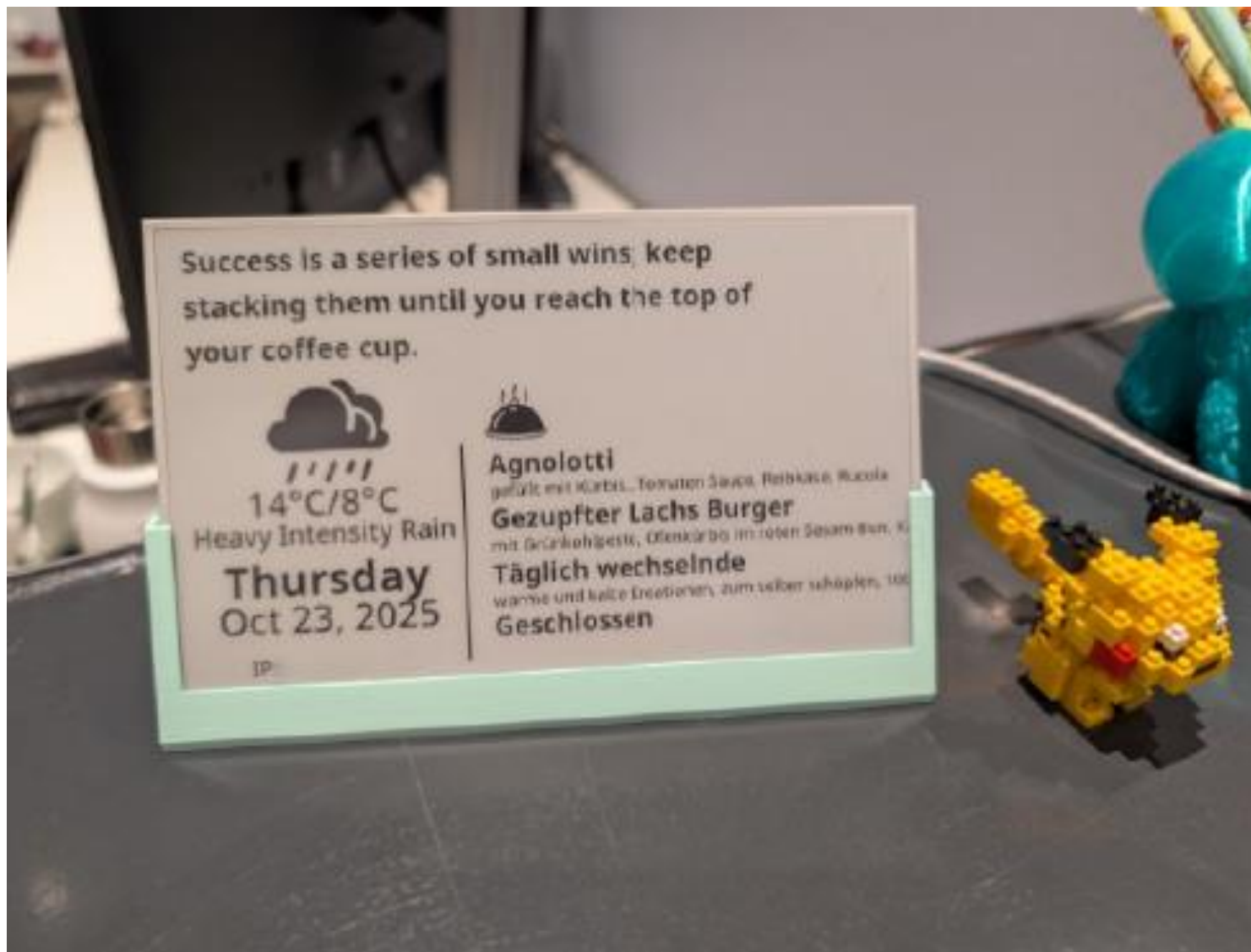


Automation

Florian Bär
MLOps

Information Technology
23.10.2024

Automation



Overview

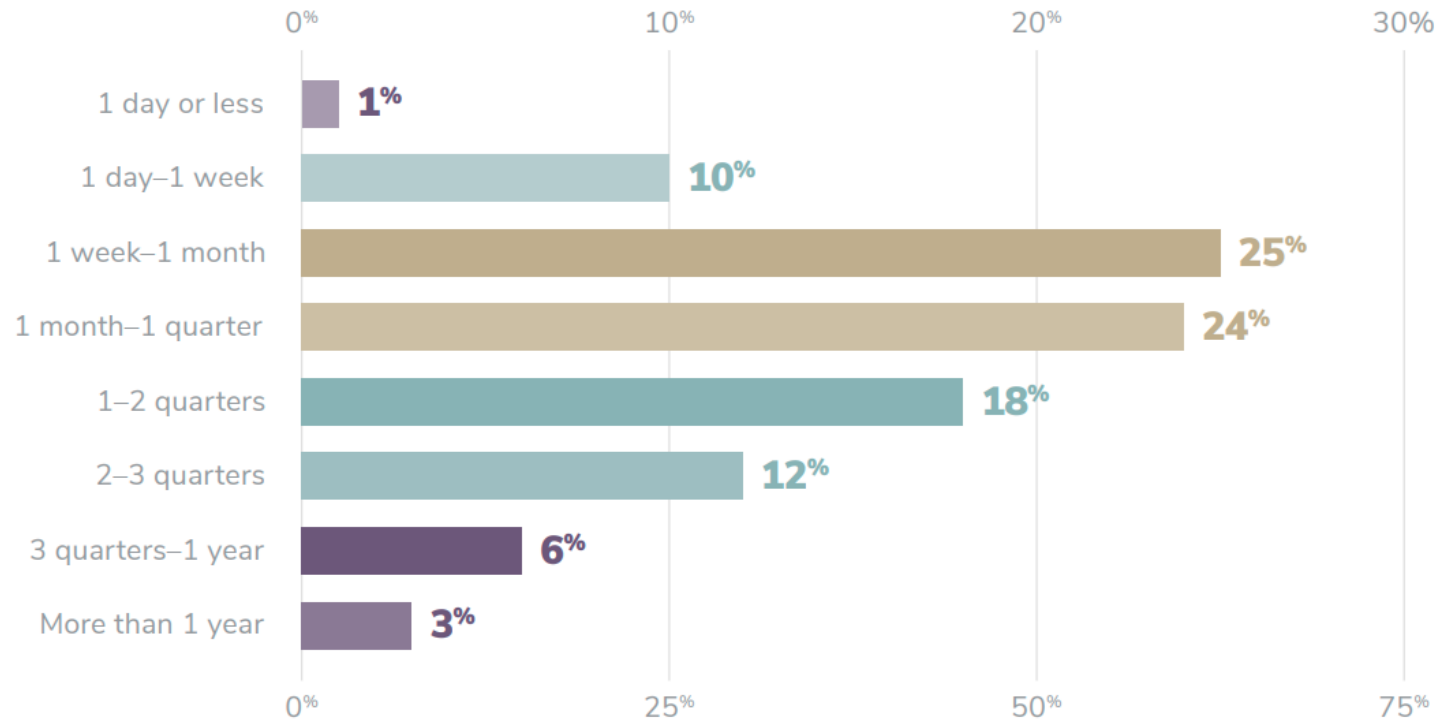
- Impact of Automation
- CI/CD pipelines
- Automation for ML

Automated Machine Learning (AutoML)

- Applied to ...
 - Code/data/model/experiment management
 - Preprocessing
 - Feature engineering
 - Model selection
 - Training
 - Evaluation
 - Deployment
 - Monitoring





Impact of Automation

Only 11% of organizations can put a model into production within a week, and 64% take a month or longer



Impact of Automation

- Aim for
 - on-demand deployments
 - updates/fixes to be integrated into products immediately
 - automatically restoring failed services
 - (obviously) reduce number of failures in production through good test coverage

Software delivery performance metric	Elite	High	Medium	Low
 Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per week and once per month	Between once per month and once every 6 months	Fewer than once per six months
 Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one hour	Between one day and one week	Between one month and six months	More than six months
 Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day	Between one day and one week	More than six months
 Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0%-15%	16%-30%	16%-30%	16%-30%

Standard Automation

- Scripts
 - Automate multiple steps instead of performing them manually
 - Popular: Shell/Bash/Make
- Cron jobs: Periodically run commands/scripts
- Uses cases
 - Create regular backups
 - Compute data statistics
 - Log rotation (prevent log files from becoming too big)

Cron

- χρόνος (chronos), the Greek primordial god of time and eternity
- Command-line utility for Unix-like operating systems
- Cron jobs defined in cron tab (cron table)

- <https://crontab.guru/>

Example:

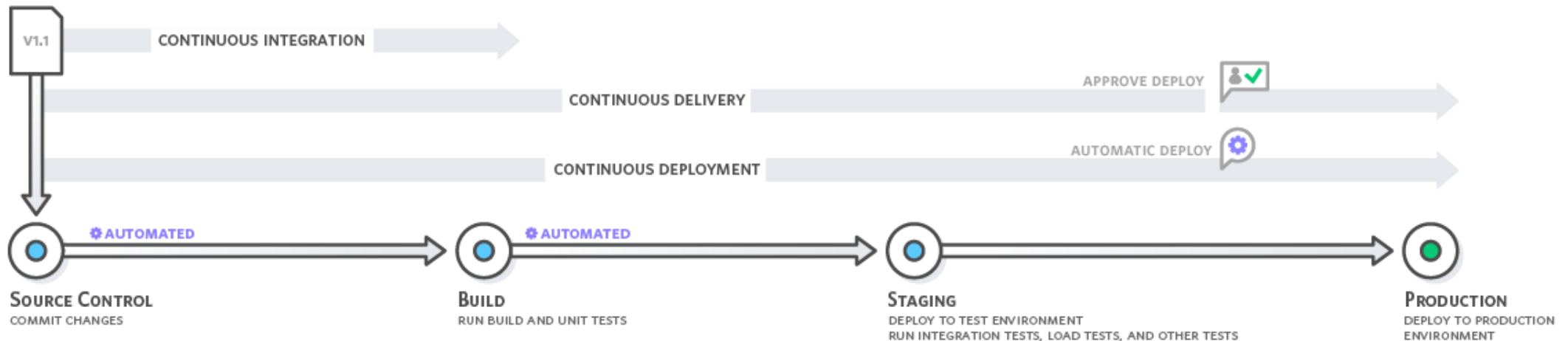
```
45 23 * * * /home/user/backup_db.sh
```

```
# minute (0-59)
# hour (0-23)
# day of the month (1-31)
# month (1-12)
# day of the week (0-6) (Sunday to Saturday;
# 7 is also Sunday on some systems)
# * * * * * <command to execute>
```

Description	Equivalent to
Run once a year at midnight of 1 January	0 0 1 1 *
Run once a month at midnight of the first day of the month	0 0 1 * *
Run once a week at midnight on Sunday	0 0 * * 0
Run once a day at midnight	0 0 * * *
Run once an hour at the beginning of the hour	0 * * * *

CI/CD

- Continuous Integration/Delivery/Deployment for code



GitHub Actions

- CI/CD platform
- Works with workflows that are triggered by events
 - Build and test a pull request
 - Automatically deploy merged pull requests
- Events can also be unrelated to code commits
 - A new issue is opened
 - Workflow: Automatically assign a label and an assignee

GitHub Actions: Workflow

- A workflow runs ≥ 1 jobs
 - In sequence or parallel
 - Each run in a separate runner
 - Jobs consist of ≥ 1 steps
 - Script or custom application
- Defined by a YAML file
 - In `.github/workflows` directory
 - [Syntax](#)
- Triggered by
 - Event
 - Manually
 - Scheduled time

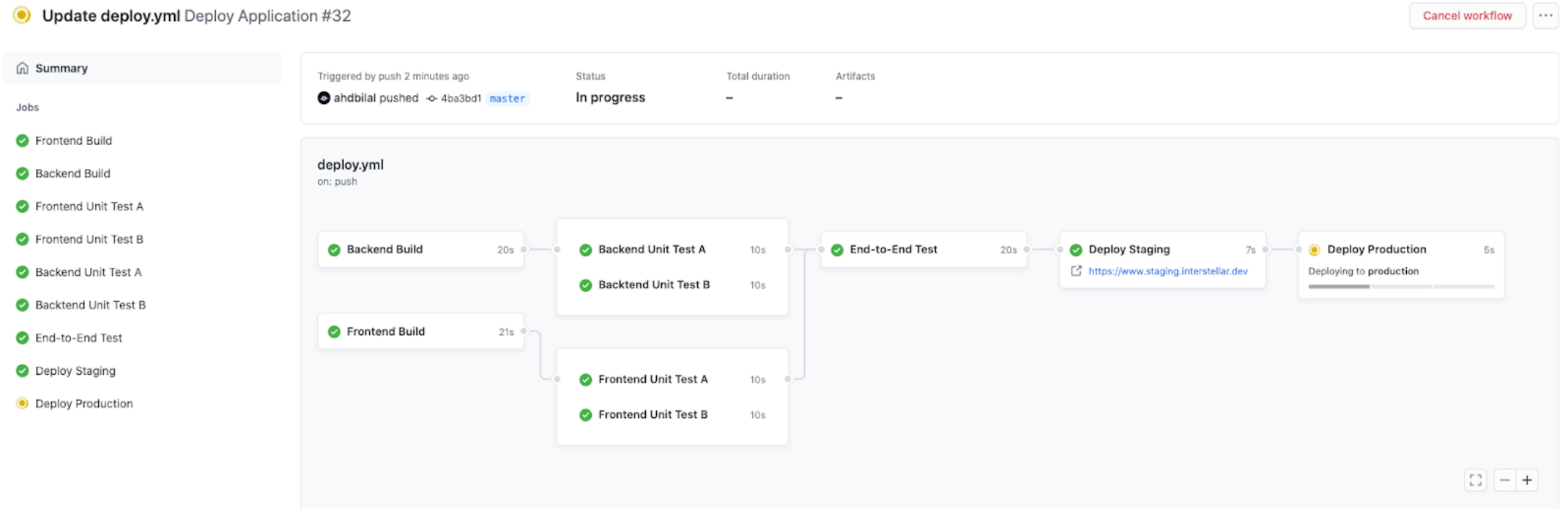
```
YAML
name: GitHub Actions Demo
run-name: ${{ github.actor }} is testing out GitHub Actions 🚀
on: [push]
jobs:
  Explore-GitHub-Actions:
    runs-on: ubuntu-latest
    steps:
      - run: echo "🎉 The job was automatically triggered by a ${{ github.event_name }} event."
      - run: echo "🐙 This job is now running on a ${{ runner.os }} server hosted by GitHub!"
      - run: echo "🔗 The name of your branch is ${{ github.ref }} and your repository is ${{ github.repository }}."
      - name: Check out repository code
        uses: actions/checkout@v4
      - run: echo "💡 The ${{ github.repository }} repository has been cloned to the runner."
      - run: echo "🖨️ The workflow is now ready to test your code on the runner."
      - name: List files in the repository
        run: |
          ls ${{ github.workspace }}
      - run: echo "🍏 This job's status is ${{ job.status }}."
```

GitHub Actions: Examples

- Generation of PDF Documents based on \LaTeX
 - <https://github.com/florianbaer/VM-Report-Template>
- Compile code (C#, Java, Go, Rust...)
- Lint / Transpile (Typescript, JavaScript, Python)
- Run Unit-Tests
 - <https://docs.github.com/en/actions/tutorials/build-and-test-code/python>
- Build Docker Image
 - <https://docs.github.com/en/actions/tutorials/publish-packages/publish-docker-images>

Workflow Visualization

- See workflow's jobs and their progress



GitHub Actions Demo

More examples at: <https://github.com/aaronwinston/open-sauced/tree/main/.github/workflows>

Continuous Machine Learning (CML)

- Open-source CI/CD Tool for ML
- Same company also created DVC
- Uses Git for experiment tracking
 - Integrated with GitHub, GitLab and BitBucket
 - Uses DVC for data and models
- Automatically runs training & evaluations for changes
 - Makes use of cloud resources and spot instances
 - Creates a report and adds it to the pull request
- Runners for cloud or on-premise

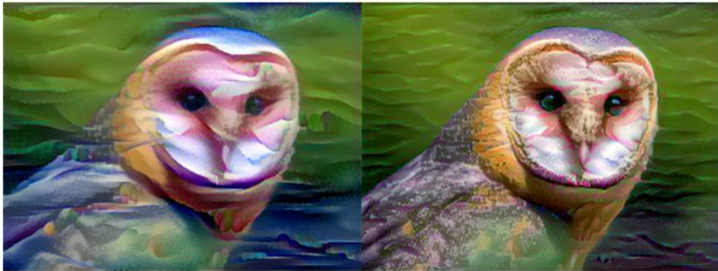
CML Comments

- Add metrics and visualizations to pull request's comments

andronovhopf commented on c5eebf3 21 hours ago

Style transfer

Workspace vs. Master



Training metrics

Path	Param	Old	New
params.yaml	iter	1	30

modify forest depth by andronovhopf

github.com/andronovhopf/example_cml/pull/4

Open modify forest depth #4
andronovhopf wants to merge 1 commit into master from experiment

Verified ✓ 3ac0171

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

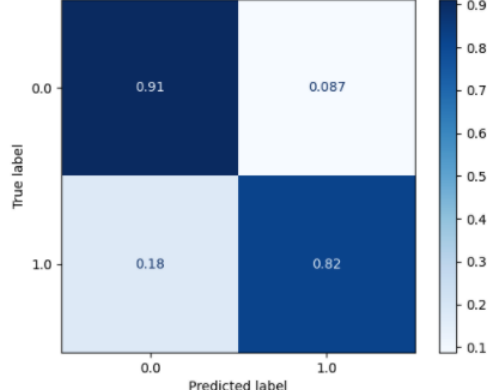
Linked issues
Successfully merging this pull request may close these issues.
None yet

Notifications
Customize
Unsubscribe
You're receiving notifications because you authored the thread.

1 participant

Lock conversation

Accuracy: 0.864



0.0 0.91 0.087 0.18 0.82 1.0

True label

Predicted label

Add more commits by pushing to the experiment branch on andronovhopf/example_cml.

✓ All checks have passed
1 successful check
Show all checks

✓ This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request

You can also open this in GitHub Desktop or view command line instructions.

CML Runners

- Execute training/evaluation in the cloud or on-prem
- Example: Launch on AWS

```
name: Train-in-the-cloud
on: [push]
jobs:
  deploy-runner:
    runs-on: ubuntu-latest
    steps:
      - uses: iterative/setup-cml@v1
      - uses: actions/checkout@v3
      - name: Deploy runner on EC2
        env:
          REPO_TOKEN: ${ secrets.PERSONAL_ACCESS_TOKEN }
          AWS_ACCESS_KEY_ID: ${ secrets.AWS_ACCESS_KEY_ID }
          AWS_SECRET_ACCESS_KEY: ${ secrets.AWS_SECRET_ACCESS_KEY }
        run: |
          cml runner launch \
            --cloud=aws \
            --cloud-region=us-west \
            --cloud-type=g4dn.xlarge \
            --labels=cml-gpu

  train-model:
    needs: deploy-runner
    runs-on: [self-hosted, cml-gpu]
    timeout-minutes: 50400 # 35 days
    container:
      image: ghcr.io/iterative/cml:0-dvc2-base1-gpu
      options: --gpus all
    steps:
      - uses: actions/checkout@v3
      - name: Train model
        env:
          REPO_TOKEN: ${ secrets.PERSONAL_ACCESS_TOKEN }
        run: |
          pip install -r requirements.txt
          python train.py
```

CML TensorBoard

- Train models, save TensorBoard logs
- Upload TensorBoard logs to <https://tensorboard.dev/>
 - Makes the logs publicly accessible
- Add a link to the pull request
- Unfortunately, TensorBoard.dev has shut down
 - Can the same be done with Weights & Biases?

```
18     - name: cml
19       env:
20         repo_token: ${ secrets.GITHUB_TOKEN }
21         TB_CREDENTIALS: ${ secrets.TB_CREDENTIALS }
22       run: |
23         pip install -r requirements.txt
24         cml-tensorboard-dev --logdir logs --md --name "Go to tensorboard" >> report.md
25         cml-send-comment report.md
26         python train.py
```

CML with DVC

- Handles data versioning/downloads
- Allows visualizing how metrics differ between commits

```
run: |
# Install requirements
pip install -r requirements.txt

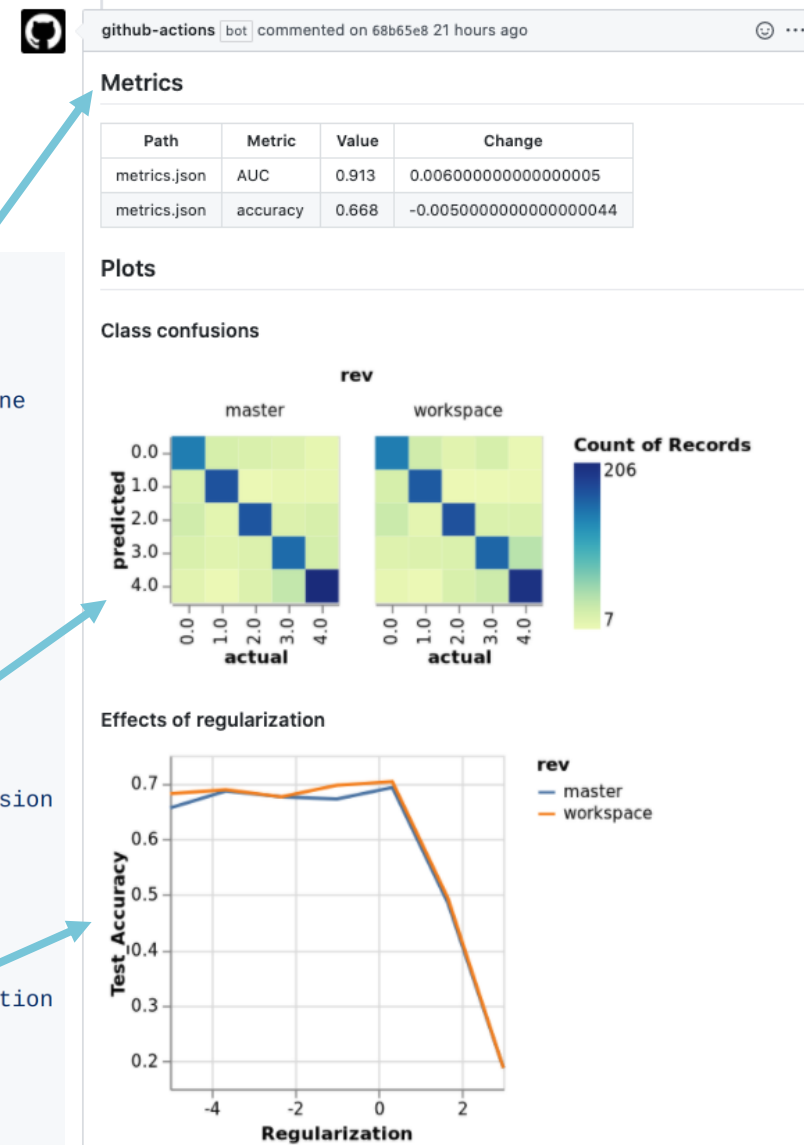
# Pull data & run-cache from S3 and reproduce pipeline
dvc pull data --run-cache
dvc repro

# Report metrics
echo "## Metrics" >> report.md
git fetch --prune
dvc metrics diff master --show-md >> report.md

# Publish confusion matrix diff
echo "## Plots" >> report.md
echo "### Class confusions" >> report.md
dvc plots diff --target classes.csv --template confusion
vl2png vega.json -s 1.5 > confusion_plot.png
echo "" >> report.md

# Publish regularization function diff
echo "### Effects of regularization" >> report.md
dvc plots diff --target estimators.csv -x Regularization
vl2png vega.json -s 1.5 > plot.png
echo "" >> report.md

cml comment create report.md
```



CML Demo

Based on: <https://cml.dev/doc/start/github>

https://github.com/florianbaer/example_cml

Continuous ... for ML

- Integration: Testing (for code, data and models)
- Delivery/Deployment: (Ready for) deployment when tests run through
- Training: Retrain/update a model when new data becomes available
 - Trigger evaluation when training finishes
- Monitoring: Track important indicators/metrics, find causes of bugs faster/preemptively

CI for ML

- Code
 - Write unit/regression tests
 - Trigger automatically with GitHub Actions/GitLab CI
- Data
 - Tests: Detection of outliers, duplicates, incomplete/invalid data
 - Compute data statistics
 - Run when a new data version is created (use CML with DVC)
 - Regularly (e.g. at midnight)
 - Compare statistics to previous versions
 - Create visualizations, dashboards, alarms (see monitoring lecture)
- Models
 - Unit tests: Test on specific inputs where the answer is known
 - Regression tests: Can the model still solve the previous test cases?
 - Run after training finishes, alongside evaluation

CD for ML

- Code
 - Use CD tools
 - Kubernetes / Openshift with ArgoCD / Flux...
- Models
 - Update deployment when tests have run through
 - Potentially perform A/B testing or shadow testing
 - See model validation lecture
 - Kubernetes: Use a rolling update of a new deployment
 - See containerization lecture
 - VertexAI / AI Studio: See docs...
 - Ray Serve: Redeploy a new configuration

Other AutoML Tools

- All big cloud providers offer “ML as a service”:
 - [Google Cloud AutoML](#), [Azure Machine Learning](#), [AWS AutoML Solutions](#), [Databricks AutoML](#)
 - With your own model: Mostly cluster management and MLOps tools
 - No-code solutions: Using their APIs for image classification, text generation, speech-to-text transcription, ...
 - Only provide data as input-output pairs
- Weights and Biases [Launch](#)
 - Start training runs locally or with different cloud providers
 - Automatically queue jobs and distribute to available resources
- auto-sklearn: Automatic scikit-learn [[GitHub](#)][[docs](#)]
- [Run:ai](#): Automatic cluster management
- Automatic deployment/serving by many others as well