TidyVerse (Data Wrangling)

Tyson Whitehead

September 19, 2023

Concepts (wikipedia.org)

Data Science An interdisciplinary academic field that uses statistics, scientific computing, scientific methods, processes, algorithms and systems to extract or extrapolate knowledge and insights from noisy, structured, and unstructured data.

Data Wrangling The process of transforming and mapping data from one "raw" data form into another format with the intent of making it more appropriate and valuable for a variety of downstream purposes such as analytics

Data analysts typically spend the majority of their time in the process of data wrangling compared to the actual analysis of the data.

Tidyverse

Opinionated collection of R packages designed for data science. All packages share an underlying

- design philosophy,
- grammar, and
- data structures.

https://www.tidyverse.org

```
library(tidyverse)
```

Packages

dplyr	a grammar of data manipulation
ggplot2	system for declaratively creating graphics
forcats	a suite of tools that solve common problems with factors
tibble	a modern reimagining of the data.frame
readr	fast and friendly way to read and write rectangular data from delimited files
stringr	a cohesive set of functions designed to make working with strings as easy as possible
tidyr	to help you create tidy data
purr	enhances R's functional programming toolkit

Example

Extract the structured Ontario aerodrome data from the sample Ontario Canadian Flight Supplement (CFS) pdf.

- total document is 899 pages (489 pages cover the aerodromes)
- data structure is implicit in layout
- includes diagrams

https://www.navcanada.ca/en/ecfs_04_en.pdf

```
pdftotext -tsv ecfs_04_en.pdf ecfs_04_en.tsv
```

New items:

`pdftotext` Poppler (PDF rendering library): utility for extracing text from PDF files.

Read data

```
basename = 'ecfs_04_en'

input = read_tsv(str_c(basename,'.txt'), quote='')
```

New items:

- `%>%` `forcats`: pass LHS as first arguement to RHS (pipe operator)
- `read_tsv` `readr`: read TSV file to tibble
- `str_c` `stringr`: join strings (vectors to vector)

Drop pdftotext groupings and compute lower-right bounds

```
input = input %>%
  mutate(x0 = left, x1 = left+width,
         y0 = top,  y1 = top+height) %>%
  filter(conf == 100,
         between(width*height / str_length(text),
                 3*3, 72*72/4)) %>%
  select(page = page_num,
         y0, y1, x0, x1, width, height,
         text)
```

New items:

- `mutate` dplyr: calculating new column vectors from existing ones
- `filter` dplyr: select rows
- `str_length` stringr: string length
- `select` dplyr: select columns

Extract page headers

```
split_y = 37

headers = input %>%
  filter(y1 <= split_y)

input = input %>%
  anti_join(headers)
```

New items:

- `anti_join` dplyr: left columns without row matches

Assign header words to line chunks

```
headers = headers %>%
  arrange(page, y1) %>%
  group_by(page) %>%
  mutate(line = cumsum(replace_na(y1-lag(y1) > 4,
                                  TRUE))) %>%
  arrange(page, line, x0) %>%
  group_by(page, line) %>%
  mutate(chunk=cumsum(replace_na(x0-lag(x1) > 2*height,
                                 TRUE)))
```

New items:

- `arrange` dplyr: order rows
- `group_by` dplyr: group rows
- `replace_na` tidyr: replace NAs
- `lag` dplyr: shift vector forward

Coalesce header words to line chunks

```
headers = headers %>%
  group_by(page, line, chunk) %>%
  summarize(text = str_flatten(text, ' '),
            x0 = min(x0), x1 = max(x1),
            y0 = min(y0), y1 = max(y1))
```

New items:

`summarize` dplyr: reduce groups

`str_flatten` stringr: flatten string (vector to scalar)

Extract page aerodrome headers

```
pages = headers %>%
  group_by(page) %>%
  filter(last(text) ==
           'AERODROME / FACILITY DIRECTORY') %>%
  summarize()

headers = headers %>%
  anti_join(pages)
```

New items:

`last` dplyr: reduce vector to last value

Extract aerodrome pages

```
pages = input %>%
  semi_join(pages)

input = input %>%
  anti_join(pages)
```

New items:

- `semi_join` left columns with only row matches

Assign page words to line chunks

```
pages = pages %>%
  arrange(page, y1) %>%
  group_by(page) %>%
  mutate(line = cumsum(replace_na(y1-lag(y1) > 4,
         TRUE))) %>%
  arrange(page,line,x0) %>%
  group_by(page,line) %>%
  mutate(chunk = cumsum(replace_na(x0-lag(x1) > 2*height,
         TRUE)))
```

Coalesce page words to line chunks

```
lines = pages %>%
  arrange(page, line, chunk, x0) %>%
  group_by(page, line, chunk) %>%
  summarize(text = str_flatten(text, ' '),
            x0 = min(x0), x1 = max(x1),
            y0 = min(y0), y1 = max(y1))
```

Get aerodrome headers and their cut lines

```
aerodrome_x0 = 27
aerodrome_x1 = 337

aerodromes =  lines %>%
  pivot_wider(names_prefix = 'chunk', names_from = chunk,
              values_from = c(text, x0, x1, y0, y1)) %>%
  filter(near(x0_chunk1, aerodrome_x0, 4),
         near(x1_chunk2, aerodrome_x1, 4),
         str_detect(text_chunk2, '^C[0-9A-Z]{3}$')) %>%
  mutate(page,name = text_chunk1, aerodrome = text_chunk2,
         y = max(y1_chunk1, y1_chunk2), .keep='none')
```

New items:

`pivot_wider` tidyr: collapse rows into columns

`near` dplyr: compare numeric vectors

`str_detect` stringr: detect pattern

Cleanup aerodrome headers

```
accs = c('CZEG','CZQM','CZQX','CZUL',
         'CZVR','CZWG','CZYZ')

aerodromes = aerodromes %>%
  filter(!aerodrome %in% accs) %>%
  ungroup() %>%
  mutate(aerodrome = fct(aerodrome)) %>%
  group_by(aerodrome) %>%
  mutate(name = first(name))
```

New items:

- `fct` forcats: create a factor
- `first` dplyr: first element

Remove aerodrome headers

```
pages = pages %>%
  anti_join(aerodromes)

lines = pages %>%
  arrange(page, line, chunk, x0) %>%
  group_by(page, line, chunk) %>%
  summarize(text = str_flatten(text, ' '),
            x0 = min(x0), x1 = max(x1),
            y0 = min(y0), y1 = max(y1))
```

Extract invisible province aerodrome lines

```
invisible = lines %>%
  mutate(aerodrome =
           str_extract(text, '^[A-Z]{2} ?(C[0-9A-Z]{3})$',
                       group = 1)) %>%
  inner_join(select(aerodromes, aerodrome, y)) %>%
  filter(near(x0, aerodrome_x0, 4),
         y1-y < 10) %>%
  select(page, line, chunk)

pages = pages %>%
  anti_join(invisible)
```

New items:

`str_extract` stringr: extract first pattern match

`inner_join` dplyr: left and right columns with only rows matches

Assign items to aerodrome by cut lines

```
items = pages %>%
  inner_join(select(aerodromes, page, aerodrome,y),
             by=join_by(page, closest(y0 > y))) %>%
  select(!y)

pages = pages %>%
  anti_join(items)
```

New items:

- `join_by` dplyr: join specification DSL
- `closest` dplyr: rolling helper

Extract labels

```
split_x = 74

labels = items %>%
  filter(x1 <= split_x)

items = items %>%
  anti_join(labels)
```

Assign label words to lines and coalesce them

```
labels = labels %>%
  arrange(page, y1) %>%
  group_by(aerodrome, page) %>%
  mutate(line = cumsum(replace_na(y1-lag(y1) > 4,
                                  TRUE))) %>%
  arrange(page, line, x0) %>%
  group_by(aerodrome, page, line) %>%
  summarize(text = str_flatten(text, ' '),
            x0 = min(x0), x1 = max(x1),
            y0 = min(y0), y1 = max(y1))
```

Assign label depths and item ids for split ones

```
labels = labels %>%
  mutate(level =
           case_when(near(x0, aerodrome_x0, 4) ~ 1L,
                     TRUE                      ~ 2L)) %>%
  arrange(page, line) %>%
  group_by(aerodrome, level) %>%
  mutate(item = cumsum(!str_detect(lag(text,
                                       default = ''),
                                   '-$')),
         text = str_remove(text, '-$'))
```

New items:

`case_when` dplyr: vectorized case

Coalesce split labels and remove continued ones

```
labels = labels %>%
  group_by(aerodrome, level, item) %>%
  summarize(text = str_flatten(text),
            page = first(page),
            line = first(line),
            x0 = min(x0), x1 = max(x1),
            y0 = min(y0), y1 = max(y1)) %>%
  filter(!str_detect(text, '\\(Cont'd\\)$'))
```

New items:

`str_remove` stringr: remove pattern match

Nest labels and get their cut lines

```
labels = labels %>%
  pivot_wider(names_from = level, names_prefix='label',
              values_from = text) %>%
  arrange(page, line) %>%
  group_by(aerodrome) %>%
  fill(label1) %>%
  ungroup() %>%
  mutate(aerodrome, page,
         across(starts_with('label'), fct),
         item = row_number(),
         y = y0-4,
         .keep = 'none')
```

New items:

- `fill` tidyr: fill missing values with previous or next
- `across` dplyr: apply function across columns
- `starts_with` dplyr: selection helper starts with

Assign items to labels by page and cut lines

```
items = items %>%
  bind_rows(select(labels, aerodrome, page, item, y)) %>%
  arrange(page, coalesce(y0, y)) %>%
  group_by(aerodrome) %>%
  fill(item) %>%
  filter(is.na(y)) %>%
  ungroup() %>%
  select(!aerodrome & !y)
```

New items:

- `bind_rows` dplyr: concatenate by row
- `coalesce` dplyr: first non-`NA` value

Assign words to line chunks and coalesce them

```
items = items %>%
  arrange(item, page, y1) %>%
  group_by(item, page) %>%
  mutate(line = cumsum(replace_na(y1-lag(y1) > 4,
                                  TRUE))) %>%
  arrange(item, page, line, x0) %>%
  group_by(item, page, line) %>%
  mutate(chunk = cumsum(replace_na(x0-lag(x1) > 2*height,
                                   TRUE))) %>%
  group_by(item, page, line, chunk) %>%
  summarize(text = str_flatten(text, ' '),
            wrap1 = first(x1-x0),
            wrap2 = first(lead(x1)-x0),
            x0 = min(x0), x1 = max(x1),
            y0 = min(y0), y1 = max(y1))
```

New items:

`lead` dplyr: shift vector back

Compute right-hand margin

```
picture_x0 = 24
picture_x1 = 201
picture_x2 = 339

items = items %>%
  left_join(select(labels, aerodrome, item)) %>%
  arrange(item, page, line, chunk) %>%
  group_by(aerodrome) %>%
  mutate(margin = cummax(if_else(x1 < picture_x1,
                                 picture_x1,
                                 picture_x2))) %>%
  ungroup() %>%
  select(!aerodrome)
```

New items:

`if_else` dplyr: vectorize if else

`left_join` dplyr: left columns augmented with right on row matches

Assign line chunks to paragraphs and coalesce them

```
final = items %>%
  arrange(item, page, line, chunk) %>%
  group_by(item, chunk) %>%
  mutate(paragraph = cumsum(replace_na(lag(margin-x1) <=
                                         wrap1+8,
                                       TRUE))) %>%
  summarize(text = str_flatten(text, ' '),
            page = first(page),
            line = first(line))
```

Compute aerodrome headers to first item gaps

```
picture_y0 = 58
picture_y1 = 191

images = items %>%
  left_join(labels) %>%
  rename(iy1 = y) %>%
  right_join(select(aerodromes, page, aerodrome, iy0 = y)) %>%
  arrange(item, page, line, chunk) %>%
  group_by(aerodrome) %>%
  summarize(page = first(page),
            iy0 = first(iy0)+1, iy1 = first(iy1))
```

New item:

`left_join` dplyr: right columns augmented with left on row matches

Compute diagram boxes

```
images = images %>%
  mutate(x0 = if_else(iy1-iy0 < 36,
                      picture_x1, picture_x0),
         x1 = picture_x2,
         y0 = iy0,
         y1 = if_else(iy1-iy0 < 36,
                      y0+picture_y1-picture_y0, iy1))
```

Get all aerodrome REF items and coalesce them

```
locations = final %>%
    arrange(item, page, line, chunk) %>%
    group_by(item) %>%
    summarize(text = str_flatten(text,' '),
              page = first(page),
              line = first(line)) %>%
    left_join(select(labels, item, aerodrome, label1, label
    filter(label1 == 'REF', is.na(label2))
```

Get latitude, longitude, and elevations strings

```
locations = locations %>%
  mutate(location =
           str_match(text,
                     str_c('N(?<latD>\\d{2})',
                           ' (?<latM>\\d{2})',
                           '(?: (?<latS>\\d{2}))?',
                           ' W(?<lonD>\\d{2,3})',
                           ' (?<lonM>\\d{2})',
                           '(?: (?<lonS>\\d{2}))?')) %>%
           as_tibble(.name_repair = 'unique'),
         elevation =
           as.integer(str_extract(text,
                                  'Elev (-?\\d+).?',
                                  1)))
```

New items:

`str_match` stringr: extract pattern matches

`as_tibble` tibble: convert to tibble

Convert to degrees and decimal degrees and write out

```
locations = locations %>%
  unpack(location) %>%
  mutate(across(starts_with('lat') | starts_with('lon'),
                as.integer),
         latitude = latD + latM/60 +
                    replace_na(latS,0)/3600,
         longitude = -lonD - lonM/60 -
                     replace_na(lonS,0)/3600) %>%
  select(aerodrome, latitude, longitude, elevation) %>%
  write_csv('aerodromes.csv')
```

New items:

`unpack`	tidyr: expand out nested columns
`write_csv`	readr: write tibble to CSV file

Generate diagram images

```
imagemagik = images %>%
    mutate(aerodrome, page,
           command =
             sprintf(str_c('magick',
                           ' -density 288',
                           ' -extract %.0fx%.0f+%.0f+%.0f'
                           ' %s.pdf\'[%.0f]\'',
                           ' %s.jpg'),
                     (x1-x0)*4, (y1-y0)*4, x0*4, y0*4,
                     basename, page-1,
                     aerodrome),
           result = map_int(command, system),
           .keep = 'none')
```

New items:

- `map_int` purr: vectorize scalar function returning integer