AN INTRODUCTION TO DATALAD

WITH A FOCUS ON ML APPLICATION

Adina Wagner
@AdinaKrik

JÜLICH
Forschungszentrum

Psychoinformatics lab,
Institute of Neuroscience and Medicine, Brain & Behavior (INM-7)
Research Center Jülich

Slides: https://github.com/datalad-handbook/course/

ACKNOWLEDGEMENTS

Software

- Michael Hanke (INM-7)
- Yaroslav Halchenko
- Joey Hess (git-annex)
- Kyle Meyer
- Benjamin Poldrack (INM-7)
- *26 additional contributors*

Documentation project

- Michael Hanke (INM-7)
- Laura Waite (INM-7)
- *28 additional contributors*

Funders

NSF 1429999

germany-usa

SPONSORED BY THE
Federal Ministry of Education and Research
BMBF 01GQ1411

JÜLICH Forschungszentrum

EUROPEAN UNION
European Regional Development Fund

cbbs
center for behavioral brain sciences

SACHSEN-ANHALT
Ministerium für Wissenschaft und Wirtschaft

Collaborators

Human Brain Project

CONP PCNO

VirtualBrainCloud

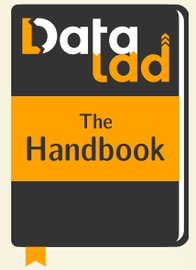
OpenNEURO

cbrain

brainlife.io

RESOURCES AND FURTHER READING

Comprehensive user documentation in the DataLad Handbook (handbook.datalad.org)



- High-level function/command overviews, Installation, Configuration, Cheatsheet

---

- Narrative-based code-along course
- Independent on background/skill level, suitable for data management novices

---

- Step-by-step solutions to common data management problems, like how to make a reproducible paper

RESOURCES AND FURTHER READING

- Specifically relevant handbook sections to ML:
    - DataLad ML example: handbook.datalad.org/r.html?ml-usecase
    - Code list with further links: handbook.datalad.org/r.html?FZJmlcode

- Not-text-based resources:
    - DataLad Youtube channel with talks and tutorials

QUESTIONS/INTERACTION

- Ask questions via chat or by speaking up at any point
- Happy to discuss specific use cases at the end
- Reach out to us via Matrix or GitHub at any later point

LIVE POLLING SYSTEM

http://etc.ch/y8uM

Direct Poll

LIVE CODING

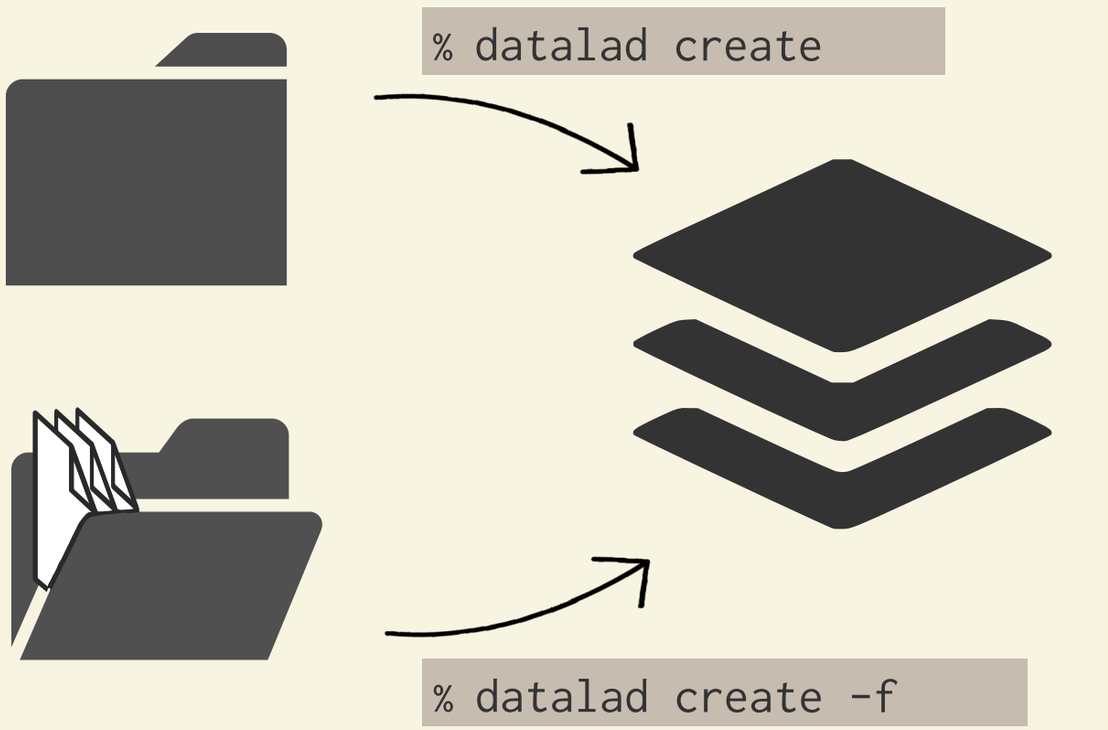
- Live-demonstration of DataLad examples and workflows throughout the talk
- Code along with copy-paste code snippets: handbook.datalad.org/r.html?FZJmlcode
- Requirements:
    - DataLad version >= 0.12 (installation instructions at handbook.datalad.org)
    - For containerized analyses: DataLad extension datalad-containers (available via pip) + Singularity

- A command-line tool, available for all major operating systems (Linux, macOS/OSX, Windows), MIT-licensed
- Build on top of Git and Git-annex
- **Allows...**
  ... version-controlling arbitrarily large content
  version control data and software alongside to code!
  ... transport mechanisms for sharing and obtaining data
  consume and collaborate on data (analyses) like software
  ... (computationally) reproducible data analysis
  Track and share provenance of all digital objects
  ... and *much* more
- Completely domain-agnostic

CORE CONCEPTS & FEATURES

EVERYTHING HAPPENS IN DATALAD DATASETS

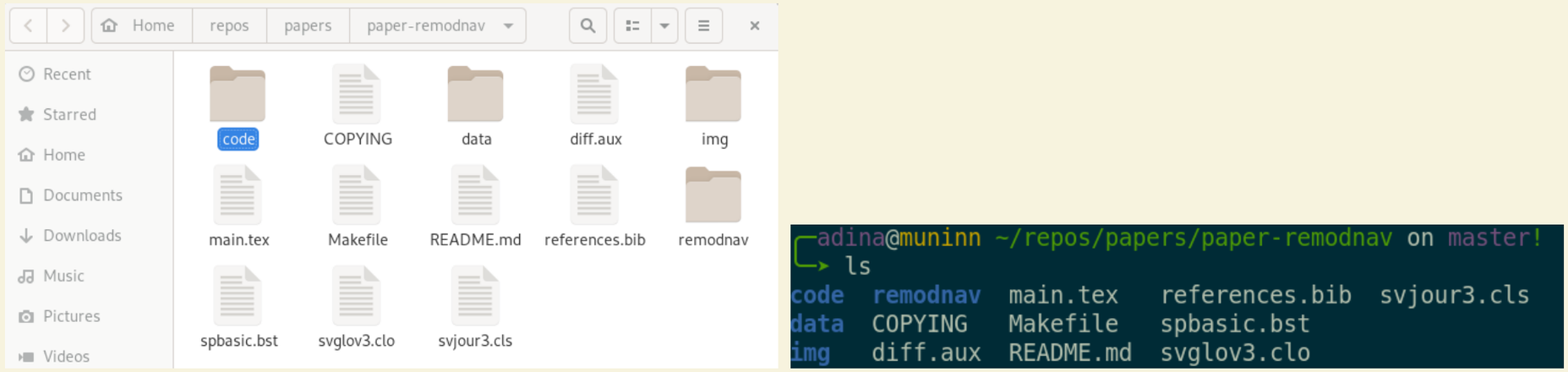
create new, empty datasets to populate...



.... or transform existing directories into datasets

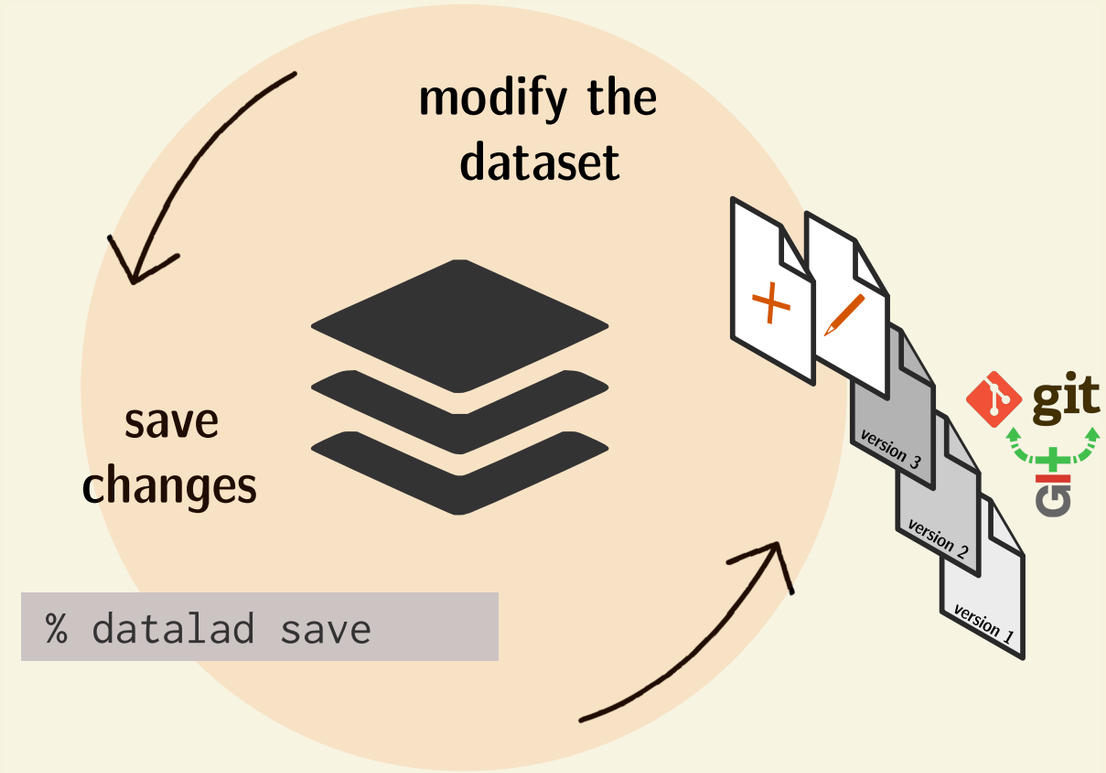
DATASET = GIT/GIT-ANNEX REPOSITORY

- content agnostic
- no custom data structures
- complete decentralization
- Looks and feels like a directory on your computer:

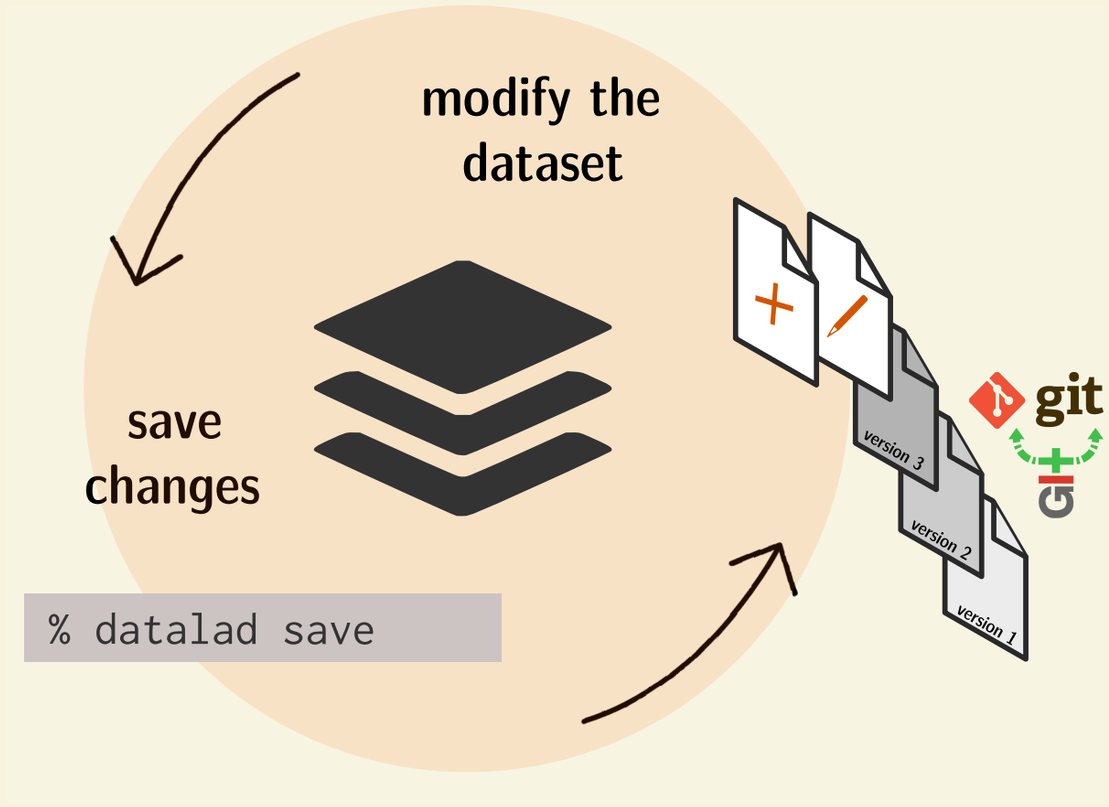


File viewer and terminal view of a DataLad dataset

VERSION CONTROL ARBITRARILY LARGE FILES

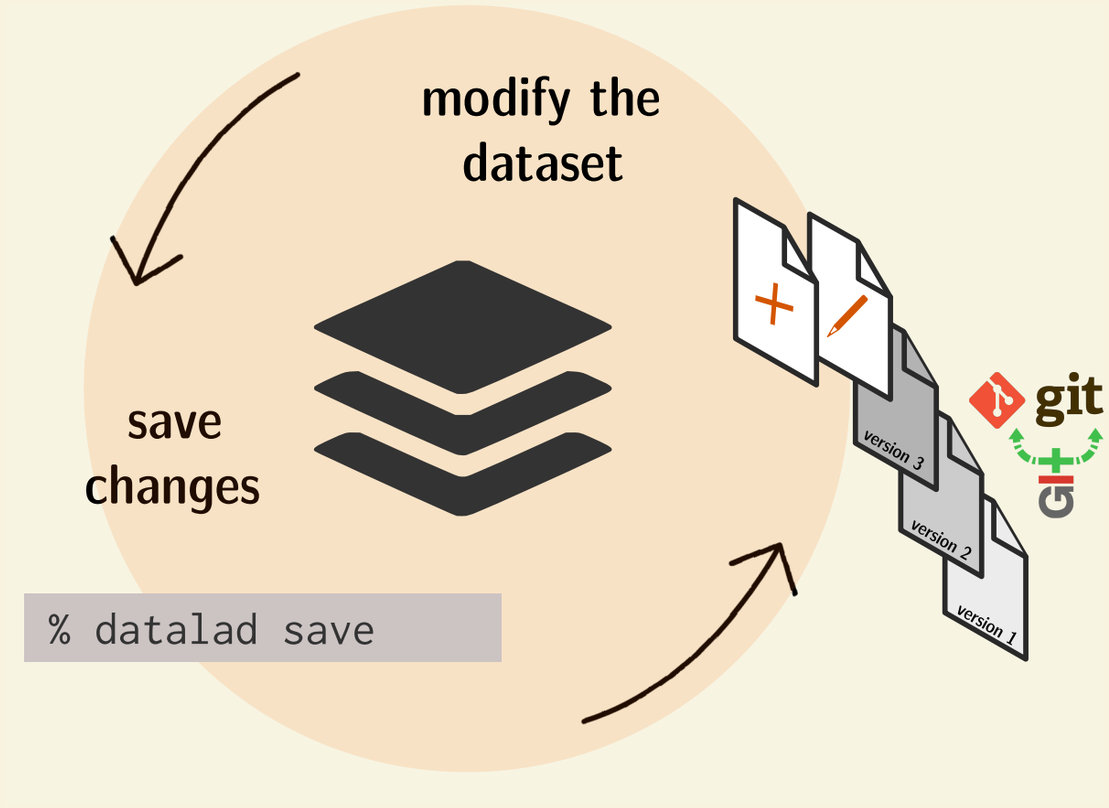


VERSION CONTROL ARBITRARILY LARGE FILES



Stay flexible:

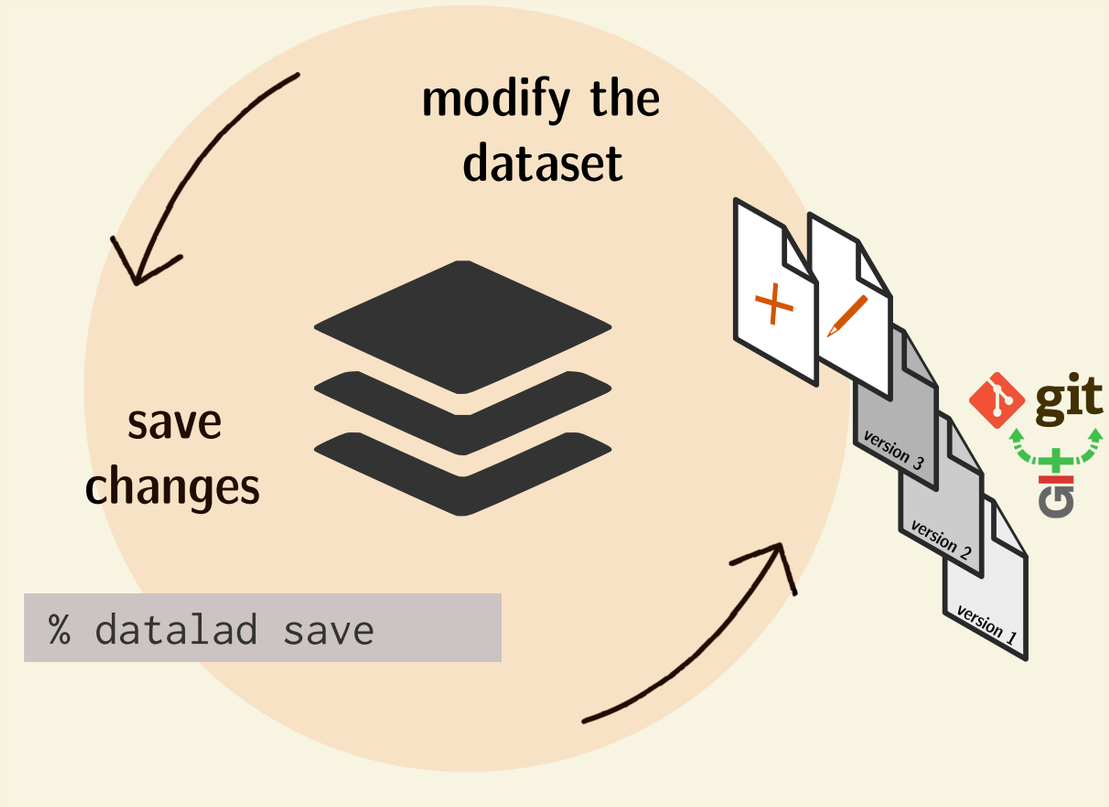
VERSION CONTROL ARBITRARILY LARGE FILES



Stay flexible:

- Non-complex DataLad core API (easy for data management novices)

VERSION CONTROL ARBITRARILY LARGE FILES



Stay flexible:

- Non-complex DataLad core API (easy for data management novices)
- Pure Git or git-annex commands (for regular Git or git-annex users, or to use specific functionality)

USE A DATASETS' HISTORY

```
2020-06-05 10:58 +0200 Adina Wagner  M─┐ [master] {upstream/master} {upstream/HEAD} Merge pull request #12 from psychoinformatics-d
2020-06-05 08:24 +0200 Adina Wagner  │ o [finalround] {upstream/finalround} add results from computing with mean instead of median
2020-06-05 09:09 +0200 Michael Hanke o │ Change wording, clarify comment
2020-06-05 07:26 +0200 Michael Hanke M─┼─┐ Merge remote-tracking branch 'gh-mine/finalround'
2020-05-28 16:39 +0200 Asim H Dar    │ o─┘ Added datalad.get() so S2SRMS() pulls data and can run standalone
2020-05-18 08:25 +0200 Adina Wagner  │ o {gh-asim/finalround} S2SRMS: example implementation of the S2SRMS method suggested by R2
2020-05-01 17:38 +0200 Adina Wagner  │ o Minor edits as suggested by reviewer 2
2020-05-29 09:04 +0200 Adina Wagner  M─┼─┐ Merge pull request #13 from psychoinformatics-de/adswa-patch-1
2020-05-24 09:15 +0200 Adina Wagner  │ │ o {upstream/adswa-patch-1} Fix installation instructions
2020-05-24 09:53 +0200 Adina Wagner  M─┼─┼─┐ Merge pull request #14 from psychoinformatics-de/bf-data
2020-05-24 09:33 +0200 Adina Wagner  │ │ │ o [bf-data] One-time datalad import
2020-05-24 09:32 +0200 Adina Wagner  │ │ │ o install and get relevant subdataset data
2020-03-18 10:19 +0100 Michael Hanke M─┼─┼─┘ Merge pull request #8 from psychoinformatics-de/adswa-patch-1
2019-12-19 10:22 +0100 Adina Wagner  │ o {gh-asim/adswa-patch-1} add sklearn to requirements
2020-03-18 10:13 +0100 Michael Hanke o │ Tune new figure caption
2020-03-18 10:03 +0100 Michael Hanke M─┼─┐ Merge pull request #11 from ElectronicTeaCup/revision_2
2020-03-18 09:59 +0100 Adina Wagner  │ M─┼─┐ [revision_2] {gh-asim/revision_2} Merge branch 'revision_2' of github.com:ElectronicTe
2020-03-18 09:58 +0100 Michael Hanke │ │ o │ Last detections
2020-03-18 09:59 +0100 Adina Wagner  │ │ o │ name parameter in caption
```

Date Author Change summary (commit message)

USE A DATASETS' HISTORY

```
2020-06-05 10:58 +0200 Adina Wagner  M─┐ [master] {upstream/master} {upstream/HEAD} Merge pull request #12 from psychoinformatics-d
2020-06-05 08:24 +0200 Adina Wagner  │ o [finalround] {upstream/finalround} add results from computing with mean instead of median
2020-06-05 09:09 +0200 Michael Hanke o │ Change wording, clarify comment
2020-06-05 07:26 +0200 Michael Hanke M─┼─┐ Merge remote-tracking branch 'gh-mine/finalround'
2020-05-28 16:39 +0200 Asim H Dar    │ o─┘ Added datalad.get() so S2SRMS() pulls data and can run standalone
2020-05-18 08:25 +0200 Adina Wagner  │ o {gh-asim/finalround} S2SRMS: example implementation of the S2SRMS method suggested by R2
2020-05-01 17:38 +0200 Adina Wagner  │ o Minor edits as suggested by reviewer 2
2020-05-29 09:04 +0200 Adina Wagner  M─┼─┐ Merge pull request #13 from psychoinformatics-de/adswa-patch-1
2020-05-24 09:15 +0200 Adina Wagner  │ │ o {upstream/adswa-patch-1} Fix installation instructions
2020-05-24 09:53 +0200 Adina Wagner  M─┼─┼─┐ Merge pull request #14 from psychoinformatics-de/bf-data
2020-05-24 09:33 +0200 Adina Wagner  │ │ │ o [bf-data] One-time datalad import
2020-05-24 09:32 +0200 Adina Wagner  │ │ │ o install and get relevant subdataset data
2020-03-18 10:19 +0100 Michael Hanke M─┼─┼─┘ Merge pull request #8 from psychoinformatics-de/adswa-patch-1
2019-12-19 10:22 +0100 Adina Wagner  │ o {gh-asim/adswa-patch-1} add sklearn to requirements
2020-03-18 10:13 +0100 Michael Hanke o │ Tune new figure caption
2020-03-18 10:03 +0100 Michael Hanke M─┼─┐ Merge pull request #11 from ElectronicTeaCup/revision_2
2020-03-18 09:59 +0100 Adina Wagner  │ M─┼─┐ [revision_2] {gh-asim/revision_2} Merge branch 'revision_2' of github.com:ElectronicTe
2020-03-18 09:58 +0100 Michael Hanke │ │ o │ Last detections
2020-03-18 09:59 +0100 Adina Wagner  │ │ o │ name parameter in caption
```

Date Author Change summary (commit message)

- reset your dataset (or subset of it) to a previous state,

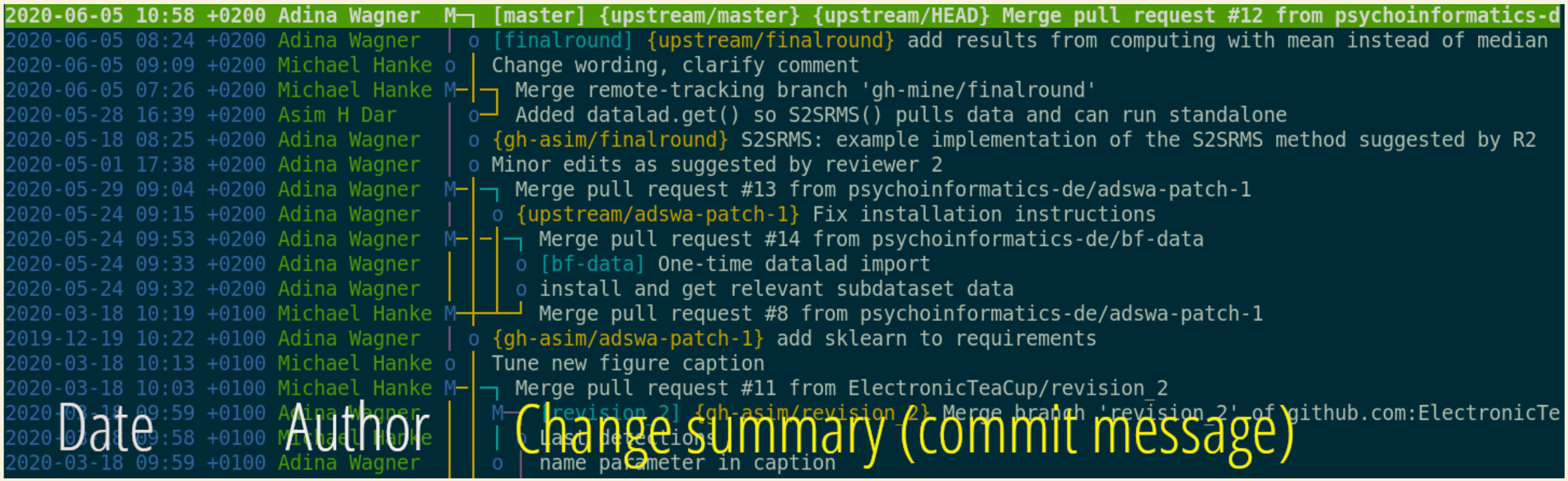
USE A DATASETS' HISTORY

```
2020-06-05 10:58 +0200 Adina Wagner  M─┐ [master] {upstream/master} {upstream/HEAD} Merge pull request #12 from psychoinformatics-d
2020-06-05 08:24 +0200 Adina Wagner  │ o [finalround] {upstream/finalround} add results from computing with mean instead of median
2020-06-05 09:09 +0200 Michael Hanke o │ Change wording, clarify comment
2020-06-05 07:26 +0200 Michael Hanke M─┼─┐ Merge remote-tracking branch 'gh-mine/finalround'
2020-05-28 16:39 +0200 Asim H Dar    │ o─┘ Added datalad.get() so S2SRMS() pulls data and can run standalone
2020-05-18 08:25 +0200 Adina Wagner  │ o {gh-asim/finalround} S2SRMS: example implementation of the S2SRMS method suggested by R2
2020-05-01 17:38 +0200 Adina Wagner  │ o Minor edits as suggested by reviewer 2
2020-05-29 09:04 +0200 Adina Wagner  M─┼─┐ Merge pull request #13 from psychoinformatics-de/adswa-patch-1
2020-05-24 09:15 +0200 Adina Wagner  │ │ o {upstream/adswa-patch-1} Fix installation instructions
2020-05-24 09:53 +0200 Adina Wagner  M─┼─┼─┐ Merge pull request #14 from psychoinformatics-de/bf-data
2020-05-24 09:33 +0200 Adina Wagner  │ │ │ o [bf-data] One-time datalad import
2020-05-24 09:32 +0200 Adina Wagner  │ │ │ o install and get relevant subdataset data
2020-03-18 10:19 +0100 Michael Hanke M─┼─┼─┘ Merge pull request #8 from psychoinformatics-de/adswa-patch-1
2019-12-19 10:22 +0100 Adina Wagner  │ o {gh-asim/adswa-patch-1} add sklearn to requirements
2020-03-18 10:13 +0100 Michael Hanke o │ Tune new figure caption
2020-03-18 10:03 +0100 Michael Hanke M─┼─┐ Merge pull request #11 from ElectronicTeaCup/revision_2
2020-03-18 09:59 +0100 Adina Wagner  │ M─┼─┐ [revision_2] {gh-asim/revision_2} Merge branch 'revision_2' of github.com:ElectronicTe
2020-03-18 09:58 +0100 Michael Hanke │ │ o │ Last detections
2020-03-18 09:59 +0100 Adina Wagner  │ │ o │ name parameter in caption
```

Date Author Change summary (commit message)

- reset your dataset (or subset of it) to a previous state,
- revert changes or bring them back,

USE A DATASETS' HISTORY



- reset your dataset (or subset of it) to a previous state,
- revert changes or bring them back,
- find out what was done when, how, why, and by whom

USE A DATASETS' HISTORY

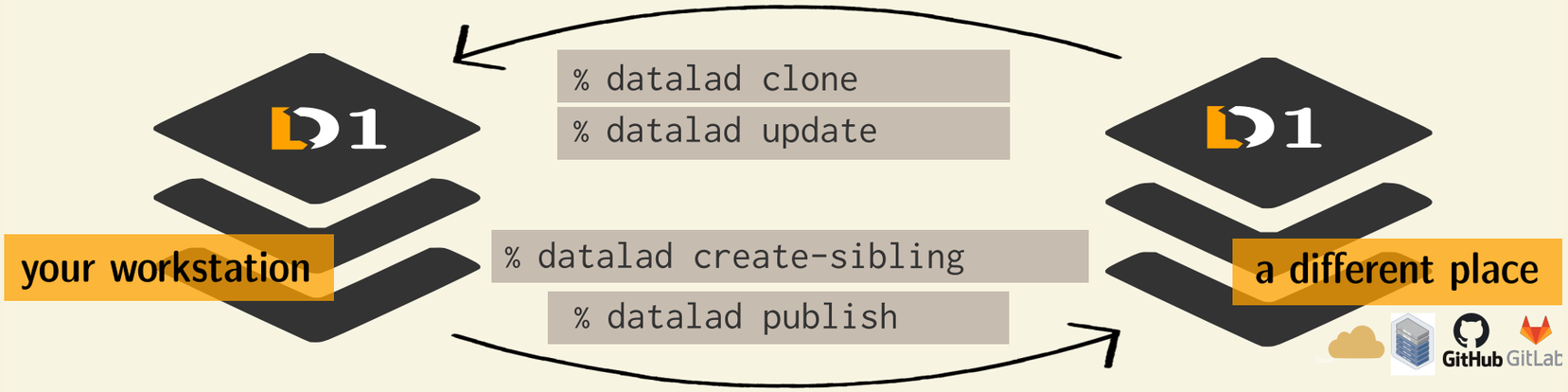
```
2020-06-05 10:58 +0200 Adina Wagner  M─┐ [master] {upstream/master} {upstream/HEAD} Merge pull request #12 from psychoinformatics-d
2020-06-05 08:24 +0200 Adina Wagner  │ o [finalround] {upstream/finalround} add results from computing with mean instead of median
2020-06-05 09:09 +0200 Michael Hanke o │ Change wording, clarify comment
2020-06-05 07:26 +0200 Michael Hanke M─┼─┐ Merge remote-tracking branch 'gh-mine/finalround'
2020-05-28 16:39 +0200 Asim H Dar    │ o─┘ Added datalad.get() so S2SRMS() pulls data and can run standalone
2020-05-18 08:25 +0200 Adina Wagner  │ o {gh-asim/finalround} S2SRMS: example implementation of the S2SRMS method suggested by R2
2020-05-01 17:38 +0200 Adina Wagner  │ o Minor edits as suggested by reviewer 2
2020-05-29 09:04 +0200 Adina Wagner  M─┼─┐ Merge pull request #13 from psychoinformatics-de/adswa-patch-1
2020-05-24 09:15 +0200 Adina Wagner  │ │ o {upstream/adswa-patch-1} Fix installation instructions
2020-05-24 09:53 +0200 Adina Wagner  M─┼─┼─┐ Merge pull request #14 from psychoinformatics-de/bf-data
2020-05-24 09:33 +0200 Adina Wagner  │ │ │ o [bf-data] One-time datalad import
2020-05-24 09:32 +0200 Adina Wagner  │ │ │ o install and get relevant subdataset data
2020-03-18 10:19 +0100 Michael Hanke M─┼─┼─┘ Merge pull request #8 from psychoinformatics-de/adswa-patch-1
2019-12-19 10:22 +0100 Adina Wagner  │ o {gh-asim/adswa-patch-1} add sklearn to requirements
2020-03-18 10:13 +0100 Michael Hanke o │ Tune new figure caption
2020-03-18 10:03 +0100 Michael Hanke M─┼─┐ Merge pull request #11 from ElectronicTeaCup/revision_2
2020-03-18 09:59 +0100 Adina Wagner  │ M─┼─┐ {revision_2] {gh-asim/revision_2} Merge branch 'revision_2' of github.com:ElectronicTe
2020-03-18 09:58 +0100 Michael Hanke │ │ o │ Last detections
2020-03-18 09:59 +0100 Adina Wagner  │ │ o │ name parameter in caption
```

Date Author Change summary (commit message)

- reset your dataset (or subset of it) to a previous state,
- revert changes or bring them back,
- find out what was done when, how, why, and by whom
- Identify precise versions: Use data in the most recent version, or the one from 2018, or...

CONSUME AND COLLABORATE

Consume existing datasets and stay up-to-date



Create sibling datasets to publish to or update from

MACHINE-READABLE, RE-EXECUTABLE PROVENANCE

capture the origin
of files obtained
from web sources

```
% datalad download-url
```

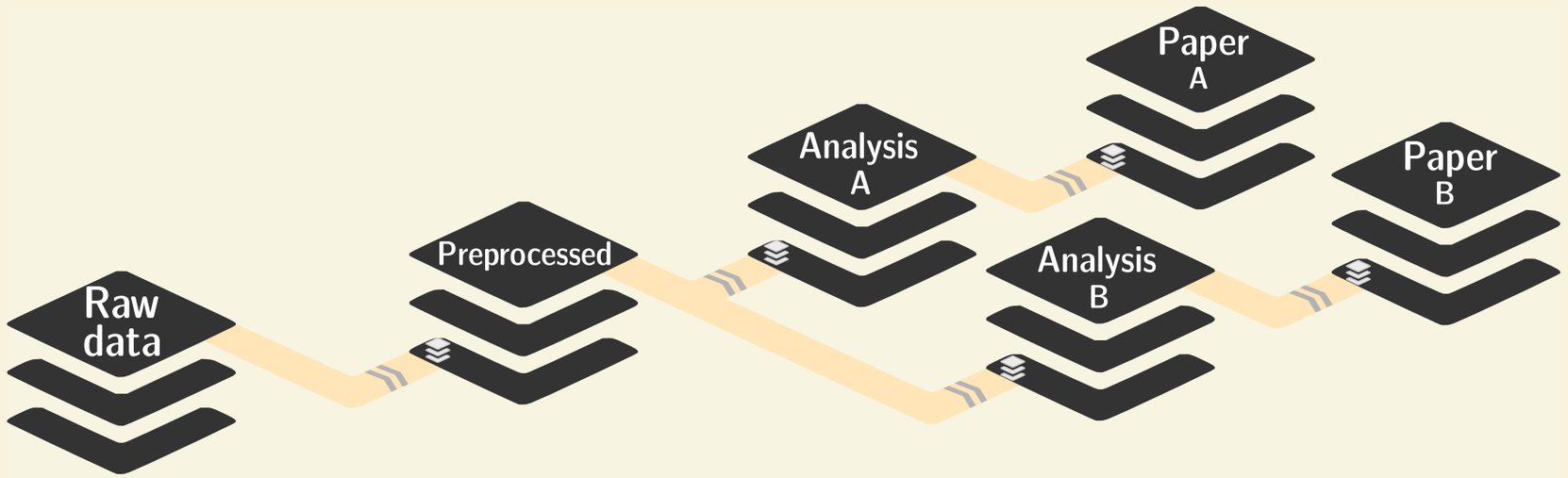
link input, code, containerized
software environments, and output,
or re-run previous executions


```
% datalad run
```

```
% datalad rerun
```

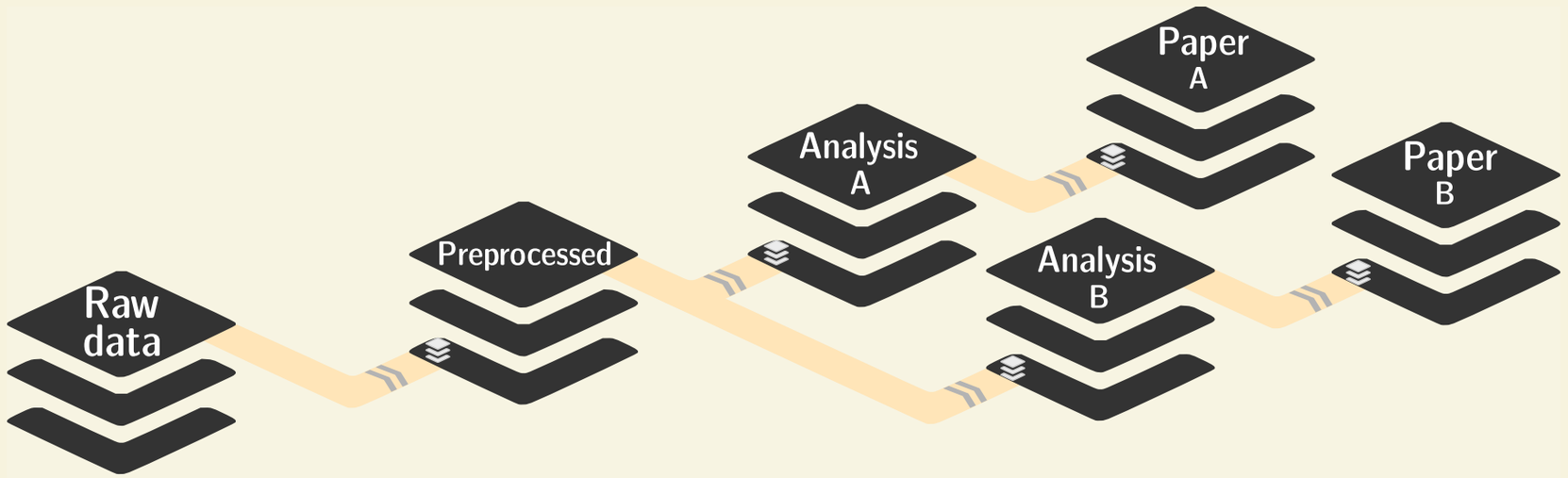
```
% datalad run-procedure
```

SEAMLESS NESTING AND DATASET LINKAGE



Nest modular datasets to create a linked hierarchy of datasets, and enable recursive operations throughout the hierarchy

SEAMLESS NESTING AND DATASET LINKAGE



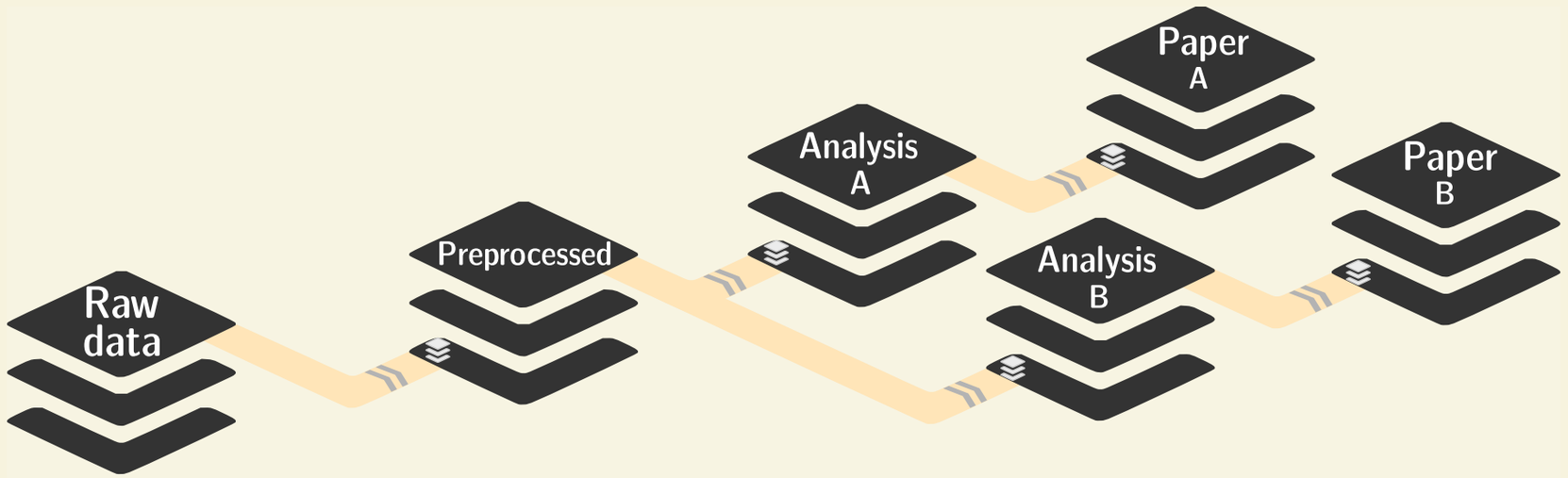
Nest modular datasets to create a linked hierarchy of datasets, and enable recursive operations throughout the hierarchy

- Overcomes scaling issues with large amounts of files

```
adina@bulk1 in /ds/hcp/super on git:master❱ datalad status --annex -r
15530572 annex'd files (77.9 TB recorded total size)
nothing to save, working tree clean
```

(github.com/datalad-datasets/human-connectome-project-openaccess)

SEAMLESS NESTING AND DATASET LINKAGE



Nest modular datasets to create a linked hierarchy of datasets, and enable recursive operations throughout the hierarchy

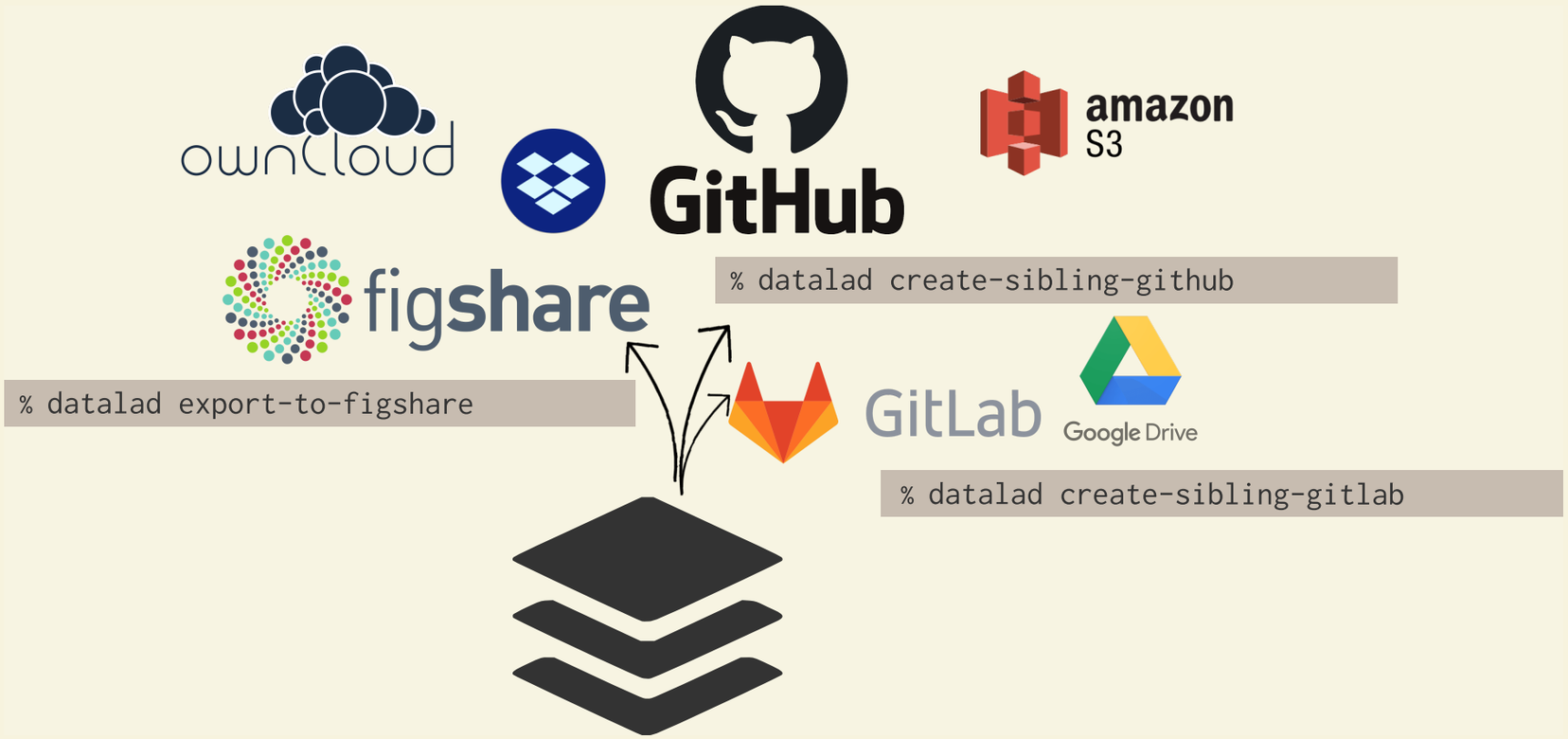
- Overcomes scaling issues with large amounts of files

```
adina@bulk1 in /ds/hcp/super on git:master❱ datalad status --annex -r
15530572 annex'd files (77.9 TB recorded total size)
nothing to save, working tree clean
```

(github.com/datalad-datasets/human-connectome-project-openaccess)

- Modularizes research components for transparency, reuse, and access management

THIRD PARTY INTEGRATIONS



Apart from **local computing infrastructure** (from private laptops to computational clusters), datasets can be hosted in major **third party repository hosting and cloud storage** services. More info: Chapter on Third party infrastructure.

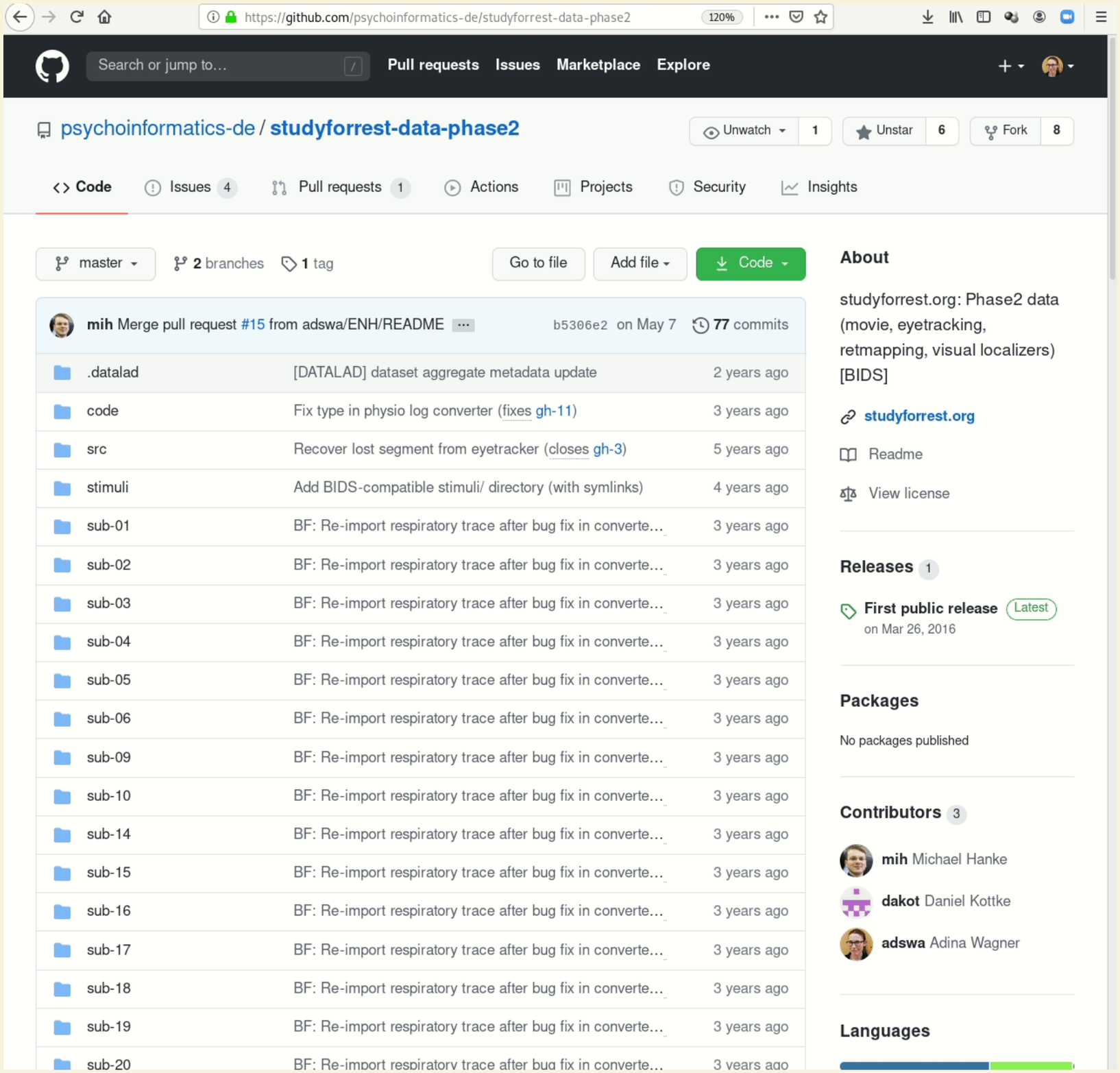
EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

- **Publish or consume datasets** via GitHub, GitLab, OSF, or similar services

EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

- **Publish or consume datasets** via GitHub, GitLab, OSF, or similar services



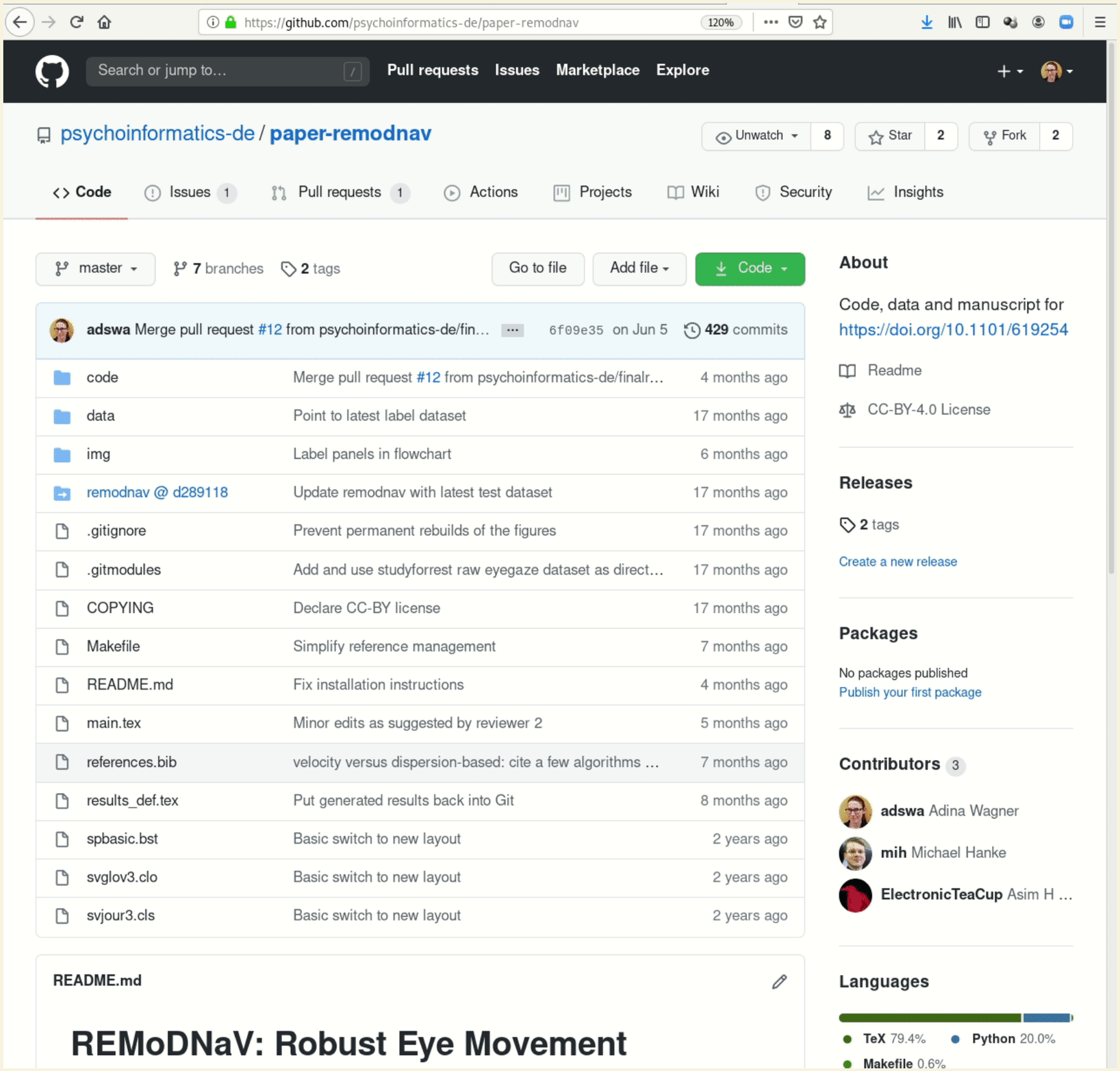
EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

- **Creating and sharing reproducible, open science**: Sharing data, software, code, and provenance

EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

- **Creating and sharing reproducible, open science**: Sharing data, software, code, and provenance



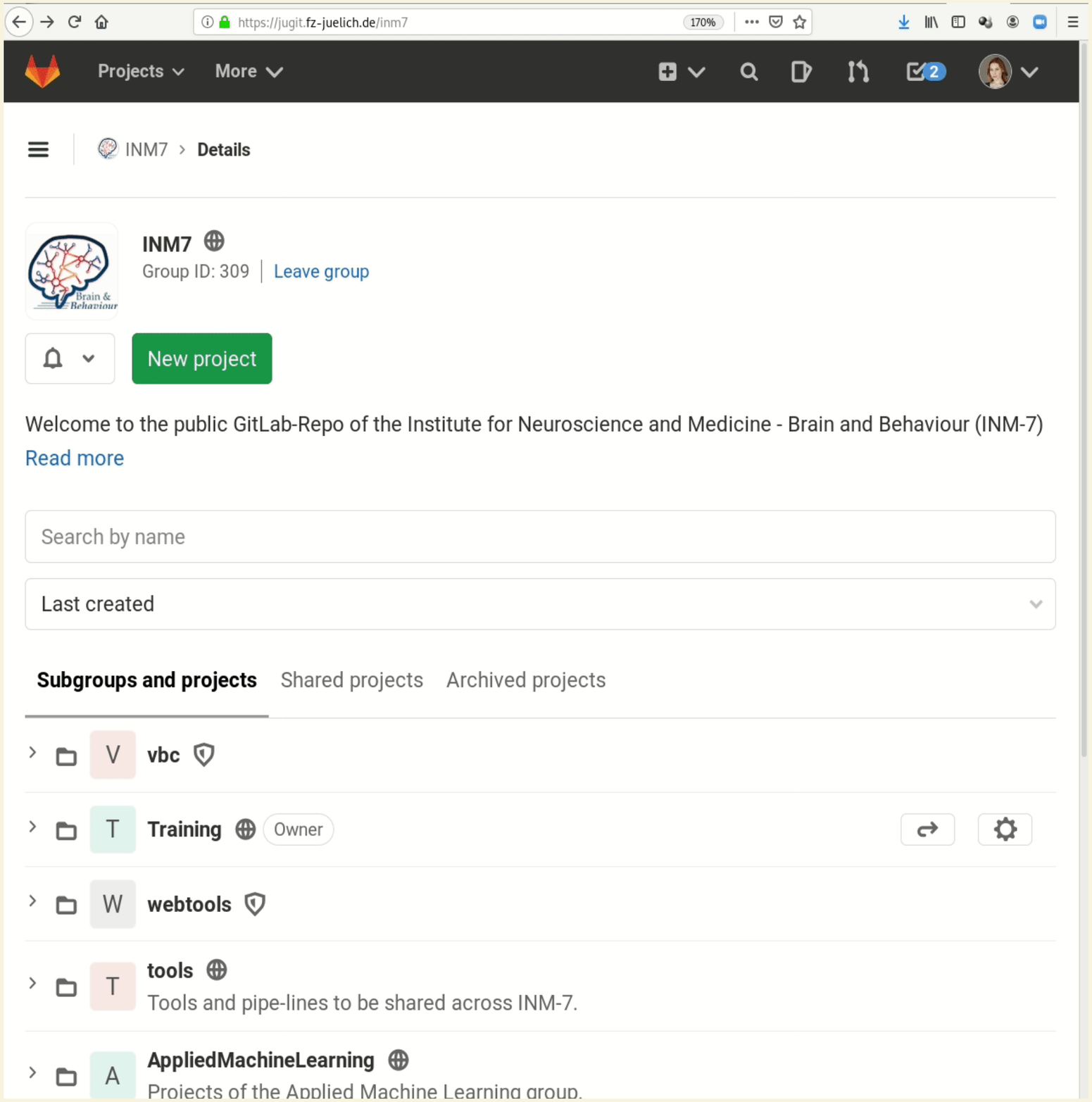
EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

- **Central data management** and archival system

EXAMPLES OF WHAT DATALAD CAN BE USED FOR:

- **Central data management** and archival system



LIVE DEMO BASICS

Code to follow along: handbook.datalad.org/r.html?FZJmlcode

DATALAD DATASETS

- DataLad's core data structure

DATALAD DATASETS

- DataLad's core data structure
    - Dataset = A directory managed by DataLad

DATALAD DATASETS

- DataLad's core data structure
    - Dataset = A directory managed by DataLad
    - Any directory of your computer can be managed by DataLad.

DATALAD DATASETS

- DataLad's core data structure
    - Dataset = A directory managed by DataLad
    - Any directory of your computer can be managed by DataLad.
    - Datasets can be *created* (from scratch) or *installed*

DATALAD DATASETS

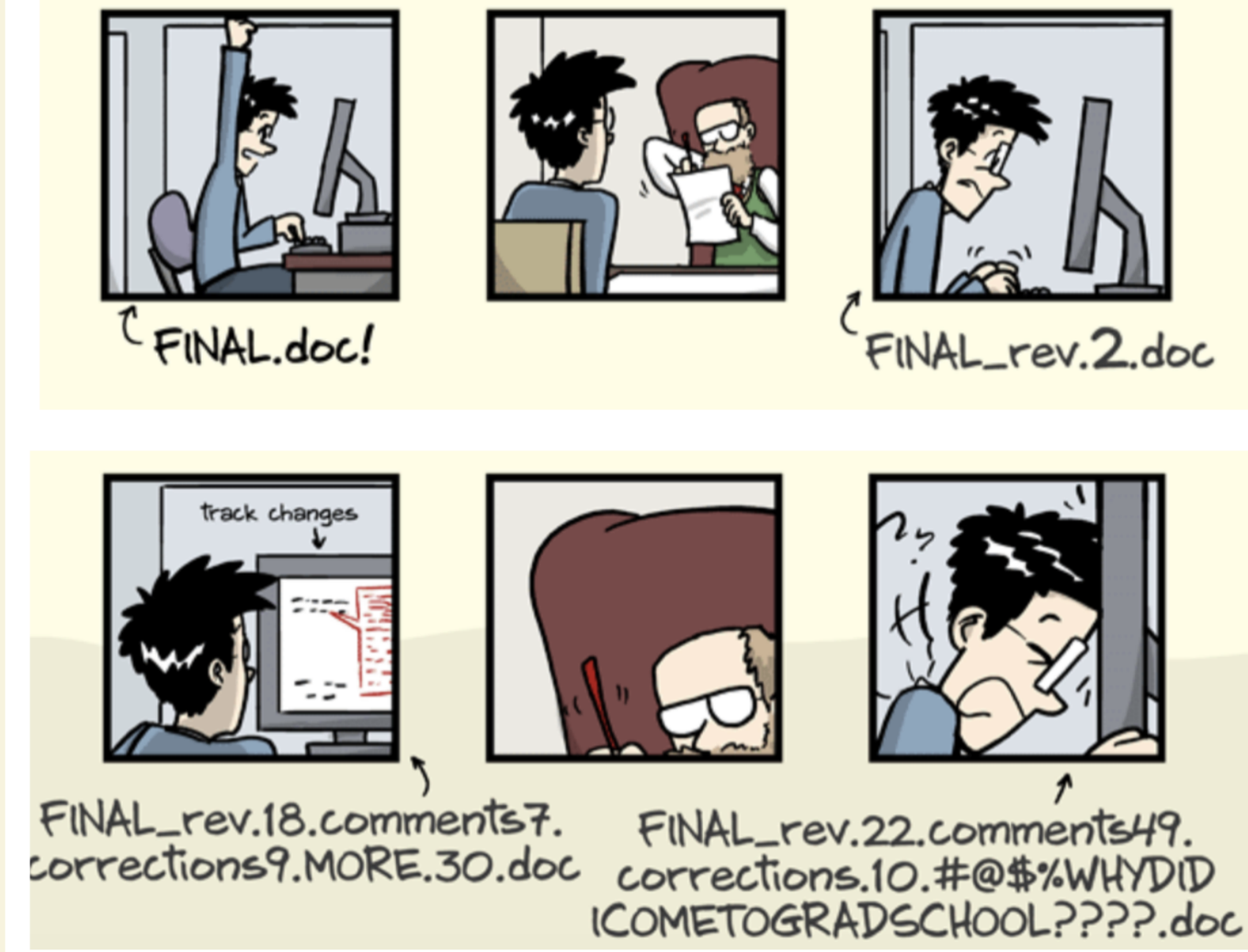
- DataLad's core data structure
    - Dataset = A directory managed by DataLad
    - Any directory of your computer can be managed by DataLad.
    - Datasets can be *created* (from scratch) or *installed*
    - Datasets can be nested: *linked subdirectories*

QUESTIONS!

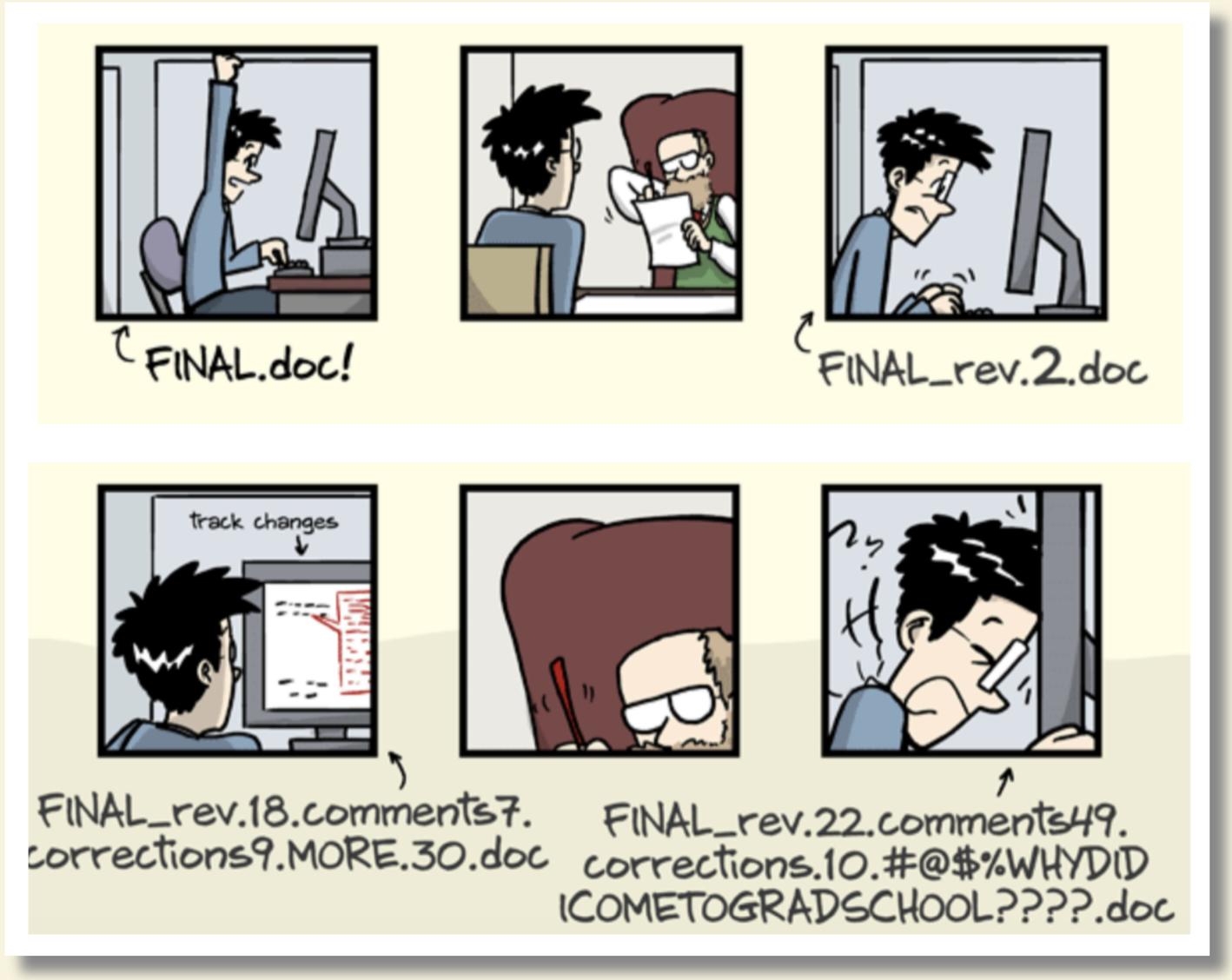
http://etc.ch/y8uM

Direct Poll

WHY VERSION CONTROL?

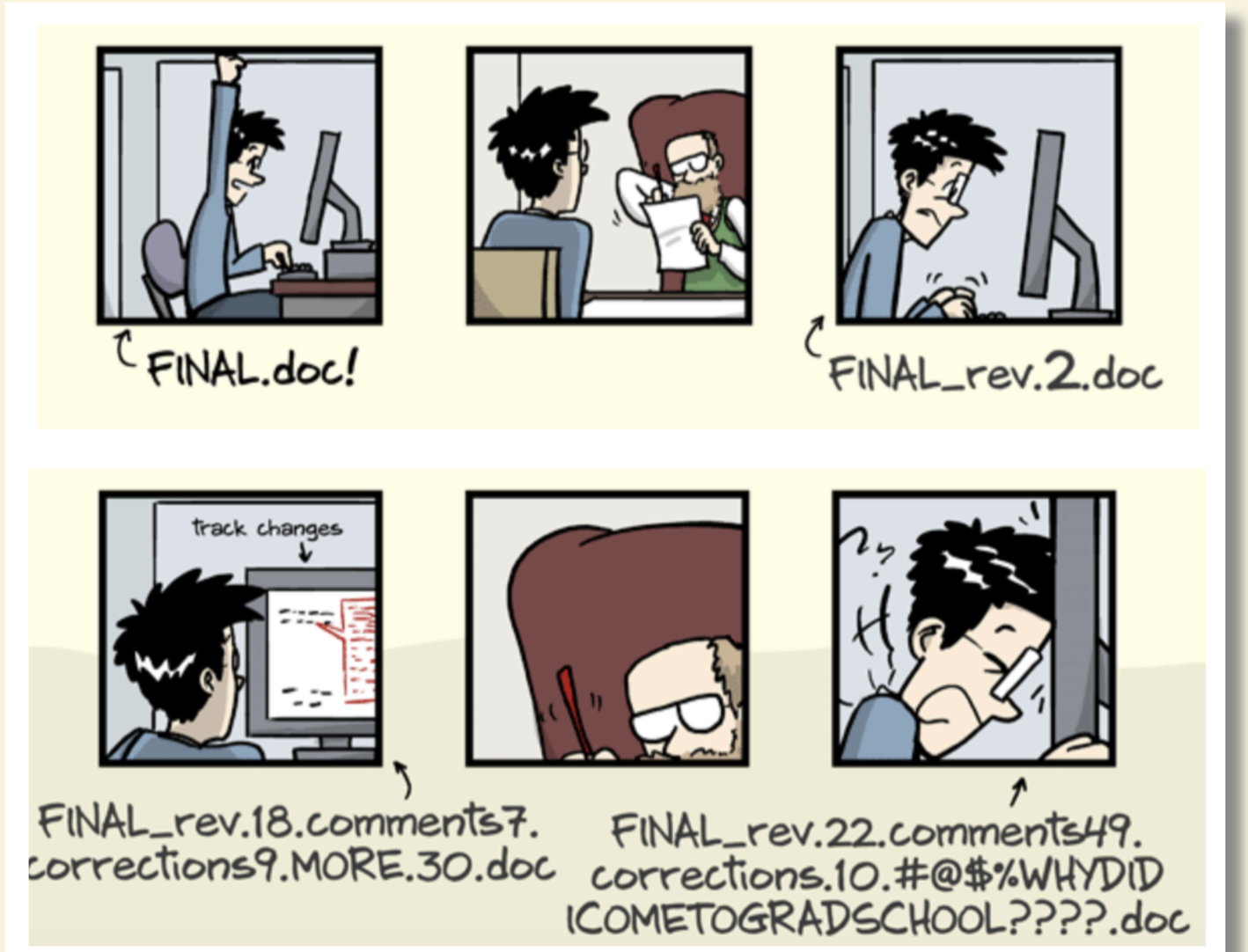


WHY VERSION CONTROL?



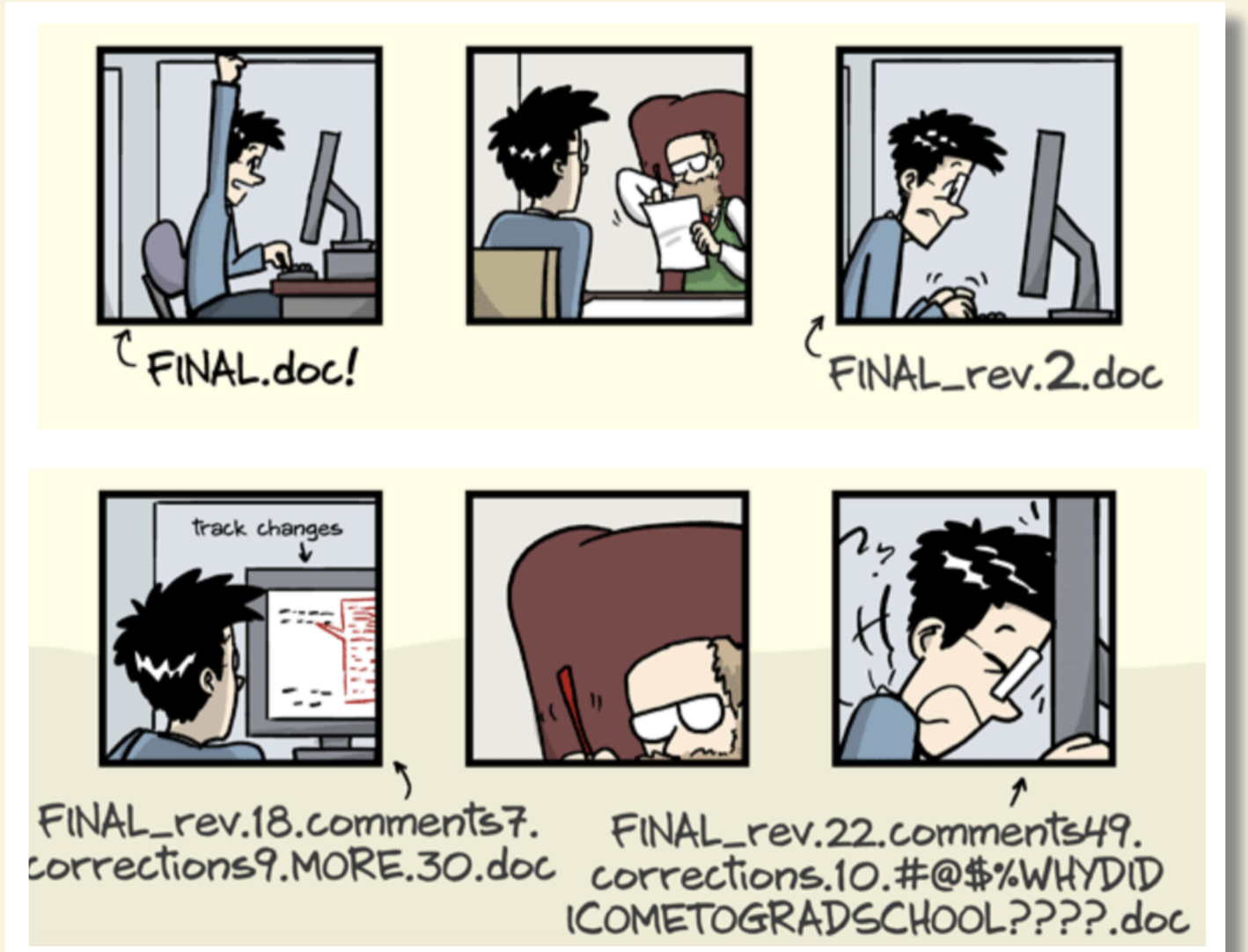
- keep things organized

WHY VERSION CONTROL?



- keep things organized
- keep track of changes

WHY VERSION CONTROL?



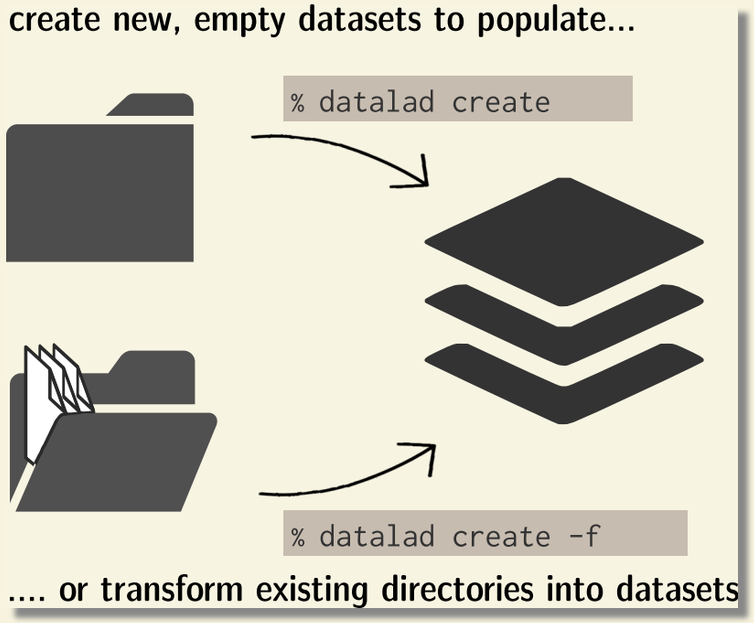
- keep things organized
- keep track of changes
- revert changes or go back to previous states

VERSION CONTROL

- DataLad knows two things: Datasets and files

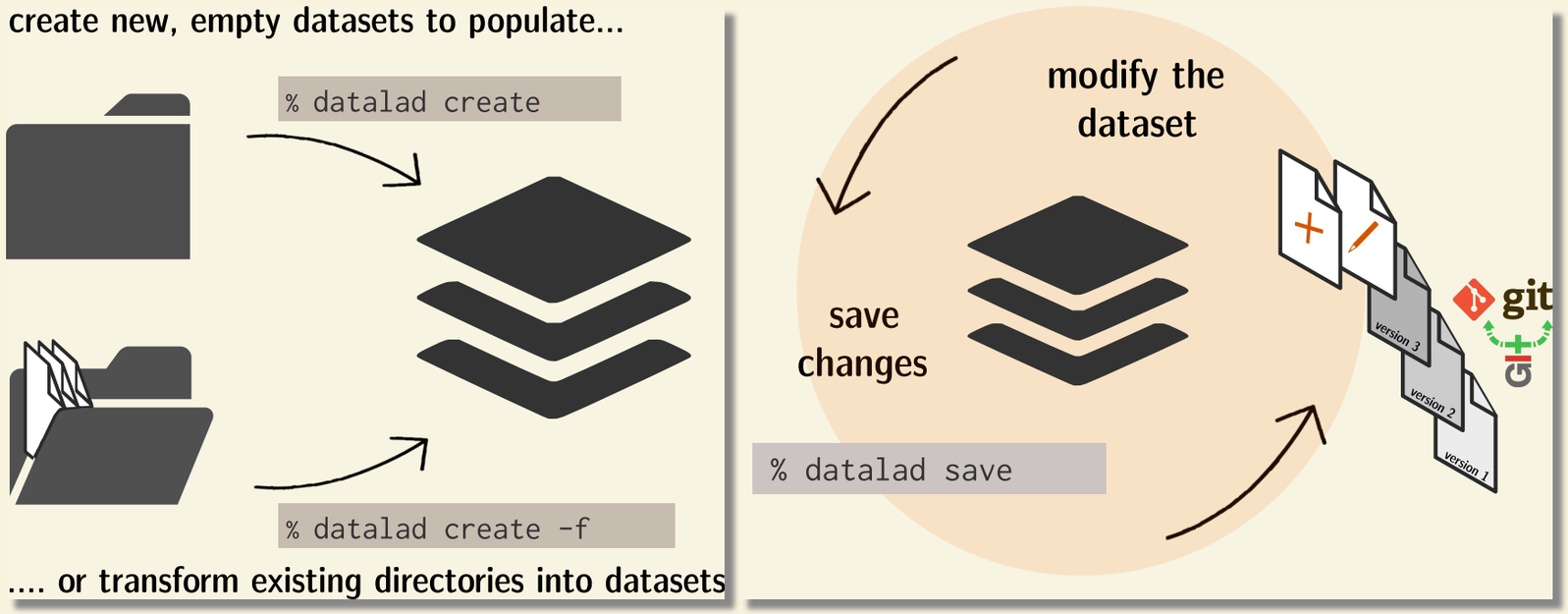
VERSION CONTROL

- DataLad knows two things: Datasets and files



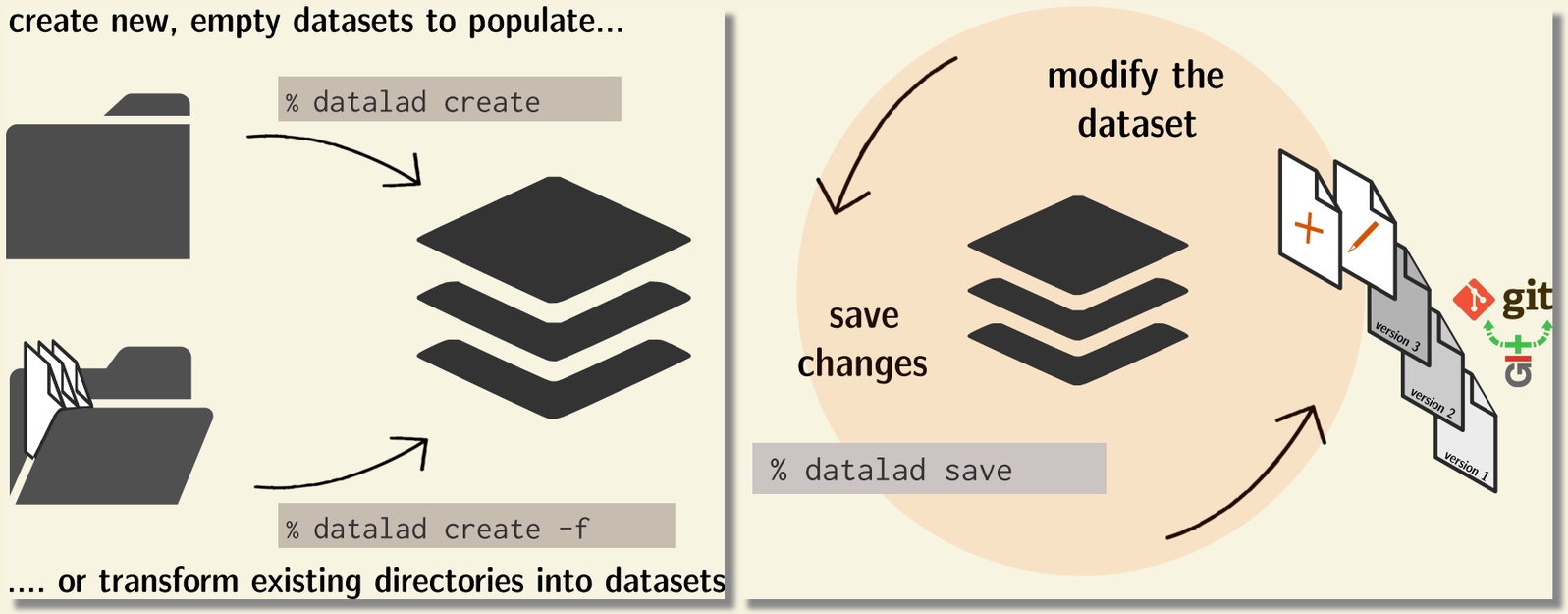
VERSION CONTROL

- DataLad knows two things: Datasets and files



VERSION CONTROL

- DataLad knows two things: Datasets and files



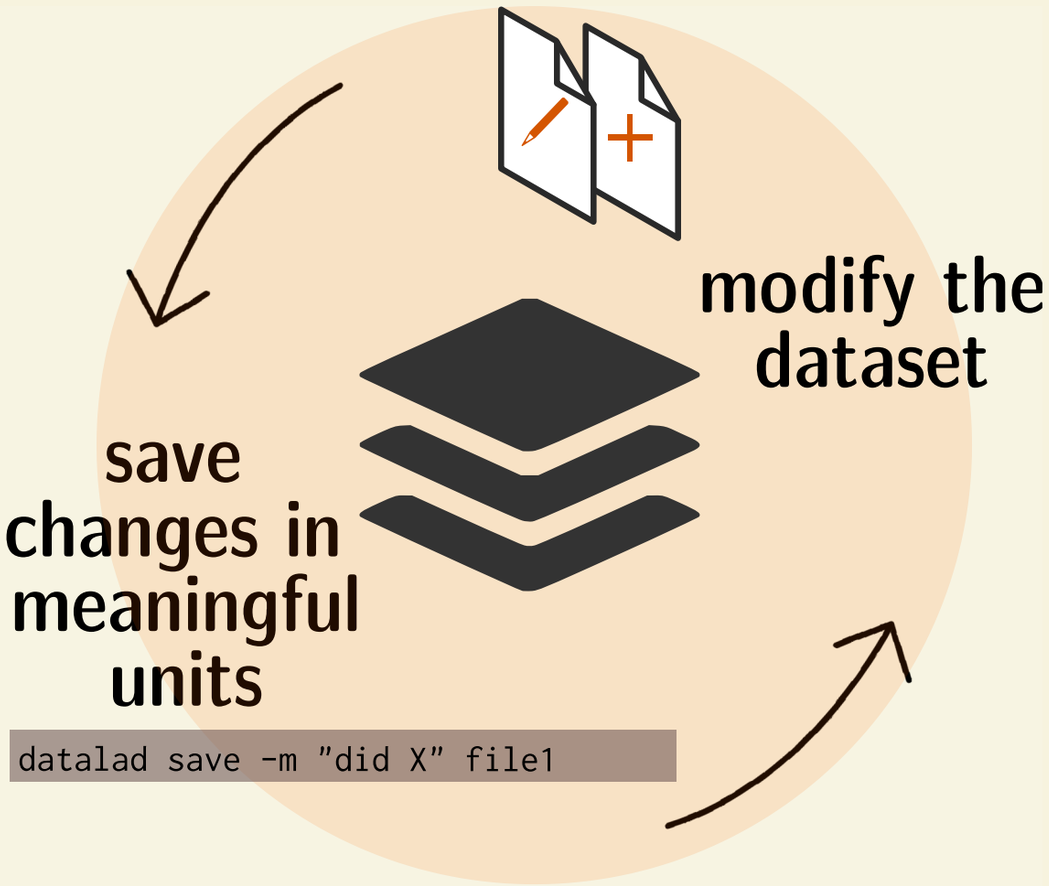
- Every file you put into a in a dataset can be easily version-controlled, regardless of size, with the same command.

LOCAL VERSION CONTROL

Procedurally, version control is easy with DataLad!

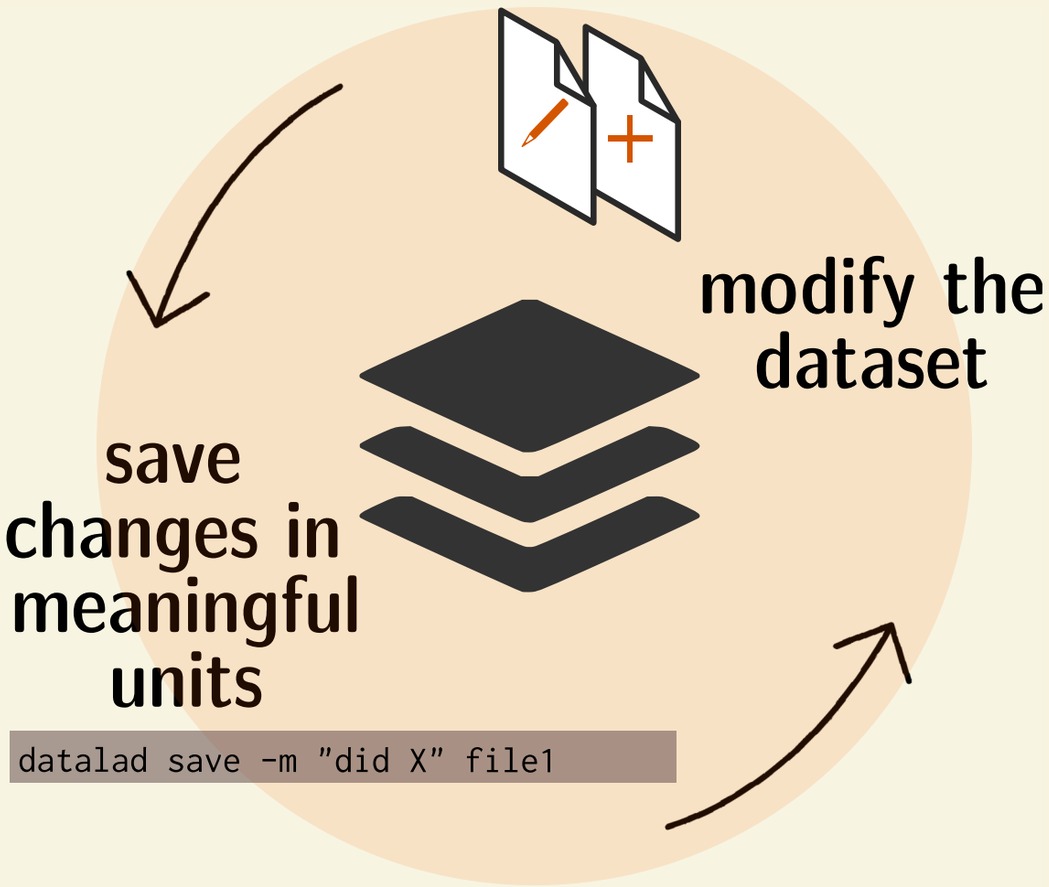
LOCAL VERSION CONTROL

Procedurally, version control is easy with DataLad!



LOCAL VERSION CONTROL

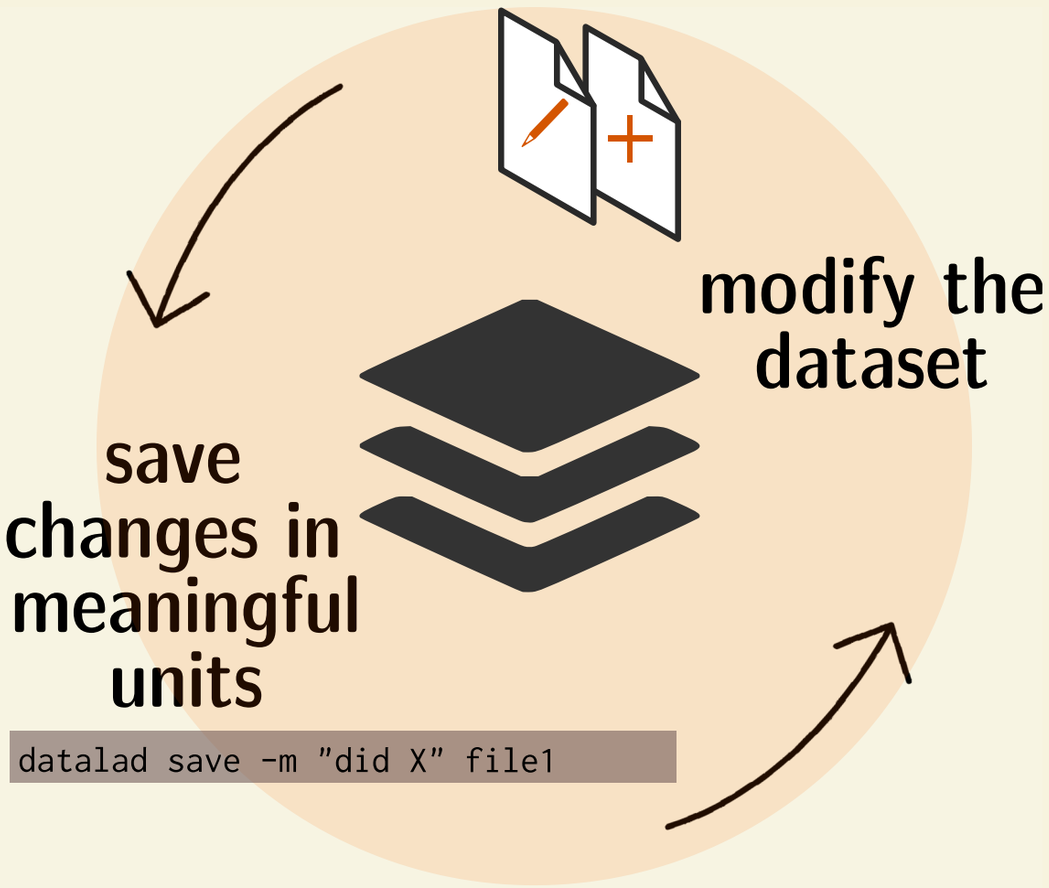
Procedurally, version control is easy with DataLad!



Advice:

LOCAL VERSION CONTROL

Procedurally, version control is easy with DataLad!

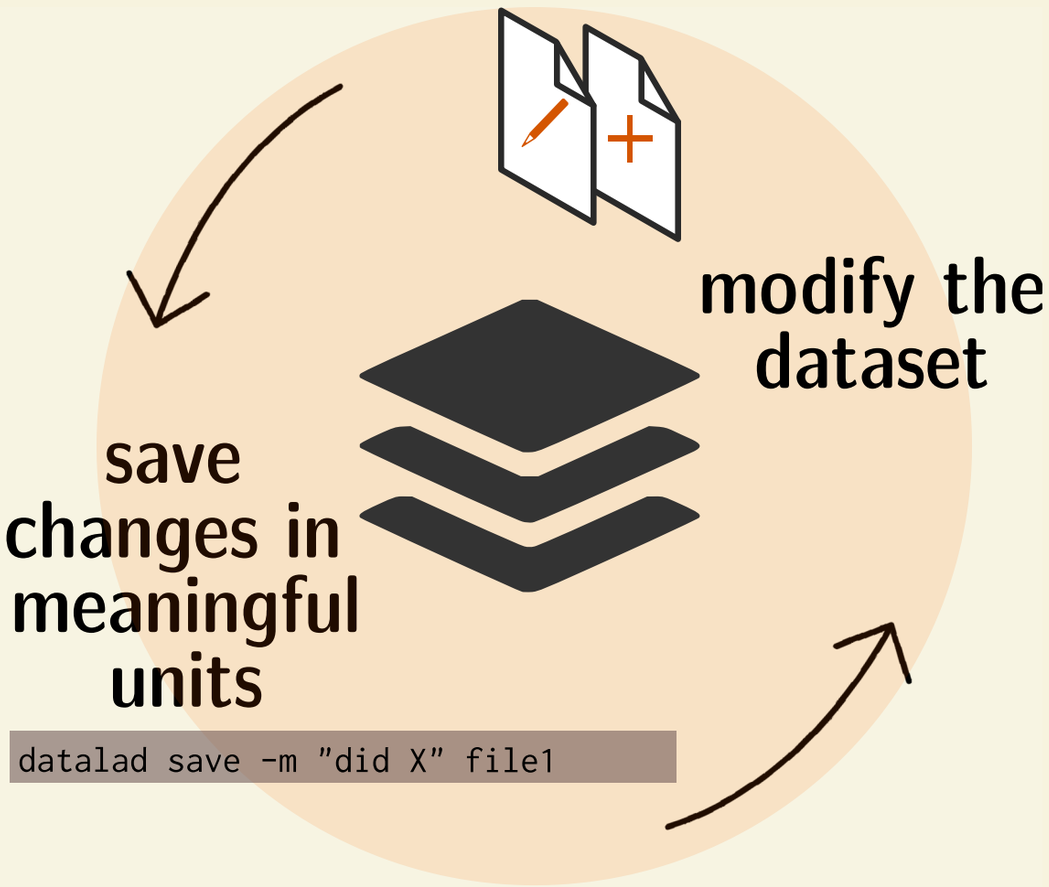


Advice:

- Save *meaningful* units of change

LOCAL VERSION CONTROL

Procedurally, version control is easy with DataLad!



Advice:

- Save *meaningful* units of change
- Attach helpful commit messages

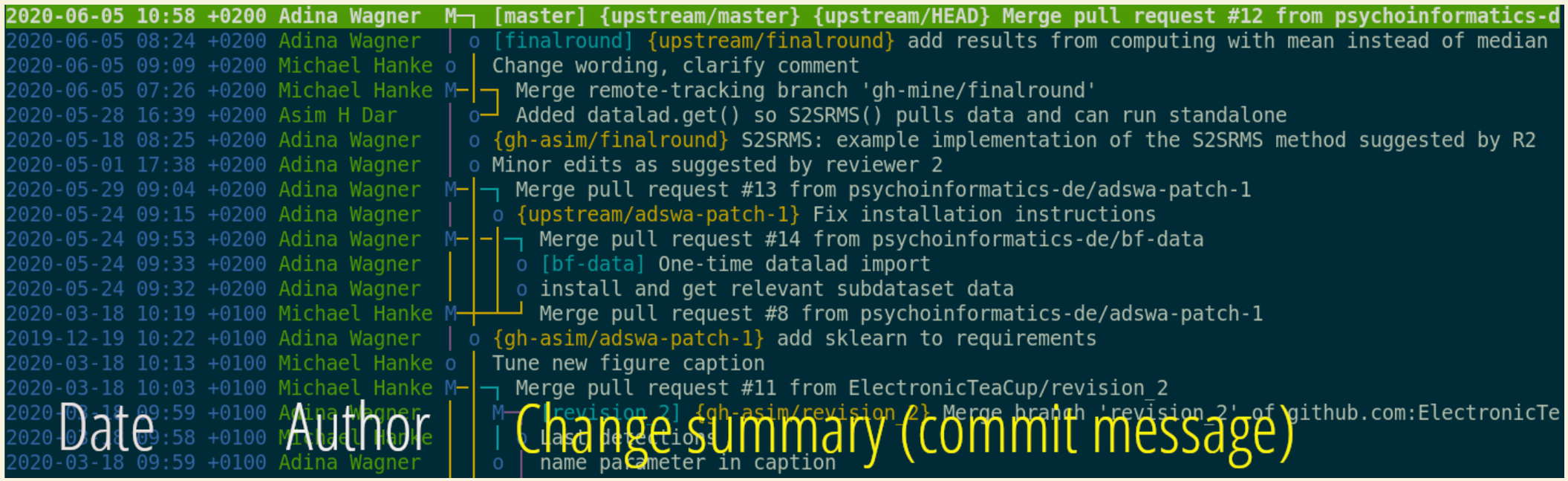
THIS MEANS: YOU CAN ALSO VERSION CONTROL DATA!

THIS MEANS: YOU CAN ALSO VERSION CONTROL DATA!

```
$ datalad save \
   -m "Adding raw data from neuroimaging study 1" \
   sub-*
add(ok): sub-1/anat/T1w.json (file)
add(ok): sub-1/anat/T1w.nii.gz (file)
add(ok): sub-1/anat/T2w.json (file)
add(ok): sub-1/anat/T2w.nii.gz (file)
add(ok): sub-1/func/sub-1-run-1_bold.json (file)
add(ok): sub-1/func/sub-1-run-1_bold.nii.gz (file)
add(ok): sub-10/anat/T1w.json (file)
add(ok): sub-10/anat/T1w.nii.gz (file)
add(ok): sub-10/anat/T2w.json (file)
add(ok): sub-10/anat/T2w.nii.gz (file)
  [110 similar messages have been suppressed]
save(ok): . (dataset)
action summary:
  add (ok: 120)
  save (ok: 1)
```

VERSION CONTROL

- Your dataset can be a complete research log, capturing everything that was done, when, by whom, and how



- Interact with the history:
  - reset your dataset (or subset of it) to a previous state,
  - throw out changes or bring them back,
  - find out what was done when, how, why, and by whom
  - Identify precise versions: Use data in the most recent version, or the one from 2018, or...
  - ...

SUMMARY - LOCAL VERSION CONTROL

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

`datalad save` records the dataset or file state to the history.

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

`datalad save` records the dataset or file state to the history.

Concise **commit messages** should summarize the change for future you and others.

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

`datalad save` records the dataset or file state to the history.

Concise **commit messages** should summarize the change for future you and others.

`datalad download-url` obtains web content and records its origin.

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

`datalad save` records the dataset or file state to the history.

Concise **commit messages** should summarize the change for future you and others.

`datalad download-url` obtains web content and records its origin.

It even takes care of saving the change.

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

`datalad save` records the dataset or file state to the history.

Concise **commit messages** should summarize the change for future you and others.

`datalad download-url` obtains web content and records its origin.

It even takes care of saving the change.

`datalad status` reports the current state of the dataset.

SUMMARY - LOCAL VERSION CONTROL

`datalad create` creates an empty dataset.

Configurations (**-c yoda**, **-c text2git**) are useful (details soon).

A dataset has a *history* to track files and their modifications.

Explore it with Git (**git log**) or external tools (e.g., **tig**).

`datalad save` records the dataset or file state to the history.

Concise **commit messages** should summarize the change for future you and others.

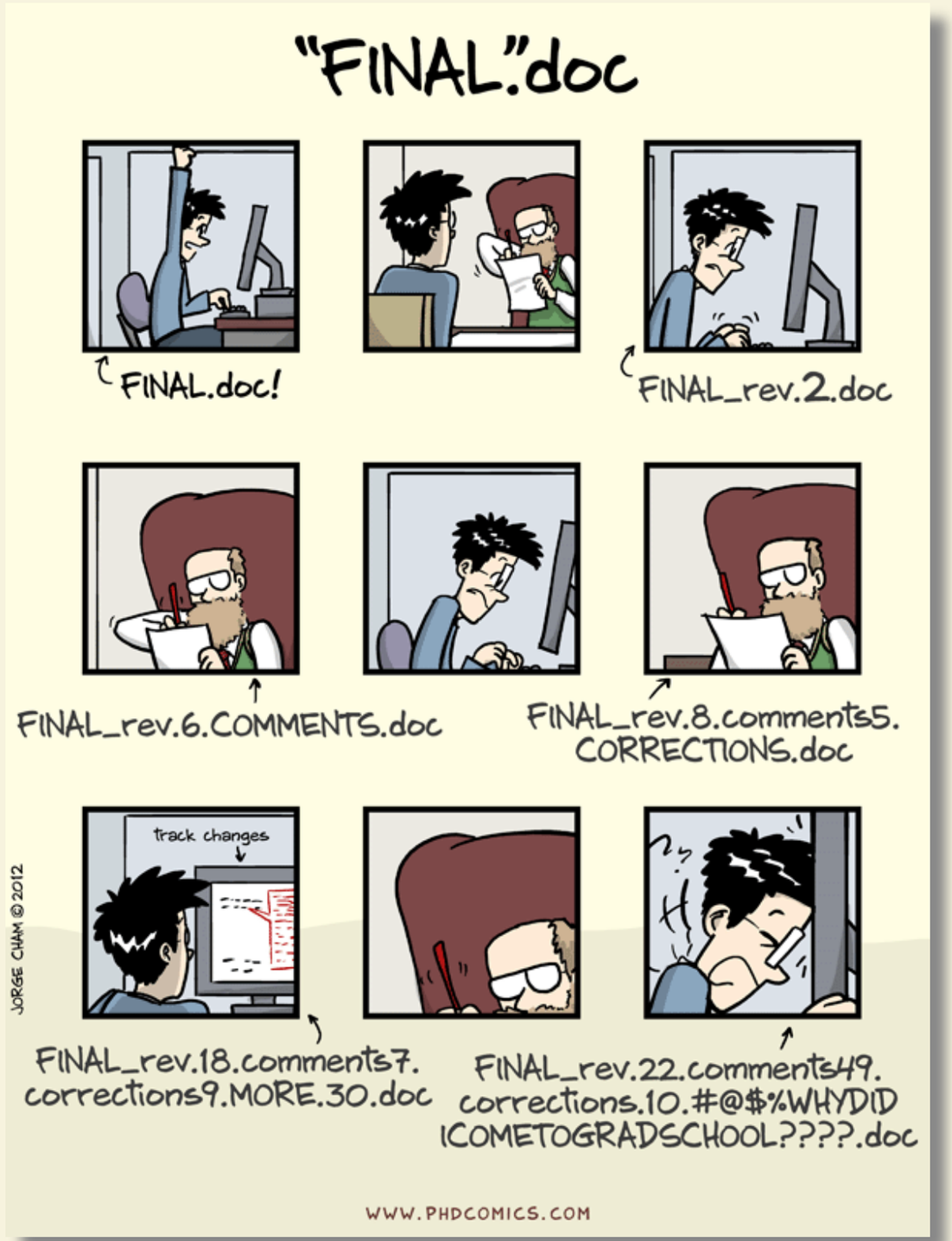
`datalad download-url` obtains web content and records its origin.

It even takes care of saving the change.

`datalad status` reports the current state of the dataset.

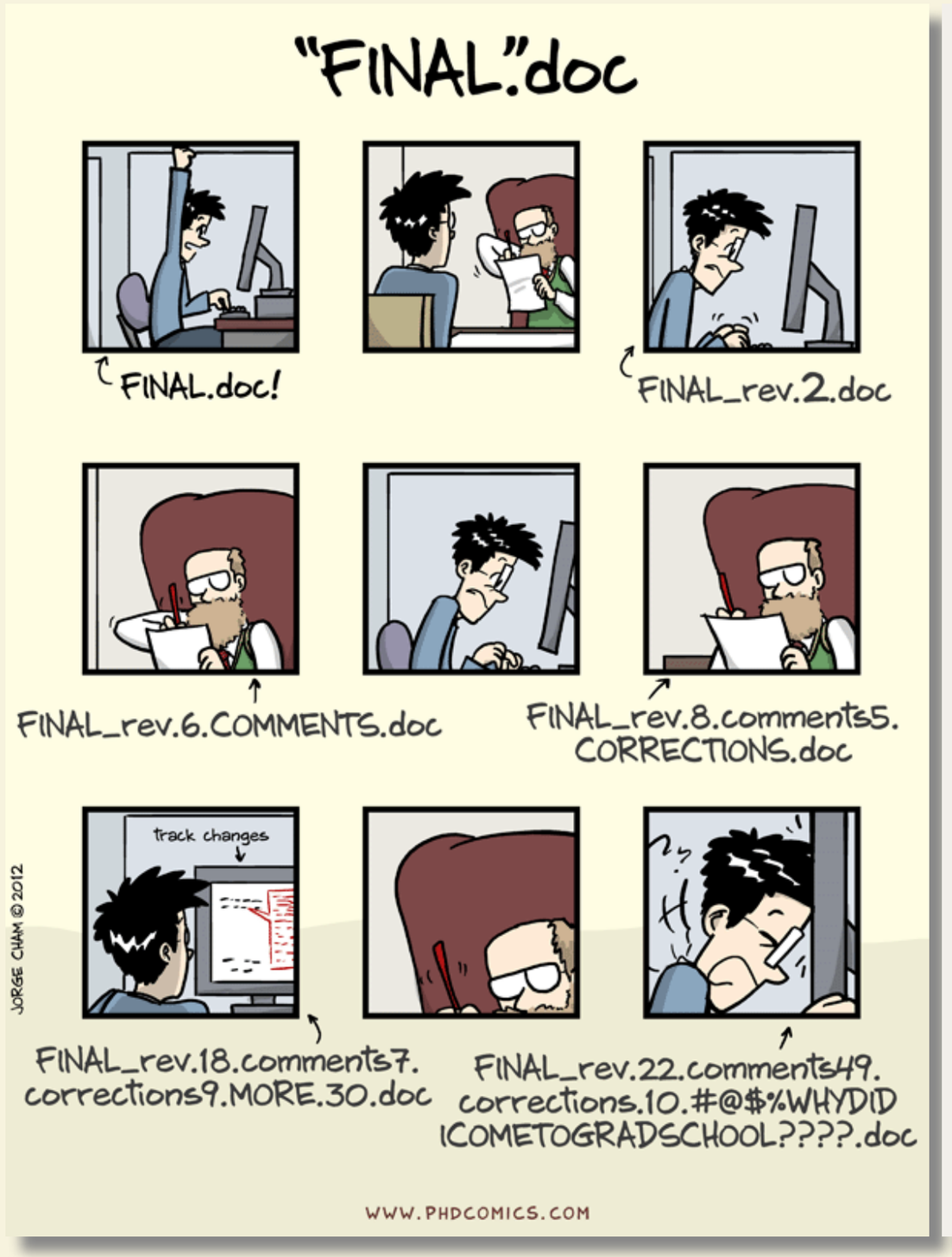
A clean dataset status (no modifications, not untracked files) is good practice.

FROM HERE


"FINAL".doc
FINAL.doc!
FINAL_rev.2.doc
FINAL_rev.6.COMMENTS.doc
FINAL_rev.8.comments5.
CORRECTIONS.doc
track changes
FINAL_rev.18.comments7.
corrections9.MORE.30.doc
FINAL_rev.22.comments49.
corrections.10.#@$%WHYDID
ICOMETOGRADSCHOOL????.doc
JORGE CHAM © 2012
WWW.PHDCOMICS.COM

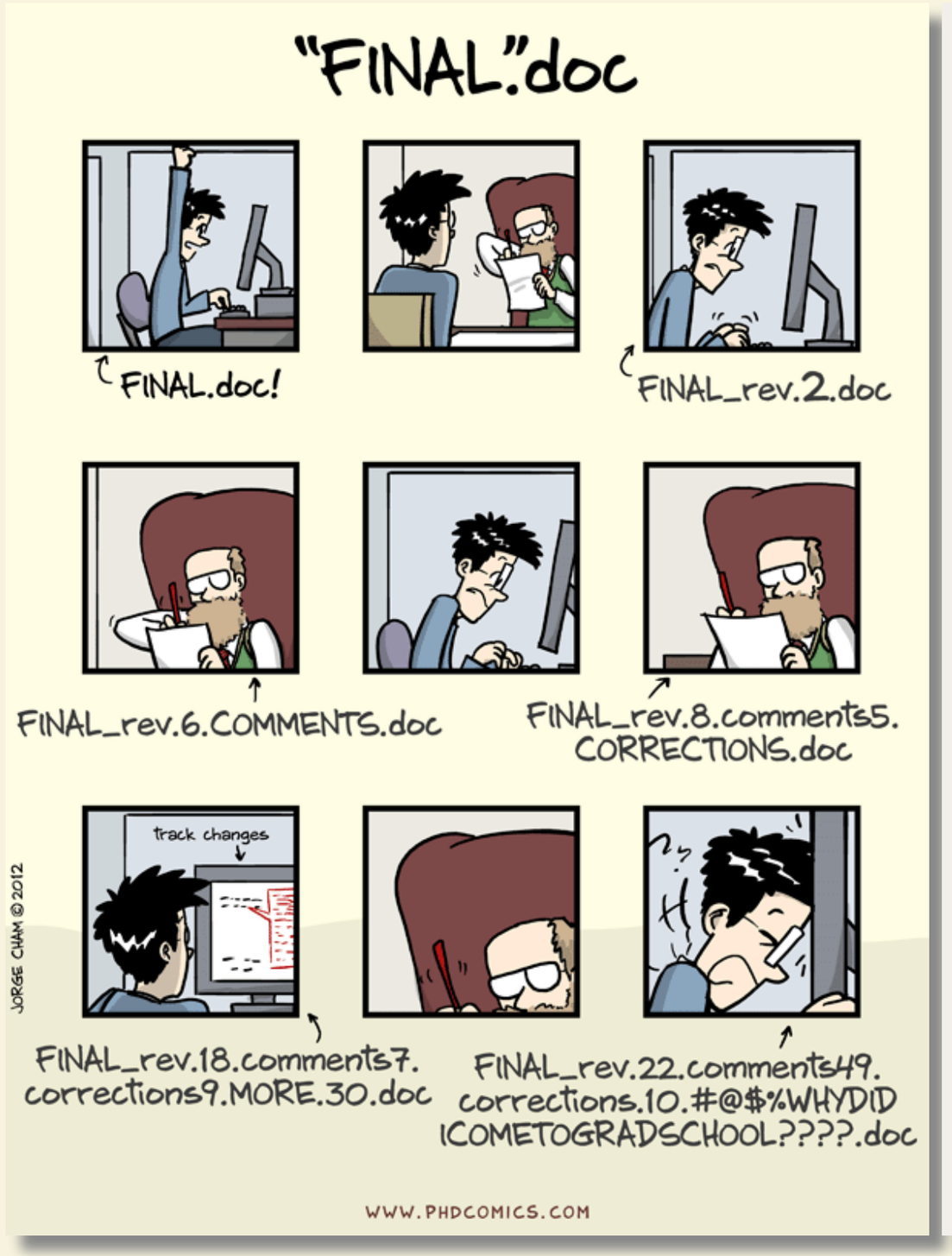
FROM HERE

TO THIS:



FROM HERE

TO THIS:



BUT: Version control is only one aspect of data management

QUESTIONS!

http://etc.ch/y8uM

Direct Poll

CONSUMING DATASETS

- A dataset can be created from scratch/existing directories:

```
$ datalad create mydataset
[INFO   ] Creating a new annex repo at /home/adina/mydataset
create(ok): /home/adina/mydataset (dataset)
```

- but datasets can also be installed from paths or from URLs:

```
$ datalad clone \
 https://github.com/datalad-datasets/human-connectome-project-openacc
 HCP
install(ok): /tmp/HCP (dataset)
```

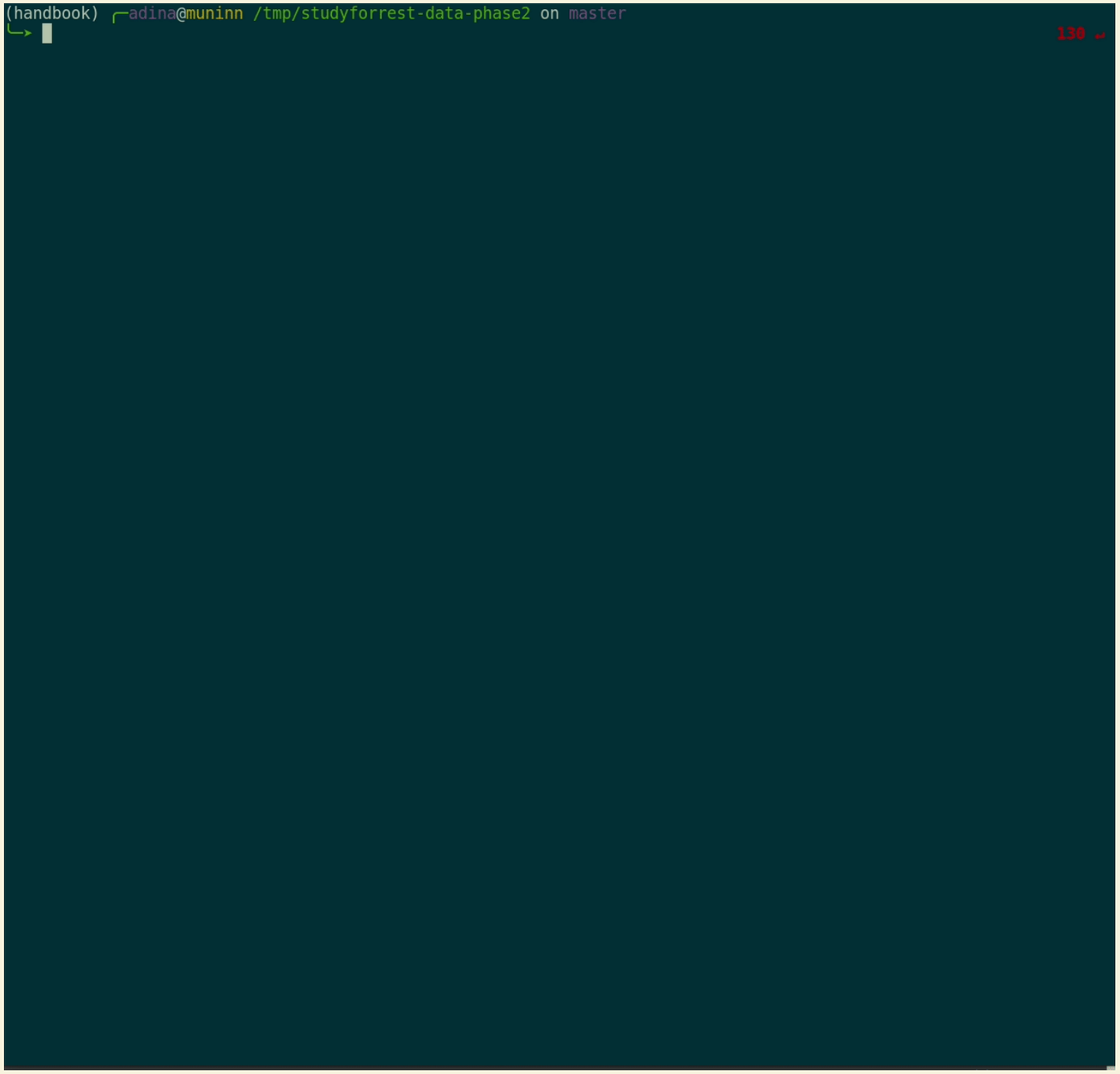
CONSUMING DATASETS

CONSUMING DATASETS

- Here's how a dataset looks after installation:

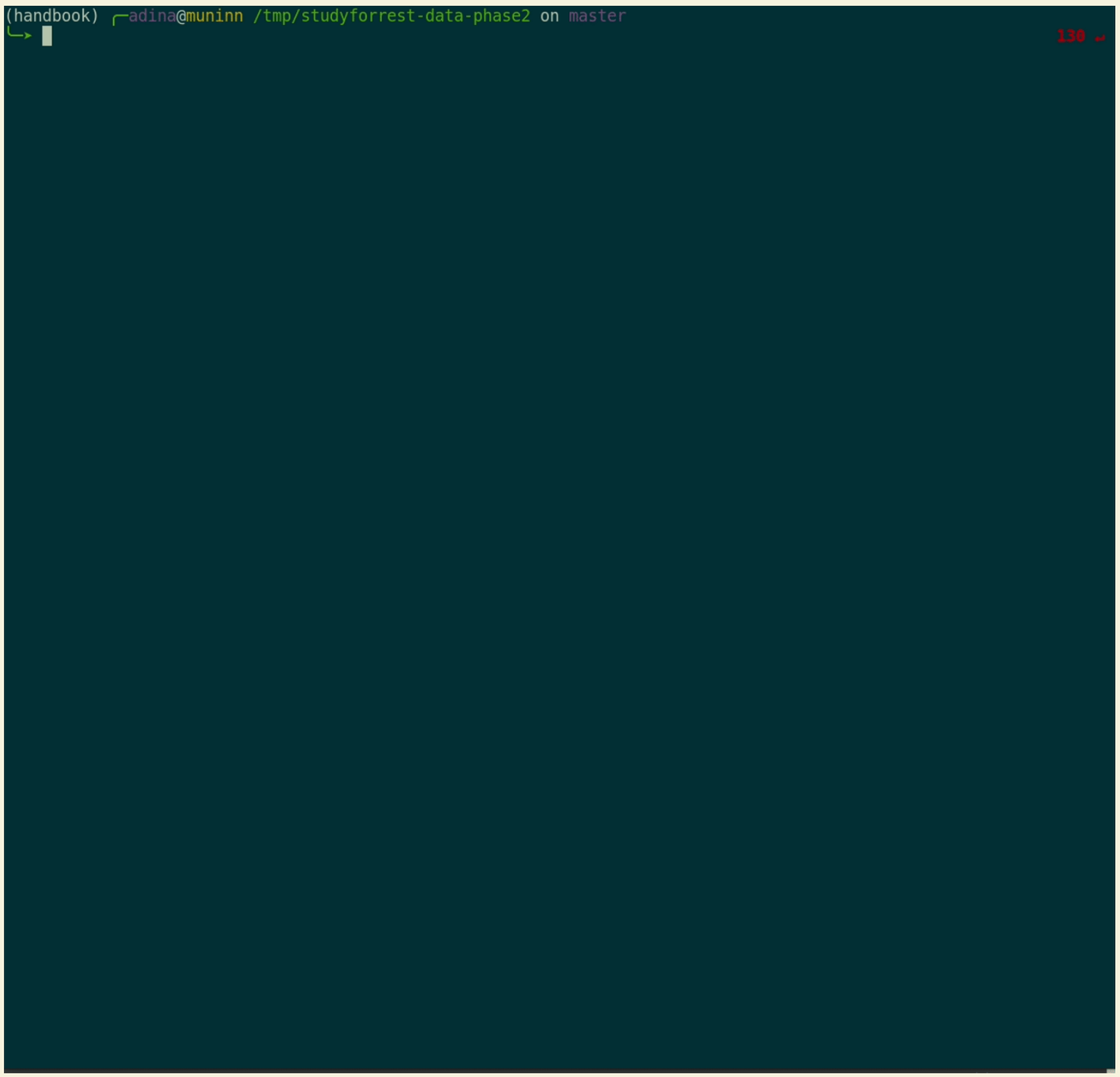
CONSUMING DATASETS

- Here's how a dataset looks after installation:



CONSUMING DATASETS

- Here's how a dataset looks after installation:



- Datasets are light-weight: Upon installation, only small files and meta data about file availability are retrieved.

PLENTY OF DATA, BUT LITTLE DISK-USAGE

PLENTY OF DATA, BUT LITTLE DISK-USAGE

- Cloned datasets are lean. "Meta data" (file names, availability) are present, but **no file content**:

PLENTY OF DATA, BUT LITTLE DISK-USAGE

- Cloned datasets are lean. "Meta data" (file names, availability) are present, but **no file content**:

```
$ datalad clone git@github.com:psychoinformatics-de/studyforrest-data-phase2.git
install(ok): /tmp/studyforrest-data-phase2 (dataset)
$ cd studyforrest-data-phase2 && du -sh
18M     .
```

PLENTY OF DATA, BUT LITTLE DISK-USAGE

- Cloned datasets are lean. "Meta data" (file names, availability) are present, but **no file content**:

```
$ datalad clone git@github.com:psychoinformatics-de/studyforrest-data-phase2.git
install(ok): /tmp/studyforrest-data-phase2 (dataset)
$ cd studyforrest-data-phase2 && du -sh
18M     .
```

- file's contents can be retrieved on demand:

PLENTY OF DATA, BUT LITTLE DISK-USAGE

- Cloned datasets are lean. "Meta data" (file names, availability) are present, but **no file content**:

```
$ datalad clone git@github.com:psychoinformatics-de/studyforrest-data-phase2.git
install(ok): /tmp/studyforrest-data-phase2 (dataset)
$ cd studyforrest-data-phase2 && du -sh
18M     .
```

- file's contents can be retrieved on demand:

```
$ datalad get sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
get(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
```

PLENTY OF DATA, BUT LITTLE DISK-USAGE

- Cloned datasets are lean. "Meta data" (file names, availability) are present, but **no file content**:

```
$ datalad clone git@github.com:psychoinformatics-de/studyforrest-data-phase2.git
install(ok): /tmp/studyforrest-data-phase2 (dataset)
$ cd studyforrest-data-phase2 && du -sh
18M     .
```

- file's contents can be retrieved on demand:

```
$ datalad get sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
get(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
```

- Have more access to your computer than you have disk-space:

PLENTY OF DATA, BUT LITTLE DISK-USAGE

- Cloned datasets are lean. "Meta data" (file names, availability) are present, but **no file content**:

```
$ datalad clone git@github.com:psychoinformatics-de/studyforrest-data-phase2.git
install(ok): /tmp/studyforrest-data-phase2 (dataset)
$ cd studyforrest-data-phase2 && du -sh
18M     .
```

- file's contents can be retrieved on demand:

```
$ datalad get sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
get(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
```

- Have more access to your computer than you have disk-space:

```
# eNKI dataset (1.5TB, 34k files):
$ du -sh
  1.5G  .
# HCP dataset (80TB, 15 million files)
$ du -sh
48G     .
```

PLENTY OF DATA, BUT LITTLE DISK-USAGE

Drop file content that is not needed:

PLENTY OF DATA, BUT LITTLE DISK-USAGE

Drop file content that is not needed:

```
$ datalad drop sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
drop(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.g
```

PLENTY OF DATA, BUT LITTLE DISK-USAGE

Drop file content that is not needed:

```
$ datalad drop sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
drop(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.g
```

When files are dropped, only "meta data" stays behind, and they can be re-obtained on demand. This allows disk-space aware computations:

PLENTY OF DATA, BUT LITTLE DISK-USAGE

Drop file content that is not needed:

```
$ datalad drop sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
drop(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.g
```

When files are dropped, only "meta data" stays behind, and they can be re-obtained on demand. This allows disk-space aware computations:

Install your input data

PLENTY OF DATA, BUT LITTLE DISK-USAGE

Drop file content that is not needed:

```
$ datalad drop sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
drop(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.g
```

When files are dropped, only "meta data" stays behind, and they can be re-obtained on demand. This allows disk-space aware computations:

Install your input data ➡ *get the data you need*

PLENTY OF DATA, BUT LITTLE DISK-USAGE

Drop file content that is not needed:

```
$ datalad drop sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
drop(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.g
```

When files are dropped, only "meta data" stays behind, and they can be re-obtained on demand. This allows disk-space aware computations:

Install your input data ➡ *get the data you need* ➡ *compute your results*

PLENTY OF DATA, BUT LITTLE DISK-USAGE

Drop file content that is not needed:

```
$ datalad drop sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.gz
drop(ok): /tmp/studyforrest-data-phase2/sub-01/ses-movie/func/sub-01_ses-movie_task-movie_run-1_bold.nii.g
```

When files are dropped, only "meta data" stays behind, and they can be re-obtained on demand. This allows disk-space aware computations:

Install your input data ➡ *get the data you need* ➡ *compute your results* ➡ *drop input data*
(and potentially all automatically re-computable results)

GIT VERSUS GIT-ANNEX

Data in datasets is either stored in Git or git-annex

By default, everything is *annexed*, i.e., stored in a dataset annex by git-annex

GIT VERSUS GIT-ANNEX

Data in datasets is either stored in Git or git-annex

By default, everything is *annexed*, i.e., stored in a dataset annex by git-annex

- With annexed data, only content identity (hash) and location information is put into Git, rather than file content. The annex, and transport to and from it is managed with **git-annex**

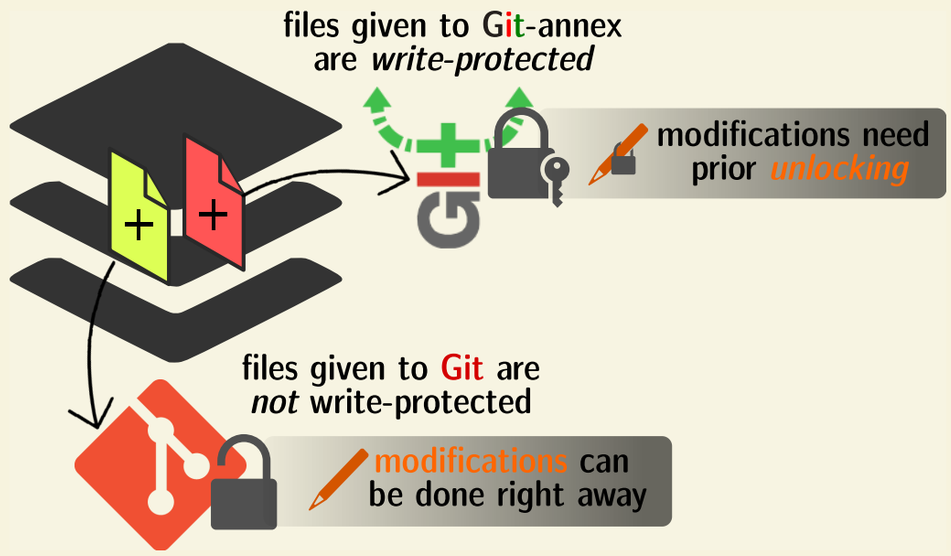
Git	**git-annex**
handles **small** files well (text, code)	handles **all** types and sizes of files well
file contents are in the Git history and will be **shared** upon git/datalad push	file contents are in the annex. Not necessarily shared
Shared with every dataset clone	**Can be kept private** on a per-file level when sharing the dataset
Useful: Small, non-binary, frequently modified, need-to-be-accessible (DUA, README) files	Useful: Large files, private files

GIT VERSUS GIT-ANNEX

Useful background information for demo later. Read this handbook chapter for details

Git and Git-annex handle files differently: annexed files are stored in an annex. File content is hashed & only content-identity is committed to Git.

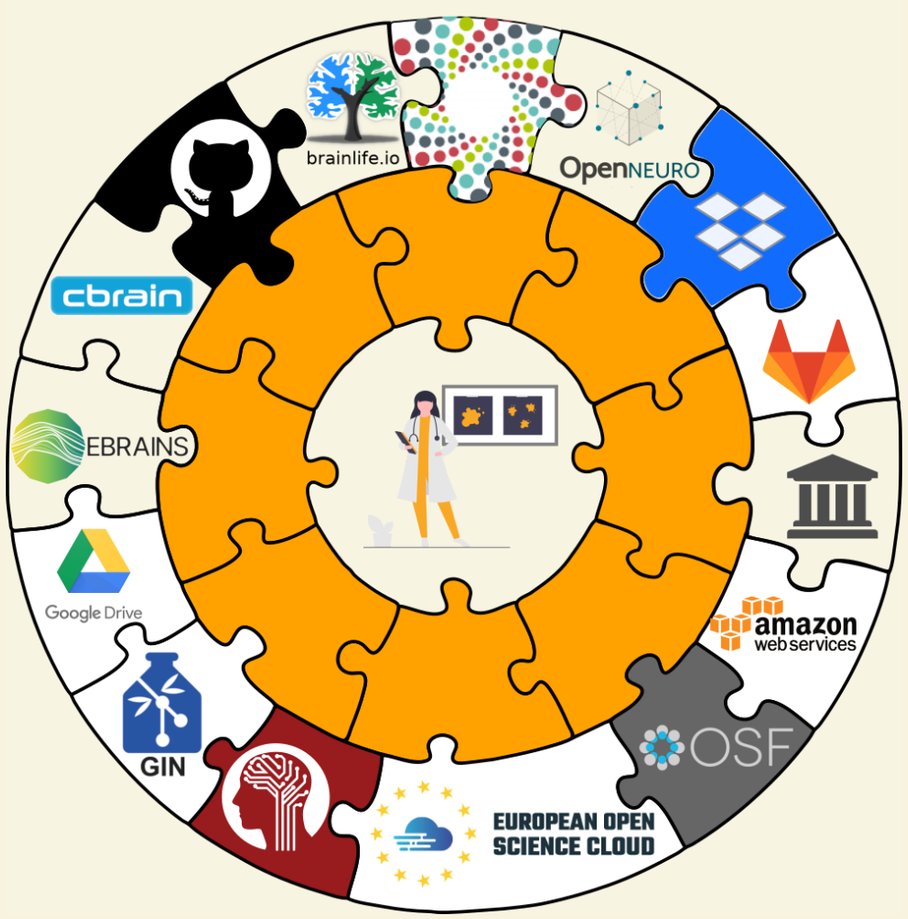
- Files stored in Git are modifiable, files stored in Git-annex are content-locked



- Annexed contents are not available right after cloning, only content- and availability information (as they are stored in Git)

GIT VERSUS GIT-ANNEX

When sharing datasets with someone without access to the same computational infrastructure, annexed data is not necessarily stored together with the rest of the dataset.



Transport logistics exist to interface with all major storage providers. If the one you use isn't supported, let us know!

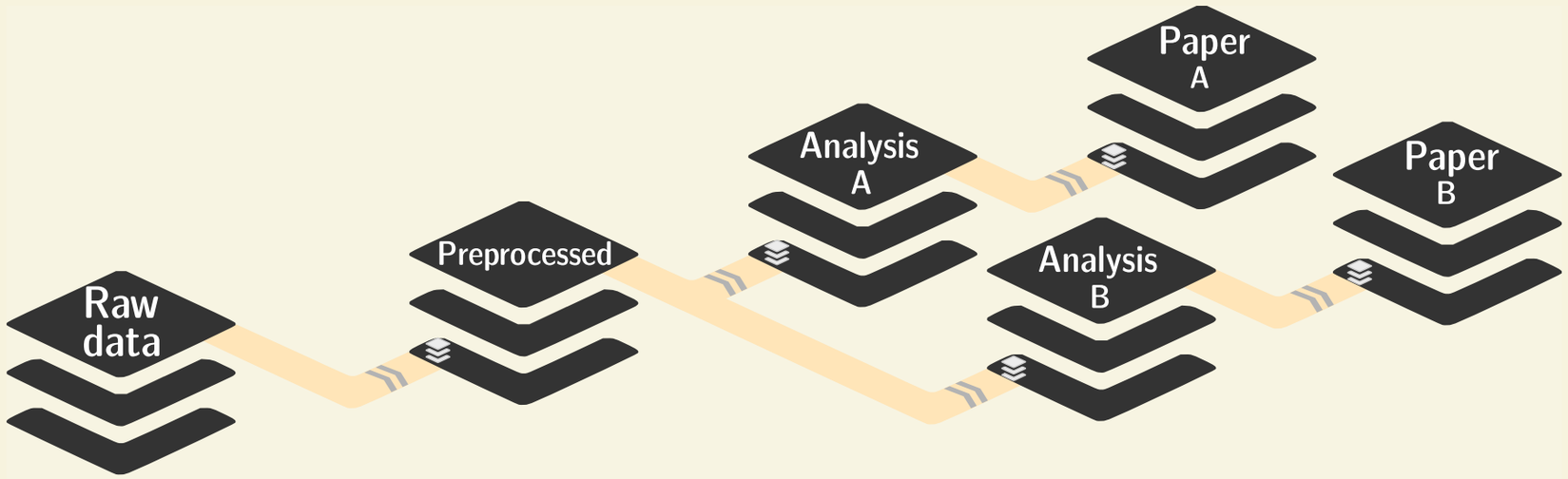
GIT VERSUS GIT-ANNEX

Users can decide which files are annexed:

- **Pre-made run-procedures**, provided by DataLad (e.g., `text2git`, `yoda`) or created and shared by users (Tutorial)
- Self-made configurations in `.gitattributes` (e.g., based on file type, file/path name, size, ...; rules and examples)
- Per-command basis (e.g., via `datalad save --to-git`)

VERSION CONTROL: NESTING

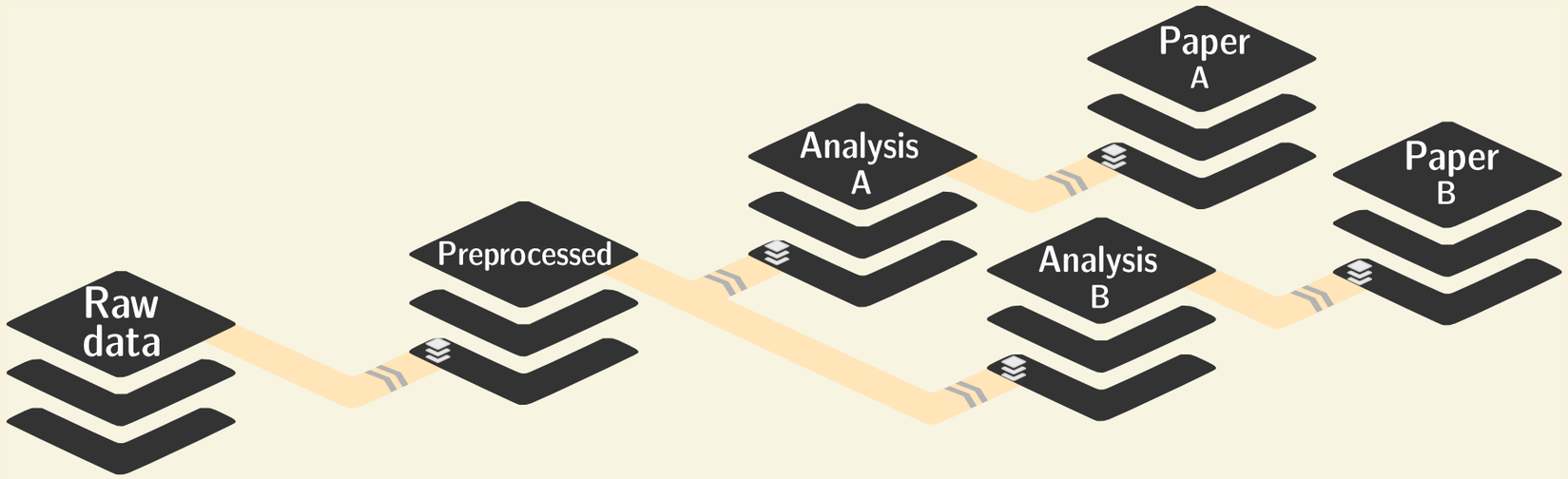
- Seamless nesting mechanisms:



Nest modular datasets to create a linked hierarchy of datasets, and enable recursive operations throughout the hierarchy

VERSION CONTROL: NESTING

- Seamless nesting mechanisms:



Nest modular datasets to create a linked hierarchy of datasets, and enable recursive operations throughout the hierarchy

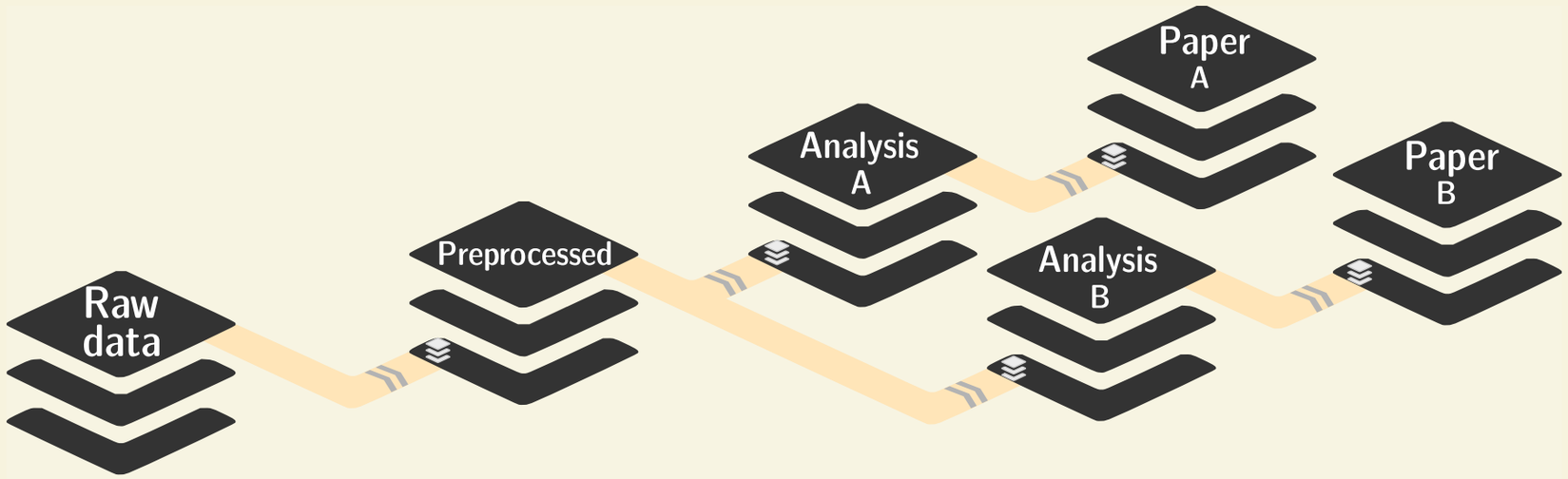
- Overcomes scaling issues with large amounts of files

```
adina@bulk1 in /ds/hcp/super on git:master❱ datalad status --annex -r
15530572 annex'd files (77.9 TB recorded total size)
nothing to save, working tree clean
```

(github.com/datalad-datasets/human-connectome-project-openaccess)

VERSION CONTROL: NESTING

- Seamless nesting mechanisms:



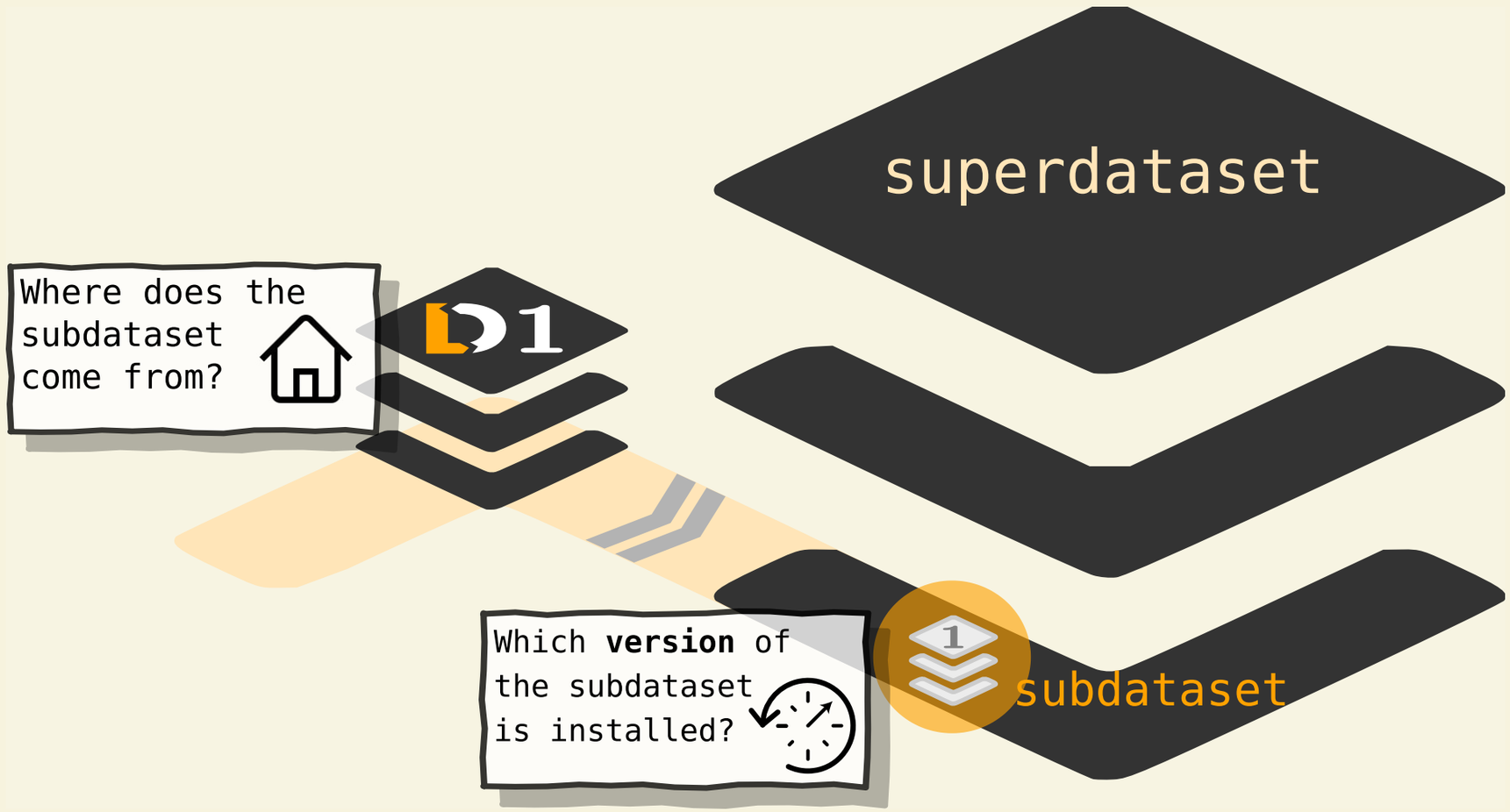
- Overcomes scaling issues with large amounts of files

```
adina@bulk1 in /ds/hcp/super on git:master❱ datalad status --annex -r
15530572 annex'd files (77.9 TB recorded total size)
nothing to save, working tree clean
```

(github.com/datalad-datasets/human-connectome-project-openaccess)

- Modularizes research components for transparency, reuse, and access management

DATASET NESTING



SUMMARY - DATASET CONSUMPTION & NESTING

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed “on its own”: Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed “on its own”: Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

Datasets can be installed as subdatasets within an existing dataset.

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed “on its own”: Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

Datasets can be installed as subdatasets within an existing dataset.

The **--dataset/-d** option needs a path to the root of the superdataset.

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed "on its own": Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

Datasets can be installed as subdatasets within an existing dataset.

The **--dataset/-d** option needs a path to the root of the superdataset.

Only small files and metadata about file availability are present locally after an install.

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed "on its own": Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

Datasets can be installed as subdatasets within an existing dataset.

The **--dataset/-d** option needs a path to the root of the superdataset.

Only small files and metadata about file availability are present locally after an install.

To retrieve actual file content of annexed files, `datalad get` downloads file content on demand.

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed "on its own": Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

Datasets can be installed as subdatasets within an existing dataset.

The **--dataset/-d** option needs a path to the root of the superdataset.

Only small files and metadata about file availability are present locally after an install.

To retrieve actual file content of annexed files, `datalad get` downloads file content on demand.

Datasets preserve their history.

SUMMARY - DATASET CONSUMPTION & NESTING

`datalad clone` installs a dataset.

It can be installed "on its own": Specify the source (url, path, ...) of the dataset, and an optional **path** for it to be installed to.

Datasets can be installed as subdatasets within an existing dataset.

The **--dataset/-d** option needs a path to the root of the superdataset.

Only small files and metadata about file availability are present locally after an install.

To retrieve actual file content of annexed files, `datalad get` downloads file content on demand.

Datasets preserve their history.

The superdataset records only the *version state* of the subdataset.

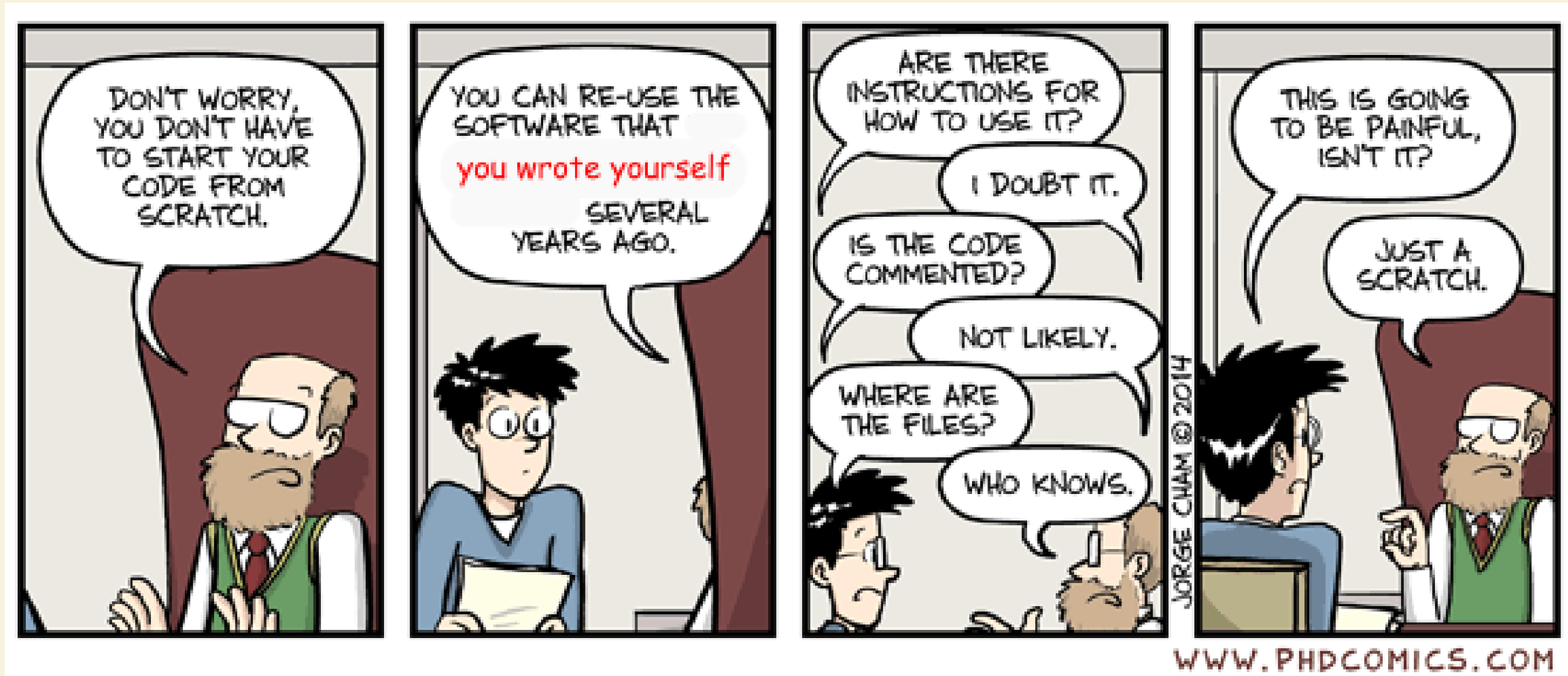
QUESTIONS!

http://etc.ch/y8uM

Direct Poll

REPRODUCIBLE DATA ANALYSIS

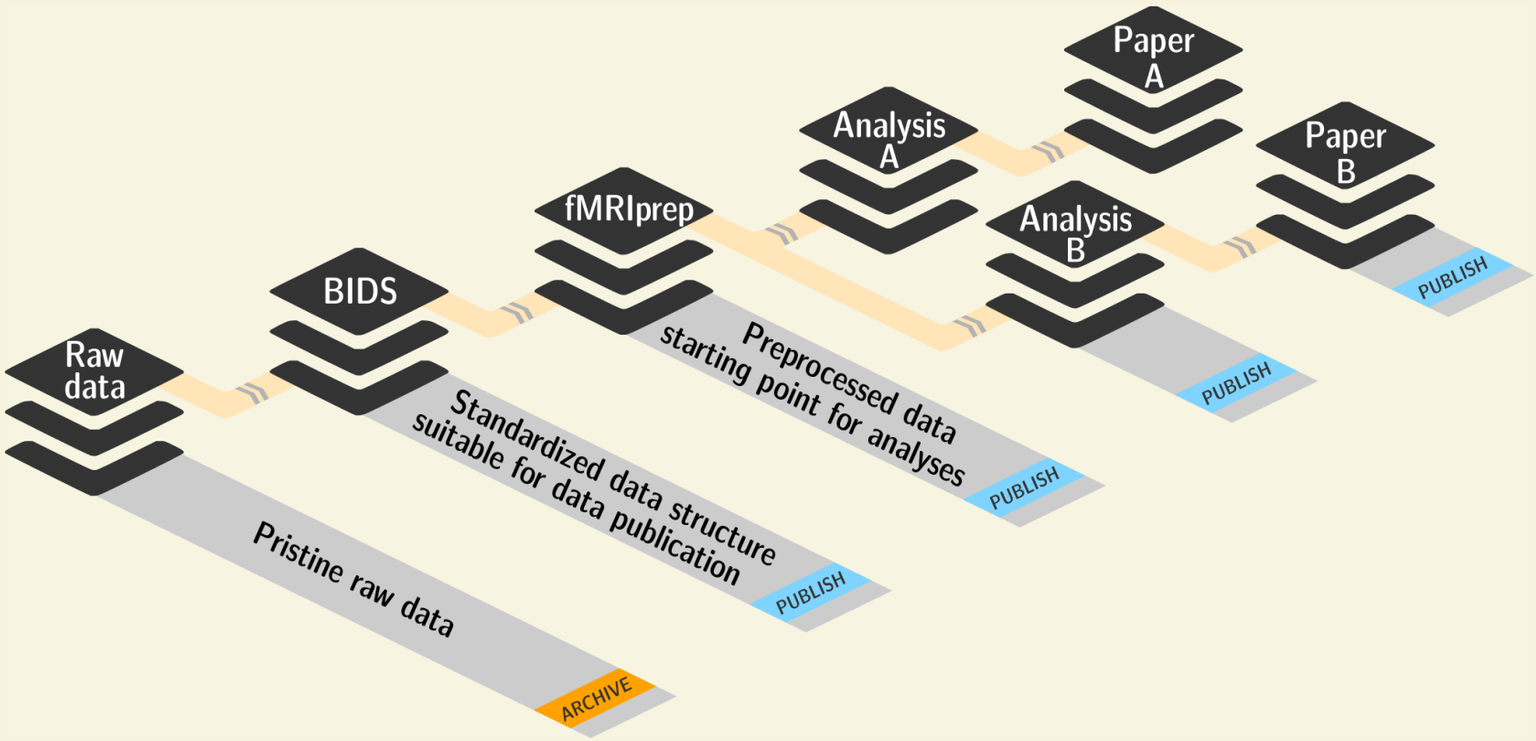
Your past self is the worst collaborator:

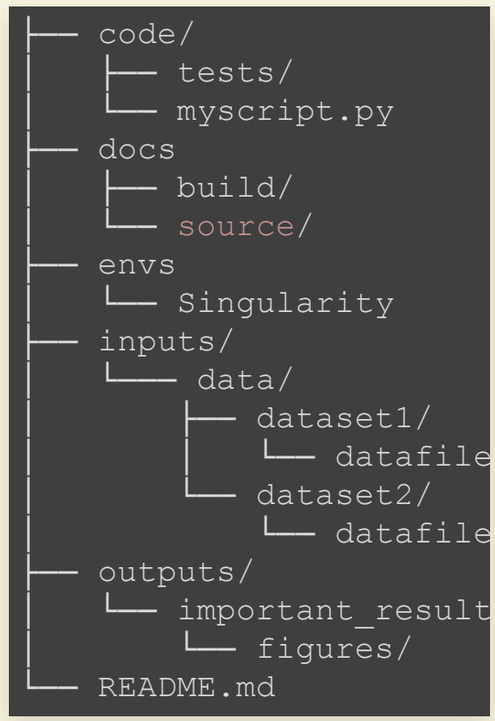


BASIC ORGANIZATIONAL PRINCIPLES FOR DATASETS

Keep everything clean and modular

- An analysis is a superdataset, its components are subdatasets, and its structure modular



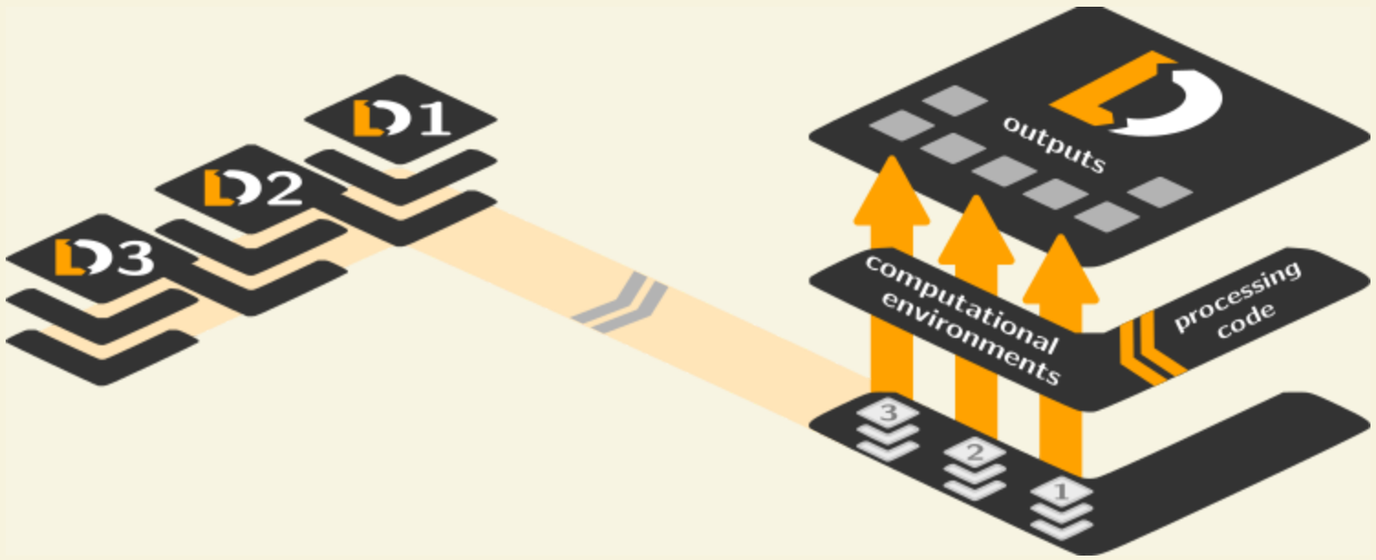


- do not touch/modify raw data: save any results/computations *outside* of input datasets
- Keep a superdataset self-contained: Scripts reference subdatasets or files with *relative paths*

BASIC ORGANIZATIONAL PRINCIPLES FOR DATASETS

Record where you got it from, where it is now, and what you do to it

- Link datasets (as subdatasets), record data origin
- Collect and store provenance of all contents of a dataset that you create

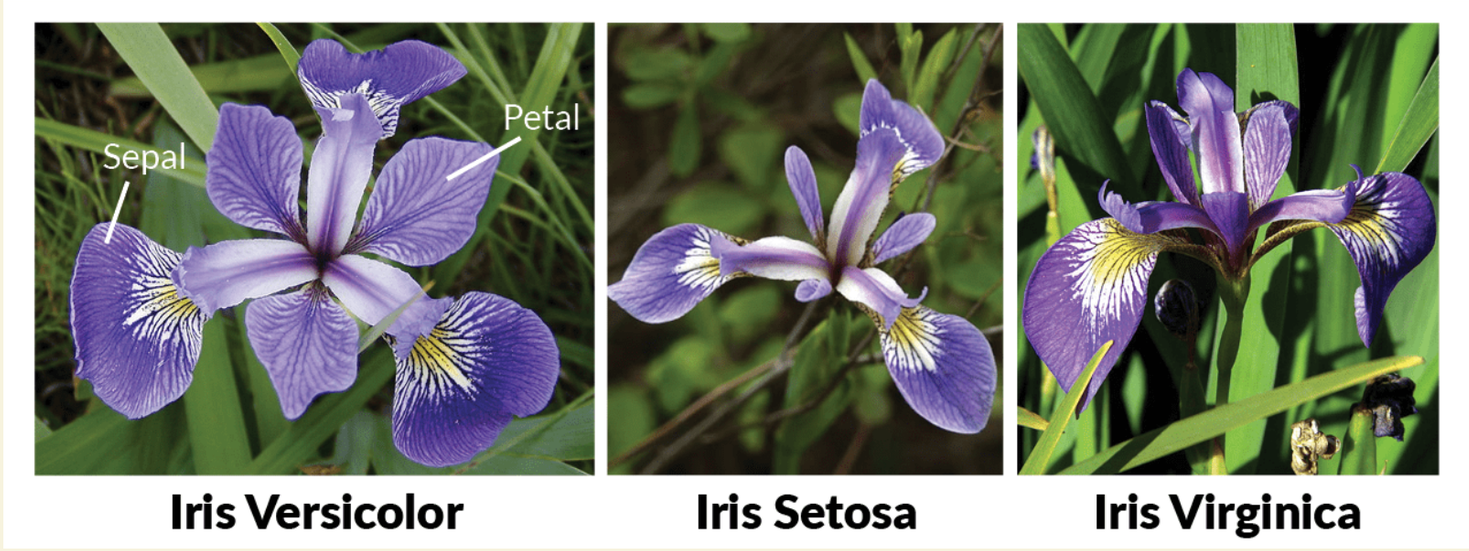


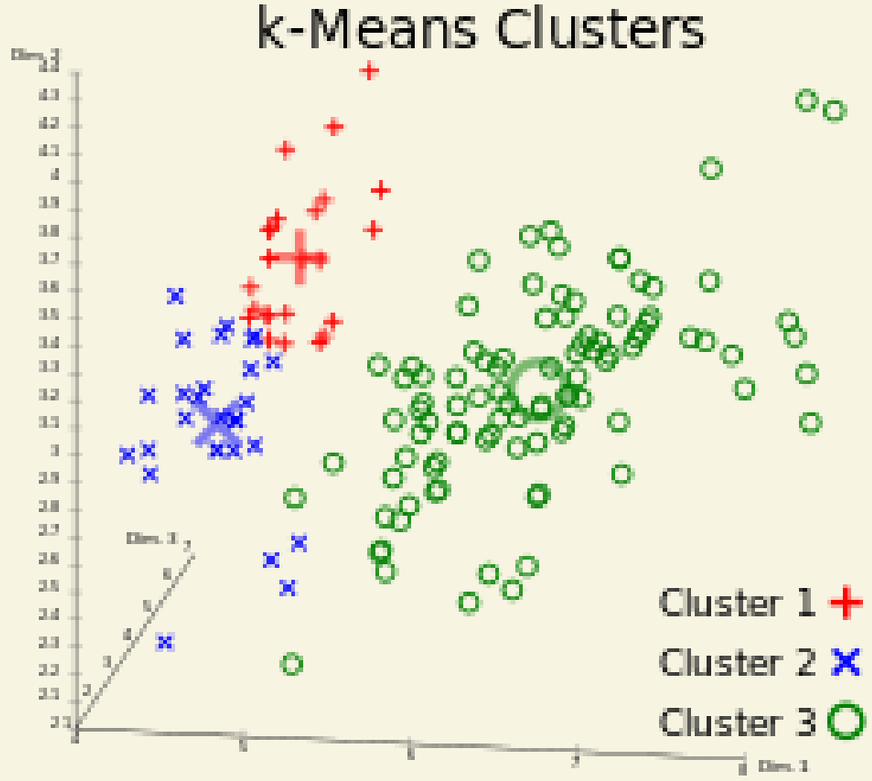
Document everything:

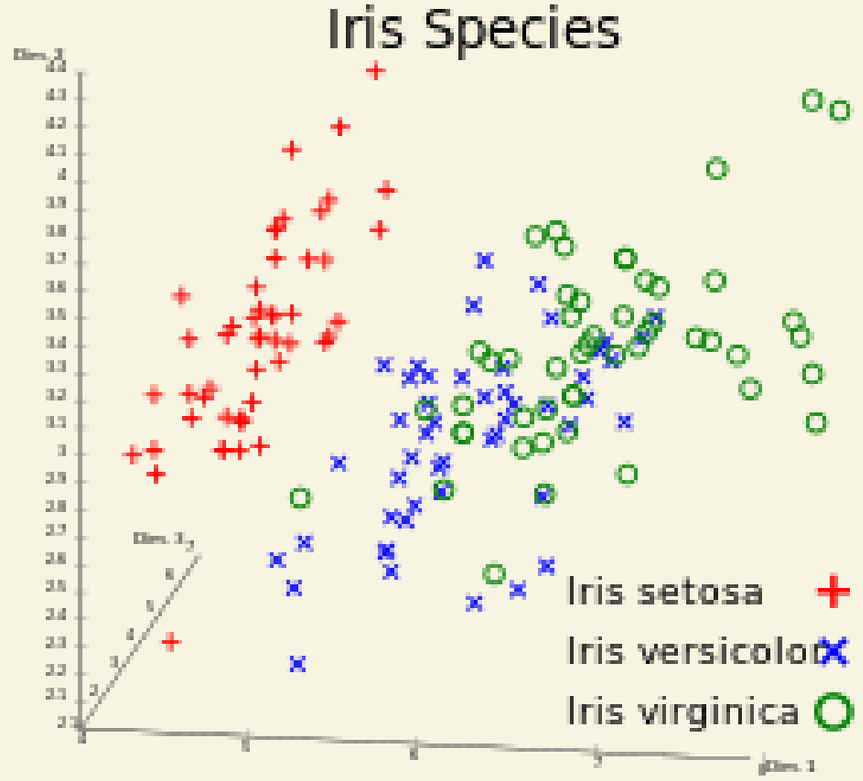
- Which script produced which output? From which data? In which software environment? …

Find out more about organizational principles in the YODA principles!

A CLASSIFICATION ANALYSIS ON THE IRIS FLOWER DATASET





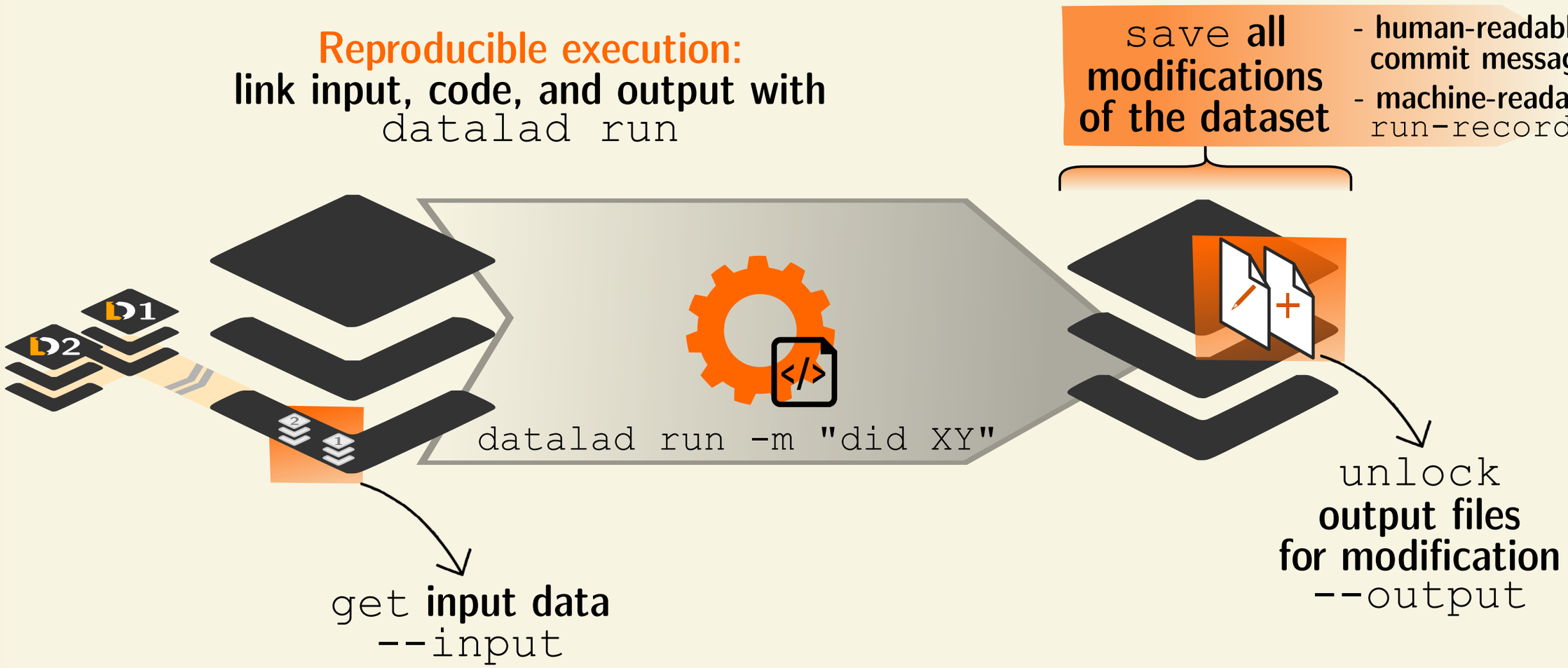


REPRODUCIBLE EXECUTION & PROVENANCE CAPTURE

datalad run

REPRODUCIBLE EXECUTION & PROVENANCE CAPTURE

datalad run



COMPUTATIONAL REPRODUCIBILITY

- Code may fail (to reproduce) if run with different software
- Datasets can store (and share) software environments (Docker or Singularity containers) and reproducibly execute code inside of the software container, capturing software as additional provenance
- DataLad extension: `datalad-container`

datalad-containers run

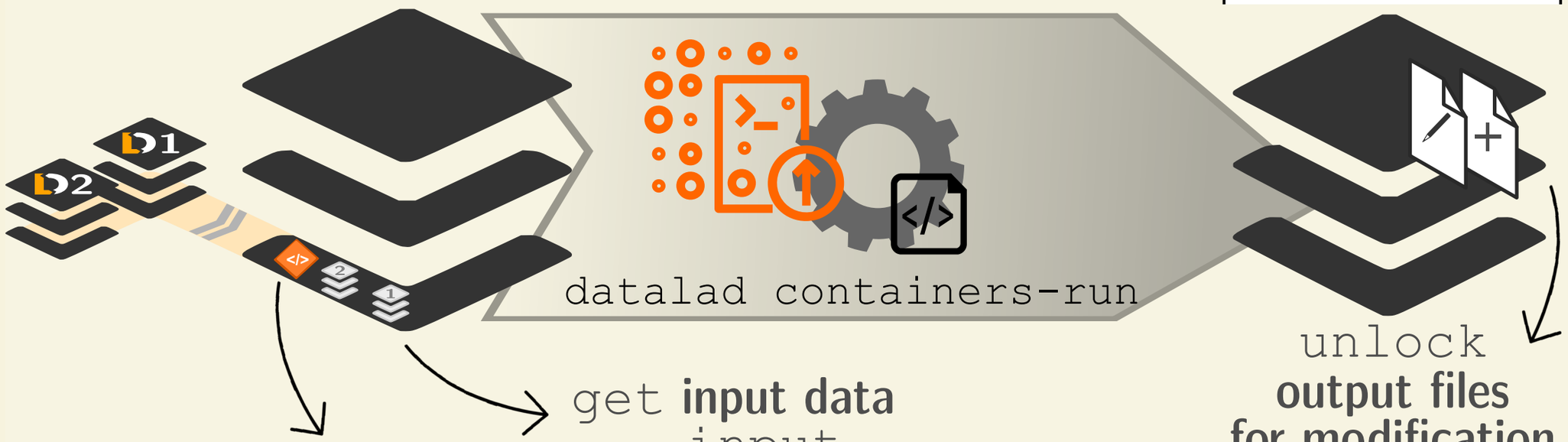
COMPUTATIONAL REPRODUCIBILITY

- Code may fail (to reproduce) if run with different software
- Datasets can store (and share) software environments (Docker or Singularity containers) and reproducibly execute code inside of the software container, capturing software as additional provenance
- DataLad extension: `datalad-container`

datalad-containers run

link input, code, output, and software with
datalad containers-run

save all
modifications
of the dataset



get input data
--input

unlock
output files
for modification
--output

get and use registered software
container for computation
--container-name

SUMMARY - REPRODUCIBLE EXECUTION

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

Use one flag per input.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

Use one flag per input.

Data/directories specified as `--output` will be unlocked for modifications prior to a rerun of the command.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

Use one flag per input.

Data/directories specified as `--output` will be unlocked for modifications prior to a rerun of the command.

Its optional to specify, but helpful for recomputations.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

Use one flag per input.

Data/directories specified as `--output` will be unlocked for modifications prior to a rerun of the command.

Its optional to specify, but helpful for recomputations.

`datalad containers-run` can be used to capture the software environment as provenance.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

Use one flag per input.

Data/directories specified as `--output` will be unlocked for modifications prior to a rerun of the command.

Its optional to specify, but helpful for recomputations.

`datalad containers-run` can be used to capture the software environment as provenance.

Its ensures computations are ran in the desired software set up. Supports Docker and Singularity containers

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

Use one flag per input.

Data/directories specified as `--output` will be unlocked for modifications prior to a rerun of the command.

Its optional to specify, but helpful for recomputations.

`datalad containers-run` can be used to capture the software environment as provenance.

Its ensures computations are ran in the desired software set up. Supports Docker and Singularity containers

`datalad rerun` can automatically re-execute run-records later.

SUMMARY - REPRODUCIBLE EXECUTION

`datalad run` records a command and its impact on the dataset.

All dataset modifications are saved - use it in a clean dataset.

Data/directories specified as `--input` are retrieved prior to command execution.

Use one flag per input.

Data/directories specified as `--output` will be unlocked for modifications prior to a rerun of the command.

Its optional to specify, but helpful for recomputations.

`datalad containers-run` can be used to capture the software environment as provenance.

Its ensures computations are ran in the desired software set up. Supports Docker and Singularity containers

`datalad rerun` can automatically re-execute run-records later.

They can be identified with any commit-ish (hash, tag, range, ...)

QUESTIONS!

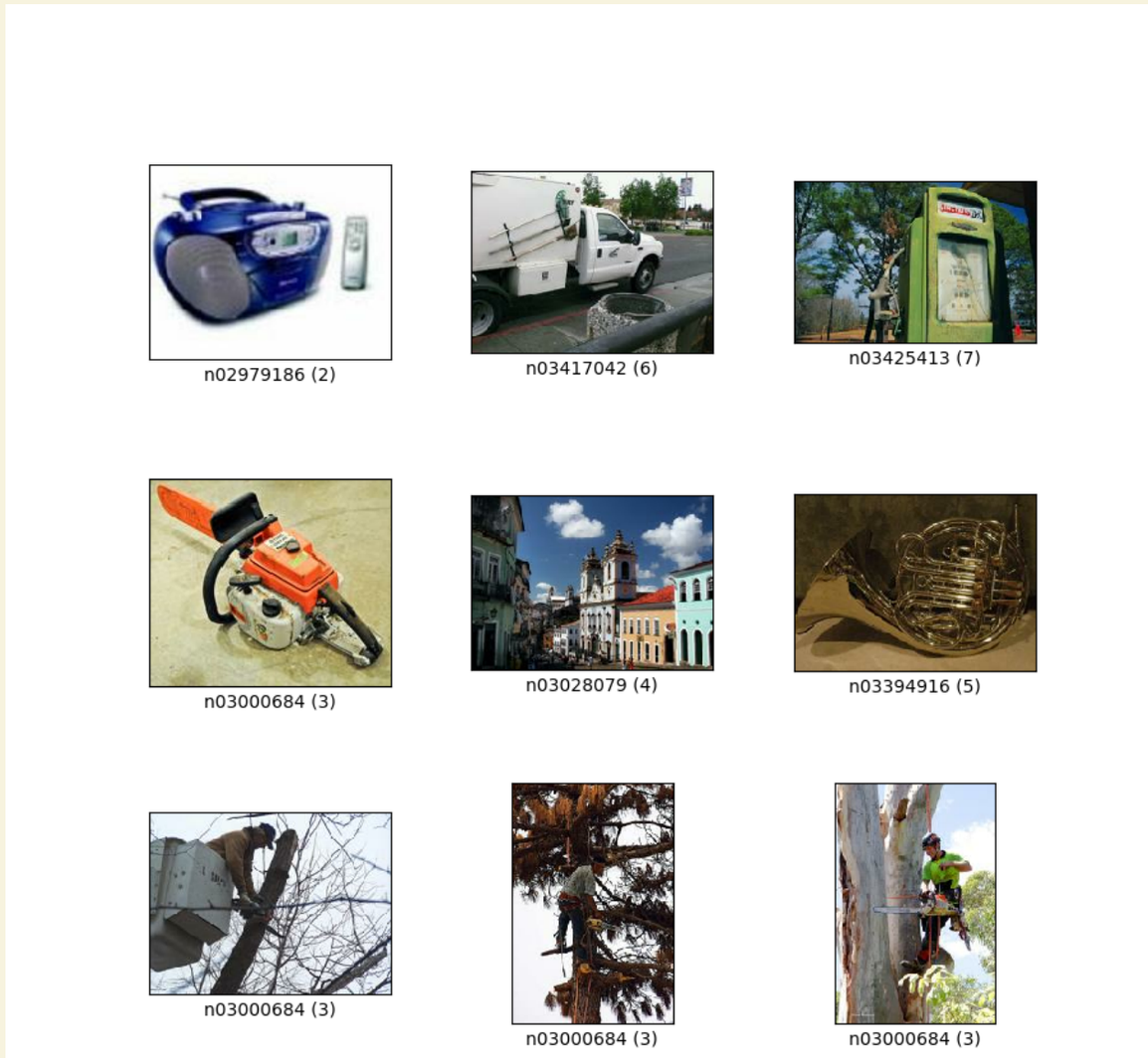
http://etc.ch/y8uM

Direct Poll

A MACHINE-LEARNING EXAMPLE

ANALYSIS LAYOUT

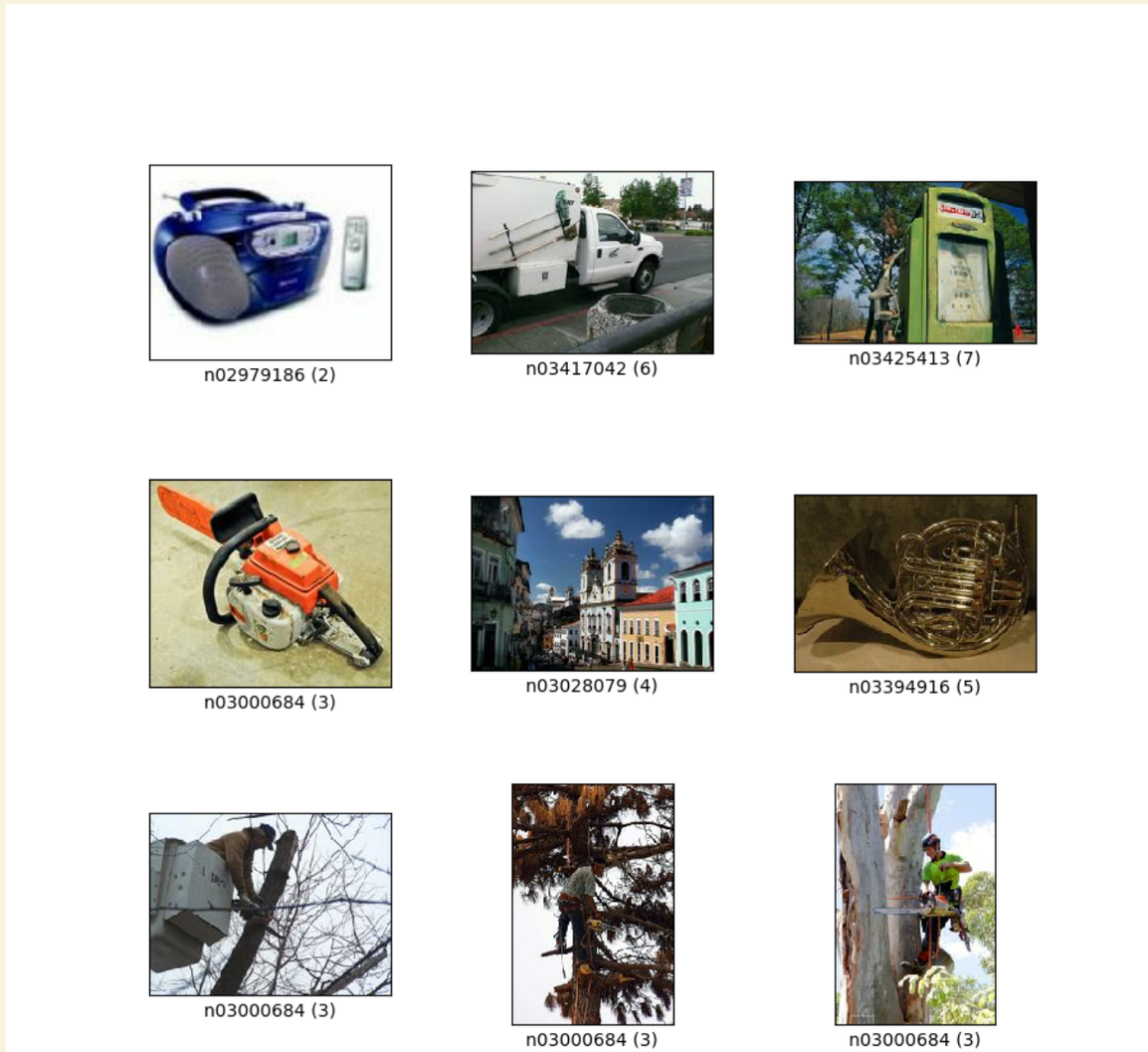
- Prepare an input data set



Imagenette dataset

ANALYSIS LAYOUT

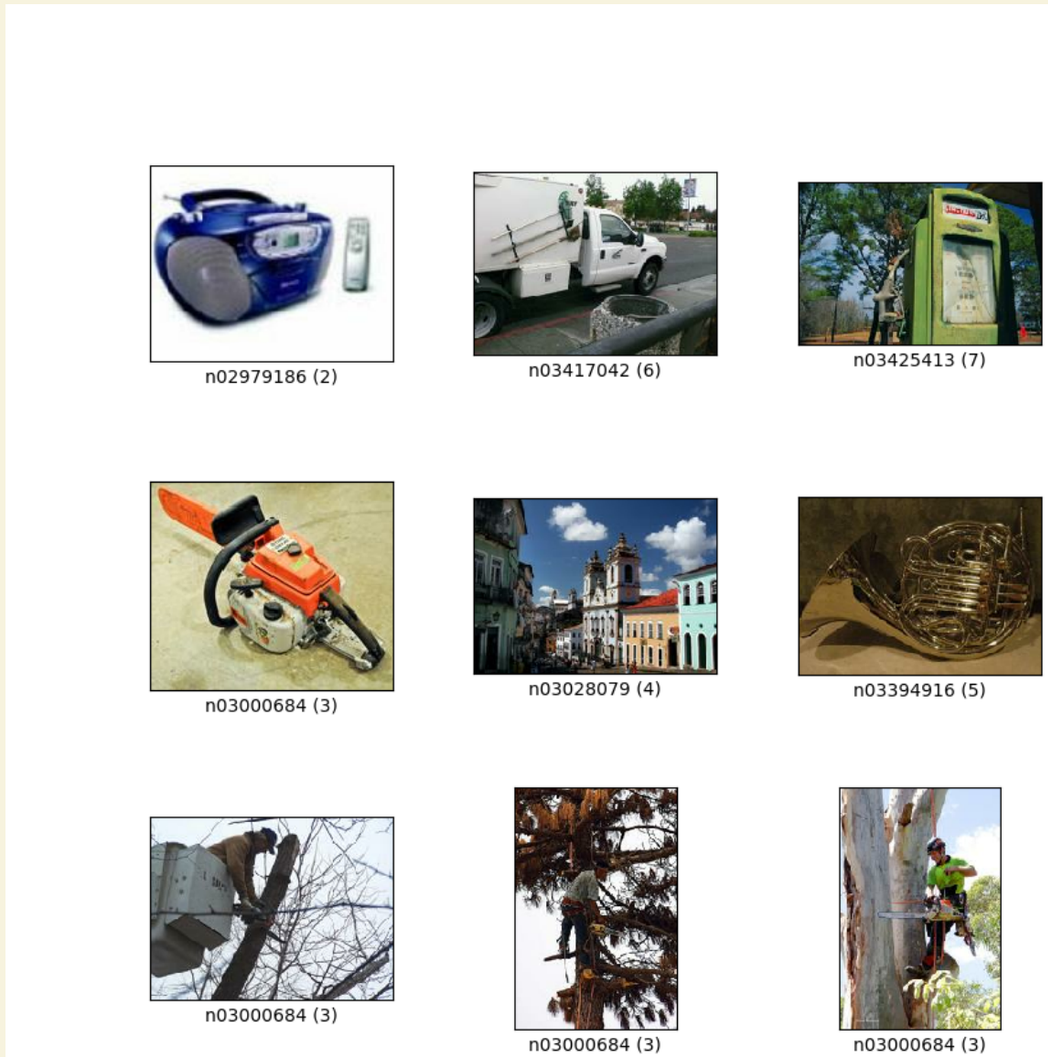
- Prepare an input data set
- Configure and setup an analysis dataset



Imagenette dataset

ANALYSIS LAYOUT

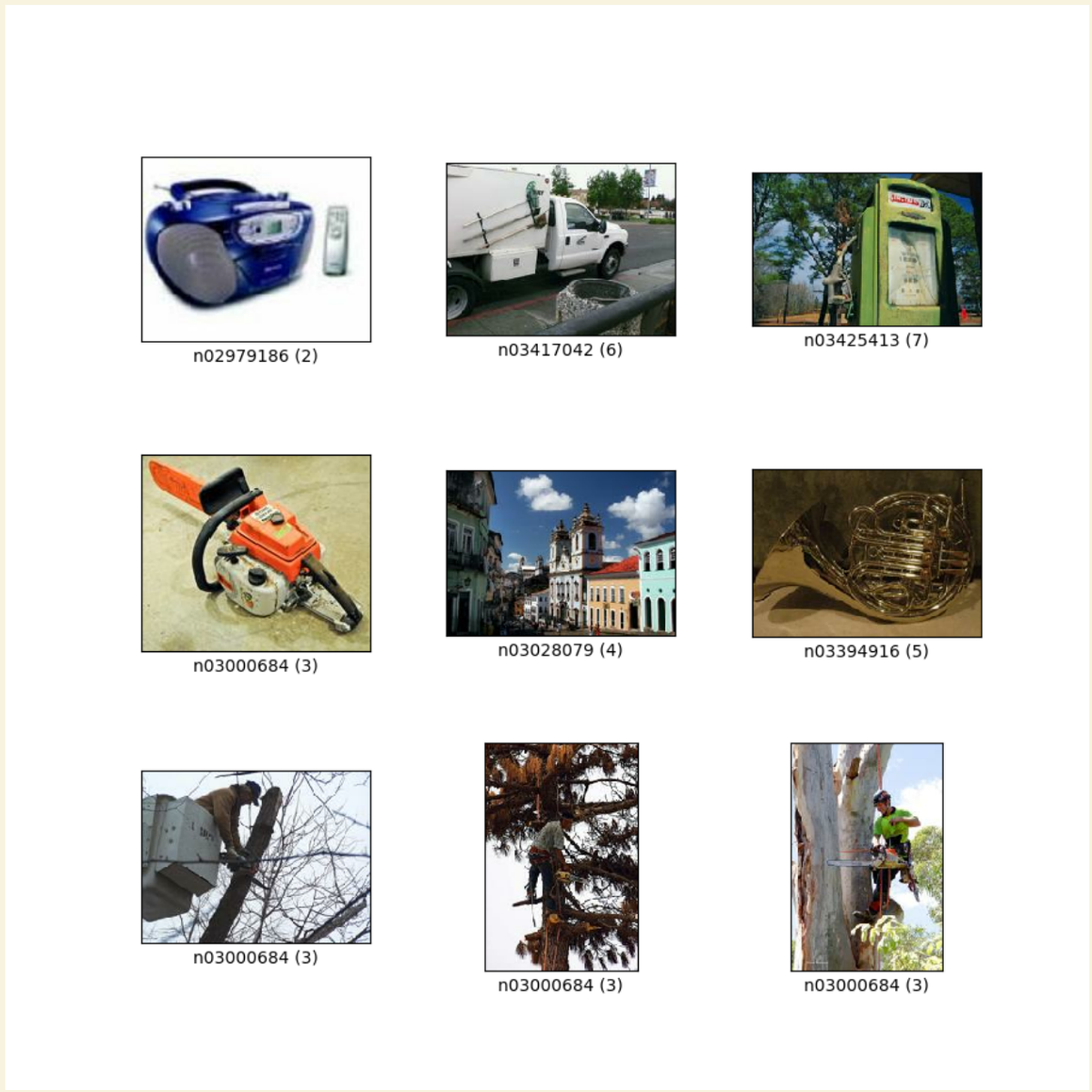
- Prepare an input data set
- Configure and setup an analysis dataset
- Prepare data



Imagenette dataset

ANALYSIS LAYOUT

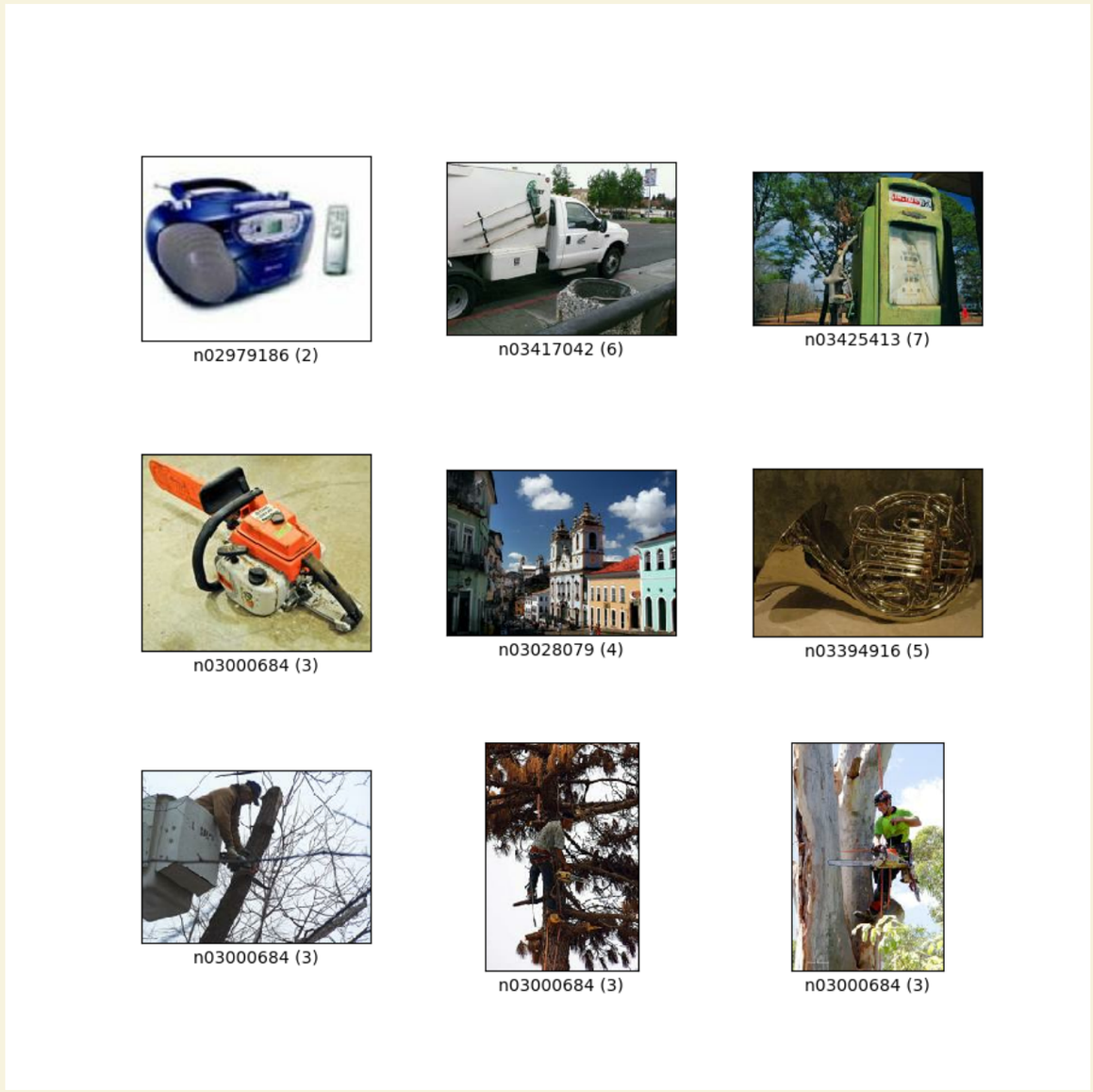
- Prepare an input data set
- Configure and setup an analysis dataset
- Prepare data
- Train models and evaluate them



Imagenette dataset

ANALYSIS LAYOUT

- Prepare an input data set
- Configure and setup an analysis dataset
- Prepare data
- Train models and evaluate them
- Compare different models, repeat with updated data



Imagenette dataset

PREPARE AN INPUT DATASET

- Create a stand-alone input dataset
- Either add data and `datalad save` it, or use commands such as `datalad download-url` or `datalad add-urls` to retrieve it from web-sources

CONFIGURE AND SETUP AN ANALYSIS DATASET

- Given the purpose of an analysis dataset, configurations can make it easier to use:
  - `-c yoda` prepares a useful structure
  - `-c text2git` keeps text files such as scripts in Git
- The input dataset is installed as a subdataset
- Required software is containerized and added to the dataset

PREPARE DATA

- Add a script for data preparation (labels train and validation images)
- Execute it using `datalad containers-run`

TRAIN MODELS AND EVALUATE THEM

- Add scripts for training and evaluation. This dataset state can be tagged to identify it easily at a later point
- Execute the scripts using `datalad containers-run`
- By dumping a trained model as a joblib object the trained classifier stays reusable

TIPS AND TRICKS FOR ML APPLICATIONS

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

Subdatasets can be registered in precise versions, and updated to the newest state

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

Subdatasets can be registered in precise versions, and updated to the newest state

Software containers aid greatly with reproducibility

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

Subdatasets can be registered in precise versions, and updated to the newest state

Software containers aid greatly with reproducibility

The correct software environment is preserved and can be shared

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

Subdatasets can be registered in precise versions, and updated to the newest state

Software containers aid greatly with reproducibility

The correct software environment is preserved and can be shared

Re-executable run-records can capture all provenance

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

Subdatasets can be registered in precise versions, and updated to the newest state

Software containers aid greatly with reproducibility

The correct software environment is preserved and can be shared

Re-executable run-records can capture all provenance

This can also capture command-line parametrization

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

Subdatasets can be registered in precise versions, and updated to the newest state

Software containers aid greatly with reproducibility

The correct software environment is preserved and can be shared

Re-executable run-records can capture all provenance

This can also capture command-line parametrization

Git workflows can be helpful elements in ML workflows

TIPS AND TRICKS FOR ML APPLICATIONS

Standalone input datasets keep input data extendable and reusable

Subdatasets can be registered in precise versions, and updated to the newest state

Software containers aid greatly with reproducibility

The correct software environment is preserved and can be shared

Re-executable run-records can capture all provenance

This can also capture command-line parametrization

Git workflows can be helpful elements in ML workflows

DataLad is no workflow manager, but by checking out out tags or branches one can switch easy and fast between results of different models

WHY USE DATALAD?

WHY USE DATALAD?

- Mistakes are not forever anymore: Easy version control, regardless of file size

WHY USE DATALAD?

- Mistakes are not forever anymore: Easy version control, regardless of file size
- Who needs short-term memory when you can have run-records?

WHY USE DATALAD?

- Mistakes are not forever anymore: Easy version control, regardless of file size
- Who needs short-term memory when you can have run-records?
- Disk-usage magic: Have access to more data than your hard drive has space

WHY USE DATALAD?

- Mistakes are not forever anymore: Easy version control, regardless of file size
- Who needs short-term memory when you can have run-records?
- Disk-usage magic: Have access to more data than your hard drive has space
- Collaboration and updating mechanisms: Alice shares her data with Bob. Alice fixes a mistake and pushes the fix. Bob says "datalad update" and gets her changes. And vice-versa.

WHY USE DATALAD?

- Mistakes are not forever anymore: Easy version control, regardless of file size
- Who needs short-term memory when you can have run-records?
- Disk-usage magic: Have access to more data than your hard drive has space
- Collaboration and updating mechanisms: Alice shares her data with Bob. Alice fixes a mistake and pushes the fix. Bob says "datalad update" and gets her changes. And vice-versa.
- Transparency: Shared datasets keep their history. No need to track down a former student, ask their project what was done.

THANK YOU FOR YOUR ATTENTION!

QUESTIONS!

http://etc.ch/y8uM

Direct Poll

MORE USEFUL INFORMATION

(BUT PLEASE ASK IF THERE'S ANYTHING ELSE YOU WANT TO KNOW)

SCALABILITY

How large can datasets get?

- In general: **Size is not a problem**, as long as large files are handled with git-annex
- Bottle-neck: **Number of files**. 100-200k per dataset!
- How to scale up? Nest datasets as subdatasets
    - Currently largest dataset: human connectome project data, 80TB, 15 million files, ~4500 subdataset
    - Currently in the making: institute archival system, 125TB
    - Currently worked towards: Processing UKBiobank data, 0.5PB

Need more information? Read the chapter on scaling up at handbook.datalad.org

TRANSFER EXISTING PROJECTS

Can existing projects become dataset?

- In general: **Yes** (but make a backup, just in case)
- `datalad create -f` can transform any directory or Git repository into a dataset
- Afterwards, untracked contents need to be saved

Need more information? Read the section on transitioning existing projects to DataLad at handbook.datalad.org

EASY STORAGE FOR ANNEX

How can I share annexed data fast and easy?

- In general: **Dozens of services are supported** via git-annex "special remote" concept, (Amazon S3, Dropbox, GDrive, private webserver...)
- Some repository hosting services also host annexed data: GIN (free, supports anonymous read access), GitHub & GitLab if used with GitLFS (non-free)
- **datalad-osf** extension for hosting datasets on the open science frameworf (OSF)

Need more information? Read the chapter on third party infrastructure at handbook.datalad.org

THANKS FOR YOUR ATTENTION!

Speak up now or reach out to us later with any questions