

Continuous Integration (CI)

02457 Machine Learning Operations

Nicki Skafte Detlefsen,

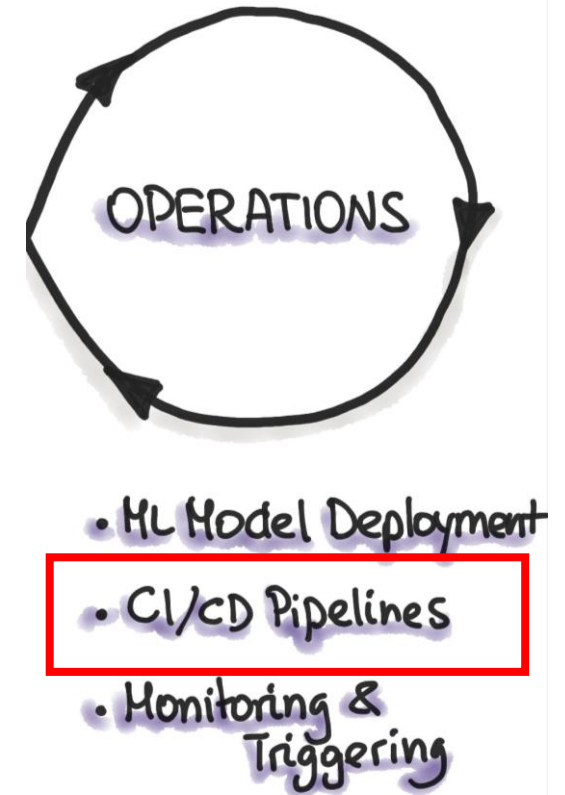
Postdoc

DTU Compute

What is continues integration?



- Software practise
 - Frequently commit code to shared repository
 - By commit sooner than later, errors are captured early
 - Make merging easier
 - Automate build + test
- App independent



What is continues deployment?



- How to get your code to the user
 - 3 phases: Testing, staging and deploying
- App dependent

Note: Not covered in this course



- ML Model Deployment
- CI/CD Pipelines
- Monitoring & Triggering

What should you know about CI?



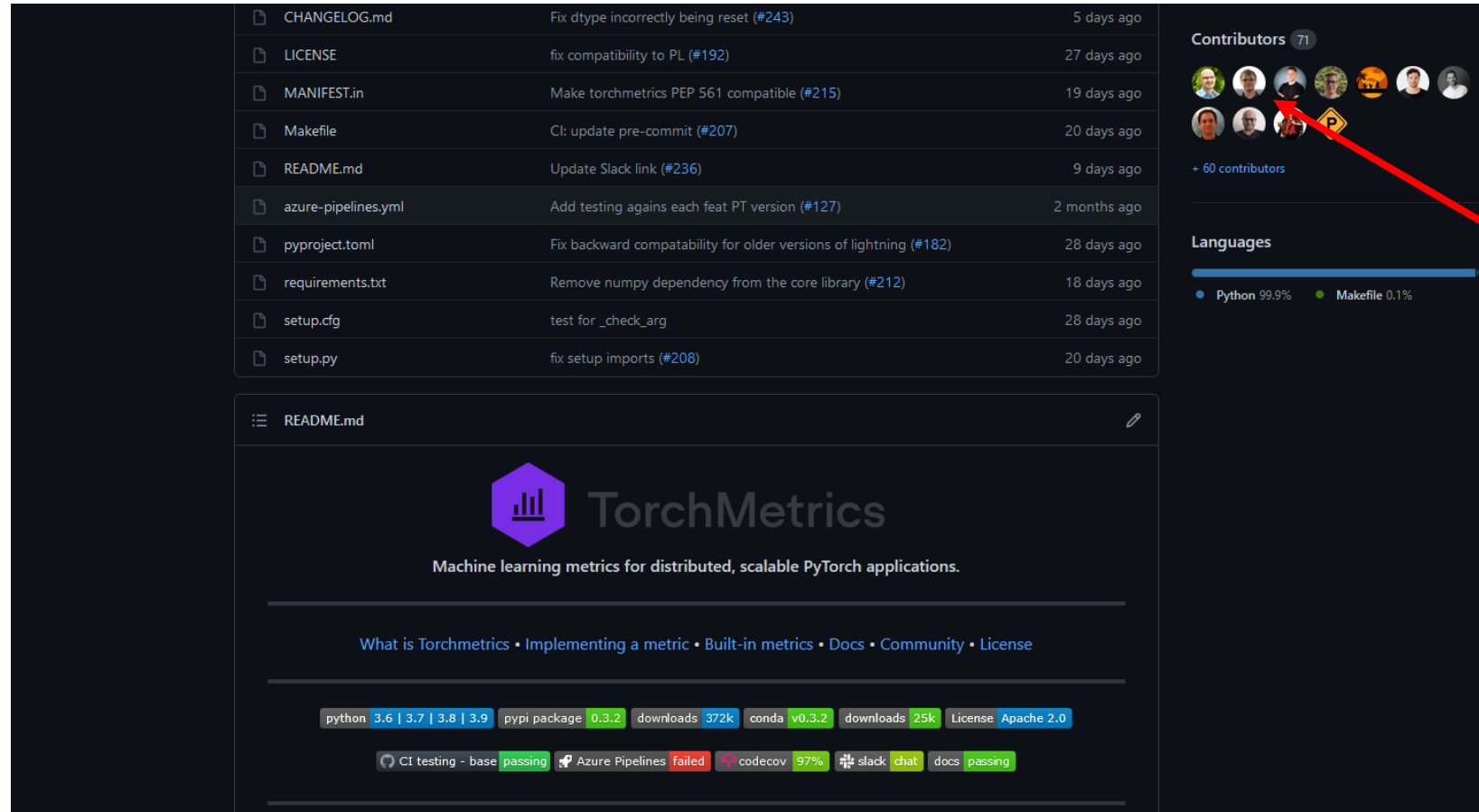
- CI is one of those topics that are best though as learning-by-doing
- If you understand how to use git, the rest is basically googling stuff

Let's look at a practical example

A small case study



All the metric in the world – now in pytorch



The screenshot shows the GitHub repository for TorchMetrics. The top section lists recent pull requests with their titles and dates. Below this is the README.md file, which features the TorchMetrics logo and a description: "Machine learning metrics for distributed, scalable PyTorch applications." The README also includes links to documentation and community resources, as well as a section for project statistics (downloads, license, CI testing status, etc.). On the right side of the repository page, there is a "Contributors" section showing 71 contributors, with a red arrow pointing to one of the contributor avatars. Below the contributors is a "Languages" section showing the distribution of code languages (Python 99.9%, Makefile 0.1%).

File	Description	Time
CHANGELOG.md	Fix dtype incorrectly being reset (#243)	5 days ago
LICENSE	fix compatibility to PL (#192)	27 days ago
MANIFEST.in	Make torchmetrics PEP 561 compatible (#215)	19 days ago
Makefile	CI: update pre-commit (#207)	20 days ago
README.md	Update Slack link (#236)	9 days ago
azure-pipelines.yml	Add testing against each feat PT version (#127)	2 months ago
pyproject.toml	Fix backward compatibility for older versions of lightning (#182)	28 days ago
requirements.txt	Remove numpy dependency from the core library (#212)	18 days ago
setup.cfg	test for _check_arg	28 days ago
setup.py	fix setup imports (#208)	20 days ago

Contributors 71

+ 60 contributors

Languages

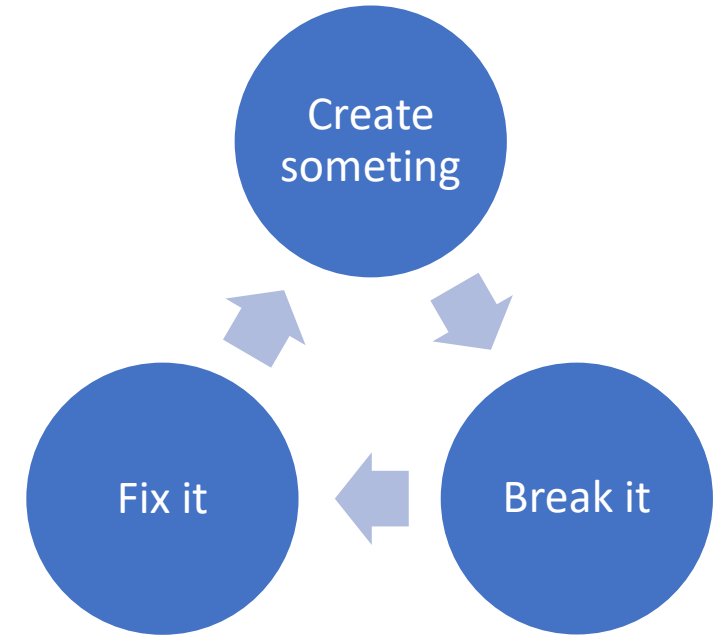
- Python 99.9%
- Makefile 0.1%

Have I seen that guy before?

CI step 1: Committing code



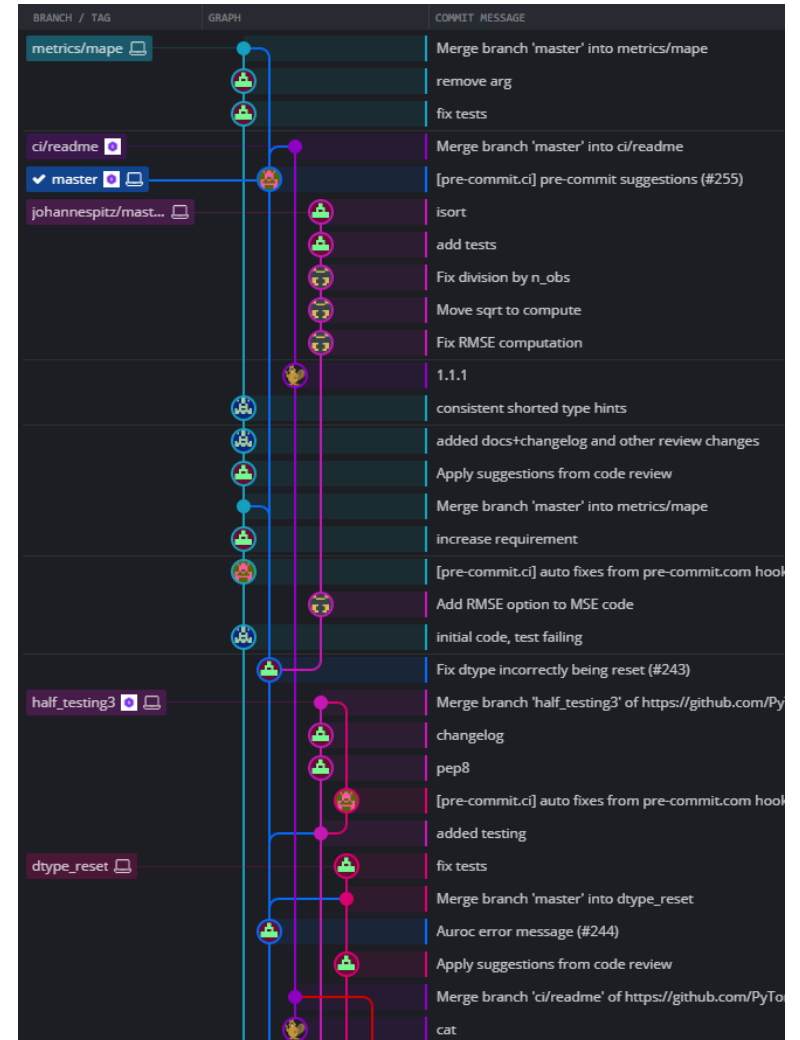
- Commit frequently
 - Catch errors sooner than later
 - Merging can be done automatically



CI step 1: Committing code



- Use branches
 - Enables parallel workflow
 - Experimental features/changes are kept away from "stable" master



CI step 1: Committing code



- Use PRs, other can review your code

A screenshot of a GitHub Pull Request (PR) interface for the repository "PyTorchLightning / metrics". The PR is titled "testing readme examples #140" and is created by "Borda". The interface shows the PR details, a list of files changed (4 files), and a diff view. Annotations are placed on the interface to guide the user through the review process:

- 1. Find PR**: A red box highlights the "Pull requests" tab in the top navigation bar.
- 2. Check changed files**: A red box highlights the "Files changed" tab in the PR header.
- 3. Make one or more comments**: A red box highlights a comment by "SkafteNicki" asking for a more descriptive name for a variable, with a suggested change shown below.
- 4. Send review**: A red box highlights the "Finish your review" button in the bottom right corner.

CI step 2: Automating stuff



- What can be automated: EVERYTHING
 - Functional tests
 - Documentation creation
 - Linters (which check style formatting)
 - Security checks
 - Code coverage
 - Custom checks...

A small case study



Ranked by importance (biased)

1. Source code

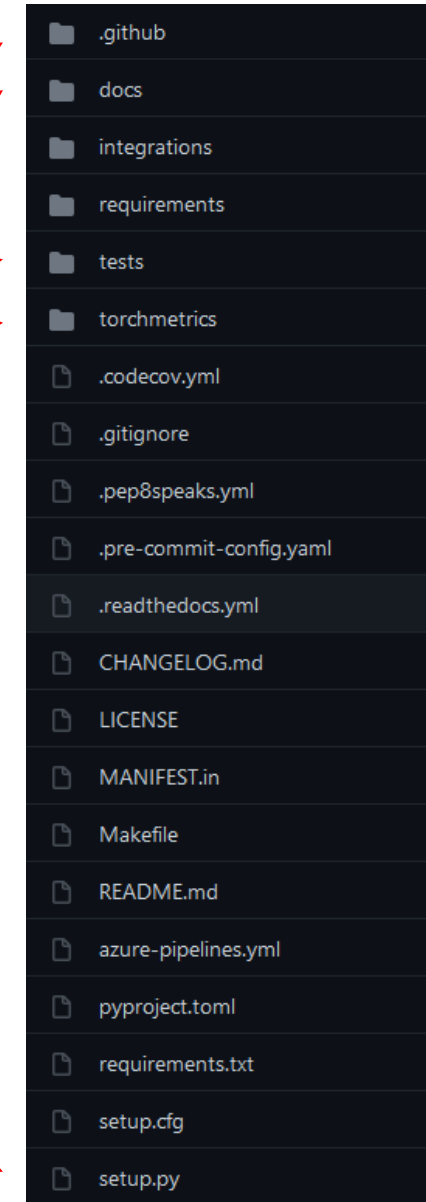
2. Tests

3. Setup

4. CI workflows


5. Documentation

6. ★



Source code



master	metrics / torchmetrics /	Go to file	Add file	...
	SkaftNicki Fix dtype incorrectly being reset (#243) ...	✓ cda5dbd	5 days ago	History
..				
classification	Allow logit input in classification metrics (#200)		6 days ago	
functional	Fix dtype incorrectly being reset (#243)		5 days ago	
regression	Add differentiability testing to more metrics [3/n] (#225)		13 days ago	
retrieval	Information Retrieval (5/5) (#160)		last month	
utilities	Allow logit input in classification metrics (#200)		6 days ago	
wrappers	Feature pre commit yaml (#145)		2 months ago	
__about__.py	Show must go on (#198)		25 days ago	
__init__.py	Add Specificity metric (#210)		15 days ago	
average.py	allow MetricCollection with args (#176)		29 days ago	
collections.py	Added add_metrics method to MetricCollection (#221)		14 days ago	
metric.py	Fix dtype incorrectly being reset (#243)		5 days ago	
py.typed	Make torchmetrics PEP 561 compatible (#215)		19 days ago	
setup_tools.py	Remove numpy dependency from the core library (#212)		18 days ago	

Test code



In total we have 6365 tests

master	metrics / tests /	Go to file	Add file	...
	SkaftNicki Fix dtype incorrectly being reset (#243) ...	✓ cda5dbd	5 days ago	History
..				
bases	Fix dtype incorrectly being reset (#243)		5 days ago	
classification	Auroc error message (#244)		5 days ago	
functional	Refactor Information Retrieval tests (#156)		last month	
helpers	Ports are reused breaking parallel tests (#226)		14 days ago	
regression	Add differentiability testing to more metrics [3/n] (#225)		13 days ago	
retrieval	Remove numpy dependency from the core library (#212)		18 days ago	
wrappers	Adds indexing operation to Metric class (#142)		2 months ago	
__init__.py	minor refactor tests (#21)		3 months ago	
test_utilities.py	more tests		2 months ago	

Test example 1



Can be simple

```
def test_warning_on_nan(tmpdir):  
    preds = torch.randint(3, size=(20, ))  
    target = torch.randint(3, size=(20, ))  
  
    with pytest.warns(  
        UserWarning,  
        match='.* nan values found in confusion matrix have been replaced with zeros.',  
    ):  
        confusion_matrix(preds, target, num_classes=5, normalize='true')
```

Test example 2



Can be very complex

```
@pytest.mark.parametrize("normalize", ['true', 'pred', 'all', None])
@pytest.mark.parametrize(
    "preds, target, sk_metric, num_classes, multilabel",
    [
        (_input_binary_prob.preds, _input_binary_prob.target, _sk_cm_binary_prob, 2, False),
        (_input_binary_logits.preds, _input_binary_logits.target, _sk_cm_binary_prob, 2, False),
        (_input_binary.preds, _input_binary.target, _sk_cm_binary, 2, False),
        (_input_mlb_prob.preds, _input_mlb_prob.target, _sk_cm_multilabel_prob, NUM_CLASSES, True),
        (_input_mlb_logits.preds, _input_mlb_logits.target, _sk_cm_multilabel_prob, NUM_CLASSES, True),
        (_input_mlb.preds, _input_mlb.target, _sk_cm_multilabel, NUM_CLASSES, True),
        (_input_mcls_prob.preds, _input_mcls_prob.target, _sk_cm_multiclass_prob, NUM_CLASSES, False),
        (_input_mcls_logits.preds, _input_mcls_logits.target, _sk_cm_multiclass_prob, NUM_CLASSES, False),
        (_input_mcls.preds, _input_mcls.target, _sk_cm_multiclass, NUM_CLASSES, False),
        (_input_mdmc_prob.preds, _input_mdmc_prob.target, _sk_cm_multidim_multiclass_prob, NUM_CLASSES, False),
        (_input_mdmc.preds, _input_mdmc.target, _sk_cm_multidim_multiclass, NUM_CLASSES, False)]
)
class TestConfusionMatrix(MetricTester):

    @pytest.mark.parametrize("ddp", [True, False])
    @pytest.mark.parametrize("dist_sync_on_step", [True, False])
    def test_confusion_matrix(
        self, normalize, preds, target, sk_metric, num_classes, multilabel, ddp, dist_sync_on_step
    ):
        self.run_class_metric_test(
            ddp=ddp,
            preds=preds,
            target=target,
            metric_class=ConfusionMatrix,
            sk_metric=partial(sk_metric, normalize=normalize),
            dist_sync_on_step=dist_sync_on_step,
            metric_args={
                "num_classes": num_classes,
                "threshold": THRESHOLD,
                "normalize": normalize,
                "multilabel": multilabel
            }
        )
```

Setup files

- Contains all information regarding the project
- Allow people to do:
python setup.py install
or if uploaded to pip
pip install my_package

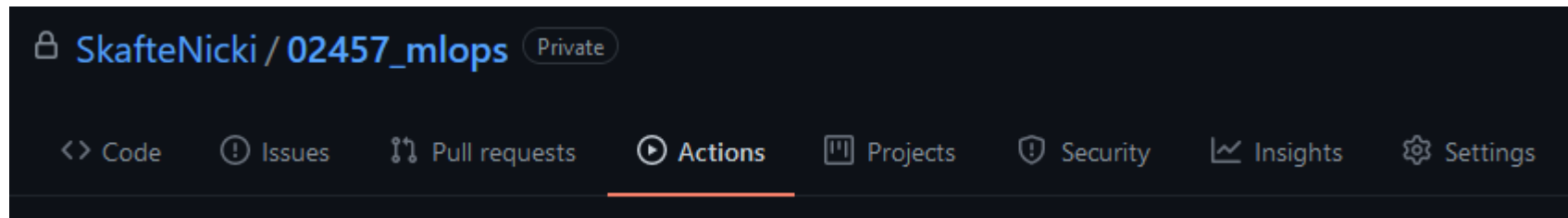
```
setup(
    name='torchmetrics',
    version=about.__version__,
    description=about.__docs__,
    author=about.__author__,
    author_email=about.__author_email__,
    url=about.__homepage__,
    download_url=os.path.join(about.__homepage__, 'archive', 'master.zip'),
    license=about.__license__,
    packages=find_packages(exclude=['tests', 'docs']),
    long_description=long_description,
    long_description_content_type='text/markdown',
    include_package_data=True,
    zip_safe=False,
    keywords=['deep learning', 'machine learning', 'pytorch', 'metrics', 'AI'],
    python_requires='>=3.6',
    setup_requires=[],
    install_requires=setup_tools._load_requirements(_PATH_ROOT),
    project_urls={
        "Bug Tracker": os.path.join(about.__homepage__, 'issues'),
        "Documentation": "https://torchmetrics.rtfd.io/en/latest/",
        "Source Code": about.__homepage__,
    },
    classifiers=[
        'Environment :: Console',
        'Natural Language :: English',
        # How mature is this project? Common values are
        # 3 - Alpha, 4 - Beta, 5 - Production/Stable
        'Development Status :: 3 - Alpha',
        # Indicate who your project is intended for
        'Intended Audience :: Developers',
        'Topic :: Scientific/Engineering :: Artificial Intelligence',
        'Topic :: Scientific/Engineering :: Image Recognition',
        'Topic :: Scientific/Engineering :: Information Analysis',
        # Pick your license as you wish
        # 'License :: OSI Approved :: BSD License',
        'Operating System :: OS Independent',
        # Specify the Python versions you support here. In particular, ensure
        # that you indicate whether you support Python 2, Python 3 or both.
        'Programming Language :: Python :: 3',
        'Programming Language :: Python :: 3.6',
        'Programming Language :: Python :: 3.7',
        'Programming Language :: Python :: 3.8',
        'Programming Language :: Python :: 3.9',
    ],
)
```



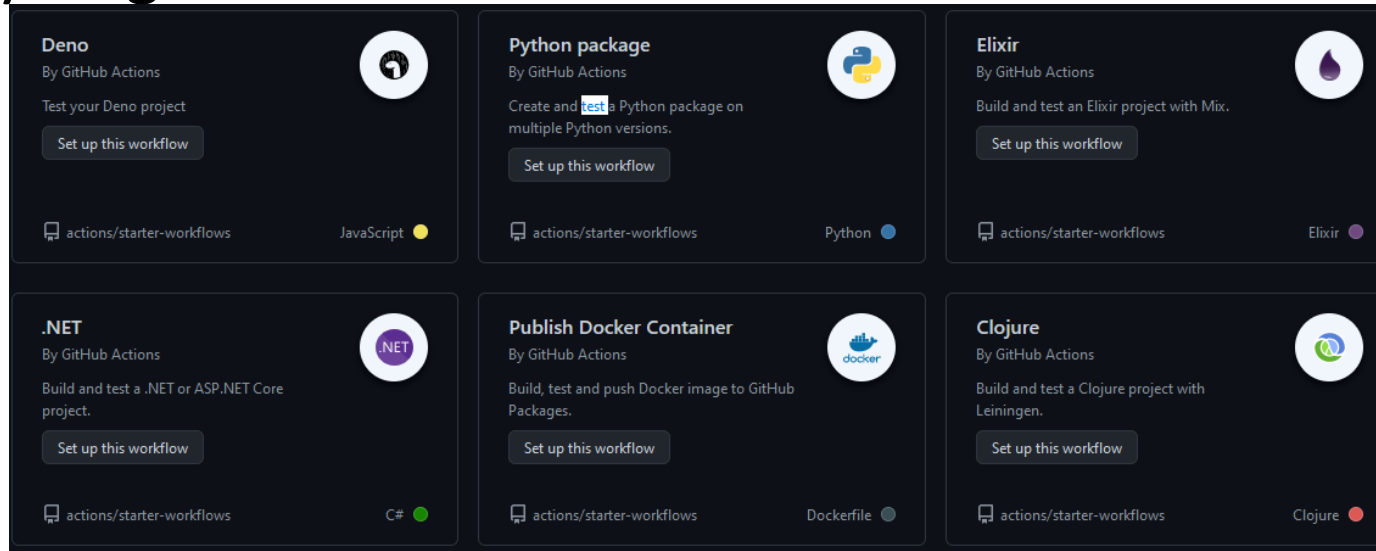
Github actions



- Build-in CI for github
- Free 2,000 automation minutes/month (public repository)



- Many ready to go workflows



The anatomy of a workflow file



- When tests should be triggered
- Define Operating system + python version
- Setup python
- Install dependencies
- Check linting (stop if errors)
- Run tests

```
02457_mlops / .github / workflows / python-package.yml in main
<> Edit new file Preview
1 name: Python package
2 on:
3   push:
4     branches: [ main ]
5   pull_request:
6     branches: [ main ]
7 jobs:
8   build:
9     runs-on: ubuntu-latest
10    strategy:
11      fail-fast: false
12      matrix:
13        python-version: [3.7, 3.8, 3.9]
14    steps:
15      - uses: actions/checkout@v2
16      - name: Set up Python ${ matrix.python-version }
17        uses: actions/setup-python@v2
18        with:
19          python-version: ${ matrix.python-version }
20      - name: Install dependencies
21        run: |
22          python -m pip install --upgrade pip
23          python -m pip install flake8 pytest
24          if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
25      - name: Lint with flake8
26        run: |
27          # stop the build if there are Python syntax errors or undefined names
28          flake8 . --count --select=E9,F63,F7,F82 --show-source --statistics
29          # exit-zero treats all errors as warnings. The GitHub editor is 127 chars wide
30          flake8 . --count --exit-zero --max-complexity=10 --max-line-length=127 --statistics
31      - name: Test with pytest
32        run: |
33          pytest
```

A small case study



master metrics / .github / workflows /

Borda CI: fix (deduplicate) release event

..

ci_install-pkg.yml	update CI install (#39)
ci_test-base.yml	CI: fix coverage config (#84)
ci_test-conda.yml	CI fix env mkl for PT 1.9 (#211)
ci_test-full.yml	fix compatibility to PL (#192)
code-format.yml	CI: publish test results (#80)
docs-check.yml	CI: update pre-commit (#207)
docs-deploy.yml	CI: update pre-commit (#207)
greetings.yml	Initial commit
release-pypi.yml	

```
- name: Install dependencies
  run: |
    python -m pip install --upgrade --user pip
    pip install --requirement ./requirements.txt --find-links https://download.pytorch.org/whl/cpu/torch_stable.html
    pip install "pytest>6.0" "pytest-cov>2.10" --upgrade-strategy only-if-needed
    python --version
    pip --version
    pip list
  shell: bash

- name: Test Package [only]
  run: |
    # NOTE: run coverage on tests does not propagate faler status for Win, https://github.com/nedbat/coveragepy/issues/1003
    python -m pytest torchmetrics -v --cov=torchmetrics --junitxml=junit/test-results-${{ runner.os }}-${{ matrix.python-version }}.xml
```

All of our workflows – 36 in total



✓ CI testing - complete
on: pull_request

✓ pytest (ubuntu-20.04, 3.6, minimal)

✓ pytest (ubuntu-20.04, 3.8, minimal)

✓ pytest (ubuntu-20.04, 3.8, latest)

✓ pytest (ubuntu-20.04, 3.9, latest)

✓ pytest (macOS-10.15, 3.6, minimal)

✓ pytest (macOS-10.15, 3.8, minimal)

✓ pytest (macOS-10.15, 3.8, latest)

✓ pytest (macOS-10.15, 3.9, latest)

✓ pytest (windows-2019, 3.6, mini...)

✓ pytest (windows-2019, 3.8, mini...)

✓ pytest (windows-2019, 3.8, latest)

✓ pytest (windows-2019, 3.9, latest)

✓ WIP

✓ WIP

✓ Check Code formatting
on: pull_request

✓ flake8

✓ imports-check-isort

✓ typing-check-mypy

✓ Docs check
on: pull_request

✓ test-docs

✓ make-docs

✓ CI testing - base
on: pull_request

✓ doctest (ubuntu-20.04, 3.7)

✓ doctest (windows-2019, 3.7)

✓ doctest (macOS-10.15, 3.7)

✓ Azure Pipelines

✗ PyTorchLightning.metrics Re-run

✓ Mergify

✓ Summary

✓ Codecov

✓ codecov/patch

✓ codecov/project

✓ PyTorch & Conda
on: pull_request

✓ conda (3.7, 1.3)

✓ conda (3.7, 1.4)

✓ conda (3.7, 1.5)

✓ conda (3.7, 1.6)

✓ conda (3.7, 1.7)

⚠ conda (3.7, 1.8)

✓ conda (3.7, 1.9)

Not github actions

✓ Install pkg
on: pull_request

✓ pkg-check

✓ pkg-install (ubuntu-20.04, 3.7)

✓ pkg-install (macOS-10.15, 3.7)

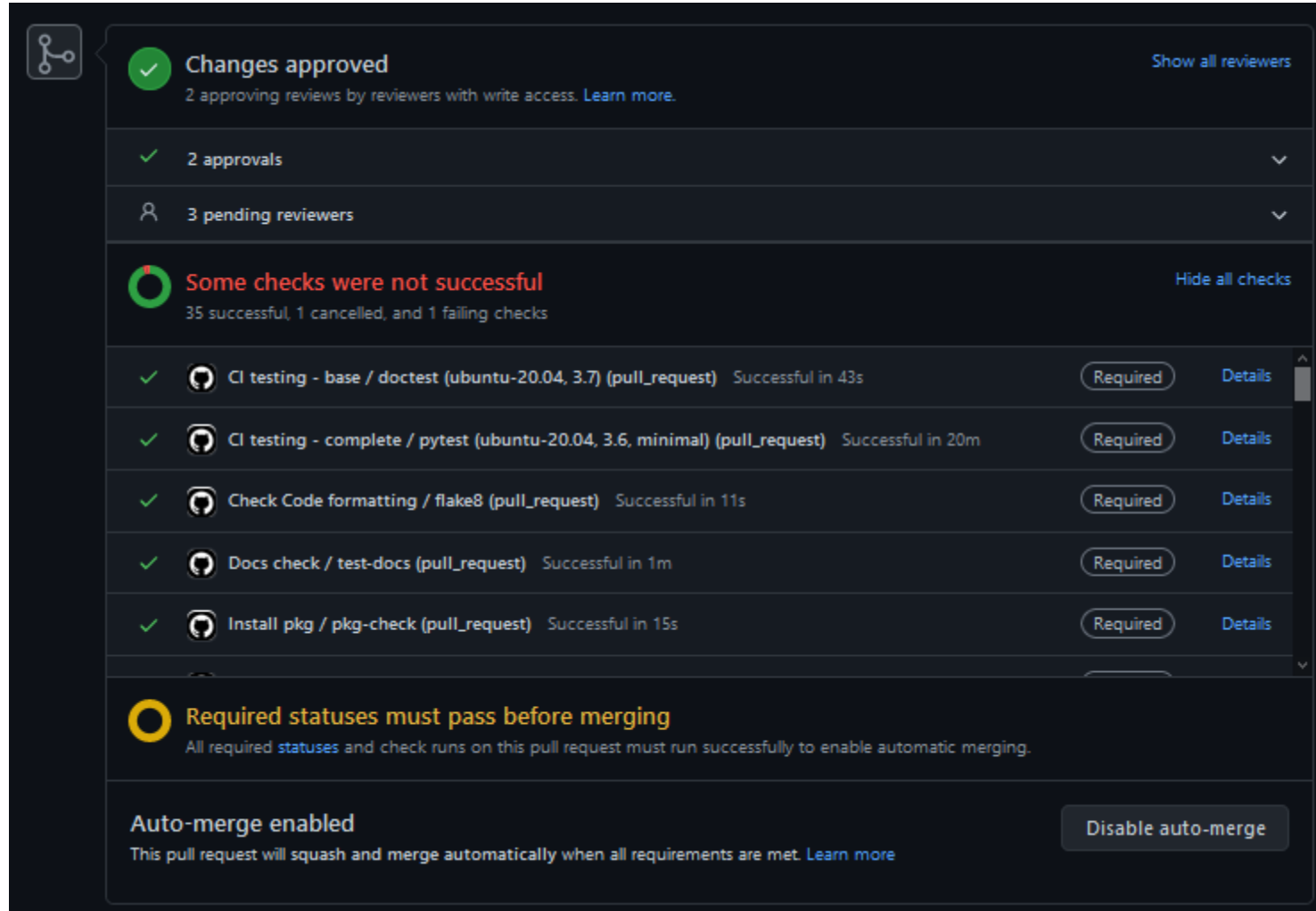
✓ pkg-install (windows-2019, 3.7)

Test combination of

- Hardware setup
- Operating system
- Python version
- Dependencies

Runs tests, docs, coverage, lintint, package install ect

Each PR triggers all tests



The screenshot displays a GitHub pull request interface. At the top, a green checkmark icon indicates 'Changes approved', with a note that 2 approving reviews by reviewers with write access are required. Below this, a summary shows '2 approvals' and '3 pending reviews'. A red circle icon with a slash indicates 'Some checks were not successful', with a note that 35 checks were successful, 1 was cancelled, and 1 was failing. A list of CI checks follows, all marked as 'Successful' and 'Required'. The checks include: 'CI testing - base / doctest (ubuntu-20.04, 3.7) (pull_request)' (43s), 'CI testing - complete / pytest (ubuntu-20.04, 3.6, minimal) (pull_request)' (20m), 'Check Code formatting / flake8 (pull_request)' (11s), 'Docs check / test-docs (pull_request)' (1m), and 'Install pkg / pkg-check (pull_request)' (15s). At the bottom, a yellow circle icon indicates 'Required statuses must pass before merging', with a note that all required statuses and check runs must run successfully to enable automatic merging. The 'Auto-merge enabled' section states that the pull request will squash and merge automatically when all requirements are met, with a 'Disable auto-merge' button.

Changes approved [Show all reviewers](#)
2 approving reviews by reviewers with write access. [Learn more.](#)

✓ 2 approvals

⋈ 3 pending reviews

⦿ **Some checks were not successful** [Hide all checks](#)
35 successful, 1 cancelled, and 1 failing checks

Check Status	Check Name	Duration	Required	Details
✓	CI testing - base / doctest (ubuntu-20.04, 3.7) (pull_request)	Successful in 43s	Required	Details
✓	CI testing - complete / pytest (ubuntu-20.04, 3.6, minimal) (pull_request)	Successful in 20m	Required	Details
✓	Check Code formatting / flake8 (pull_request)	Successful in 11s	Required	Details
✓	Docs check / test-docs (pull_request)	Successful in 1m	Required	Details
✓	Install pkg / pkg-check (pull_request)	Successful in 15s	Required	Details

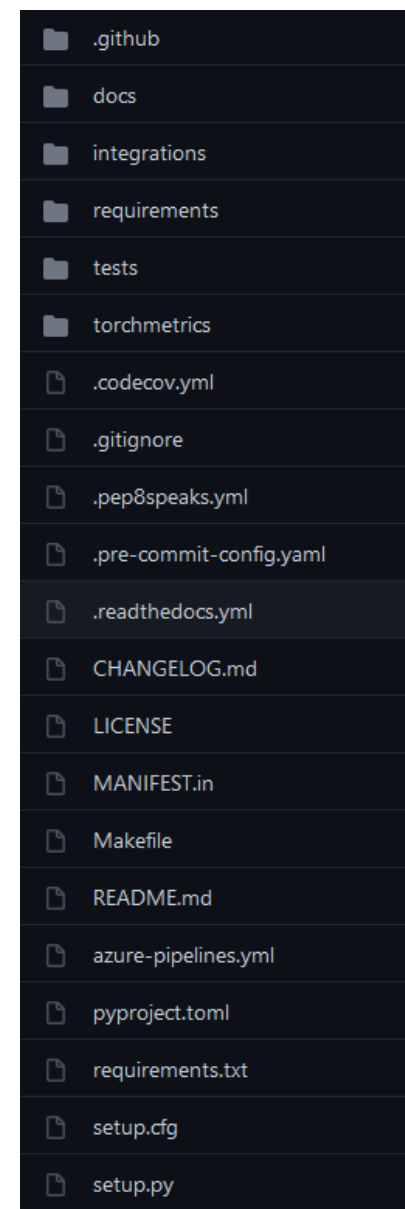
⦿ **Required statuses must pass before merging**
All required [statuses](#) and check runs on this pull request must run successfully to enable automatic merging.

Auto-merge enabled [Disable auto-merge](#)
This pull request will squash and merge automatically when all requirements are met. [Learn more](#)

Other files to consider



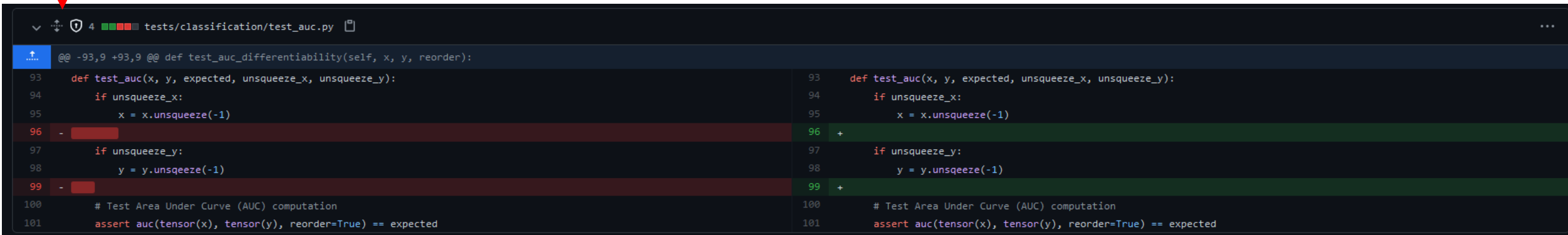
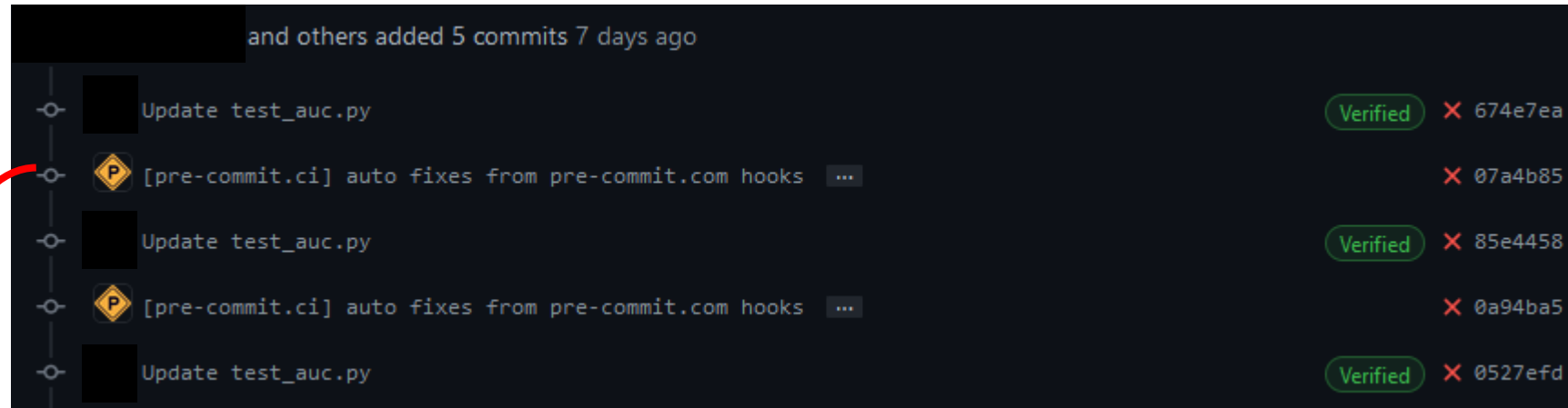
- requirements.txt
- LICENSE
- README.md
- .gitignore



Advanced: Use bots



Bots can take care of tedious task for you (like linting)



The future is here



Gabriele Petronella
@gabro27

So this just happened:

- a bot found a vulnerability in a dependency
- a bot sent a PR to fix it
- the CI verified the PR
- a bot merged it
- a bot celebrated the merge with a GIF

