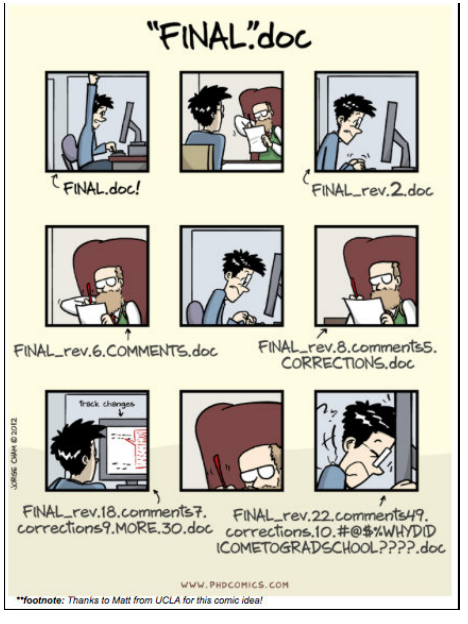
W4995 Applied Machine Learning

Git, Github and Testing

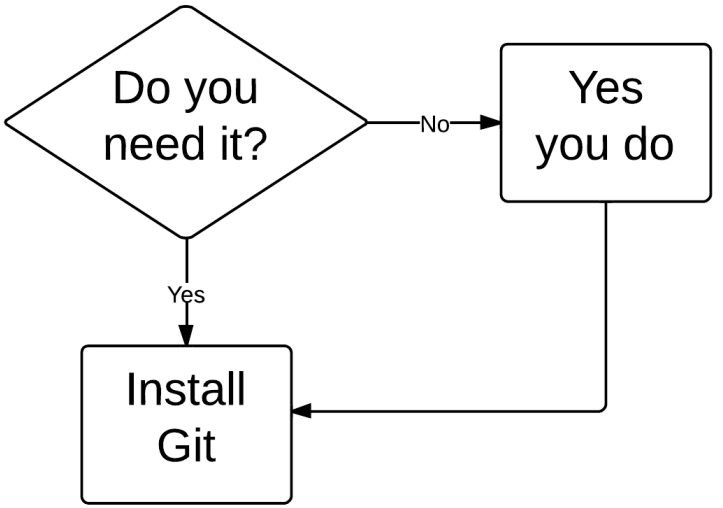
01/28/19

Andreas C. Müller

Why do I care?


"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
WWW.PHDCOMICS.COM
**footnote: Thanks to Matt from UCLA for this comic idea!

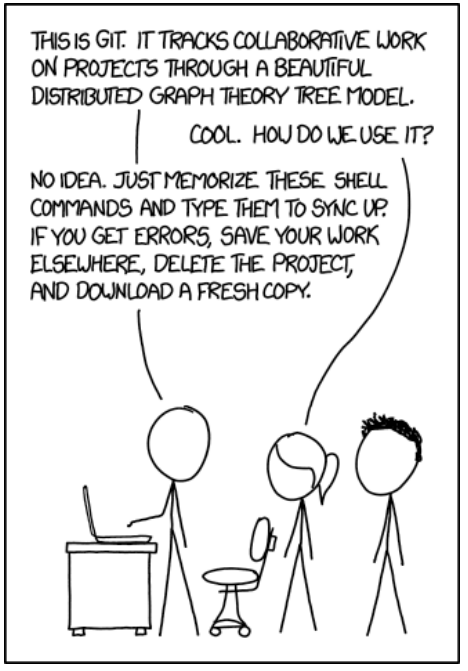
Version Control Flowchart



Git and Github

https://guides.github.com/

http://rogerdudler.github.io/git-guide/


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.


Discard usability
Obtain arduousness

Configuration

```
git config --global user.name "Andreas Mueller"
git config --global user.email "acm2248@columbia.edu"
git config --global color.ui "auto"
git config --global core.editor "vim"
```

Show your configuration:

```
git config --list
```

Creating a repository

```
$ mkdir homework1
$ cd homework1
```

```
$ git init
```

```
> Initialized empty Git repository in /tmp/homework1/.git/
```

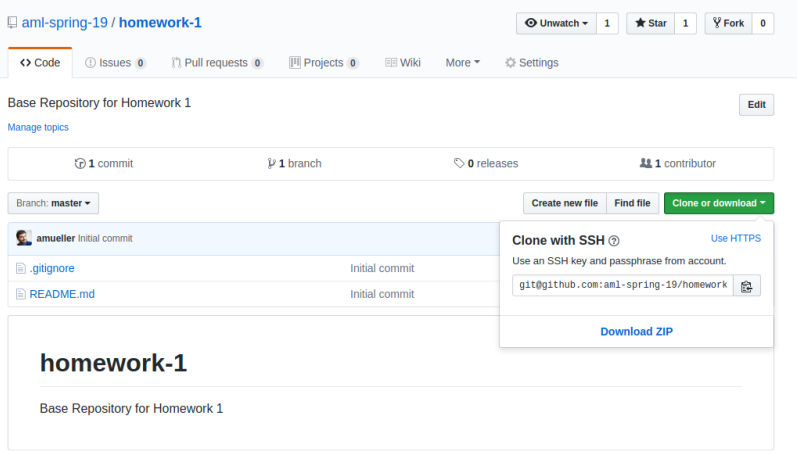
```
$ ls -a
```

```
.  ..  .git
```

```
$ git status
```

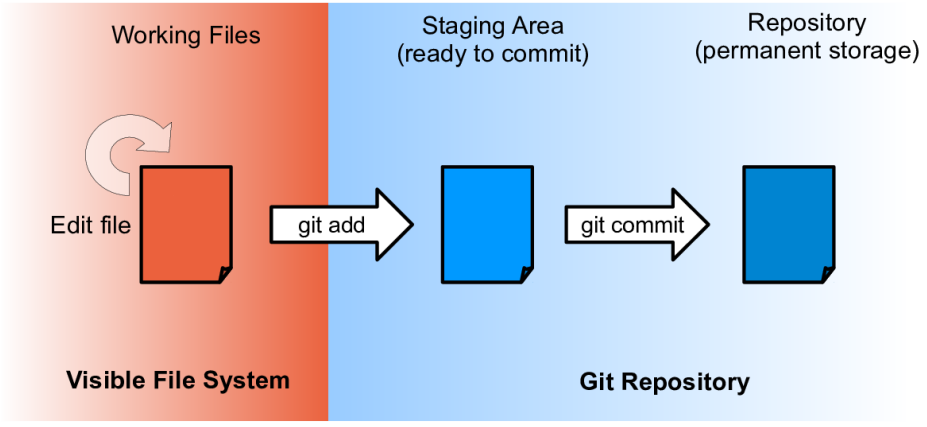
```
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

Cloning a repository




```
$ git clone git@github.com:aml-sprint-19/homework-1
```

Workflow?


Working Files
Staging Area
(ready to commit)
Repository
(permanent storage)
Edit file
git add
git commit
Visible File System
Git Repository

Changing a File

```
$ echo "print('Hello world!')" >> task1.py
```

```
$ git status
```

```
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    task1.py

nothing added to commit but untracked files present (use "git add" to track)
```

```
$ git add task1.py
```

Viewing your history

```
$ git commit -m "say hello"
```

```
[master (root-commit) 7d139fb] say hello
 1 file changed, 1 insertion(+)
 create mode 100644 task1.py
```

```
$ git log
```

```
commit 7d139fb317ecfa7d629654b709747e91ececa444 (HEAD -> master)
Author: Andreas Mueller <andreas.mueller@columbia.edu>
Date:   Mon Jan 28 12:37:29 2019 -0500

    say hello
```

A note on viewing changes

Changes between working directory and what was last staged

```
git diff
```

Changes between staging area and last commit

```
git diff --staged
```

Referencing different versions

- Shorthand for different versions of a repository (refers to commits)
  - Current Version (most recent commit): `HEAD`
  - Version before current: `HEAD~1`
  - Version before that: `HEAD~2`
- Each of these also has a commit hash
  - use `git log` to get appropriate hash

Exploring History

Changes made in the last commit

```
git diff HEAD~1
```

Changes made in the last 2 commits

```
git diff HEAD~2
```

Changes made since commit hash...

```
git diff 0b0d55e
```

Recovering Older Versions of Files

- Overwrite task1.py:

```
$ echo "print('goodbye, cruel world!')" >  task1.py
$ cat task1.py
```

```
print('goodbye, cruel world!')
```

Recover last recorded version:

```
$ git checkout HEAD task1.py
$ cat task1.py
```

```
print('Hello world!')
```

Recovering Older Versions of whole repo

```
$ echo "print('goodbye, cruel world!')" >  task1.py
$ git add task1.py
$ git commit -m " saying goodbye"
```

```
[master 95f3b39]  saying goodbye
 1 file changed, 1 insertion(+), 1 deletion(-)
```

$ git log

```
commit 95f3b39402451c474e3887be94d2cc37a11be511 (HEAD -> master)
Author: Andreas Mueller <andreas.mueller@columbia.edu>
Date:   Mon Jan 28 12:44:52 2019 -0500

     saying goodbye

commit 7d139fb317ecfa7d6206f4b70974e01eeea444
```

Resetting:

```
$ git reset --hard HEAD~1
```

```
HEAD is now at 7d139fb say hello
```

Branches

```
$ git checkout -b "new_feature"
```

```
Switched to a new branch 'new_feature'
```

make some changes, add, commit...

Moving between branches:

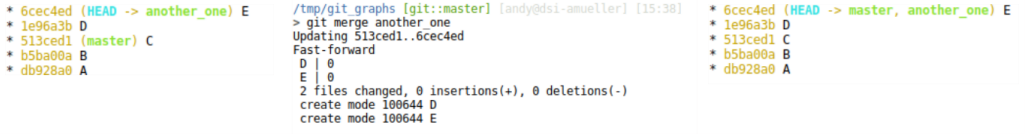
```
$ git checkout master
```

```
Switched to branch 'master'
```

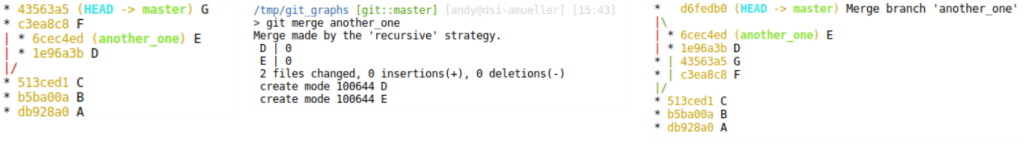
changes are not present in master..

Merge

- Fast-forward merge:



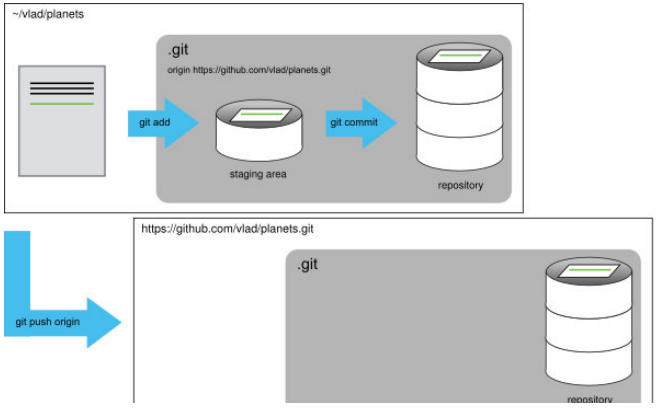
- Merge-commits:



Collaborating using git & github

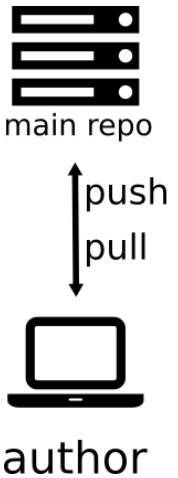
Remote repository

- central location everyone can see
- requires network ?
- github, bitbucket
- public vs private

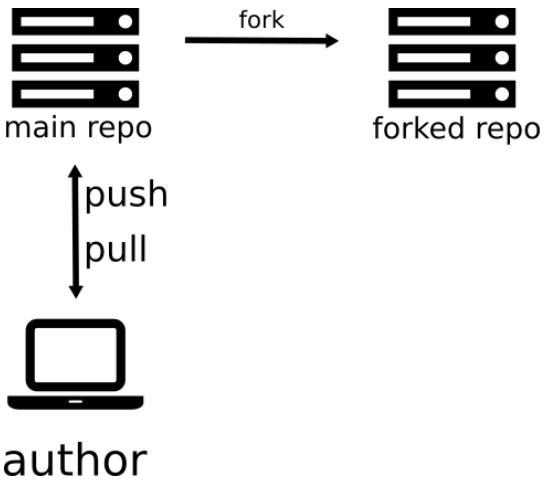


Github

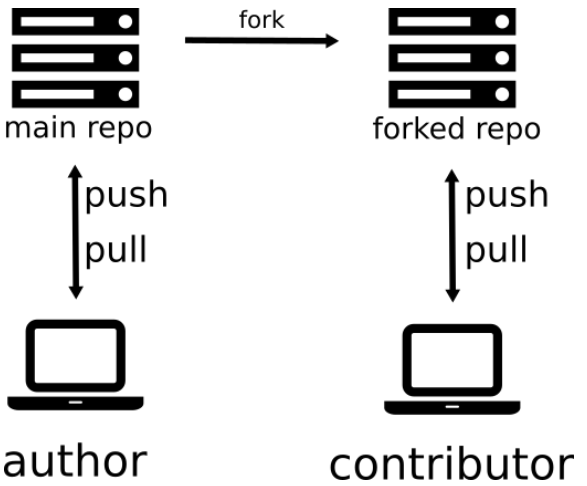
Github pull request workflow



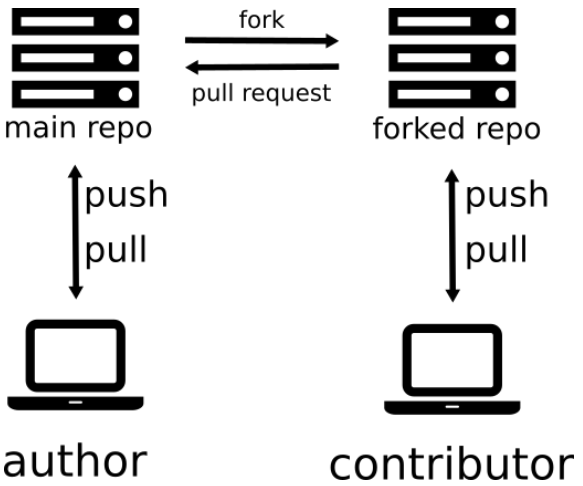
Github pull request workflow



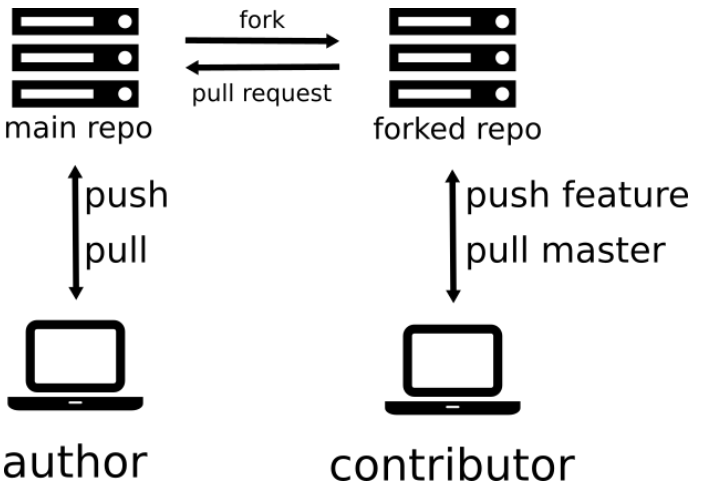
Github pull request workflow



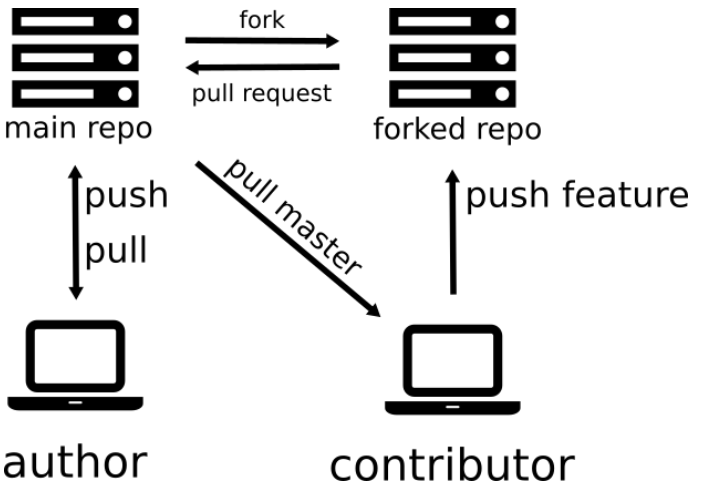
Github pull request workflow



Github pull request workflow



Github pull request workflow



Typical Workflow

- Clone
- Branch
- Add, commit, add, commit, add, commit, ...
- Merge / rebase
- Push

Some tips

- `$ git status`
- Install shell plugins for status and branch (`oh-my-zsh`)

```
/home/andy/checkout/scikit-learn [git::master *] [andy@dsi-amueller] [16:35]
> 
```

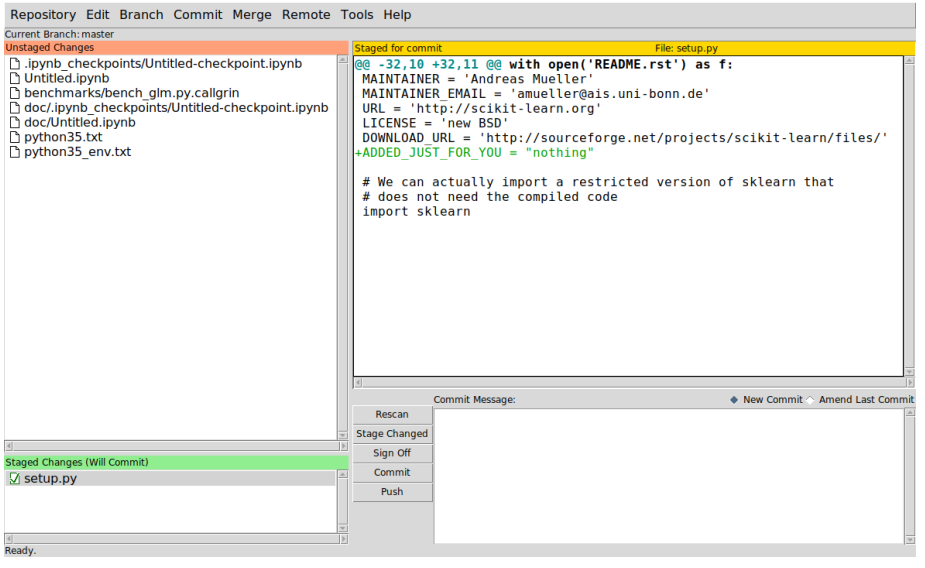
- Set editor, pager and diff-tool (check out meld!)
- Use `.gitignore`

Git log

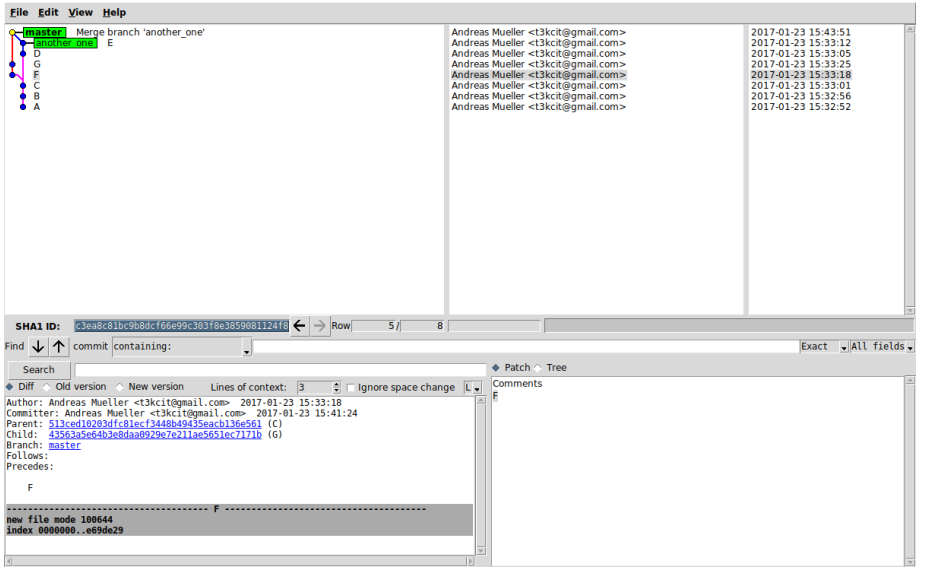
```
/home/andy/checkout/scikit-learn [git::master *] [andy@dsi-amueller] [16:43]
> git log --oneline --decorate --all --graph -n 50
* 9616ac7 (HEAD -> master, upstream/master) CI remove obsolete comment
* 7978119 [MRG] #8218: in FAQ, link deep learning question to GPU question (#8220)
* be305ce TST/FIX Add check for estimator: parameters not modified by `fit` (#7846)
* aaebee1 FIX Issue #8173 - pass n_neighbors in MI computation (#8181)
* 4826883 Call sorted on lfw folder path contents (#7648)
* 4907029 [MRG+3] FIX Memory leak in MAE; Use safe_realloc; Acquire GIL only when raising; Propagate all errors to python interpreter level (#7811) (#8002)
* 08772cd FIX Ensure coef_ is an ndarray when fitting LassoLars (#8160)
* 5d6460d MNT/BLD Use GitHub's merge refs to test PRs on CircleCI (#8211)
* 66443aa [MRG+1] Add prominent mention of Laplacian Eigenmaps (#8155)
* bddda7b [MRG + 2] [MAINT] Update to Sphinx-Gallery 0.1.7 (#7986)
* aea6462 [MRG+1] Fixes #8198 - error in datasets.make_moons (#8199)
* 0eb33ad TRAVIS fix flake8_diff.sh check_files (#8208)
* eabaef3 DOC add missing parentheses in TfidfTransformer docstring
* 3dc8d2f DOC additional fixes to 20 newsgroups to prevent TypeError (#8204)
* cbddb92 removed stray space in '__main__ ' (#8203)
* ca687ba Upgrade html documentation to jQuery v3.1.1 (#8145)
* 84cc67b Clarify error message for min_samples_split. (#8167)
* 2a1408a fixing typo in cs_mse_path_ deprecation (#8176)
* fd84a56 [MRG + 1] Fix the cross_val_predict function for method='predict_proba' (#7889)
* 4910e11 DOC Fix link (#8171)
* 8998856 Fix Ridge floating point instability (#8154)
* 2f7f5a1 [MRG + 1] Add fowlkess-mallows and other supervised cluster metrics to SCORERS dict so it can be used in hyper-param search (#8117)
* 8695ff5 [MRG + 1] add partial_fit to multioutput module (#8054)
* 0b02125 [MRG] FIX Avoid default mutable argument in constructor of AgglomerativeClustering (#8153)
* d0ce0d9 [MRG+2] Avoid failure in first iteration of RANSAC regression (#7914)
* e874398 [MRG+1] DOC: complete list of online learners (#8152)
* e0c60fe FIX sphinx gallery rendering of plot_digits_pipe example
* 84349a7 [MRG+1] Deprecate ridge_alpha param on SparsePCA.transform() (#8137)
* c49ced9 DOC: updating GridSearchCV's n_jobs parameter (#8106)
* 2cb7e47 [MRG+1] fowlkes_mallows_score: more unit tests (Fixes #8101) (#8140)
* 543b056 [MRG+1] Add DBSCAN support for additional metric params (#8139)
* d7e77ce [MRG+1] Fix "cite us" link in sidebar (#8142)
* 288827b [MRG] update copyright years for 2017 (#8138)
* e2adbb7 DOC Fix typo in FAQ (#8132)
* 986a49b FIX Split data using _safe_split in _permutaion_test_score (#5697)
* ab1c4d4 [MRG+1] Catch cases for different class size in MLPClassifier with warm start (#7976)  (#8035)
| * dcf24b9 (origin/repr_give_up, repr_give_up) i have no idea what I'm doing
| * b25fa4b pep8
| * 8b30325 playing around, then giving up
| | * 0d1398a (upstream/ignore_lambda_to_diff_errors) MNT Ignore E731: Use a def instead of lambda
| |/
|/|
* | d97d13e DOC add sklearn-crfsuite to related projects (#7878)
* | fcb706a [MRG+3] Fused types for MultiTaskElasticNet (#8061)
* | 096a9cb [MRG] MAINT Python 3.6 fixes (#8123)
* | e088685 DOC Fix indentation errors and username links (#8121)
* | 4f3c60c [MRG+2] FIX IsolationForest(max_features=0.8).predict(X) fails input validation (#5757)
```


I DON'T DO GIT LOG ALL THE TIME,
BUT WHEN I DO JUST REMEMBER
(A DOG) --ALL --DECORATE --ONELINE --GRAPH
memegenerator.net

git gui


Repository Edit Branch Commit Merge Remote Tools Help
Current Branch: master
Unstaged Changes
.ipynb_checkpoints/Untitled-checkpoint.ipynb
Untitled.ipynb
benchmarks/bench_glm.py.callgrin
doc/.ipynb_checkpoints/Untitled-checkpoint.ipynb
doc/Untitled.ipynb
python35.txt
python35_env.txt
Staged for commit
File: setup.py
@@ -32,10 +32,11 @@ with open('README.rst') as f:
MAINTAINER = 'Andreas Mueller'
MAINTAINER_EMAIL = 'amueller@ais.uni-bonn.de'
URL = 'http://scikit-learn.org'
LICENSE = 'new BSD'
DOWNLOAD_URL = 'http://sourceforge.net/projects/scikit-learn/files/'
+ADDED_JUST_FOR_YOU = "nothing"
# We can actually import a restricted version of sklearn that
# does not need the compiled code
import sklearn
Commit Message:
New Commit
Amend Last Commit
Rescan
Stage Changed
Sign Off
Commit
Push
Staged Changes (Will Commit)
setup.py
Ready.

gitk

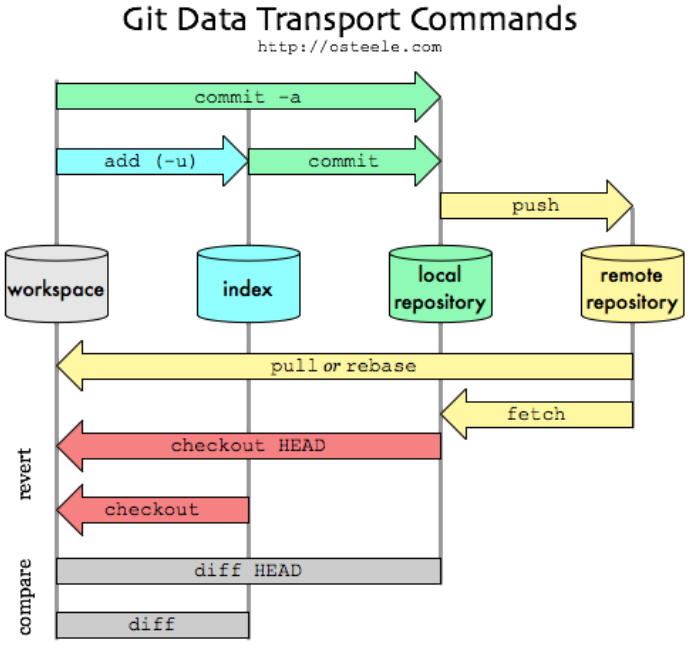

File Edit View Help
master Merge branch 'another_one'
another one E
D
G
F
C
B
A
Andreas Mueller <t3kcit@gmail.com>
2017-01-23 15:43:51
2017-01-23 15:33:12
2017-01-23 15:33:05
2017-01-23 15:33:25
2017-01-23 15:33:18
2017-01-23 15:33:01
2017-01-23 15:32:56
2017-01-23 15:32:52
SHA1 ID: c3ea8c81bc9b8dcf66e99c303f8e3859081124f8
Row 5 / 8
Find commit containing:
Exact
All fields
Search
Patch Tree
Diff Old version New version Lines of context: 3 Ignore space change
Author: Andreas Mueller <t3kcit@gmail.com> 2017-01-23 15:33:18
Commiter: Andreas Mueller <t3kcit@gmail.com> 2017-01-23 15:41:24
Parent: 513ced10203dfc81ecf3448b49435eacb136e561 (C)
Child: 43563a5e64b3e8daa0929e7e211ae5651ec7171b (G)
Branch: master
Follows:
Precedes:
F
---------- F ----------
new file mode 100644
index 0000000..e69de29
Comments
F

Understanding Git

- Working directory
- Repository (Commit graph, history)
- Index (Staging Area)
- Branches
- Head

git Commands

- `git add`
  puts files from working director into staging area (index) If not tracked so far, adds them to tracked files.
- `git commit`
  commits files from staging area (index) to repository, moves current branch with HEAD
- `git checkout [<commit>] [<file>]`
  Set `<file>` in working directory to state at `<commit>` and stages it.
- `git checkout [-b] <branch>`
  moves HEAD to `<branch>` (-b creates it), changes content of working dir
- `git reset --soft <commit>`
  moves HEAD to `<commit>` (takes the current branch with it)
- `git reset --mixed <commit>`
  moves HEAD to `<commit>`, changes index to be at `<commit>` (but not working directory)
- `git reset --hard <commit>`
  moved HEAD to `<commit>`, changes index and working tree to `<commit>`.


Git Data Transport Commands
http://osteele.com
commit -a
add (-u)
commit
push
workspace
index
local repository
remote repository
pull or rebase
fetch
checkout HEAD
checkout
diff HEAD
diff
revert
compare

reflog

```
$ git reflog
```

```
d6fedb0 HEAD@{0}: merge another_one: Merge made by the 'recursive' strategy.
43563a5 HEAD@{1}: rebase finished: returning to refs/heads/master
43563a5 HEAD@{2}: rebase: G
c3ea8c8 HEAD@{3}: rebase: F
513ced1 HEAD@{4}: rebase: checkout 513ced1
4db426c HEAD@{5}: merge feature: Fast-forward
b5ba00a HEAD@{6}: checkout: moving from master to master
b5ba00a HEAD@{7}: reset: moving to HEAD~3
6cec4ed HEAD@{8}: merge another_one: Fast-forward
513ced1 HEAD@{9}: checkout: moving from another_one to master
6cec4ed HEAD@{10}: reset: moving to 6cec4ed
4db426c HEAD@{11}: checkout: moving from feature to another_one
4db426c HEAD@{12}: checkout: moving from master to feature
513ced1 HEAD@{13}: reset: moving to HEAD~4
4db426c HEAD@{14}: checkout: moving from feature to master
4db426c HEAD@{15}: checkout: moving from master to feature
4db426c HEAD@{16}: commit: G
125e957 HEAD@{17}: commit: F
6cec4ed HEAD@{18}: commit: E
1e96a3b HEAD@{19}: commit: D
513ced1 HEAD@{20}: commit: C
b5ba00a HEAD@{21}: commit: B
db928a0 HEAD@{22}: commit (initial): A
```

```
> git log --oneline --graph --decorate --all
*   d6fedb0 (HEAD -> master) Merge branch 'another_one'
|\
| * 6cec4ed (another_one) E
| * 1e96a3b D
* | 43563a5 G
* | c3ea8c8 F
|/
* 513ced1 C
* b5ba00a B
* db928a0 A
```

Git for ages 4 and up:

https://www.youtube.com/watch?v=1ffBJ4sVUb4

(with play-doh!)

Unit Tests and integration tests

Why test?

- Ensure that code works correctly.
- Ensure that changes don’t break anything.
- Ensure that bugs are not reintroduced.
- Ensure robustness to user errors.
- Ensure code is reachable.

Test-driven development?

Types of tests

- Unit tests – function does the right thing.
- Integration tests – system / process does the right thing.
- Non-regression tests – bug got removed (and will not be reintroduced).

How to test?

- pytest – http://doc.pytest.org
- Searches for all test * methods.
- Reports nice errors!
- Dig deeper: http://pybites.blogspot.com/2011/07/behind-scenes-of-pytests-new-assertion.html

Example

```
 (x):
     x + 2


inc        inc

           ():
     inc(3) == 4
```

```
> py.test test_sample.py
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-2.9.2, py-1.4.31, pluggy-0.3.1
rootdir: /tmp, inifile:
plugins: cov-2.4.0, timeout-1.2.0
collected 1 items

test_sample.py F

=================================== FAILURES ===================================
_____________________________ test_answer _____________________________
```

Example

```
      (x):
          x + 1


    inc         inc


               ():
          inc(3) == 4
```

```
> py.test test_sample.py
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-2.9.2, py-1.4.31, pluggy-0.3.1
rootdir: /tmp, inifile:
plugins: cov-2.4.0, timeout-1.2.0
```

Test coverage

```
 (x):
x < 0:
        0
    x + 1


 (x):
     x - 1
```

```
inc        inc

      ():
      inc(3) == 4
```

```
> py.test
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-2.9.2, py-1.4.31, pluggy-0.3.1
rootdir: /tmp/myproj, inifile:
plugins: cov-2.4.0, timeout-1.2.0
collected 1 items

test_inc.py .

=========================== 1 passed in 0.01 seconds ===========================
```

Test coverage

```
 (x):
x < 0:
        0
    x + 1


 (x):
     x - 1
```

```
inc        inc

      ():
      inc(3) == 4
```

```
/tmp/myproj [andy@dsi-amueller] [10:51]
> py.test --cov inc
========================== test session starts ===========================
platform linux -- Python 3.5.2, pytest-2.9.2, py-1.4.31, pluggy-0.3.1
rootdir: /tmp/myproj, inifile:
plugins: cov-2.4.0, timeout-1.2.0
collected 1 items

test_inc.py .

---------- coverage: platform linux, python 3.5.2-final-0 ----------
Name     Stmts   Miss  Cover
--------------------------
inc.py       6      2    67%
```

HTML report

```
$ pytest --cov inc --cov-report=html
```

Coverage report: 67%

Module ↓	*statements*	*missing*	*excluded*	*coverage*
inc.py	6	2	0	67%
Total	**6**	**2**	**0**	**67%**

coverage.py v4.2, created at 2017-01-23 10:53

Coverage for **inc.py** : 67%

6 statements | 4 run | 2 missing | 0 excluded

```
1  # inc.py
2  def inc(x):
3      if x < 0:
4          return 0
5      return x + 1
6
7
8  def dec(x):
9      return x - 1
```

Continuous integration (with GitHub)

What is Continuous integration?

- Run command on each commit (or each PR).
- Unit testing and integration testing.
- Can act as a build-farm (for binaries or documentation).
- requires clear declaration of dependencies.
- Build matrix: Can run on many environments.
- Standard serviced: TravisCI, Appveyor, Azure Pipelines and CircleCI

Benefits of CI

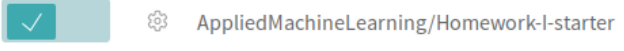
- Can run on many systems
- Can’t forget to run it
- Contributor doesn’t need to know details
- Can enforce style
- Can provide immediate feedback
- Protects the master branch (if run on PR)

What does it do?

- Triggered at each commit / push
- Sets up a virtual machine with your configuration.
- Pulls the current branch.
- Runs command. Usually: install, then test.
- Reports success / Failure to github.

Setting up TravisCI

- Create account linked to your github account: http://travis-ci.org



- Check out docs at https://docs.travis-ci.com

```
language: python
python:
  - "2.7"
  - "3.4"
  - "3.5"
  - "3.6"
  - "3.7"
  - "nightly"

install:
  - pip install -r requirements.txt

script:
  - pytest
```

Using Travis

- Triggered any time you push a change
- Integrated with Pull requests
- Try a pull request on your own repository!

Questions ?