Effective Programming Practices for Economists

Data management with pandas

Data management: Tips, tricks, and advanced topics

Janoś Gabler and Hans-Martin von Gaudecker

Motivation

- When to select the sample for analysis?
- When to use metadata in a programmatic way?
- What to do when combining variables into one?
- What if variables change over time?

Selecting the sample for analysis

- Upfront restrictions you will not even touch in robustness checks: Very beginning or at the end of the data management pipeline
- Other restrictions: Missing values in covariates, robustness checks, etc.
  - Impose at the end of the data management pipeline
  - Or set up a custom loading function that you always use, which takes the path to the data and the sample definition as arguments
  - Always be explicit!

Using metadata programmatically

nlsy_name	readable_name	label
C0000100	childid	id code of child
C0564000	anxiety_mood	ch has sud chgs in mood/feelng
C0564100	anxiety_complain	ch cmplns no one loves him/her
C0564400	anxiety_fearful	ch is too fearful or anxious
C0565300	anxiety_worthless	ch feels worthless or inferior
C0565900	anxiety_sad	child is unhappy/sad/depressed
C0780800	anxiety_mood	ch has sud chgs in mood/feelng
…	…	…

Read into DataFrame called `metadata` , with `nlsy_name` as the index.

Using metadata programmatically

```python
bpi_subscale = "anxiety"
bpi_subscale_items = {
    new: old
    for old, new in metadata["readable_name"].items(
    if new.startswith(f"{bpi_subscale}_")
}
for old, new in bpi_subscale_items.items():
    df[new] = clean_item(raw[old])
```

```python
df["anxiety_mood"] = clean_item(raw["C0564000"])
df["anxiety_complain"] = clean_item(raw["C0564100"])
df["anxiety_fearful"] = clean_item(raw["C0564400"])
df["anxiety_worthless"] = clean_item(raw["C0565300"]
df["anxiety_sad"] = clean_item(raw["C0565900"])
df["anxiety_mood"] = clean_item(raw["C0780800"])
```

How to combine variables into one?

SOEP asks for transfer receipt in

- Person-level data (pl)
- Personal calendar data (pkal)

Would have three variables:

```
df["transfer_receipt_pl"]
df["transfer_receipt_pkal"]
df["transfer_receipt"]
```

where the third is obtained by a function taking the first two as arguments

What if variables change over time?

- Panel stability vs. questions that do not yield the expected information (any more)
- End up with changes in variable coding over time
- Two strategies:

1. One time series variable, harmonize by case distinctions in a single cleaning function
2. Keep differently-named variables for each time point with a cleaning function each, then combine into one variable (same strategy as in the previous slide)