I stopped caring about the estimates that occasionally raid the internet about how much time data scientists spend on data wrangling VS modelling. The answer is: *probably* a lot and likely more than originally planned (*probably* indicating here dependency on the state and richness of input data and the intended application for it, ekhem).

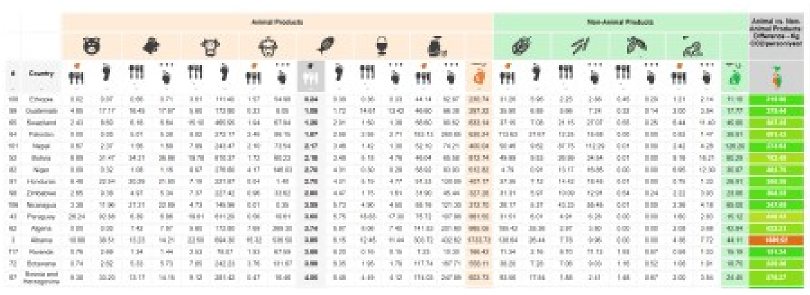
Still, the right tools can go a long way in achieving the desired result in the time frame that can surprise even the most optimistic of us. Needless to say, R is an excellent example of that right tool :).

Goal

I want to be able to analyse the data on annual CO2 emissions per person for 130 nations worldwide published by Food and Agriculture Organization of the United Nations (FAO), that are nicely summarised on nu3 website.

Challenge

The data resides in the HTML table that has notoriously messy headers. Look and weep:


Animal Products
Non-Animal Products
Country

But fear not! There's nothing that R can't fix in a blink of an eye. And here's how:

Solution

First, load a handful of classic R packages:

- `{rvest}` for web-scraping
- `{dplyr}` for data-wrangling
- `{tidyr}` for data transformation
- `{stringr}` for string manipulation
- `{janitor}` for clean headers that your OCD will love you for

```
library(rvest)
library(dplyr)
library(tidyr)
library(stringr)
library(janitor)
```

Next, scrape the content of the website and extract the HTML table:

```
url <- "https://www.nu3.de/blogs/nutrition/food-carbon-footprint-index-2018"

# scrape the website
url_html <- read_html(url)

# extract the HTML table
whole_table <- url_html %>%
 html_nodes('table') %>%
 html_table(fill = TRUE) %>%
 .[[1]]
```

```
str(whole_table)
```

```
## 'data.frame':    133 obs. of  27 variables:
##  $ X1 : chr  "" "" "#" "1" ...
##  $ X2 : chr  "" "" "Country" "Argentina" ...
##  $ X3 : chr  "Animal Products" "" "" "10.51" ...
##  $ X4 : chr  "Animal Products" "" "" "37.20" ...
##  $ X5 : chr  "Animal Products" "" "" "38.66" ...
##  $ X6 : chr  "Animal Products" "" "" "41.53" ...
##  $ X7 : chr  "Animal Products" "" "" "55.48" ...
##  $ X8 : chr  "Animal Products" "" "" "1712.00" ...
##  $ X9 : chr  "Animal Products" "" "" "1.56" ...
##  $ X10: chr  "Animal Products" "" "" "54.63" ...
##  $ X11: chr  "Animal Products" "" "" "4.36" ...
##  $ X12: chr  "Animal Products" "" "" "6.96" ...
##  $ X13: chr  "Animal Products" "" "" "11.39" ...
##  $ X14: chr  "Animal Products" "" "" "10.46" ...
##  $ X15: chr  "Animal Products" "" "" "195.08" ...
##  $ X16: chr  "Animal Products" "" "" "277.87" ...
##  $ X17: chr  "Animal Products" "" "" "2140.65" ...
##  $ X18: chr  "Non-Animal Products" "" "" "103.11" ...
##  $ X19: chr  "Non-Animal Products" "" "" "19.66" ...
##  $ X20: chr  "Non-Animal Products" "" "" "8.77" ...
##  $ X21: chr  "Non-Animal Products" "" "" "11.22" ...
##  $ X22: chr  "Non-Animal Products" "" "" "0.00" ...
##  $ X23: chr  "Non-Animal Products" "" "" "0.00" ...
##  $ X24: chr  "Non-Animal Products" "" "" "0.49" ...
##  $ X25: chr  "Non-Animal Products" "" "" "0.87" ...
##  $ X26: chr  "Non-Animal Products" "" "" "31.75" ...
##  $ X27: chr  "Animal vs. Non-Animal Products:" "Difference - Kg CO2/person
/year" "" "2108.90" ...
```

The state of column names can be summarised in only one word here: messmessmessmessmessmessmessmess….!!

Before we tidy them up, let’s scrap the headers and isolate just the content of the table:

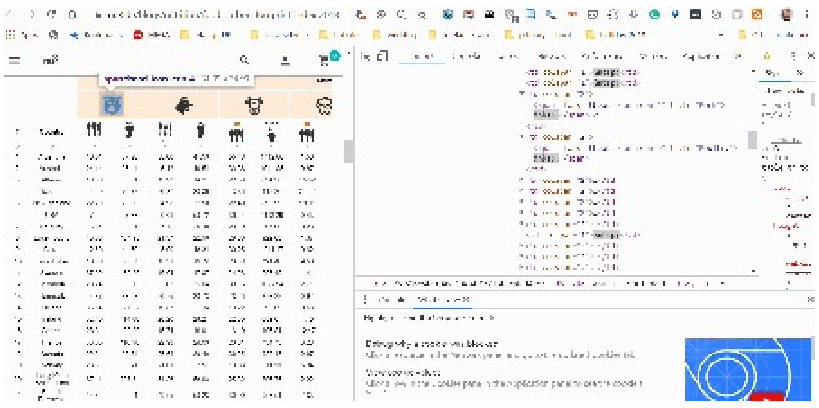
```
# tidying the table
table_content <- whole_table %>%
 select(-X1) %>% # remove redundant column
 filter(!dplyr::row_number() %in% 1:3) # remove redundant rows

head(table_content)
```

```
##             X2    X3    X4    X5    X6    X7      X8    X9    X10   X11
## 1   Argentina 10.51 37.20 38.66 41.53 55.48 1712.00  1.56  54.63  4.36
## 2   Australia 24.14 85.44 46.12 49.54 33.86 1044.85  9.87 345.65 17.69
## 3     Albania 10.88 38.51 13.23 14.21 22.50  694.30 15.32 536.50  3.85
## 4     Iceland 21.69 76.77 26.87 28.86 13.36  412.26 21.12 739.62 74.41
## 5 New Zealand 22.29 78.90 34.98 37.58 22.49  693.99 18.91 662.23 20.36
## 6         USA 27.64 97.83 50.01 53.72 36.24 1118.29  0.43  15.06 12.35
##      X12   X13   X14    X15    X16     X17    X18   X19   X20   X21  X22
## 1   6.96 11.39 10.46 195.08 277.87 2140.65 103.11 19.66  8.77 11.22 0.00
## 2  28.25  8.51  7.82 234.49 334.01 1895.55  70.46 13.44 11.03 14.12 0.19
## 3   6.15 12.45 11.44 303.72 432.62 1733.73 138.64 26.44  7.78  9.96 0.00
## 4 118.81  8.24  7.57 225.82 321.66 1705.55  72.92 13.91  3.89  4.98 0.11
## 5  32.51  9.91  9.10 137.25 195.50 1709.80  76.91 14.67  9.16 11.72 0.44
## 6  19.72 14.58 13.39 254.69 362.78 1680.79  80.43 15.34  6.88  8.80 0.04
##    X23  X24   X25   X26     X27
## 1 0.00 0.49  0.87 31.75 2108.90
## 2 0.09 8.73 15.45 43.09 1852.46
## 3 0.00 4.36  7.72 44.11 1689.62
## 4 0.05 3.88  6.87 25.80 1679.75
## 5 0.20 8.22 14.55 41.14 1668.67
## 6 0.02 7.86 13.91 38.07 1642.72
```

A bit better. It's worth noting that we're dealing with 26 variables that need column names. Now, how can we extract these?

When inspecting the HTML code behind the table (**Right Click => Inspect => Navigate to the part of the code that highlights the table headers**), you will notice that all the "icon headers" belong to `thead-icon` class that has an attribute `title` with the very column names that we're interested in.



`{rvest}` package makes it super easy to extract these using `html_nodes()` and `html_attr()` functions:

```
raw_headers <- url_html %>%
 html_nodes(".thead-icon") %>%
 html_attr('title')

raw_headers[1:35]
```

```
##  [1] "Schwein"
##  [2] "Geflügel"
##  [3] "Rind"
##  [4] "Lamm- und Ziegenfleisch"
##  [5] "Fisch"
##  [6] "Eier"
##  [7] "Milch - inkl. Käse"
##  [8] "Durchschnitt Kg CO2/Person/Jahr"
##  [9] "Weizen und Weizenerzeugnisse"
## [10] "Reis"
## [11] "Sojabohnen"
## [12] "Nüsse - inkl. Erdnussbutter"
## [13] "Durchschnitt Kg CO2/Person/Jahr"
## [14] "Supplied for Consumption (kg/person/year)"
## [15] "Kg CO2/person/year"
## [16] "Animal vs. Non-Animal Products: Difference - Kg CO2/person/year"
## [17] "Pork"
## [18] "Poultry"
## [19] "Beef"
## [20] "Lamb & Goat"
## [21] "Fish"
## [22] "Eggs"
## [23] "Milk - inc. cheese"
## [24] "Wheat and Wheat Products"
## [25] "Rice"
## [26] "Soybeans"
## [27] "Nuts inc. Peanut Butter"
## [28] "Supplied for Consumption (kg/person/year)"
## [29] "Kg CO2/person/year"
## [30] "Supplied for Consumption (kg/person/year)"
## [31] "Kg CO2/person/year"
## [32] "Supplied for Consumption (kg/person/year)"
## [33] "Kg CO2/person/year"
## [34] "Supplied for Consumption (kg/person/year)"
## [35] "Kg CO2/person/year"
```

As you can see, this way we extracted a few additional titles, but it's easy to select the correct ones from here. First, let's get the bottom-most names that refer to the actual column names:

```
tidy_bottom_header <- raw_headers[28:length(raw_headers)]
tidy_bottom_header[1:10]
```

```
##  [1] "Supplied for Consumption (kg/person/year)"
##  [2] "Kg CO2/person/year"
##  [3] "Supplied for Consumption (kg/person/year)"
##  [4] "Kg CO2/person/year"
##  [5] "Supplied for Consumption (kg/person/year)"
##  [6] "Kg CO2/person/year"
##  [7] "Supplied for Consumption (kg/person/year)"
##  [8] "Kg CO2/person/year"
##  [9] "Supplied for Consumption (kg/person/year)"
## [10] "Kg CO2/person/year"
```

Then, the middle ones:

```
raw_middle_header <- raw_headers[17:27]
raw_middle_header
```

```
##  [1] "Pork"                        "Poultry"
##  [3] "Beef"                        "Lamb & Goat"
##  [5] "Fish"                        "Eggs"
##  [7] "Milk - inc. cheese"          "Wheat and Wheat Products"
##  [9] "Rice"                        "Soybeans"
## [11] "Nuts inc. Peanut Butter"
```

And finally, let's organise them in a way that the column names reflect the variable order:

```
tidy_headers <- c(
 rep(raw_middle_header[1:7], each = 2),
 "animal_total",
 rep(raw_middle_header[8:length(raw_middle_header)], each = 2),
 "non_animal_total",
 "country_total")

tidy_headers
```

```
##  [1] "Pork"                        "Pork"
##  [3] "Poultry"                     "Poultry"
##  [5] "Beef"                        "Beef"
##  [7] "Lamb & Goat"                 "Lamb & Goat"
##  [9] "Fish"                        "Fish"
## [11] "Eggs"                        "Eggs"
## [13] "Milk - inc. cheese"          "Milk - inc. cheese"
## [15] "animal_total"                "Wheat and Wheat Products"
## [17] "Wheat and Wheat Products"    "Rice"
## [19] "Rice"                        "Soybeans"
## [21] "Soybeans"                    "Nuts inc. Peanut Butter"
## [23] "Nuts inc. Peanut Butter"  "non_animal_total"
## [25] "country_total"
```

I must say, this does the job but it is the part of the code that I'm least happy with, as it requires a lot of manual effort and thus is prone to errors. If you know better ways how to abstract/automate it, feel free to share it in the comments below!

Ok, over the first hurdle! Now, we want to retain the information about both, the type of food that the value refers too, as well as the metric that it describes (consumption or CO2 emmissions). Let's combine the two and make them column names:

```
combined_colnames <- paste(tidy_headers, tidy_bottom_header, sep = ';')
colnames(table_content) <- c("Country", combined_colnames)
glimpse(table_content[, 1:10])
```

```
## Observations: 130
## Variables: 10
## $ Country                                                   "Argen...
## $ `Pork;Supplied for Consumption (kg/person/year)`          "10.51...
## $ `Pork;Kg CO2/person/year`                                 "37.20...
## $ `Poultry;Supplied for Consumption (kg/person/year)`       "38.66...
## $ `Poultry;Kg CO2/person/year`                              "41.53...
## $ `Beef;Supplied for Consumption (kg/person/year)`          "55.48...
## $ `Beef;Kg CO2/person/year`                                 "1712....
## $ `Lamb & Goat;Supplied for Consumption (kg/person/year)`  "1.56"...
## $ `Lamb & Goat;Kg CO2/person/year`                          "54.63...
## $ `Fish;Supplied for Consumption (kg/person/year)`          "4.36"...
```

I know what you're thinking – this is faaaar from perfect (or pretty, for that matter), but we're only 2 lines of code away to having something much more digestible. Long live `{tidyr}` !!

```
long_table <- table_content %>%
 # make column names observations of Category variable
 tidyr::pivot_longer(cols = -Country, names_to = "Category", values_to =
"Values") %>%
 # separate food-related information from the metric
 tidyr::separate(col = Category, into = c("Food Category", "Metric"), sep = ';')

glimpse(long_table)
```

```
## Observations: 3,250
## Variables: 4
## $ Country          "Argentina", "Argentina", "Argentina", "Argent...
## $ `Food Category`  "Pork", "Pork", "Poultry", "Poultry", "Beef", ...
## $ Metric           "Supplied for Consumption (kg/person/year)", "...
## $ Values           "10.51", "37.20", "38.66", "41.53", "55.48", "...
```

Almost there! I'm still not happy with the `Metric` variable that ideally should be split into two separate columns for an easier comparison of consumption and CO2 emissions between the countries:

```
tidy_table <- long_table %>%
   tidyr::pivot_wider(names_from = Metric, values_from = Values) %>%
   janitor::clean_names('snake')

glimpse(tidy_table)
```

```
## Observations: 1,820
## Variables: 4
## $ country                                  "Argentina", "Argentin...
## $ food_category                            "Pork", "Poultry", "Be...
## $ supplied_for_consumption_kg_person_year  "10.51", "38.66", "55....
## $ kg_co2_person_year                       "37.20", "41.53", "171...
```

Great stuff. Now we only need some final touches renaming the long column names and removing the instances of summarised data – these will be easy to calculate, if we need to, based on the final dataset:

```
final_table <- tidy_table %>%
 rename(consumption = 3,
         co2_emmission = 4) %>%
 filter(!stringr::str_detect(food_category, "total"))

head(final_table, 20)
```

```
## # A tibble: 20 x 4
##    country   food_category            consumption co2_emmission
##
##  1 Argentina Pork                     10.51       37.20
##  2 Argentina Poultry                  38.66       41.53
##  3 Argentina Beef                     55.48       1712.00
##  4 Argentina Lamb & Goat              1.56        54.63
##  5 Argentina Fish                     4.36        6.96
##  6 Argentina Eggs                     11.39       10.46
##  7 Argentina Milk - inc. cheese       195.08      277.87
##  8 Argentina Wheat and Wheat Products 103.11      19.66
##  9 Argentina Rice                     8.77        11.22
## 10 Argentina Soybeans                 0.00        0.00
## 11 Argentina Nuts inc. Peanut Butter  0.49        0.87
## 12 Australia Pork                     24.14       85.44
## 13 Australia Poultry                  46.12       49.54
## 14 Australia Beef                     33.86       1044.85
## 15 Australia Lamb & Goat              9.87        345.65
## 16 Australia Fish                     17.69       28.25
## 17 Australia Eggs                     8.51        7.82
## 18 Australia Milk - inc. cheese       234.49      334.01
## 19 Australia Wheat and Wheat Products 70.46       13.44
## 20 Australia Rice                     11.03       14.12
```

Voila! We've got what we wanted with a few lines of code and basic knowledge of HTML. The whole data-wrangling process took only a fraction of what I expected it to take and now the data is tidy and 'analysis-ready'.

Do you know any examples of tricky-to-tidy data? Make sure to share your knowledge – and frustrations, why not!