

# Day6 - Continuous Integration

02476 Machine Learning Operations

Nicki Skafte Detlefsen, Associate Professor, DTU Compute

January 2026

50 people and 5 groups are marked in my little black book...

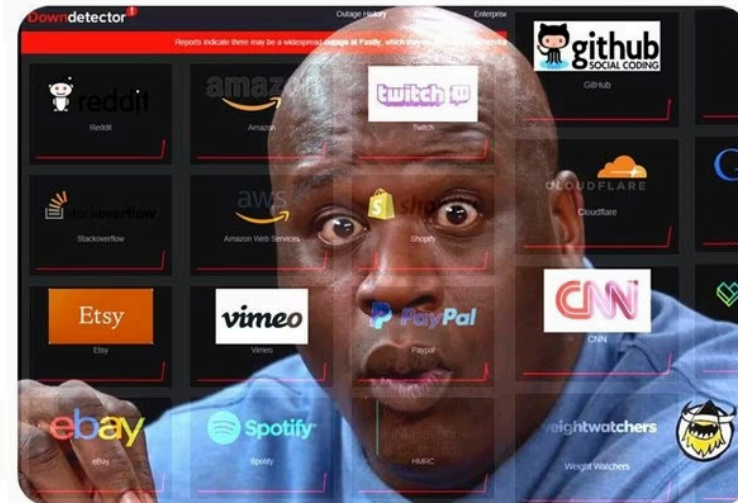


# Why you should care about today

3 years ago, the day before this lecture, the internet went down for a couple of hours because someone f..ked up their continues integration at **Fastly**.

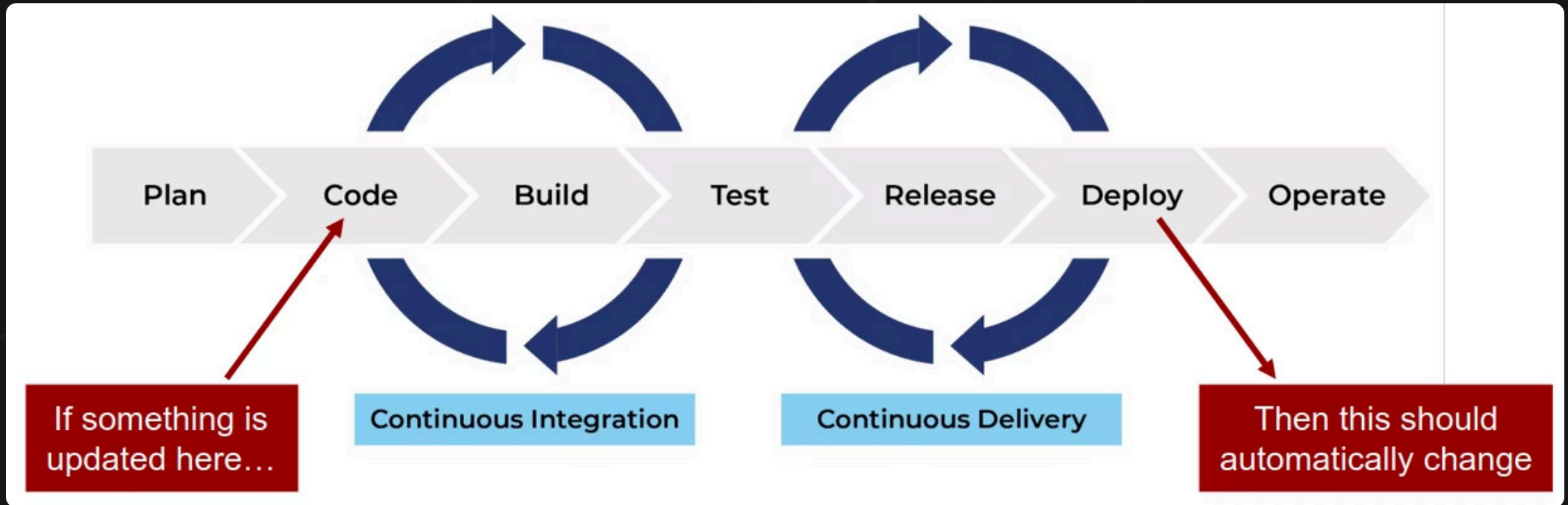
Dev at Fastly : I'll just push this small change to production

Dev at Fastly 2 seconds later:



# Continues X

🔥 Term refers to a set of software practices for automating tedious tasks and make sure changed in a pipeline are continuously propagated through the pipeline



# The different types

CI	CD	CML
Continues Integration	Continues Deployment	Continues Machine Learning
💡 How to automatically secure that code does not break during development?	💡 How to get your code/application to the user automatically? + monitor life cycle	💡 How to automatically retrain machine learning models when data and code changes?
💻 App independent concept	💻 App dependent concept	💻 Specific to ML applications



# MLOps levels

The **Maturity model** overall describes the DevOps practices to run a successful MLOps environment.

Intended to identify gaps in an existing organization's attempt to implement such an environment.

💡 Estimate the scope of the work for new engagements.

💡 Establish realistic success criteria.

💡 Identify deliverables you'll hand over at the conclusion of the engagement.

Level	Description	Highlights	Technology
0	No MLOps	<ul style="list-style-type: none"><li>• Difficult to manage full machine learning model lifecycle</li><li>• The teams are disparate and releases are painful</li><li>• Most systems exist as "black boxes," little feedback during/post deployment</li></ul>	<ul style="list-style-type: none"><li>• Manual builds and deployments</li><li>• Manual testing of model and application</li><li>• No centralized tracking of model performance</li><li>• Training of model is manual</li></ul>
1	DevOps but no MLOps	<ul style="list-style-type: none"><li>• Releases are less painful than No MLOps, but rely on Data Team for every new model</li><li>• Still limited feedback on how well a model performs in production</li><li>• Difficult to trace/reproduce results</li></ul>	<ul style="list-style-type: none"><li>• Automated builds</li><li>• Automated tests for application code</li></ul>
2	Automated Training	<ul style="list-style-type: none"><li>• Training environment is fully managed and traceable</li><li>• Easy to reproduce model</li><li>• Releases are manual, but low friction</li></ul>	<ul style="list-style-type: none"><li>• Automated model training</li><li>• Centralized tracking of model training performance</li><li>• Model management</li></ul>
3	Automated Model Deployment	<ul style="list-style-type: none"><li>• Releases are low friction and automatic</li><li>• Full traceability from deployment back to original data</li><li>• Entire environment managed: train &gt; test &gt; production</li></ul>	<ul style="list-style-type: none"><li>• Integrated A/B testing of model performance for deployment</li><li>• Automated tests for all code</li><li>• Centralized tracking of model training performance</li></ul>
4	Full MLOps Automated Operations	<ul style="list-style-type: none"><li>• Full system automated and easily monitored</li><li>• Production systems are providing information on how to improve and, in some cases, automatically improve with new models</li><li>• Approaching a zero-downtime system</li></ul>	<ul style="list-style-type: none"><li>• Automated model training and testing</li><li>• Verbose, centralized metrics from deployed model</li></ul>

# This lecture: continues integration

Core task:

🔥 How to automatically secure that code does not break during development? 🔥

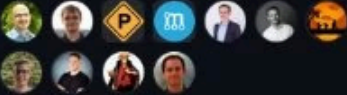
3 steps to do this:

- 💡 Use version control: Frequently committing code to a shared repository
- 💡 Write (unit)test for your code: Should capture unwanted bugs in your code
- 💡 Automate build + testing: Automatically run test so code cannot be merged without working

# A small case study for continuous integration

MANIFEST.in	update CI	6 months ago
Makefile	rename tests/ (#1091)	5 months ago
README.md	Code cleaning after classifcation refactor 2/n (#1252)	7 days ago
pyproject.toml	CI: re-use checks (#1261)	2 months ago
requirements.txt	Set minimum pytorch version to 1.8 + cleanup (#1263)	2 months ago
setup.cfg	CI: re-use checks (#1261)	2 months ago
setup.py	CI: Enable testing with Python 3.10 (#1132)	5 months ago

Contributors 169




+ 158 contributors

Languages

Python 99.9%

Other 0.1%

README.md



## TorchMetrics

Machine learning metrics for distributed, scalable PyTorch applications.

[What is Torchmetrics](#) • [Implementing a metric](#) • [Built-in metrics](#) • [Docs](#) • [Community](#) • [License](#)

python

3.7 | 3.8 | 3.9 | 3.10

pypi package

0.10.3

downloads

28M

conda

v0.10.3

downloads

585k

License

Apache 2.0

CI testing - complete

Azure Pipelines

succeeded

codecov

39%

slack

chat

docs

passing

DOI

10.5281/zenodo.5844769

JOSS

10.21105/joss.04101

pre-commit.ci

passed



# CI step 1: version control

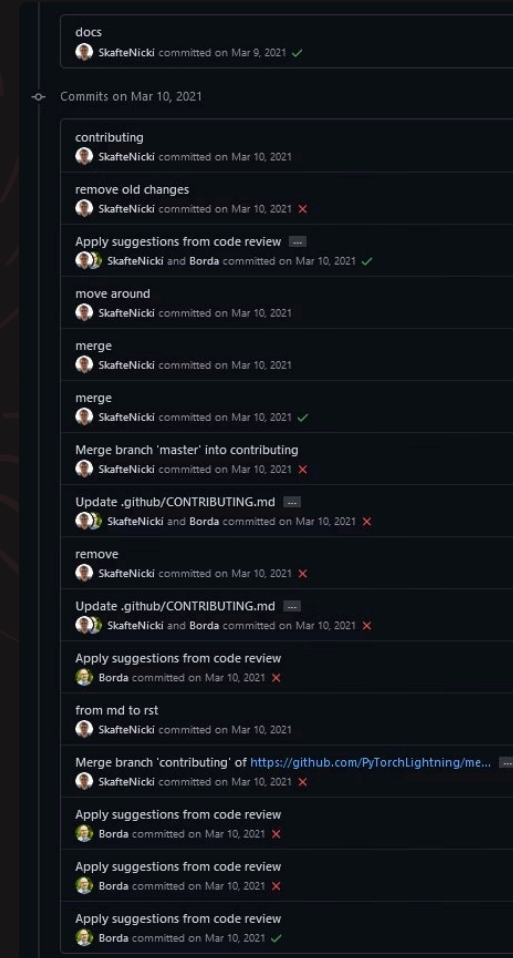
User version control:

- 💡 Code changes are tracked
- 💡 Branches for parallel work

Commit frequently:

- 💡 Catch errors sooner than later
- 💡 Revert back easily to when things were working
- 💡 Merging can be done automatically

Create it → Break it → Fix it →...



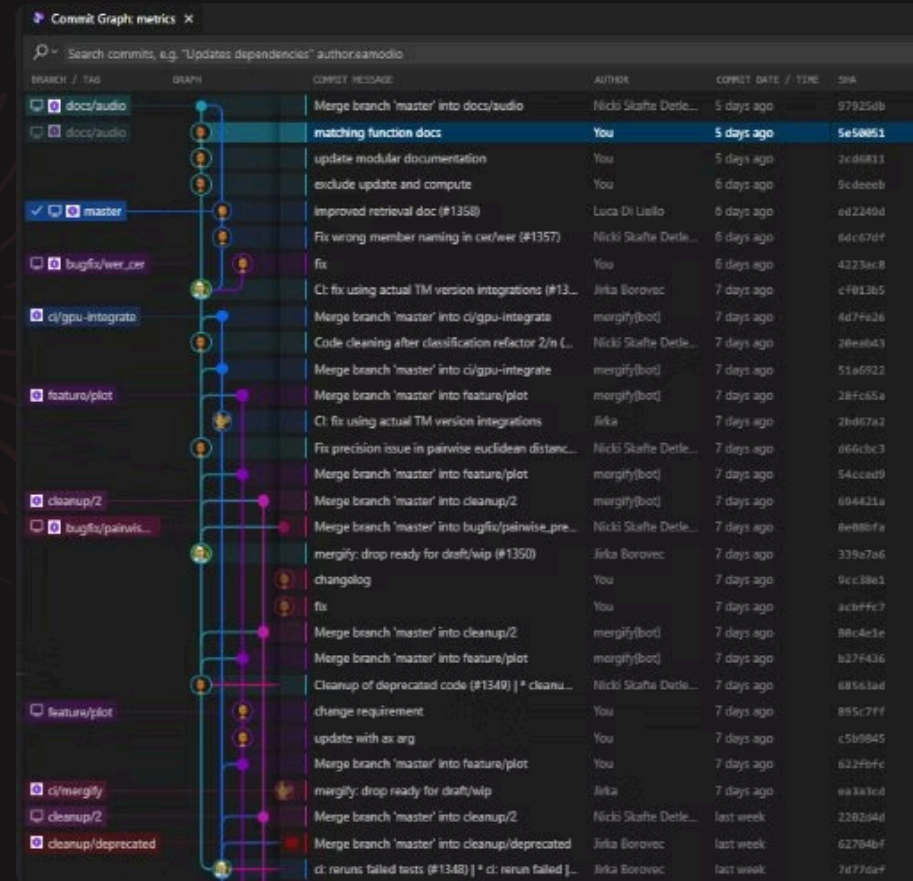
# CI step 1: Use branches

Parallel workflow

Experimental features changes are kept away from master/main

Recommend extensions for VS code:

- GitLens
- Git Graph
- GitHub Pull Requests



# CI step 1: Use pull requests

⚠ No commit can be pushed to master without being in a pull request

The screenshot displays a GitHub pull request for the repository 'Lightning-AI/metrics'. The title of the pull request is 'Added MinkowskiDistance support #1362'. The interface is annotated with four numbered steps:

- 1. Find PR**: A red box highlights the 'Pull requests' tab in the repository's navigation bar.
- 2. Check changes**: A red box highlights the 'Files changed' tab, which shows a list of modified files in the 'functional' directory.
- 3. Make one or more comments**: A red box highlights the comment section at the bottom of the page, showing a comment from 'Skeffmcd' and a reply input field.
- 4. Send review**: A red box highlights the 'Review changes' button in the top right corner of the pull request view.

The pull request details include a description of the changes, a list of files changed, and a diff view showing the code modifications. The comment section shows a user named 'Skeffmcd' commenting on the pull request, and a reply input field is visible.

# CI step 1: pre-commit

✓ Check that everything is up to standard before commits are created

```
! .pre-commit-config.yaml x
! .pre-commit-config.yaml
1  default_language_version:
2    python: python3
3
4  repos:
5    - repo: https://github.com/pre-commit/pre-commit-hooks
6      rev: v4.4.0
7      hooks:
8        - id: end-of-file-fixer
9        - id: trailing-whitespace
10       # - id: check-json
11       # - id: check-yaml
12       - id: check-toml
13       - id: check-docstring-first
14       - id: check-executables-have-shebangs
15       - id: check-case-conflict
16       - id: detect-private-key
17
18     - repo: https://github.com/astral-sh/ruff-pre-commit
19       rev: v0.1.3
20       hooks:
21         - id: ruff
22         args: [--fix, --exit-non-zero-on-fix]
23
24     - repo: https://github.com/astral-sh/ruff-pre-commit
25       rev: v0.1.3
26       hooks:
27         - id: ruff-format
28
29     - repo: https://github.com/codespell-project/codespell
30       rev: v2.2.5
31       hooks:
32         - id: codespell
33         additional_dependencies: [tomli]
34
```

```
dtu_mlops on main [!?!] via v3.11.5 @mlops
> git commit -m "implementation of client"
fix end of files.....Failed
- hook id: end-of-file-fixer
- exit code: 1
- files were modified by this hook

Fixing s8_monitoring/exercise_files/client.py

trim trailing whitespace.....Passed
check toml.....(no files to check)Skipped
check docstring is first.....Passed
check that executables have shebangs.....Passed
check for case conflicts.....Passed
detect private key.....Passed
ruff.....Failed
- hook id: ruff
- exit code: 1
- files were modified by this hook

s8_monitoring\exercise_files\client.py:17:12: S113 Probable use of requests call without timeout
Found 2 errors (1 fixed, 1 remaining).

ruff-format.....Passed
codespell.....Passed
markdownlint-docker.....(no files to check)Skipped
```

# CI step 2: write tests

Tests are the cornerstones of continuous integration

💡 *unit tests* are arguable the most important.

💡 A single unittest, tests a small part of your code

💡 By testing code in small pieces, bugs are easier to find

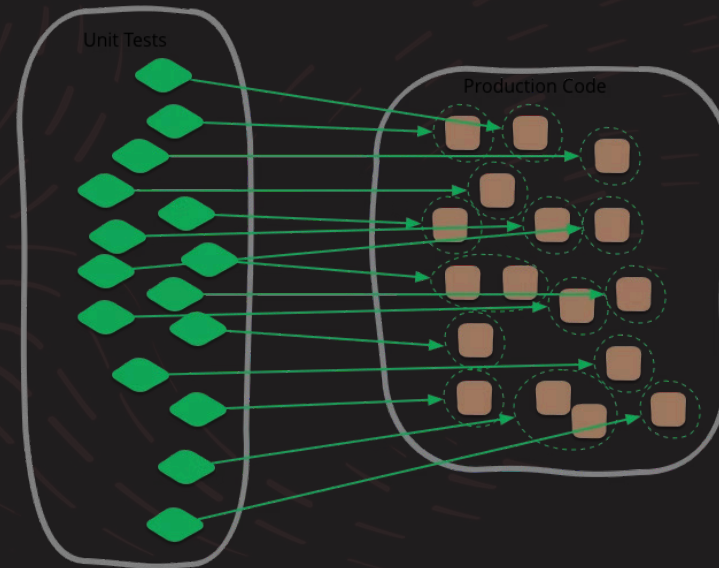
Other test types worth considering:

🔥 Integration tests

🔥 Regression tests

🔥 Performance tests

🔥 Security tests



# CI step 2: write tests

💡 By Python convention your source code should either be

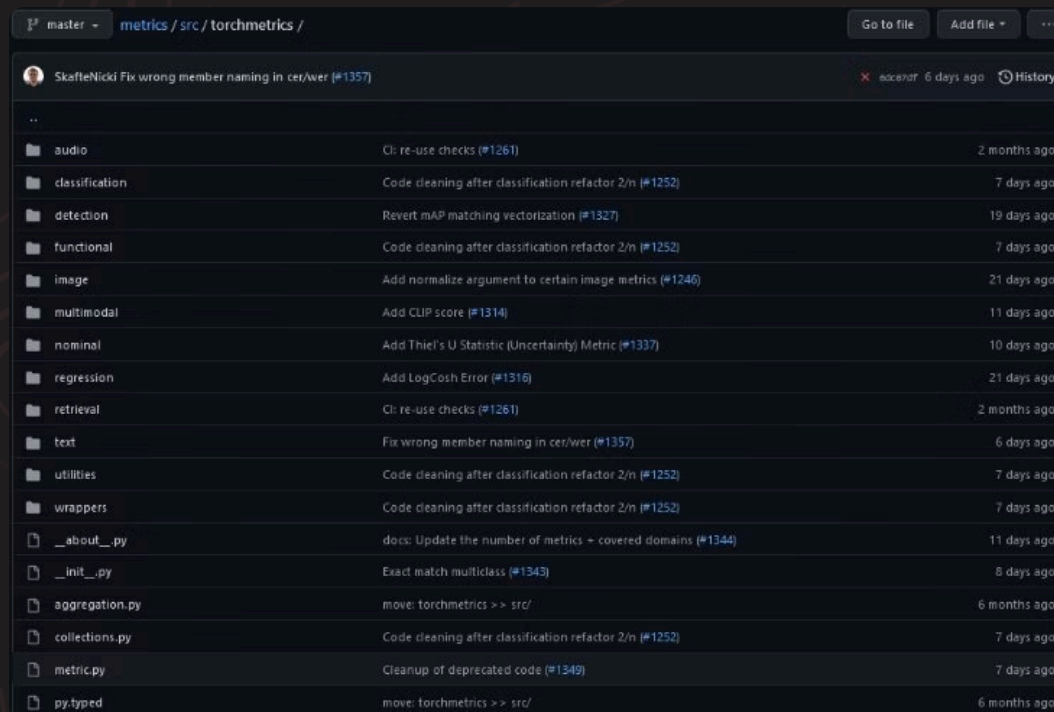
src/<project\_name>

(src-layout)

or

<project\_name>

(flat-layout)



The screenshot shows a GitHub repository view for the 'metrics / src / torchmetrics /' directory. The table lists files and folders with their commit history:

File/Folder	Commit Message	Time Ago
..		
audio	Cl: re-use checks (#1261)	2 months ago
classification	Code cleaning after classification refactor 2/n (#1252)	7 days ago
detection	Revert mAP matching vectorization (#1327)	19 days ago
functional	Code cleaning after classification refactor 2/n (#1252)	7 days ago
image	Add normalize argument to certain image metrics (#1246)	21 days ago
multimodal	Add CLIP score (#1314)	11 days ago
nominal	Add Thiel's U Statistic (Uncertainty) Metric (#1337)	10 days ago
regression	Add LogCosh Error (#1316)	21 days ago
retrieval	Cl: re-use checks (#1261)	2 months ago
text	Fix wrong member naming in cer/wer (#1357)	6 days ago
utilities	Code cleaning after classification refactor 2/n (#1252)	7 days ago
wrappers	Code cleaning after classification refactor 2/n (#1252)	7 days ago
__about__.py	docs: Update the number of metrics + covered domains (#1344)	11 days ago
__init__.py	Exact match multiclass (#1343)	8 days ago
aggregation.py	move: torchmetrics >> src/	6 months ago
collections.py	Code cleaning after classification refactor 2/n (#1252)	7 days ago
metric.py	Cleanup of deprecated code (#1349)	7 days ago
py.typed	move: torchmetrics >> src/	6 months ago

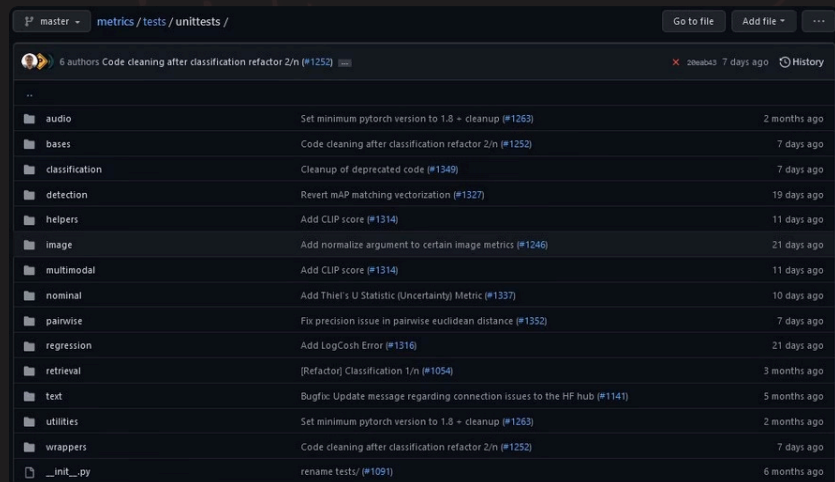


# CI step 2: write tests

For tests, the convention is to either place the tests in a separate tests folder, or put the tests in the same folder as the function/class/submodule they are testing.

```
|—— README.md
|—— src/
|   |—— __init__.py
|   |—— important_functions.py
|—— tests/
|   |—— __init__.py
|   |—— test_important_functions.py
```

```
|—— README.md
|—— src/
|   |—— __init__.py
|—— submodule/
|   |—— __init__.py
|   |—— important_functions.py
|   |—— test_important_functions.py
```



The screenshot shows a GitHub repository interface for the 'metrics / tests / unittests /' directory. It displays a commit history table with columns for commit details, commit message, and time since commit. The table lists various commits across different submodules like audio, bases, classification, detection, helpers, image, multimodal, nominal, pairwise, regression, retrieval, test, utilities, wrappers, and \_\_init\_\_.py.

Commit	Message	Time
6 authors	Code cleaning after classification refactor 2/n (#1252)	7 days ago
..		
audio	Set minimum pytorch version to 1.8 + cleanup (#1263)	2 months ago
bases	Code cleaning after classification refactor 2/n (#1252)	7 days ago
classification	Cleanup of deprecated code (#1349)	7 days ago
detection	Revert map matching vectorization (#1327)	19 days ago
helpers	Add CLIP score (#1314)	11 days ago
image	Add normalize argument to certain image metrics (#1246)	21 days ago
multimodal	Add CLIP score (#1314)	11 days ago
nominal	Add Thiel's U Statistic (Uncertainty) Metric (#1337)	10 days ago
pairwise	Fix precision issue in pairwise euclidean distance (#1352)	7 days ago
regression	Add LogCosh Error (#1316)	21 days ago
retrieval	[Refactor] Classification 1/n (#1054)	3 months ago
test	Bugfix: Update message regarding connection issues to the HF hub (#1141)	5 months ago
utilities	Set minimum pytorch version to 1.8 + cleanup (#1263)	2 months ago
wrappers	Code cleaning after classification refactor 2/n (#1252)	7 days ago
__init__.py	rename tests/ (#1091)	6 months ago

# CI step 2: write tests

💡 In python, we recommend using the **pytest** framework.

💡 Test are simple functions that start with *test\_* and uses *assert*

```
import torch
from torch.nn.functional import mse_loss

def test_mse_loss_zeros():
    # (0 - 0)**2 = 0
    assert mse_loss(torch.zeros(1,), torch.zeros(1,)) == 0

def test_mse_loss_ones():
    # (1 - 0)**2 = 1
    assert mse_loss(torch.ones(1,), torch.zeros(1,)) == 0
```

# CI step 2: write tests

Test 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')
```

# CI step 2: write test

or complicated

```
@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
            }
        )
```

# Exercise 1: The data contract

Imagine you are building a model to predict **House Prices**. Aside from 'the code doesn't crash,' list 3 automated checks you would run on the **input data** before training starts to ensure the data is healthy."

Example dataset:



 [www.kaggle.com](https://www.kaggle.com)

**Housing Prices Dataset**

Housing Prices Prediction – Regression Problem



# Solution: Exercise 1 – The Data Contract

Automated checks for healthy input data before model training:



## Data Type & Range Validation

Confirm numerical features like 'square footage' and 'number of bedrooms' fall within plausible ranges (e.g., non-negative) and maintain their expected data types.



## Missing Values Threshold

Verify that no essential columns (e.g., 'price', 'address') contain missing values above a predefined, acceptable percentage, preventing incomplete data from skewing results.



## Categorical Consistency

Validate categorical features such as 'neighborhood' or 'property type' against a list of known, expected values to catch typos or new, unhandled categories.



## Exercise 2: The Smoke Test

You've just finished training a new model version in your CI pipeline. What is the very first thing you should do with that model file before saving it to a Model Registry? Write down one 'sanity check'.

# Solution: Exercise 2 - The Smoke Test

## Model Output Validation

Load the newly trained model and perform a prediction on a small, known-good data sample. Verify that the model's output is not empty or erroneous, its data type matches expectations (e.g., float, integer), and it falls within a sensible range for the problem (e.g., a positive house price, a probability between 0 and 1).

# Exercise 3: ML Bug Hunting

A developer pushes code that passes all unit tests, but the model's accuracy on the test set drops from 95% to 40%.  
Discuss with your partner: Should the CI pipeline 'fail' this build? How would you write a test to catch this automatically?"

# Solution: Exercise 3 - ML Bug Hunting

## Implement a Model Performance Test

Introduce an automated regression test in the CI pipeline. This test should:

- Load the newly trained model.
- Evaluate its performance on a stable, versioned test dataset.
- Compare key metrics (e.g., accuracy, F1-score) against a predefined threshold (e.g., accuracy  $\geq 90\%$ ).
- Fail the build if performance falls below the threshold, blocking deployment.

# CI step 2: execute locally

```
(lightning) C:\Users\nsde\Documents\metrics>pytest tests\unittests\regression\test_mean_error.py
===== test session starts =====
platform win32 -- Python 3.8.13, pytest-6.2.5, py-1.11.0, pluggy-1.0.0
rootdir: C:\Users\nsde\Documents\metrics, configfile: setup.cfg
plugins: cov-4.0.0, doctestplus-0.12.1, timeout-2.1.0
collected 116 items

tests\unittests\regression\test_mean_error.py sssssssssssss.....ssssssssssssss.....xxxxxxxx.....

===== warnings summary =====
..\..\Anaconda3\envs\lightning\lib\site-packages\_pytest\config\_init_.py:1183
C:\Users\nsde\Anaconda3\envs\lightning\lib\site-packages\_pytest\config\_init_.py:1183: PytestDeprecationWarning: The --strict option is deprecated, use --strict-markers instead.
  self.issue_config_time_warning(


-- Docs: https://docs.pytest.org/en/stable/warnings.html
===== 80 passed, 28 skipped, 8 xfailed, 1 warning in 16.44s =====
```








- Test passed
- F Test failed
- s Test skipped (`pytest.skipif`, `pytest.skip`)
- X Test was expected to fail (`pytest.xfail`)

Do you remember to do this before each commit?

Let's automate doing it instead

# CI step 3: Automating stuff

What can be automated: EVERYTHING 

-  Unit testing
-  Integration testing
-  Documentation creation
-  Linters (style formatting)
-  Security checks
-  Code coverage
-  Custom checks...

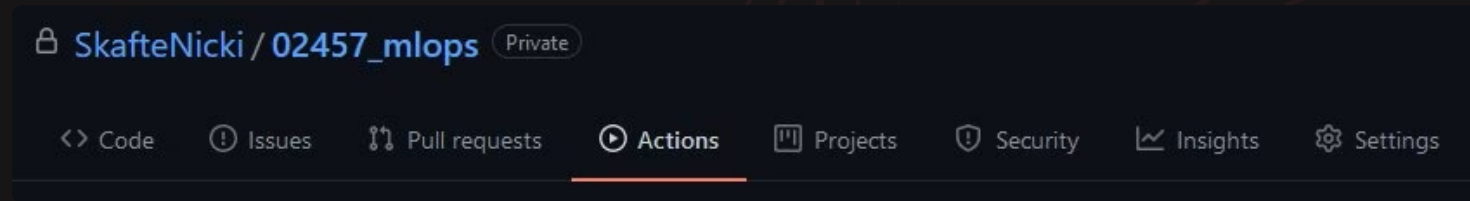
Only your imagination is the limit...



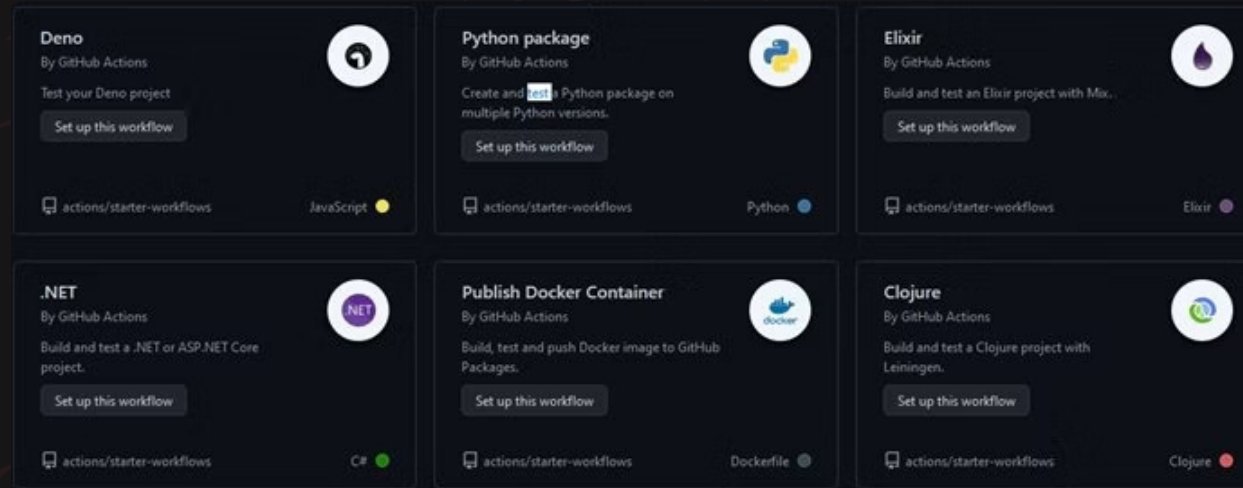
# CI step 3: Github actions

Build-in continuous integration in Github.

Free 2000 automation minutes/month (public repository)



Many ready to go workflows



# CI step 3: workflow files

Workflow files are a set of instructions that should be executed on a virtual machine hosted by Github

You can have one or many workflow files (runs in parallel)

When should workflow be triggered

Define OS + python

Clones code

Setup Python

Install dependencies

Check formatting

Run tests

```
1  name: Python package
2
3  on:
4    push:
5      branches: [ main ]
6    pull_request:
7      branches: [ main ]
8
9  jobs:
10   build:
11
12     runs-on: ubuntu-latest
13     strategy:
14       matrix:
15         python-version: ["3.7", "3.8", "3.9", "3.10"]
16
17     steps:
18       - uses: actions/checkout@v3
19       - name: Set up Python ${ matrix.python-version }
20         uses: actions/setup-python@v4
21         with:
22           python-version: ${ matrix.python-version }
23       - name: Install dependencies
24         run: |
25           python -m pip install --upgrade pip
26           pip install flake8 pytest
27           pip install -r requirements.txt
28           python setup.py install
29       - name: Lint with flake8
30         run: |
31           flake8 src/
32       - name: Test with pytest
33         run: |
34           pytest tests/
35
```

# CI step 3: workflow files

master metrics / .github / workflows /

Go to file Add file ...

Borda ci: reruns failed tests (#1348) 7d7daf 7 days ago History

ci-checks.yml	CI: merge conda and full tests (#1324)	18 days ago
ci-docker.yml	ci: reruns failed tests (#1348)	7 days ago
ci-integrate.yml	ci: reruns failed tests (#1348)	7 days ago
ci-tests-full.yml	ci: reruns failed tests (#1348)	7 days ago
docs-check.yml	Add CLIP score (#1314)	11 days ago
focus-diff.yml	Bump actions/setup-python from 2 to 4 (#1169)	4 months ago
creating.yml	Initial commit	7 years ago

- 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 failer 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
```

# CI step 3: workflow files

✓ 43 checks in total

Test a combination of

- 💡 Hardware setup
- 💡 Operating system
- 💡 Python version
- 💡 Core dependencies

Runs unit tests, build documentation, test coverage, linting of code, package installer etc.

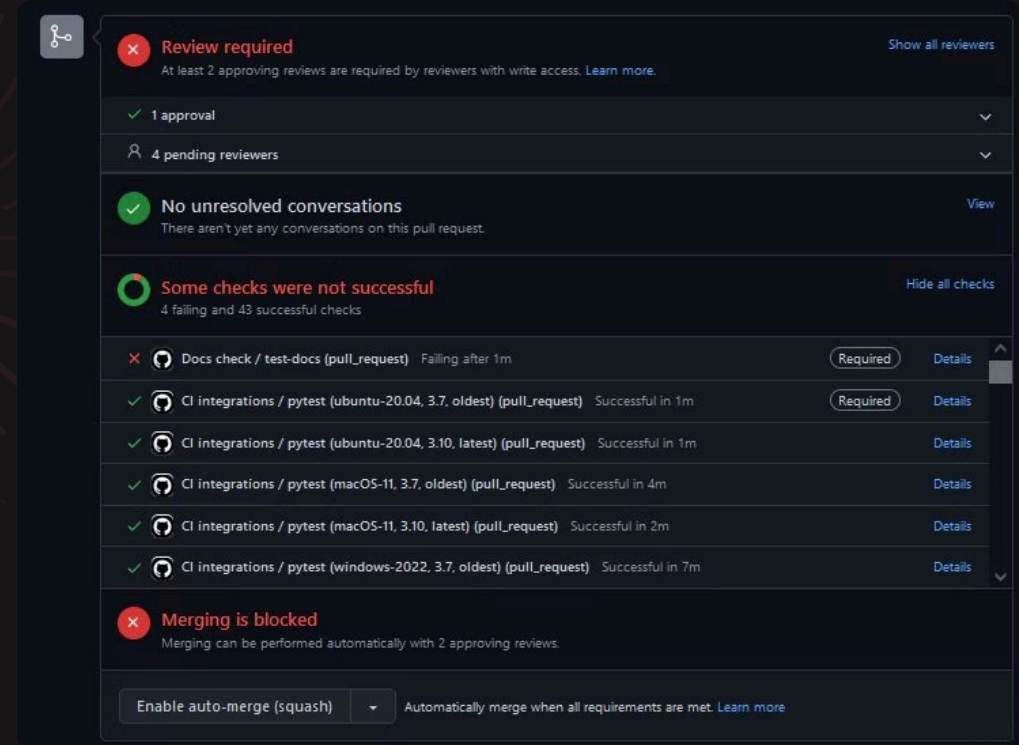
✓ Mergify	✓ CI testing - complete on: pull_request	1	✓ General checks on: pull_request	2
✓ Summary	✓ check-diff / eval-diff		✓ check-code / mypy	
✓ WIP	✗ pytest (ubuntu-20.04, 3.8, 1.9.1)		✗ check-code / pre-commit	
✓ WIP	✗ pytest (ubuntu-20.04, 3.8, 1.10.2)		✓ check-schema / schema	
✓ Docs check on: pull_request	✗ pytest (ubuntu-20.04, 3.8, 1.11.0)		✓ check-package / pkg-check (ub...	
✓ test-docs	✗ pytest (ubuntu-20.04, 3.8, 1.12.1)		✓ check-package / pkg-check (ub...	
✓ make-docs	✗ pytest (ubuntu-22.04, 3.8, 1.13.0)		✓ check-package / pkg-check (ma...	
✓ paper-JOSS	✗ pytest (ubuntu-22.04, 3.10, 1.13...		✓ check-package / pkg-check (ma...	
✓ paper-cite	✗ pytest (macOS-11, 3.8, 1.13.0)		✓ check-package / pkg-check (ma...	
✓ CI integrations on: pull_request	✗ pytest (macOS-11, 3.9, 1.13.0)		✓ check-package / pkg-check (wi...	
✓ pytest (ubuntu-20.04, 3.7, oldest)	✗ pytest (windows-2022, 3.8, 1.13.0)		✓ check-package / pkg-check (wi...	
✓ pytest (ubuntu-20.04, 3.10, latest)	✗ pytest (windows-2022, 3.9, 1.13.0)		✓ check-package / pkg-check (wi...	
✓ pytest (macOS-11, 3.7, oldest)	✗ pytest (ubuntu-20.04, 3.7, 1.8.1,...)			
✓ pytest (macOS-11, 3.10, latest)	✗ pytest (ubuntu-20.04, 3.8, 1.8.1,...)		✓ Azure Pipelines	
✓ pytest (windows-2022, 3.7, olde...	✗ pytest (macOS-11, 3.7, 1.8.1, old...		✗ Lightning-AI.metrics Re-run	
✓ pytest (windows-2022, 3.10, lat...	✗ pytest (macOS-11, 3.8, 1.8.1, old...		✗ Lightning-AI.metrics (pytest PyT... Re-run	
✓ pytest (3.10, latest, ubuntu-22.04)	✗ pytest (windows-2019, 3.7, 1.8.1...		✗ Lightning-AI.metrics (pytest PyT... Re-run	
✓ pytest (3.10, latest, macOS-12)	✗ pytest (windows-2019, 3.8, 1.8.1...			

# CI step 3: Code is checked before merging

Branch protection rules:

- ⚠️ All/some tests should pass
- ⚠️ At least x core members need to approve
- ⚠️ Comments should be taken care of

View more [here](#)



The screenshot displays the GitHub pull request status bar, which outlines the requirements for merging code. It includes sections for approvals, pending reviews, unresolved conversations, and a list of CI checks. A 'Review required' message states that at least two approving reviews are needed. The 'No unresolved conversations' section indicates that there are no pending discussions. The 'Some checks were not successful' section shows that 4 checks failed and 43 were successful. The failed checks include 'Docs check / test-docs (pull\_request)' which is failing after 1 minute. The successful checks include 'CI integrations / pytest' on various operating systems and Python versions. At the bottom, a 'Merging is blocked' message states that merging can be performed automatically with 2 approving reviews. There is also an option to 'Enable auto-merge (squash)' which would automatically merge when all requirements are met.

**Review required** [Show all reviewers](#)  
At least 2 approving reviews are required by reviewers with write access. [Learn more.](#)

✓ 1 approval  
4 pending reviewers

✓ No unresolved conversations [View](#)  
There aren't yet any conversations on this pull request.

⚠️ Some checks were not successful [Hide all checks](#)  
4 failing and 43 successful checks

Check Status	Check Name	Result	Details
✗	Docs check / test-docs (pull_request)	Failing after 1m	<a href="#">Details</a>
✓	CI integrations / pytest (ubuntu-20.04, 3.7, oldest) (pull_request)	Successful in 1m	<a href="#">Details</a>
✓	CI integrations / pytest (ubuntu-20.04, 3.10, latest) (pull_request)	Successful in 1m	<a href="#">Details</a>
✓	CI integrations / pytest (macOS-11, 3.7, oldest) (pull_request)	Successful in 4m	<a href="#">Details</a>
✓	CI integrations / pytest (macOS-11, 3.10, latest) (pull_request)	Successful in 2m	<a href="#">Details</a>
✓	CI integrations / pytest (windows-2022, 3.7, oldest) (pull_request)	Successful in 7m	<a href="#">Details</a>

✗ Merging is blocked  
Merging can be performed automatically with 2 approving reviews.

Enable auto-merge (squash) [Automatically merge when all requirements are met. Learn more](#)

# CI step 3: Automate tedious tasks with bots

justusschock and others added 5 commits 7 days ago

- Update test\_auc.py Verified ✗ 674e7ea
- [pre-commit.ci] auto fixes from pre-commit.com hooks ✗ 07a4b85
- Update test\_auc.py Verified ✗ 85e4458
- [pre-commit.ci] auto fixes from pre-commit.com hooks ✗ 0a94ba5
- Update test\_auc.py Verified ✗ 0527efd

tests/classification/test\_auc.py

```
@@ -93,9 +93,9 @@ def test_auc_differentiability(self, x, y, reorder):
```

```
93 def test_auc(x, y, expected, unsqueeze_x, unsqueeze_y):
94     if unsqueeze_x:
95         x = x.unsqueeze(-1)
96 -
97     if unsqueeze_y:
98         y = y.unsqueeze(-1)
99 -
100     # Test Area Under Curve (AUC) computation
101     assert auc(tensor(x), tensor(y), reorder=True) == expected
```

```
93 def test_auc(x, y, expected, unsqueeze_x, unsqueeze_y):
94     if unsqueeze_x:
95         x = x.unsqueeze(-1)
96 +
97     if unsqueeze_y:
98         y = y.unsqueeze(-1)
99 +
100     # Test Area Under Curve (AUC) computation
101     assert auc(tensor(x), tensor(y), reorder=True) == expected
```



# CI step 3: Automate tedious tasks with bots

🤖 Dependabot can help auto checking new releases of dependencies in your project

The image shows a GitHub pull request (PR) titled "build(deps): update kornia requirement from <0.7.1, >=0.6.7 to >=0.6.7, <0.7.2 in /requirements #2293". The PR is created by dependabot and is currently open. A red arrow points from the PR title to the code diff, highlighting the update for kornia.

**PR Details:**

- Title:** build(deps): update kornia requirement from <0.7.1, >=0.6.7 to >=0.6.7, <0.7.2 in /requirements #2293
- Author:** dependabot
- Branch:** dependabot-gh-requirements-kornia-gt-0.6.7-and-lt-0.7.2
- Files changed:** 1 file changed
- Commits:** 1 commit
- Labels:** Ready To Go
- Assignees:** No one assigned
- Projects:** None yet
- Milestones:** No milestone
- Development:** Successfully merging this pull request
- Notifications:** You're receiving notifications for this pull request
- Participants:** 2 participants
- Lock conversation:** Lock conversation

**Code Diff:**

```
requirements/image_test.txt
@@ -2,7 +2,7 @@
2  # in case you want to preserve/enforce restrictions on the latest
3  # compatible version, add "strict" as an in-line comment
4  scikit-image >=0.19.0, <=0.21.0
5  - kornia >=0.6.7, <0.7.1
6  + kornia >=0.6.7, <0.7.2
7  pytorch-msssim ==1.0.0
8  sewar >=0.4.4, <=0.4.6
9  numpy <1.25.0
```

**dependabot.yml file content:**

```
# Basic dependabot.yml file with
# minimum configuration for two package managers

version: 2
updates:
  # Enable version updates for python
  - package-ecosystem: "pip"
    # Look for a `requirements` in the `root` directory
    directory: "/requirements"
    # Check for updates once a week
    schedule:
      interval: "weekly"
```

# Summary of continues integration

Use version control



Write (unit-)test for your code



Automate build + test



# The agents are here

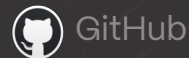
Agents can help you solve issues, review code and solve bugs...

SkaftaNicki/dtu\_mlops

## #480 [bug] CLI exercise classifier dataset mismatch

0 comments

 iarata opened on January 6, 2026




GitHub

## [bug] CLI exercise classifier dataset mismatch · Issue #480 · S...

In exercise of Command line arguments at  
s2\_organisation\_and\_version\_control/cli/, part 3 mentions: ... trains a...

# A few corrections to last weeks template

SkaftaNicki/mlops\_template




## #93 Code helpers + updated workflows for uv

0 comments


0 reviews

23 files

+518 -65

 **SkaftaNicki** • January 12, 2026

14 commits

 GitHub

Code helpers + updated workflows for uv by SkaftaNicki · Pull ...

This pull request introduces support for an optional &quot;coding agent&quot; feature in the project template, adds a specialized...

- Smaller updates to agents
- Specific works for pip and uv workflows
- Correction of Docker uv files

# Meme of the day



**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

