



Structure of a Data Analysis

Part 2

Roger D. Peng, Associate 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

An example

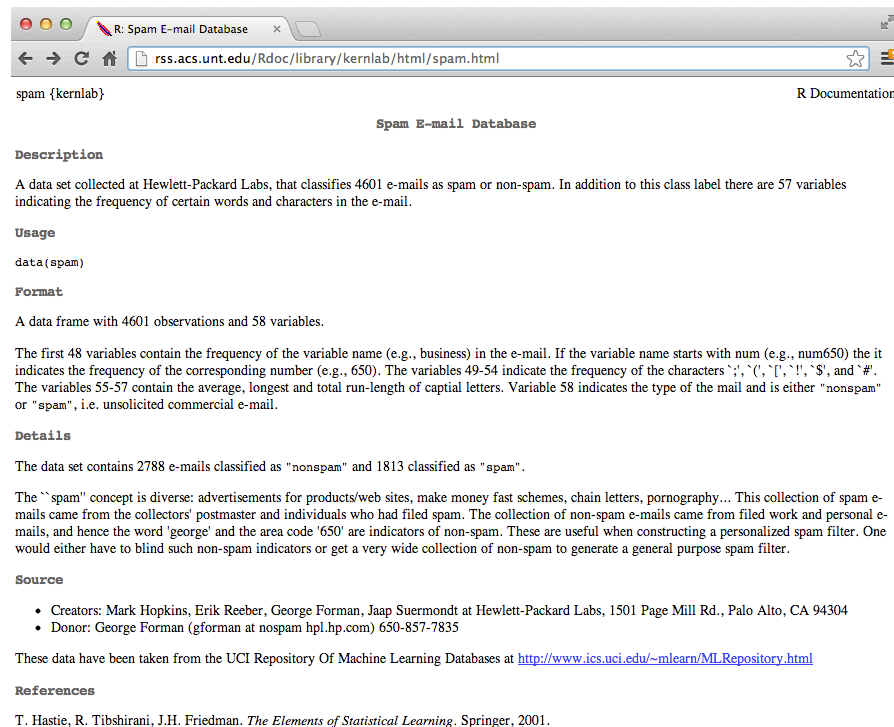
Start with a general question

Can I automatically detect emails that are SPAM or not?

Make it concrete

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

Our data set



The screenshot shows a web browser window with the title "R: Spam E-mail Database". The address bar displays the URL "rss.acs.unt.edu/Rdoc/library/kernlab/html/spam.html". The page content includes the following sections:

- spam {kernlab}** (top left)
- R Documentation** (top right)
- Spam E-mail Database** (center header)
- 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 capital letters. Variable 58 indicates the type of the mail and is either "nospam" or "spam", i.e. unsolicited commercial e-mail.
- Details**

The data set contains 2788 e-mails classified as "nospam" 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/~mllearn/MLRepository.html>
- References**

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

<http://search.r-project.org/library/kernlab/html/spam.html>

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"
## [55] "capitalAve"    "capitalLong"    "capitalTotal"
## [58] "type"
```


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
## 12     0  0.00 0  0      0      0  0  0      0      0 0.00      0
## 14     0  0.00 0  0      0      0  0  0      0      0 0.00      0
```

Summaries

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

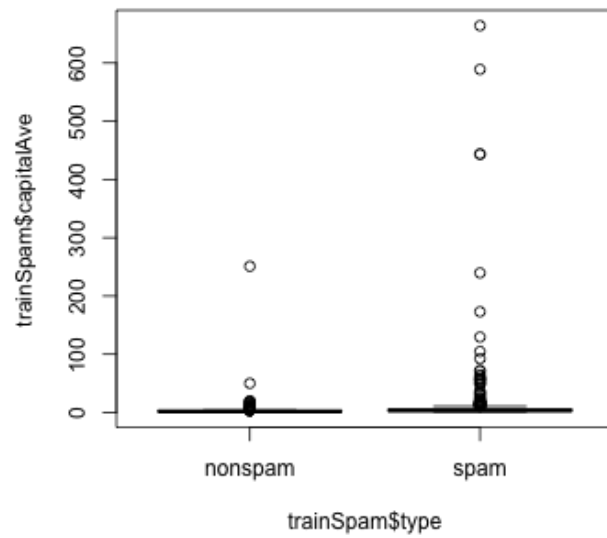
```
##
```

```
## nonspam    spam
```

```
##      1381     906
```

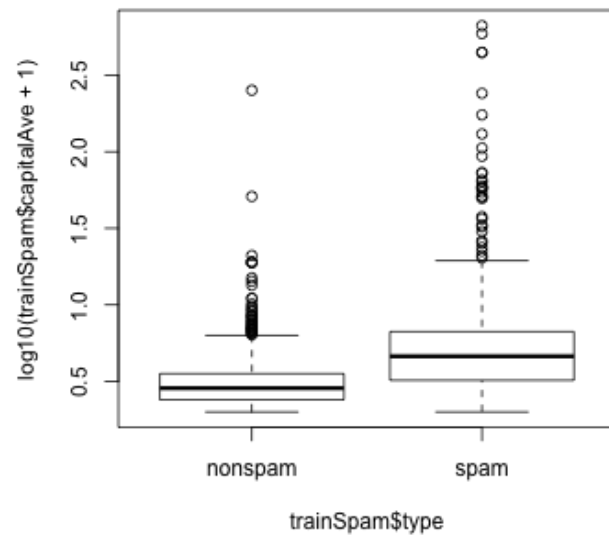
Plots

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



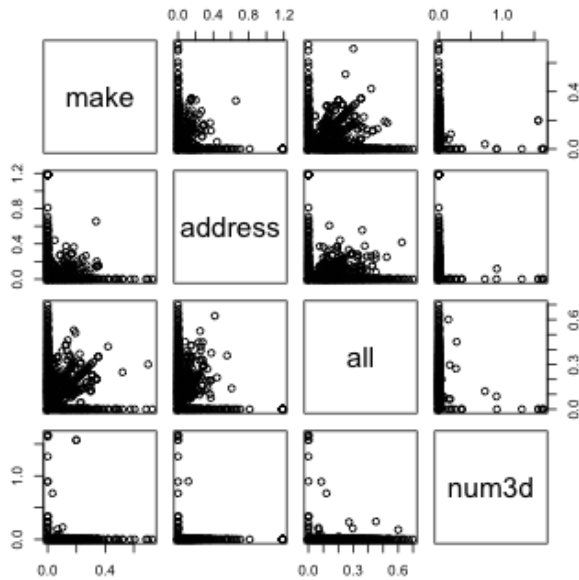
Plots

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



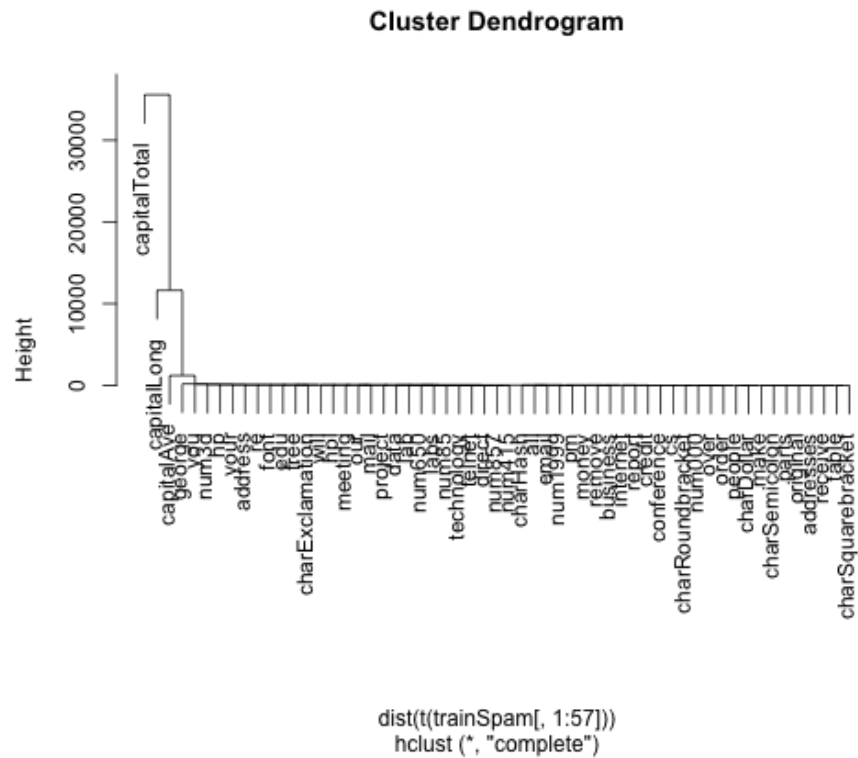
Relationships between predictors

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



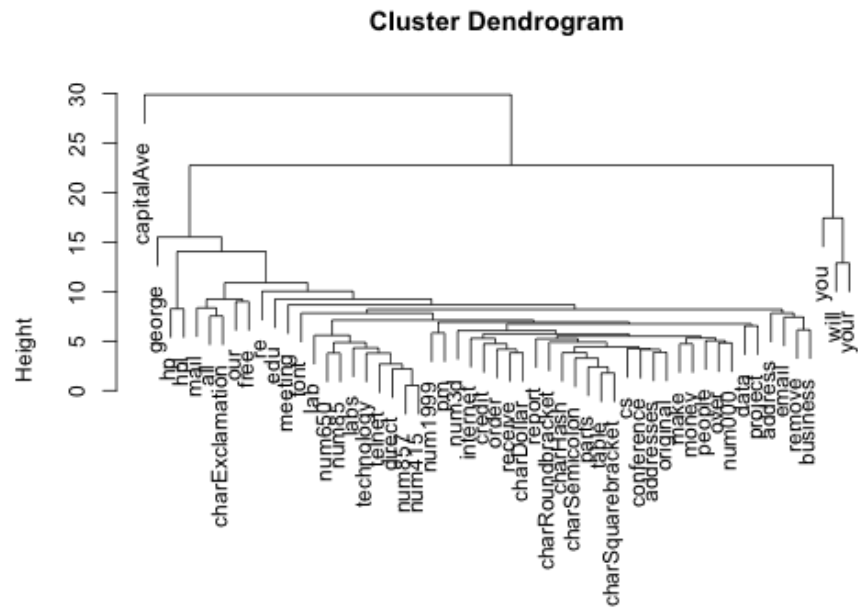
Clustering

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



New clustering

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



```
dist(t(log10(trainSpam[, 1:55] + 1)))  
hclust (*, "complete")
```

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 = reformulate(names(trainSpam)[i], response = "numType")
  glmFit = glm(lmFormula, family = "binomial", data = trainSpam)
  cvError[i] = cv.glm(trainSpam, glmFit, costFunction, 2)$delta[2]
}

## Which predictor has minimum cross-validated error?
names(trainSpam)[which.min(cvError)]

## [1] "charDollar"
```

Get a measure of uncertainty

```
## Use the best model from the group
predictionModel = glm(numType ~ charDollar, family = "binomial", data = trainSpam)

## Get predictions on the test set
predictionTest = predict(predictionModel, testSpam)
predictedSpam = rep("nonspam", dim(testSpam)[1])

## Classify as `spam' for those with prob > 0.5
predictedSpam[predictionModel$fitted > 0.5] = "spam"
```

Get a measure of uncertainty

```
## Classification table  
table(predictedSpam, testSpam$type)
```

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

```
## Error rate  
(61 + 458)/(1346 + 458 + 61 + 449)
```

```
## [1] 0.2243
```

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 characters 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

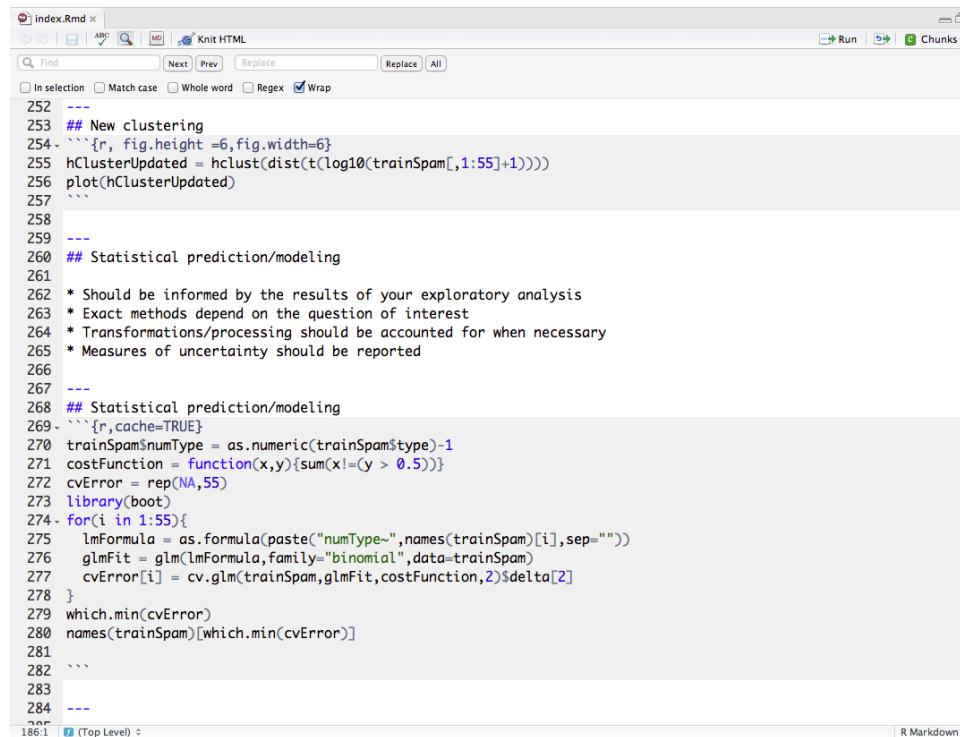
Synthesize/write-up results

- 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



```
252 ---
253 ## New clustering
254 ```{r, fig.height = 6, fig.width = 6}
255 hClusterUpdated = hclust(dist(t(log10(trainSpam[,1:55]+1))))
256 plot(hClusterUpdated)
257 ```
258 ---
259 ---
260 ## Statistical prediction/modeling
261
262 * Should be informed by the results of your exploratory analysis
263 * Exact methods depend on the question of interest
264 * Transformations/processing should be accounted for when necessary
265 * Measures of uncertainty should be reported
266 ---
267 ---
268 ## Statistical prediction/modeling
269 ```{r, cache=TRUE}
270 trainSpam$numType = as.numeric(trainSpam$type)-1
271 costFunction = function(x,y){sum(x!=(y > 0.5))}
272 cvError = rep(NA,55)
273 library(boot)
274 for(i in 1:55){
275   lmFormula = as.formula(paste("numType~",names(trainSpam)[i],sep=""))
276   glmFit = glm(lmFormula,family="binomial",data=trainSpam)
277   cvError[i] = cv.glm(trainSpam,glmFit,costFunction,2)$delta[2]
278 }
279 which.min(cvError)
280 names(trainSpam)[which.min(cvError)]
281 ```
282 ---
283 ---
284 ---
```