam indicators or get a very wide collection of non-spam to generate a general purpose spam filter.

Source

- Creators: Mark Hopkins, Erik Reeber, George Forman, Jaap Suermondt at Hewlett-Packard Labs, 1501 Page Mill Rd., Palo Alto, CA 94304
- Donor: George Forman (gforman at nospam hpl.hp.com) 650-857-7835

These data have been taken from the UCI Repository Of Machine Learning Databases at http://www.ics.uci.edu/~mlearn/MLRepository.html

References

T. Hastie, R. Tibshirani, J.H. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

http://rss.acs.unt.edu/Rdoc/library/kernlab/html/spam.html

5/24

Subsampling our data set

We need to generate a test and training set (prediction)

```
# If it isn't installed, install the kernlab package
library(kernlab)
data(spam)
# Perform the subsampling
set.seed(3435)
trainIndicator = rbinom(4601, size = 1, prob = 0.5)
table(trainIndicator)
```

```
## trainIndicator
##    0    1
## 2314 2287
```

```
trainSpam = spam[trainIndicator == 1, ]
testSpam = spam[trainIndicator == 0, ]
```

Exploratory data analysis

- Look at summaries of the data
- Check for missing data
- Create exploratory plots
- Perform exploratory analyses (e.g. clustering)

Names

```
names(trainSpam)
```

```
##  [1] "make"              "address"           "all"
##  [4] "num3d"             "our"               "over"
##  [7] "remove"            "internet"          "order"
## [10] "mail"              "receive"           "will"
## [13] "people"            "report"            "addresses"
## [16] "free"              "business"          "email"
## [19] "you"               "credit"            "your"
## [22] "font"              "num000"            "money"
## [25] "hp"                "hpl"               "george"
## [28] "num650"            "lab"               "labs"
## [31] "telnet"            "num857"            "data"
## [34] "num415"            "num85"             "technology"
## [37] "num1999"           "parts"             "pm"
## [40] "direct"            "cs"                "meeting"
## [43] "original"          "project"           "re"
## [46] "edu"               "table"             "conference"
## [49] "charSemicolon"     "charRoundbracket"  "charSquarebracket"
## [52] "charExclamation"   "charDollar"        "charHash"
```

Head

```
head(trainSpam)
```

```
##    make address  all num3d  our over remove internet order mail receive
## 1  0.00    0.64 0.64     0 0.32 0.00   0.00        0  0.00 0.00    0.00
## 7  0.00    0.00 0.00     0 1.92 0.00   0.00        0  0.00 0.64    0.96
## 9  0.15    0.00 0.46     0 0.61 0.00   0.30        0  0.92 0.76    0.76
## 12 0.00    0.00 0.25     0 0.38 0.25   0.25        0  0.00 0.00    0.12
## 14 0.00    0.00 0.00     0 0.90 0.00   0.90        0  0.00 0.90    0.90
## 16 0.00    0.42 0.42     0 1.27 0.00   0.42        0  0.00 1.27    0.00
##    will people report addresses free business email  you credit your font
## 1  0.64   0.00      0         0 0.32        0  1.29 1.93   0.00 0.96    0
## 7  1.28   0.00      0         0 0.96        0  0.32 3.85   0.00 0.64    0
## 9  0.92   0.00      0         0 0.00        0  0.15 1.23   3.53 2.00    0
## 12 0.12   0.12      0         0 0.00        0  0.00 1.16   0.00 0.77    0
## 14 0.00   0.90      0         0 0.00        0  0.00 2.72   0.00 0.90    0
## 16 0.00   0.00      0         0 1.27        0  0.00 1.70   0.42 1.27    0
##    num000 money hp hpl george num650 lab labs telnet num857 data num415
## 1       0  0.00  0   0      0      0   0    0      0      0 0.00      0
## 7       0  0.00  0   0      0      0   0    0      0      0 0.00      0
## 9       0  0.15  0   0      0      0   0    0      0      0 0.15      0
```

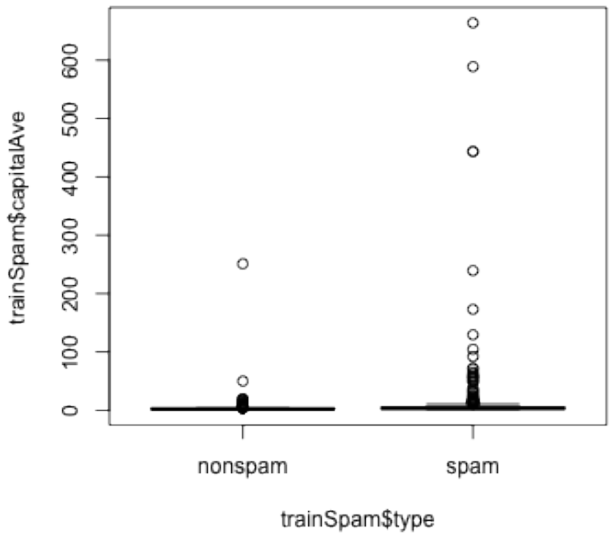
Summaries

```
table(trainSpam$type)
```

```
##
## nonspam    spam
##    1381     906
```

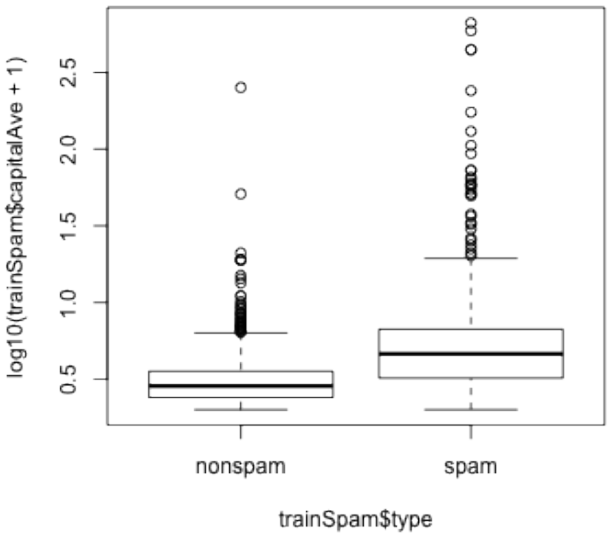
Plots

```
plot(trainSpam$capitalAve ~ trainSpam$type)
```



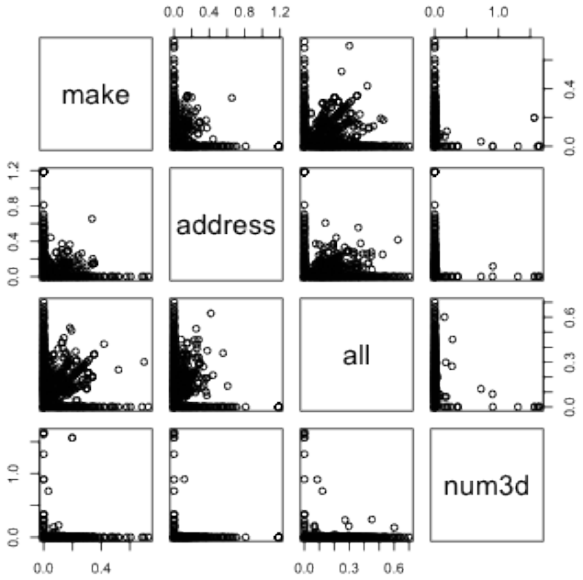
Plots

```
plot(log10(trainSpam$capitalAve + 1) ~ trainSpam$type)
```



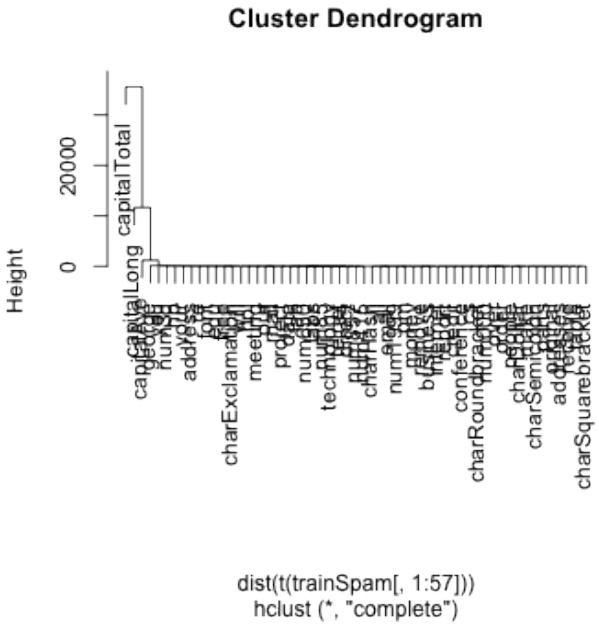
Relationships between predictors

```
plot(log10(trainSpam[, 1:4] + 1))
```



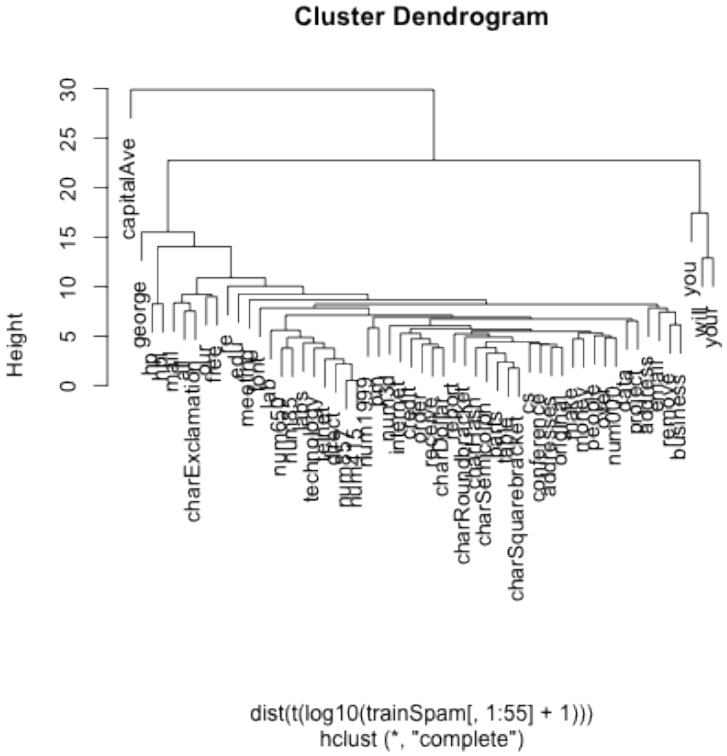
Clustering

```
hCluster = hclust(dist(t(trainSpam[, 1:57])))
plot(hCluster)
```



New clustering

```
hClusterUpdated = hclust(dist(t(log10(trainSpam[, 1:55] + 1))))
plot(hClusterUpdated)
```



Statistical prediction/modeling

- Should be informed by the results of your exploratory analysis
- Exact methods depend on the question of interest
- Transformations/processing should be accounted for when necessary
- Measures of uncertainty should be reported

Statistical prediction/modeling

```
trainSpam$numType = as.numeric(trainSpam$type) - 1
costFunction = function(x, y) {
    sum(x != (y > 0.5))
}
cvError = rep(NA, 55)
library(boot)
for (i in 1:55) {
    lmFormula = as.formula(paste("numType~", names(trainSpam)[i], sep = ""))
    glmFit = glm(lmFormula, family = "binomial", data = trainSpam)
    cvError[i] = cv.glm(trainSpam, glmFit, costFunction, 2)$delta[2]
}
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

Get a measure of uncertainty

```
predictionModel = glm(numType ~ charDollar, family = "binomial", data = trainSpam)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
predictionTest = predict(predictionModel, testSpam)
predictedSpam = rep("nonspam", dim(testSpam)[1])
predictedSpam[predictionModel$fitted > 0.5] = "spam"
table(predictedSpam, testSpam$type)
```

```
##
## predictedSpam nonspam spam
##       nonspam    1346  458
##       spam         61  449
```

```
(61 + 458)/(1346 + 458 + 61 + 449)
```

Interpret results

- Use the appropriate language
  - describes
  - correlates with/associated with
  - leads to/causes
  - predicts
- Give an explanation
- Interpret coefficients
- Interpret measures of uncertainty

Our example

- The fraction of charcters that are dollar signs can be used to predict if an email is Spam
- Anything with more than 6.6% dollar signs is classified as Spam
- More dollar signs always means more Spam under our prediction
- Our test set error rate was 22.4%

Challenge results

- Challenge all steps:
  - Question
  - Data source
  - Processing
  - Analysis
  - Conclusions
- Challenge measures of uncertainty
- Challenge choices of terms to include in models
- Think of potential alternative analyses

Synthesize/write-up results

- Lead with the question
- Summarize the analyses into the story
- Don't include every analysis, include it
  - If it is needed for the story
  - If it is needed to address a challenge
- Order analyses according to the story, rather than chronologically
- Include "pretty" figures that contribute to the story

In our example

- Lead with the question
  - Can I use quantitative characteristics of the emails to classify them as SPAM/HAM?
- Describe the approach
  - Collected data from UCI -> created training/test sets
  - Explored relationships
  - Choose logistic model on training set by cross validation
  - Applied to test, 78% test set accuracy
- Interpret results
  - Number of dollar signs seems reasonable, e.g. "Make money with Viagra $ $ $ $!"
- Challenge results
  - 78% isn't that great
  - I could use more variables
  - Why logistic regression?

Create reproducible code