

Data Cleaning Part 1

Data Wrangling in R

Data Cleaning

In general, data cleaning is a process of investigating your data for inaccuracies, or recoding it in a way that makes it more manageable.

MOST IMPORTANT RULE - LOOK AT YOUR DATA!

Read in the UFO dataset

Read in data or download from: http://sisbid.github.io/Data-Wrangling/data/ufo/ufo_data_complete.csv.gz

```
ufo <- read_delim("https://sisbid.github.io/Data-Wrangling/
```

Warning: One or more parsing issues, call ``problems()`` on y
e.g.:

```
dat <- vroom(...)  
problems(dat)
```

Rows: 88875 Columns: 11

-- Column specification -----

Delimiter: ","

chr (10): datetime, city, state, country, shape, duration

dbl (1): duration (seconds)

i Use ``spec()`` to retrieve the full column specification for

i Specify the column types or set ``show_col_types = FALSE``

The “problems”

You saw warning messages when reading in this dataset. We can see these with the `problems()` function from `readr`.

If we scroll through we can see some interesting notes.

```
p <-problems(ufo)
p %>% glimpse()
```

Rows: 200

Columns: 5

```
$ row      <int> 878, 1713, 1815, 2858, 3734, 4756, 5389, 5
$ col      <int> 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12
$ expected <chr> "11 columns", "11 columns", "11 columns",
$ actual   <chr> "12 columns", "12 columns", "12 columns",
$ file     <chr> "", "", "", "", "", "", "", "", "", "", ""
```

Any unique problems?

```
count(p, expected, actual)
```

```
# A tibble: 5 x 3
```

	expected	actual	n
	<chr>	<chr>	<int>
1	11 columns	"12 columns"	196
2	a double	"0.5`"	1
3	a double	"2`"	1
4	a double	"2631600 "	1
5	a double	"8`"	1

The “problems”

Go look at the raw data around these rows.

4/10/2005 22:52 santa cru
38453 el zumbador (el

```
colnames(ufo)
```

[1]	"datetime"	"city"	"state"
[4]	"country"	"shape"	"duration"
[7]	"duration (hours/min)"	"comments"	"date po"
[10]	"latitude"	"longitude"	

Reading in again

Now we have a chance to keep but clean these values!

```
url <- "https://sisbid.github.io/Data-Wrangling/data/ufo/ufo.csv"
ufo <- read_csv(url, col_types = cols(`duration (seconds)` =
```

Warning: One or more parsing issues, call `problems()` on your object.
e.g.:

```
dat <- vroom(...)
problems(dat)
```

Look at the problems again

```
p <- problems(ufo)
```

```
count(p, expected, actual)
```

```
# A tibble: 1 x 3
```

	expected	actual	n
	<chr>	<chr>	<int>
1	11 columns	12 columns	196

Drop the remaining shifted problematic rows for now

Multiply by negative one to drop the rows. Use the Slice function to “select” those rows based on the index.

```
(pull(p, row) *-1) %>% head()
```

```
[1] -878 -1713 -1815 -2858 -3734 -4756
```

```
ufo_clean <- ufo %>% slice((pull(p, row))*-1)
```

Checking

```
nrow(ufo) - nrow(ufo_clean)
```

```
[1] 196
```

```
count(p, expected, actual)
```

```
# A tibble: 1 x 3
```

expected	actual	n
<chr>	<chr>	<int>

1 11 columns	12 columns	196
--------------	------------	-----

Clean names with the `clean_names()` function from the `janitor` package

```
colnames(ufo_clean)
```

```
[1] "datetime"          "city"              "state"  
[4] "country"           "shape"             "duration"  
[7] "duration (hours/min)" "comments"          "date posted"  
[10] "latitude"          "longitude"
```

```
ufo_clean <- clean_names(ufo_clean)  
colnames(ufo_clean)
```

```
[1] "datetime"          "city"              "state"  
[4] "country"           "shape"             "duration_seconds"  
[7] "duration_hours_min" "comments"          "date_posted"  
[10] "latitude"          "longitude"
```

Recoding Variables

Exact Swaps - recode function

```
within mutate...
```

```
recode(variable, value_old = value_new,  
        other_value_old = other_value_new)
```

```
ufo_clean %>%  
  mutate(country =  
    recode(country, gb = "Great Britain")) %>%  
  glimpse()
```

```
Rows: 88,679
```

```
Columns: 11
```

```
$ datetime      <chr> "10/10/1949 20:30", "10/10/1949  
$ city          <chr> "san marcos", "lackland afb", "c  
$ state         <chr> "tx", "tx", NA, "tx", "hi", "tn"  
$ country       <chr> "us", NA, "Great Britain", "us"  
$ shape         <chr> "cylinder", "light", "circle", "  
$ duration_seconds <chr> "2700", "7200", "20", "20", "900  
$ duration_hours_min <chr> "45 minutes", "1-2 hrs", "20 sec  
$ comments      <chr> "This event took place in early
```

Exact Swaps - recode function

```
ufo_clean %>% mutate(country =  
  recode(country,  
    gb = "Great Britain",  
    us = "United States")) %>%  
  glimpse()
```

Rows: 88,679

Columns: 11

\$ datetime	<chr> "10/10/1949 20:30", "10/10/1949
\$ city	<chr> "san marcos", "lackland afb", "c
\$ state	<chr> "tx", "tx", NA, "tx", "hi", "tn"
\$ country	<chr> "United States", NA, "Great Brit
\$ shape	<chr> "cylinder", "light", "circle", "
\$ duration_seconds	<chr> "2700", "7200", "20", "20", "900
\$ duration_hours_min	<chr> "45 minutes", "1-2 hrs", "20 sec
\$ comments	<chr> "This event took place in early
\$ date_posted	<chr> "4/27/2004", "12/16/2005", "1/21
\$ latitude	<chr> "29.8830556", "29.38421", "53.2"
\$ longitude	<chr> "-97.0411111", "-99.5810888", "

How many countries?

```
ufo_clean %>% count(country)
```

```
# A tibble: 6 x 2
```

```
  country      n
```

```
  <chr>    <int>
```

```
1 au         592
```

```
2 ca        3259
```

```
3 de         111
```

```
4 gb        2043
```

```
5 us       70146
```

```
6 <NA>     12528
```

case_when() regions to create a new variable based on conditions of other variables

```
case_when(test ~ value if test is true,  
          test2 ~ vlue if test2 is true,  
          TRUE ~ value if all above tests are not true) # de
```

```
ufo_clean <- ufo_clean %>% mutate(  
  region = case_when(  
    country %in% c("us", "ca") ~ "North America",  
    country %in% c("de") ~ "Europe",  
    country %in% "gb" ~ "Great Britain",  
    TRUE ~ "Other"  
  ))  
ufo_clean %>% select(country, region) %>% head()
```

```
# A tibble: 6 x 2
```

```
  country region
```

```
<chr>    <chr>
```

```
1 us      North America
```

```
2 <NA>    Other
```


case_when - another example

The TRUE value can also just be the original values.

```
ufo_clean %>% mutate(country = case_when(  
  country == "gb" ~ "Great Britain",  
  country == "us" ~ "United States",  
  country == "au" ~ "Australia",  
  country == "DE" ~ "Germany",  
  TRUE ~ country))%>%  
  
glimpse()
```

Rows: 88,679

Columns: 12

\$ datetime	<chr> "10/10/1949 20:30", "10/10/1949
\$ city	<chr> "san marcos", "lackland afb", "c
\$ state	<chr> "tx", "tx", NA, "tx", "hi", "tn"
\$ country	<chr> "United States", NA, "Great Brit
\$ shape	<chr> "cylinder", "light", "circle", "
\$ duration_seconds	<chr> "2700", "7200", "20", "20", "900
\$ duration_hours_min	<chr> "45 minutes", "1-2 hrs", "20 sec

Strange country values

Sometimes country is NA even though state is known. A conditional more flexible recoding would be helpful...

```
head(ufo_clean)
```

```
# A tibble: 6 x 12
```

	datetime	city	state	country	shape	durat~1	durat~2	co
	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
1	10/10/1949	~ san	tx	us	cyli~	2700	45 min~	TH
2	10/10/1949	~ lack~	tx	<NA>	light	7200	1-2 hrs	19
3	10/10/1955	~ ches~	<NA>	gb	circ~	20	20 sec~	Gr
4	10/10/1956	~ edna	tx	us	circ~	20	1/2 ho~	My
5	10/10/1960	~ kane~	hi	us	light	900	15 min~	AS
6	10/10/1961	~ bris~	tn	us	sphe~	300	5 minu~	My

```
# ... with 2 more variables: longitude <chr>, region <chr>
#   variable names 1: duration_seconds, 2: duration_hours_m
#   4: date_posted, 5: latitude
```

Deeper look

Looking at city... it seems like many of these are in fact in the US.

```
ufo_clean %>% filter(state == "tx") %>% count(country, state)
```

```
# A tibble: 2 x 3
```

	country	state	n
	<chr>	<chr>	<int>
1	us	tx	3734
2	<NA>	tx	307

```
ufo_clean %>% filter(state == "tx" & is.na(country)) %>% select(city)
```

```
# A tibble: 307 x 1
```

	city
	<chr>
1	lackland afb
2	mercedies
3	texas city/galveston
4	houston/tomball
5	bettendorf

Checkin Utah as well

```
ufo_clean %>% filter(state == "ut") %>% count(country, state)
```

```
# A tibble: 2 x 3
```

	country	state	n
	<chr>	<chr>	<int>
1	us	ut	658
2	<NA>	ut	140

```
ufo_clean %>% filter(state == "ut" & is.na(country)) %>% summarise(city = paste(unique(city)))
```

```
# A tibble: 140 x 1
```

	city
	<chr>
1	canyonlands np
2	ogden/clinton
3	sandy
4	salt lake valley
5	sandy
6	duchenne

Get US States

```
ufo_clean %>% filter(country == "us") %>%  
  count(state) %>%  
  pull(state)
```

```
[1] "ak" "al" "ar" "az" "ca" "co" "ct" "dc" "de" "fl" "ga"  
[16] "in" "ks" "ky" "la" "ma" "md" "me" "mi" "mn" "mo" "ms"  
[31] "nh" "nj" "nm" "nv" "ny" "oh" "ok" "or" "pa" "pr" "ri"  
[46] "ut" "va" "vt" "wa" "wi" "wv" "wy"
```

```
US_states <- ufo_clean %>%  
  filter(country == "us") %>%  
  count(state) %>%  
  pull(state)
```

Get Canada States

```
ufo_clean %>% filter(country == "ca") %>%  
  count(state) %>%  
  pull(state)
```

```
[1] "ab" "bc" "mb" "nb" "nf" "ns" "nt" "on" "pe" "pq" "qc"  
[16] NA
```

```
CA_states <- ufo_clean %>%  
  filter(country == "ca") %>%  
  count(state) %>%  
  pull(state)
```

Get Great Britan states

```
ufo_clean %>% filter(country == "gb") %>%  
  count(state) %>%  
  pull(state)
```

```
[1] "bc" "la" "ms" "nc" "ns" "nt" "ri" "sk" "tn" "wv" "yt"
```

```
GB_states <- ufo_clean %>%  
  filter(country == "gb") %>%  
  count(state) %>%  
  pull(state)
```

A small overlap with us states.

Get DE states

```
ufo_clean %>% filter(country == "de") %>%  
  count(state) %>%  
  pull(state)
```

```
[1] NA
```


Get AU states

```
ufo_clean %>% filter(country == "au") %>%  
  count(state) %>%  
  pull(state)
```

```
[1] "al" "dc" "nt" "oh" "sa" "wa" "yt" NA
```

```
AU_states <- ufo_clean %>%  
  filter(country == "au") %>%  
  count(state) %>%  
  pull(state)
```

Some overlap with US states.

Get just unique

```
US_states <- US_states[!(US_states %in% AU_states)]  
US_states <- US_states[!(US_states %in% GB_states)]  
US_states <- US_states[!(US_states %in% CA_states)]
```

```
AU_states <- AU_states[!(AU_states %in% US_states)]  
AU_states <- AU_states[!(AU_states %in% GB_states)]  
AU_states <- AU_states[!(AU_states %in% CA_states)]
```

```
CA_states <- CA_states[!(CA_states %in% US_states)]  
CA_states <- CA_states[!(CA_states %in% GB_states)]  
CA_states <- CA_states[!(CA_states %in% CA_states)]
```

```
GB_states <- GB_states[!(GB_states %in% US_states)]  
GB_states <- GB_states[!(GB_states %in% AU_states)]  
GB_states <- GB_states[!(GB_states %in% CA_states)]
```

How often do rows have a value for country but not the US?

```
ufo_clean %>%  
  filter(country != "us" & !is.na(country)) %>%  
  count(country)
```

```
# A tibble: 4 x 2  
  country      n  
  <chr>    <int>  
1 au         592  
2 ca        3259  
3 de         111  
4 gb        2043
```

more complicated case_when

Let's make an assumption that if the state value is within the data as a state for a specific country that it comes from that country for the sake of illustration.

```
ufo_clean <- ufo_clean %>% mutate(prob_country =  
  case_when((is.na(country) & state %in% c(US_states))  
            (is.na(country) & state %in% c(CA_states))  
            (is.na(country) & state %in% c(AU_states))  
            (is.na(country) & state %in% c(GB_states))  
            TRUE ~ country))
```

results

```
count(ufo_clean, prob_country)
```

```
# A tibble: 9 x 2
```

	prob_country	n
	<chr>	<int>
1	au	592
2	Australia	699
3	ca	3259
4	de	111
5	gb	2043
6	Great Britain	5395
7	United States	5897
8	us	70146
9	<NA>	537

results

Take a look at those NAs.

```
ufo_clean %>% filter(is.na(prob_country))
```

```
# A tibble: 537 x 13
```

	datetime	city	state	country	shape	durat~1	durat~2	co
	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>	<chr>
1	10/10/1979~	sadd~	ab	<NA>	tria~	270	4.5 or~	L
2	10/10/1987~	mani~	on	<NA>	disk	600	10/mins	We
3	10/10/1990~	pens~	sa	<NA>	unkn~	180	1/2 hr.	i
4	10/10/1999~	st. ~	pq	<NA>	egg	5400	90 min.	ea
5	10/10/2012~	moun~	on	<NA>	tria~	30	30 sec~	Or
6	10/1/2005 ~	mont~	qc	<NA>	chan~	0	don~	I
7	10/1/2005 ~	port~	mb	<NA>	other	30	30 sec~	St
8	10/12/2007~	mont~	nb	<NA>	oval	0	an goi~	8
9	10/12/2010~	port~	nf	<NA>	light	900	10 -15~	mo
10	10/13/2006~	simc~	on	<NA>	circ~	30	30 sec~	Ho

```
# ... with 527 more rows, 3 more variables: longitude <chr>  
# prob_country <chr>, and abbreviated variable names 1: c
```

We could confirm with city info and latitude and longitude

```
ufo_clean %>% filter(country == "de") %>%  
  pull(city)
```

```
[1] "berlin (germany)"  
[2] "berlin (germany)"  
[3] "obernheim (germany)"  
[4] "ottersberg (germany)"  
[5] "urbach (germany)"  
[6] "bremen (30 km south of) (germany)"  
[7] "sembach (germany)"  
[8] "magdeburg (germany)"  
[9] "neuruppin (germany)"  
[10] "lampertheim (germany)"  
[11] "ramstein (germany)"  
[12] "bremen (germany)"  
[13] "nurenborg (germany)"  
[14] "senftenberg (germany)"  
[15] "schwalmtal (germany)"  
[16] "neuenhagen (germany)"
```

Even more specific

```
ufo_clean <- ufo_clean %>% mutate(prob_country =  
  case_when(  
    (is.na(country) & state %in% c(US_states)) |  
    country == "us" ~ "United States",  
    (is.na(country) & state %in% c(CA_states)) |  
    country == "ca" ~ "Canada",  
    (is.na(country) & state %in% c(AU_states)) |  
    country == "au" ~ "Australia",  
    (is.na(country) & state %in% c(GB_states)) |  
    country == "gb" ~ "Great Britain",  
    country == "de" ~ "Germany",  
    TRUE ~ country))
```

We would want to confirm what we recoded with the cities and latitude and longitude, especially to deal with the overlaps in the state lists.

Check counts

```
ufo_clean %>%  
  count(country, prob_country)
```

```
# A tibble: 9 x 3
```

	country	prob_country	n
	<chr>	<chr>	<int>
1	au	Australia	592
2	ca	Canada	3259
3	de	Germany	111
4	gb	Great Britain	2043
5	us	United States	70146
6	<NA>	Australia	699
7	<NA>	Great Britain	5395
8	<NA>	United States	5897
9	<NA>	<NA>	537

Summary

- ▶ `recode` makes exact swaps
- ▶ `case_when` can use conditionals, need to specify what value for if no conditions are met (can be the original value of a variable if we use the variable name).

Lab

<https://sisbid.github.io/Data-Wrangling/labs/data-cleaning-lab.Rmd>