?

defend your code

Virtual Environments

What is a virtual environment?

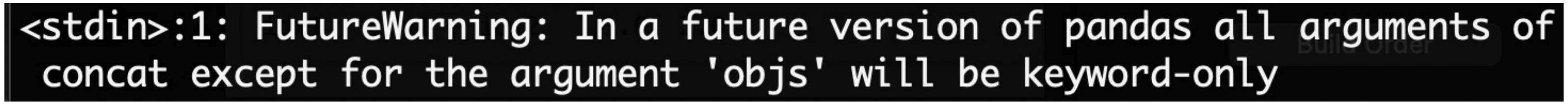
A semi-isolated python environment -> you cannot access packages (libraries and their dependencies) installed in other environments.

packages are installed inside a project-specific virtual environment folder (not added to general python path)

If something goes wrong with one environment, it does not affect the others.

Why environments?

Avoid errors when working on multiple projects / updating your Python packages

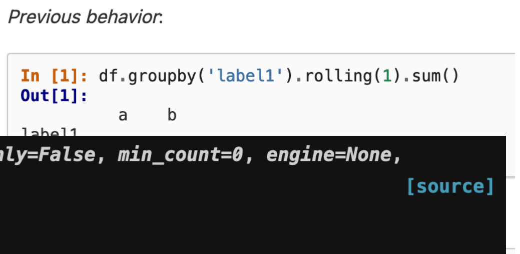


```
<stdin>:1: FutureWarning: In a future version of pandas all arguments of
 concat except for the argument 'objs' will be keyword-only
```

—> if you keep updating your python packages, you will run into issues

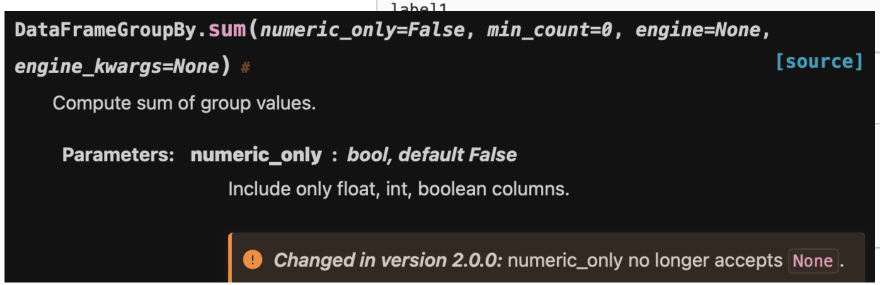
code errors

unexpected results



Previous behavior:

```
In [1]: df.groupby('label1').rolling(1).sum()
Out[1]:
            a     b
label1
```



```
DataFrameGroupBy.sum(numeric_only=False, min_count=0, engine=None,
engine_kwargs=None) #                                        [source]
```

Compute sum of group values.

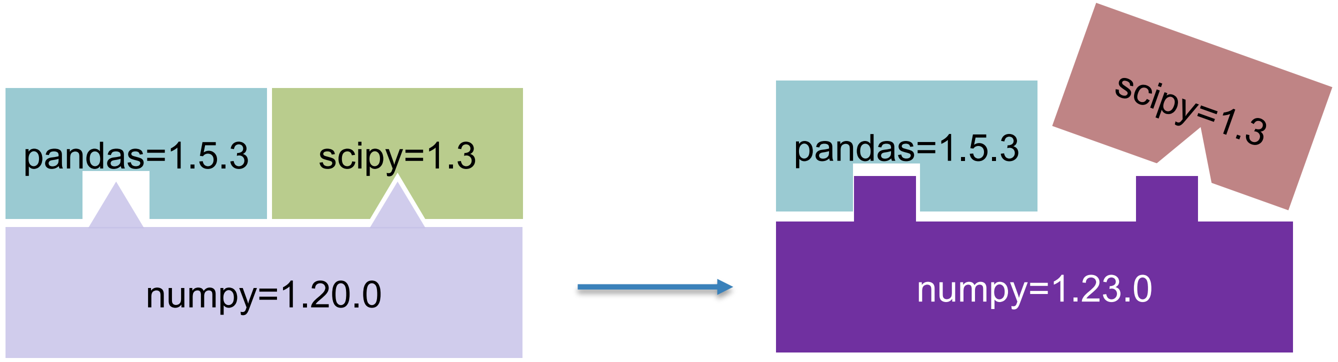
Parameters: **numeric_only** : ***bool, default False***

Include only float, int, boolean columns.

Changed in version 2.0.0: numeric_only no longer accepts `None`.

Why environments?

Avoid importing errors when working on multiple projects / updating your Python packages



Virtual Environments

Create and activate a virtual environment following the directions in **Exercise 5a Virtual Environments.md**

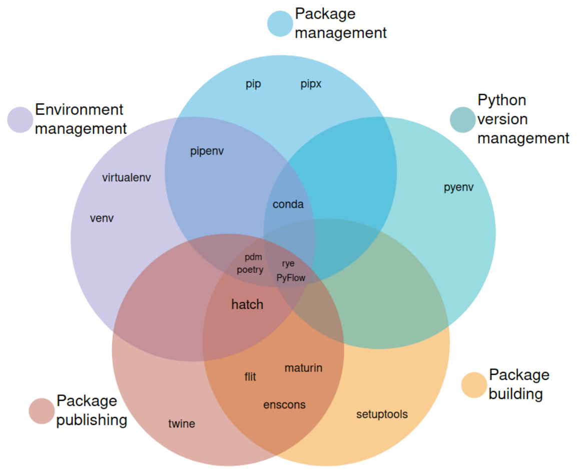
Notes

Environment Managers

venv - current standard recommended by Python

poetry - super useful (if it works)
pyenv - multiple different Pythons

etc



a description of the chaos:
https://chriswarrick.com/blog/2023/01/15/how-to-improve-python-packaging/
https://alpopkes.com/posts/python/packaging_tools/

Why environments?

Avoid errors when working on multiple projects / updating your Python packages

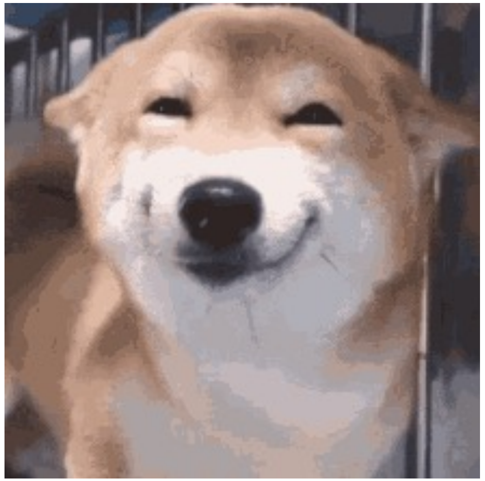
Increased reproducibility: give yourself / other people the exact instructions **and** tools to run your code (cluster, collaboration)

Additional advantages

The main advantage is that you can start over if something goes wrong and you have broken nothing!

Our goal

1. Local importing
   → review and best practices
2. Editable installations
   → avoid importing errors
3. Python package structure and code organization
   → organize folders and files in a standardized way
4. Environments
   → avoid and alleviate package installation problems
5. Documentation and formatting tools
   → make code more readable and usable



? readability