Stat405

String processing

Hadley Wickham

1. Motivation: classify spam
2. String basics
3. stringr package

Projects

Project 1

- Will grade this week
- I only have 10 electronic reports. Please email asap if you haven’t already.
- Please let me know if you want your team to stay together: http://bit.ly/SZYF5e

Project 2

- Due Oct 23
- http://projects.propublica.org/emails/
- Not traditional dataset, but increasingly common. We’ll learn useful techniques in the coming weeks
- Experimental!

Email

Motivation

For project, you’ll need to tease emails apart and extract their contents.

Today, we’ll learn the structure of emails, and some string processing basics.

```
contents <- readRDS(”email.rds”)
str(contents)
head(contents)
contents[1]
cat(contents[1], ”\n”)
```

```
Received: from NAHOU-MSMBX07V.corp.enron.com ([192.168.110.98]) by NAHOU-
MSAPP01S.corp.enron.com with Microsoft SMTPSVC(5.0.2195.2966);
      Mon, 1 Oct 2001 14:39:38 -0500
MIME-Version: 1.0
Content-Type: text/plain;
Content-Transfer-Encoding: binary
Subject: MERCATOR ENERGY INCORPORATED and ENERGY WEST INCORPORATED
Date: Mon, 1 Oct 2001 14:39:38 -0500
Message-ID: <053C29CC8315964CB98E1BD5BD48E3080B522E@NAHOU-MSMBX07V.corp.enron.com>
Thread-Index: AcFKsM5wASRhZ102QuKXl3U5Ww741Q==
X-Priority: 1
Priority: Urgent
Importance: high
From: "Landau, Georgi" <Georgi.Landau@ENRON.com>
To: "Bailey, Susan" <Susan.Bailey@ENRON.com>,
     "Boyd, Samantha" <Samantha.Boyd@ENRON.com>,
     "Heard, Marie" <Marie.Heard@ENRON.com>,
     "Jones, Tana" <Tana.Jones@ENRON.com>,
     "Panus, Stephanie" <Stephanie.Panus@ENRON.com>
Return-Path: Georgi.Landau@ENRON.com
```

Please check your records and let me know if you have any kind of documentation evidencing a merger indicating that Mercator Energy Incorporated merged with and into Energy West Incorporated?

I am unable to find anything to substantiate this information.

```
...
Date: Sun, 11 Nov 2001 11:55:05 -0500
Message-Id: <200503101247.j2ACloAq014654@host.high-host.com>
To: benrobinson13@shaw.ca
Subject: LETS DO THIS TOGTHER
From: ben1 <ben_1_wills@yahoo.com.au>
X-Priority: 3 (Normal)
CC:
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-Mailer: RLSP Mailer

Dear Friend,

Firstly, not to caurse you embarrassment, I am Barrister Benson Wills, a
Solicitor at law and the personal attorney to late Mr. Mark Michelle a
National of France, who used to be a private contractor with the Shell
Petroleum Development Company in Saudi Arabia, herein after shall be
referred to as my client. On the 21st of April 2001, him and his wife with
their three children were involved in an auto crash, all occupants of the
vehicle unfortunately lost their lives.
```

Structure of an email

Headers give metadata

Blank line

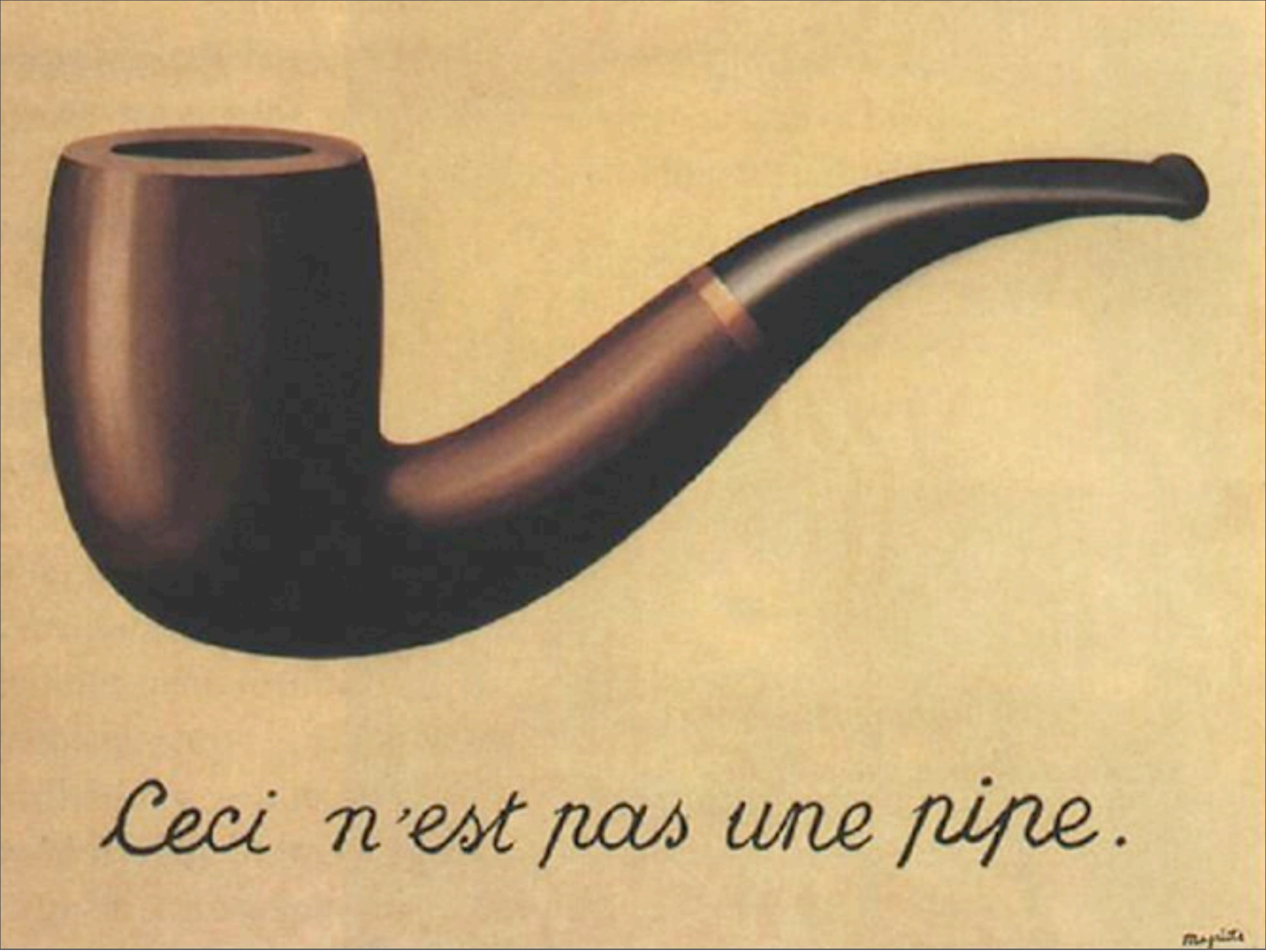
Body of email

Other major complication is attachments and alternative content, but we’ll ignore those for class

Tasks

- Split into header and contents
- Split header into fields
- Make variable that might be interesting to explore

String basics


Ceci n'est pas une pipe.
Magritte

http://www.youtube.com/watch?v=ejweI0EQpX8

Your turn

Try to create a character string that contains just one quotation mark:

"

You have 30 seconds.

```
# Special characters
a <- "\""
b <- "\\"
c <- "a\nb\nc"

a              # displays the representation
cat(a, "\n") # displays the actual string
b
cat(b, "\n") # "\n" tells R to start a new line
c
cat(c, "\n")
```

```
contents[1]

# Does exactly the same thing
# (unless you’re inside a function)
print(contents[1])

cat(contents[1], ”\n”)
```

Special characters

- Use \ to “escape” special characters
  - `\”` = ”
  - `\n` = new line
  - `\\` = \
  - `\t` = tab
- ?Quotes for more

Displaying strings

`print()` will display quoted form.

`cat()` will display actual contents (but need to add a newline, `\n`, to the end).

`message()` sends a message to the user. Can turn off with `suppressMessages()`

Your turn

Create a string for each of the following strings:

```
:-\
```

```
(^_^")
```

```
@_'-'
```

```
\m/
```

Create a multiline string.

Compare the output from `print()` and `cat()`

```
a <- ":-\\"
b <- "(^_^\")"
c <- "@_'-'"
d <- "\\m/"
e <- "This string\ngoes over\nmultiple lines"

a; b; c; d; e
cat(str_c(a, b, c, d, e, "\n", sep = "\n"))
```

stringr
(Back to the problem)

```
install.packages("stringr")
library(stringr)

help(package = "stringr")
# lists all functions in a package
# all functions in stringr start with str_

apropos("str_")
# lists all functions with names containing
# specified characters
```

Header vs. content

Need to split the string into two pieces, based on the the **location** of double line break:

```
str_locate(string, pattern)
```

Need two substrings, one to the right and one to the left:

```
str_sub(string, start, end)
```

```
str_locate("great", "a")
str_locate("fantastic", "a")
str_locate("super", "a")

superlatives <- c("great", "fantastic", "super")
res <- str_locate(superlatives, "a")
str(res) # matrix
str(str_locate_all(superlatives, "a")) # list

str_sub("testing", 1, 3)
str_sub("testing", start = 4) # by default goes to end
str_sub("testing", end = 4) # by default starts at 1

input <- c("abc", "defg")
str_sub(input, c(2, 3))
```

Your turn

Use `str_locate()` to identify the location of the blank line. (Hint: a blank line is a newline immediately followed by another newline)

Split the emails into header and content with `str_sub()`

Make sure to check your results.

```r
breaks <- str_locate(contents, ”\n\n”)

# Extract headers and bodies
header <- str_sub(contents, end = breaks[, 1])
body <- str_sub(contents, start = breaks[, 2])

# Is everything ok with breaks?
```

Headers

- Each header starts at the beginning of a new line
- Each header is composed of a name and contents, separated by a colon

```
h <- header[2]

# Does this work?
str_split(h, ”\n”)[[1]]

# Why / why not?
# How could you fix the problem?
```

```
lines <- str_split(h, "\n")

# because str_split returns a list with one element
# for each input string
lines <- lines[[1]]
continued <- str_sub(lines, 1, 1) %in% c(" ", "\t")

# This is a useful trick!
groups <- cumsum(!continued)

fields <- rep(NA, max(groups))
for (i in seq_along(fields)) {
  fields[i] <- str_c(lines[groups == i], collapse = "\n")
}
# Or
tapply(lines, groups, str_c, collapse = "\n")
```

Your turn

Write a small function that given a single header field splits it into name and contents. Do you want to use `str_split()`, or `str_locate()` & `str_sub()`?

Remember to get the algorithm working before you write the function

```
test1 <- "Sender: <Lighthouse@independent.org>"

test2 <- "Subject: Alice: Where is my coffee?"
```

```
f1 <- function(input) {
  str_split(input, ": ")[[1]]
}

f2 <- function(input) {
  colon <- str_locate(input, ": ")
  c(
    str_sub(input, end = colon[, 1] - 1),
    str_sub(input, start = colon[, 2] + 1)
  )
}

f3 <- function(input) {
  str_split_fixed(input, ": ", 2)[1, ]
}
```

Next steps

We split the content into header and body. And split up the header into fields. Both of these tasks used fixed strings.

What if the pattern we need to match is more complicated?