Why code?

Joseph Ciesielski
12/10/2019

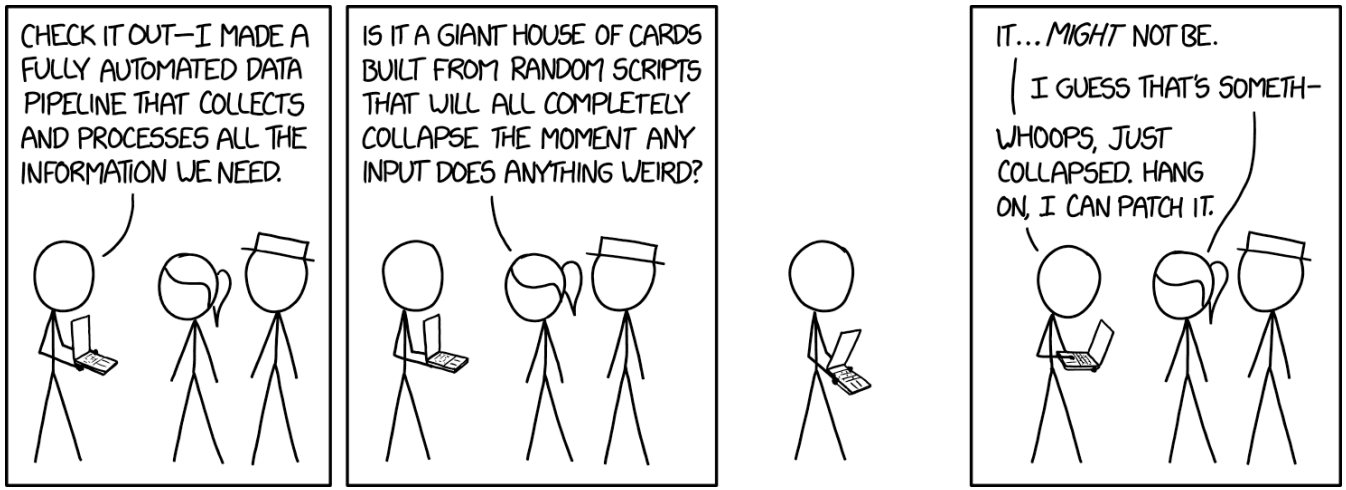
Grounding

1. Data analysis offers the power to critically examine and improve our organizations and advance their missions

2. The way we do data analysis isn't (always) conducive to this

3. Using code can help

A story


CHECK IT OUT—I MADE A FULLY AUTOMATED DATA PIPELINE THAT COLLECTS AND PROCESSES ALL THE INFORMATION WE NEED.
IS IT A GIANT HOUSE OF CARDS BUILT FROM RANDOM SCRIPTS THAT WILL ALL COMPLETELY COLLAPSE THE MOMENT ANY INPUT DOES ANYTHING WEIRD?
IT... MIGHT NOT BE.
I GUESS THAT'S SOMETH-
WHOOPS, JUST COLLAPSED. HANG ON, I CAN PATCH IT.

Four tables

Ran 30+ reports and organized into four tables

1. Characteristics
2. Activity
3. Test scores
4. Outcomes

Allows me to do any analysis I need and not rely on specific reports

Characteristics

- ID
- name
- race
- gender
- program
- program start date

Activity

```
## # A tibble: 1,500 x 3
##       id activity        date
##    <int> <chr>           <date>
##  1    76 education       2017-04-21
##  2    58 case management 2017-05-02
##  3    62 case management 2018-01-19
##  4    87 case management 2017-11-26
##  5    63 case management 2015-09-21
##  6    74 employment      2016-02-13
##  7    62 employment      2016-10-13
##  8    50 employment      2019-10-13
##  9    87 case management 2018-02-20
## 10     7 employment      2018-11-05
## # . with 1,490 more rows
```

Test scores

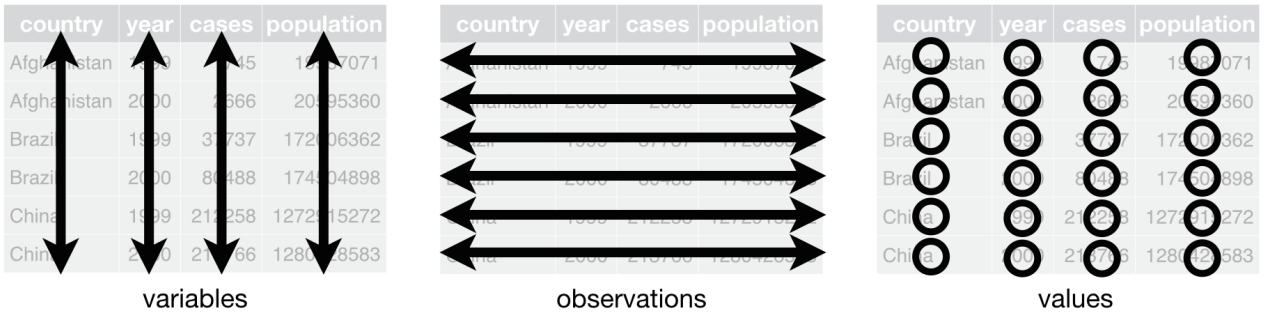
```
## # A tibble: 200 x 4
##       id test  test_date  score
##    <int> <chr> <date>     <dbl>
##  1    92 TABE  2018-01-01 27.6
##  2    90 GED   2018-04-18 40.1
##  3    52 GED   2017-09-10 34.1
##  4    93 TABE  2019-10-31 71.1
##  5    76 GED   2016-06-23 73.2
##  6    23 GED   2018-12-06 38.0
##  7    23 TABE  2015-07-12 44.0
##  8    13 GED   2018-06-16 50.6
##  9    19 GED   2019-08-17  3.78
## 10    68 GED   2018-06-23 17.2
## # . with 190 more rows
```

Outcomes

```
## # A tibble: 50 x 3
##       id outcome_date outcome
##    <int> <date>       <chr>
##  1    24 2019-06-06   education
##  2    65 2016-01-30   education
##  3     4 2018-10-10   employment
##  4    73 2017-07-02   education
##  5    36 2018-10-07   employment
##  6    99 2016-06-16   employment
##  7    63 2015-05-31   education
##  8    63 2017-04-22   employment
##  9     8 2019-04-21   education
## 10    10 2016-12-31   employment
## # . with 40 more rows
```

Tidy data

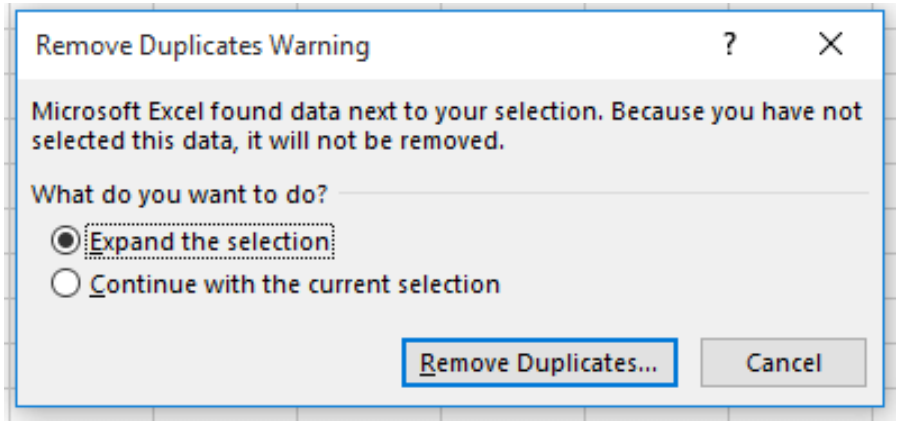
https://vita.had.co.nz/papers/tidy-data.pdf



- Recreated relational databases that worked for me
- If you can't use code, can still get data in this format
  - load into Tableau, Power BI, etc. for reproducible analysis

Why is code helpful?

Provenance



https://www.youtube.com/watch?v=cpbtcsGE0OA

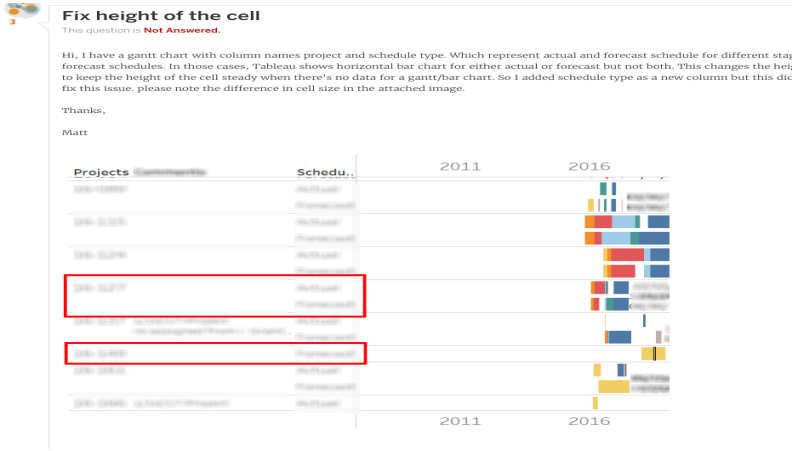
Reproducible

- Shareable
- Modifiable
- Updatable

Plain Text

Copy and paste

non-pasteable



Fix height of the cell

This question is Not Answered.

Hi, I have a gantt chart with column names project and schedule type. Which represent actual and forecast schedule for different stag forecast schedules. In those cases, Tableau shows horizontal bar chart for either actual or forecast but not both. This changes the hei to keep the height of the cell steady when there's no data for a gantt/bar chart. So I added schedule type as a new column but this dic fix this issue. please note the difference in cell size in the attached image.

Thanks,

Matt

Projects Schedu..

2011 2016

2011 2016

pasteable

1262

By using the `merge` function and its optional parameters:

Inner join: `merge(df1, df2)` will work for these examples because R automatically joins the frames by common variable names, but you would most likely want to specify `merge(df1, df2, by = "CustomerId")` to make sure that you were matching on only the fields you desired. You can also use the `by.x` and `by.y` parameters if the matching variables have different names in the different data frames.

Outer join: `merge(x = df1, y = df2, by = "CustomerId", all = TRUE)`

Left outer: `merge(x = df1, y = df2, by = "CustomerId", all.x = TRUE)`

Right outer: `merge(x = df1, y = df2, by = "CustomerId", all.y = TRUE)`

Cross join: `merge(x = df1, y = df2, by = NULL)`

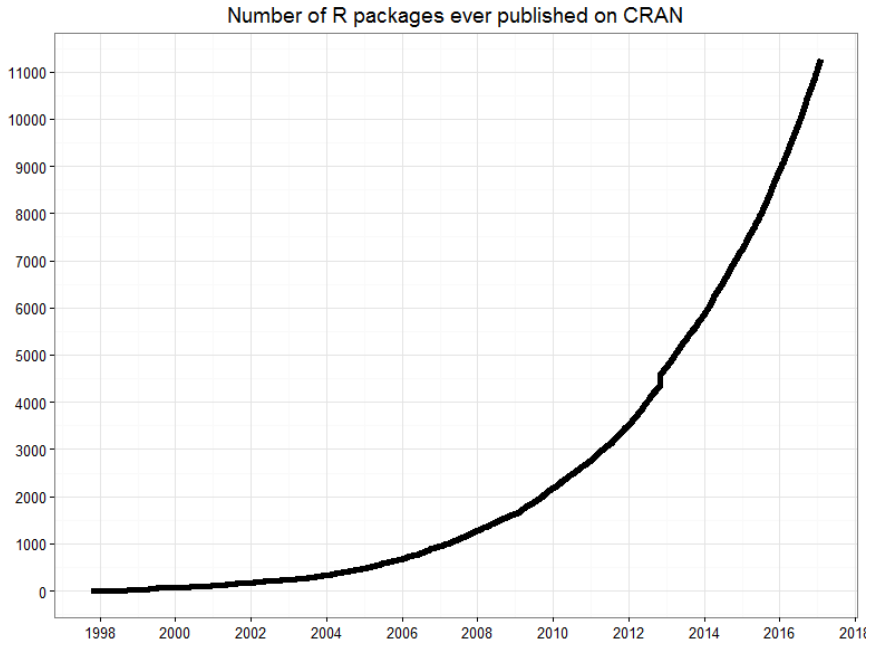
~~Just as with the inner join, you would probably want to explicitly pass "CustomerId" to R as the matching variable.~~ I think it's almost always best to explicitly state the identifiers on which you want to merge; it's safer if the input data.frames change unexpectedly and easier to read later on.

You can merge on multiple columns by giving `by` a vector, e.g., `by = c("CustomerId", "OrderId")`.

If the column names to merge on are not the same, you can specify, e.g., `by.x = "CustomerId_in_df1", by.y = "CustomerId_in_df2"` where `CustomerId_in_df1` is the name of the column in the first data frame and `CustomerId_in_df2` is the name of the column in the second data frame. (These can also be vectors if you need to merge on multiple columns.)

Google / www.stackoverflow.com

Open Source


Number of R packages ever published on CRAN
11000
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
1998
2000
2002
2004
2006
2008
2010
2012
2014
2016

So what?

Go beyond counting

Ask different kinds of questions

1. What did … ?
2. Why did … ?
3. What will … ?

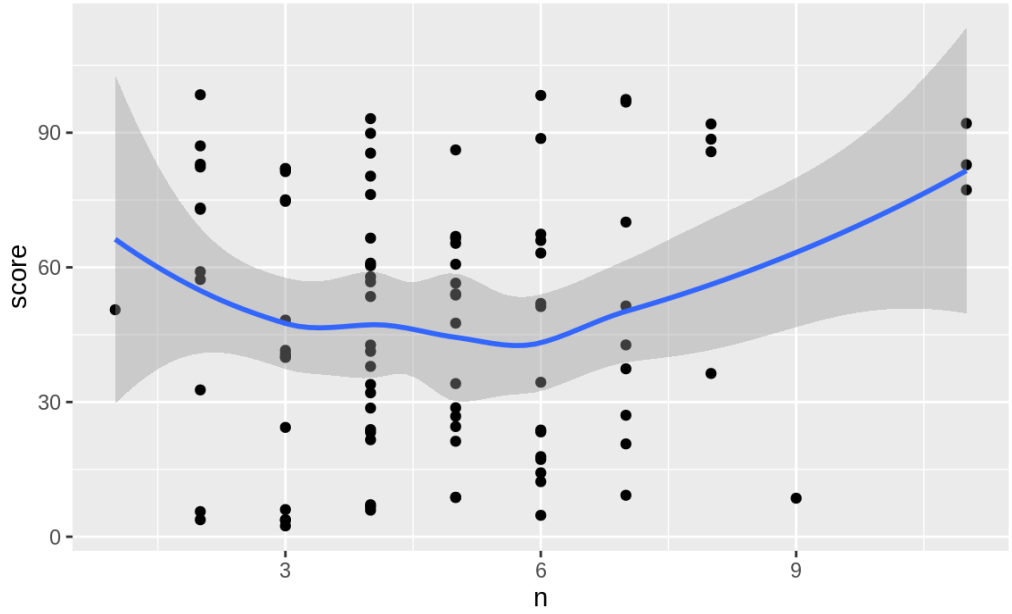
Look at relationships

```
library(tidyverse)

test_cm <- act %>%
    filter(activity == "case management") %>%
    count(id) %>%
    inner_join(test, by = "id") %>%
    filter(test == "GED")

test_cm
```

```
## # A tibble: 98 x 5
##       id     n test  test_date  score
##    <int> <int> <chr> <date>     <dbl>
##  1     2     3 GED   2019-10-21 40.8
##  2     3     7 GED   2019-05-17 97.4
##  3     6     8 GED   2016-12-18 88.6
##  4     7     6 GED   2017-06-21 52.0
##  5     8     4 GED   2015-10-18 60.9
##  6     8     4 GED   2015-11-10 41.3
##  7     9     4 GED   2019-07-07  5.92
##  8     9     4 GED   2016-02-24 66.5
##  9    10     5 GED   2019-08-05 56.5
```

Look at relationships

```
test_cm %>%
    ggplot(aes(n, score)) +
        geom_point() +
        geom_smooth()
```



Regression

```
cm_model <- lm(score ~ n, data = test_cm)

summary(cm_model)
```

```
##
## Call:
## lm(formula = score ~ n, data = test_cm)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -46.426 -24.733   0.883  27.176  53.421
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   42.176      7.442   5.667 1.52e-07 ***
## n              1.426      1.437   0.993    0.323
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 28.87 on 96 degrees of freedom
## Multiple R-squared:  0.01016,    Adjusted R-squared:  -0.0001525
## F-statistic: 0.9852 on 1 and 96 DF,  p-value: 0.3234
```

Prediction

```
library(randomForest)

randomForest(outcome ~ ., data = outcomes)
```

Where to start with coding?

Languages

- SQL and (R or Python)

Resources

- R for Data Science and learning community
- Google
- Twitter
- SQL is easiest to learn at work

Focus on questions that matter

- Incentives are not designed to critically examine our work
- Using code and good data organization principles allows us to focus energy on asking deeper questions and investigating relationships

Contact Info

Joe Ciesielski

jtcies@gmail.com