Effective Programming Practices for Economists

Data management with pandas

Imperative data cleaning

Janoś Gabler and Hans-Martin von Gaudecker

Survey of course participants

	Q001	Q002	Q003
0	strongly disagree	agree	python
1	strongly agree	strongly agree	Python
2	-77	disagree	R
3	agree	-77	Python
4	-99	-99	Python
5	nan	strongly agree	Python
6	neutral	strongly agree	Python
7	disagree	agree	python
8	strongly agree	-99	PYTHON
9	agree	-99	Ypthon

- Q001: I am a coding genius
- Q002: I learned a lot
- Q003: What is your favourite language?

- -77 not readable
- -99 no reply

- All variables are strings after running

```
df = pd.read_csv("survey.csv")
```

Cleaning 1: New names

```
new_names = {
    "Q001": "coding_genius",
    "Q002": "learned_a_lot",
    "Q003": "favorite_language",
}
df = df.rename(columns=new_names)
```

Cleaning 2: Agreement scales

```
for var in ["coding_genius", "learned_a_lot"]:
    df[var] = df[var].replace({"-77": pd.NA, "-99": pd.NA})
    categories = ["strongly disagree", "disagree", "neutral", "agree", "strongly agree"]
    dtype = pd.CategoricalDtype(categories=categories, ordered=True)
    df[var] = df[var].astype(dtype)
```

Cleaning 3: Favorite language

```
df["favorite_language"] = df["favorite_language"].replace({"-77": pd.NA, "-99": pd.NA})
df["favorite_language"] = df["favorite_language"].str.lower().str.strip()
df["favorite_language"] = df["favorite_language"].replace("ypthon", "python")
df["favorite_language"] = df["favorite_language"].astype(pd.CategoricalDtype())
```

Result

	coding_genius	learned_a_lot	favorite_language
0	strongly disagree	agree	python
1	strongly agree	strongly agree	python
2	NaN	disagree	r
3	agree	NaN	python
4	NaN	NaN	python
5	NaN	strongly agree	python
6	neutral	strongly agree	python
7	disagree	agree	python
8	strongly agree	NaN	python
9	agree	NaN	python

```
>>> df.dtypes
coding_genius     category
learned_a_lot     category
favorite_language category
dtype: object

>>> df["coding_genius"].cat.categories
[
    'strongly disagree',
    'disagree',
    'neutral',
    'agree',
    'strongly agree'
]

>>> df["favorite_language"].cat.categories
['python', 'r']
```

Result is fine, but...

```
new_names = {
    "Q001": "coding_genius",
    "Q002": "learned_a_lot",
    "Q003": "favorite_language",
}
df = df.rename(columns=new_names)

# Clean the two variables with agreement scales
for var in ["coding_genius", "learned_a_lot"]:
    df[var] = df[var].replace({"-77": pd.NA, "-99": pd.NA})
    categories = ["strongly disagree", "disagree", "neutral", "agree", "strongly agree"]
    dtype = pd.CategoricalDtype(categories=categories, ordered=True)
    df[var] = df[var].astype(dtype)

# Clean the favorite language variable
df["favorite_language"] = df["favorite_language"].replace({"-77": pd.NA, "-99": pd.NA})
df["favorite_language"] = df["favorite_language"].str.lower().str.strip()
df["favorite_language"] = df["favorite_language"].replace("ypthon", "python")
df["favorite_language"] = df["favorite_language"].astype(pd.CategoricalDtype())
```

Result is fine, but...

- The variables inside `df` change many times but keep their name
- There are many invalid intermediate states of df where variables already have their final names. This is especially dangerous if code is spread across multiple cells, let alone multiple files.
- The global namespace is cluttered with helper variables like `var`, `categories`, and `dtype`.
- The code has no natural structure. We need comments to get some orientation.
- The only way to re-use code across variables is to include it in a loop. Hence, the two agreement questions have to be cleaned at the same time, whether they are related or not.
- We either had to repeat the name `favorite_language` multiple times or use method chaining. Which is hard to read and debug.