Einführung in MLOps

05 Codespaces und Docker

Tobias Mérinat
teaching2025@fsck.ch

Lucerne University of
Applied Sciences and Arts
HOCHSCHULE
LUZERN

Department of Information Technology
Lucerne University of Applied Sciences and Arts
6343 Rotkreuz, Switzerland

14. und 15. Februar 2025

Arbeitsumgebung: Github Codespaces

- Wir führen die Übungen in der Cloud auf Github Codespaces durch
- Codespaces bietet eine Linux-basierte Entwicklungsumgebung, auf die via Browser zugegriffen werden kann
- Anforderungen:
  - Github Account
  - Genug Quota (Free Quota reicht aus, wenn der Space heute Abend gestoppt wird)
- Alle Übungen funktionieren jedoch auch unabhängig von Codespaces, z.B. lokal auf dem Laptop (gutes Docker-Knowhow vorausgesetzt)

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN

Codespaces

- Hauptseite: `https://github.com/codespaces`
- Beispiel-Repo (nur für diese Demo verwendet)
- Verfügbare Quota: `https://github.com/settings/billing/summary`

Docker

- In den Übungen verwenden wir *Docker Container*, um verschiedene Services sauber getrennt aufzusetzen.
- Docker Container sind eine Art leichtgewichtige virtuelle Maschinen
- Docker Container können miteinander kommunizieren
- In einem Container kann beispielsweise eine Webapplikation laufen, in einem anderen deren Datenbank

Docker Images

- Ein Docker Image definiert, wie ein Container aussieht
- Das ‘Dockerfile’ listet auf, wie ein Image gebaut wird.
  - Basis-Image
  - installierte Pakete
  - ins Image zu kopierende Files
  - zu startende Prozesse
- Images sind also die Vorlage für Container
- Images können aus einem Repository geladen oder lokal gebaut werden

Dockerfile Beispiel

```
FROM python:3.12.2-slim-bookworm
RUN pip install jupyter
EXPOSE 8888
ENTRYPOINT ["/usr/local/bin/jupyter-notebook"]
```

Docker Container

- Ein Container ist eine laufende Instanz eines Images
- In einem Container läuft ein Linux Betriebsystem . . .
- . . . auf welchem dann Dienste und Applikationen laufen können

docker-compose.yml

- Das File docker-compose.yml definiert, wie ein Dienst aussieht
  - welche Container laufen
  - aus welchen Images die Container instanziiert werden
  - welche Ports zugänglich sind
  - welche Verzeichnisse des Hosts in einem Container verfügbar sind
  - Umgebungsvariablen können gesetzt werden
  - wie mit einem Container kommuniziert werden kann

docker-compose.yml Beispiel

```
services:
   development_env:
      container_name: development_env
      image: ghcr.io/tobycheese/development_env:latest
      build: .
      environment:
        - GIT_PYTHON_REFRESH=quiet
      ports:
         - 127.0.0.1:8080:8888
      volumes:
         - ../../notebooks:/notebooks
      networks:
         - development
         - production
```

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN

Commandline

Wir verwenden zwei Befehle, um mit Docker zu interagieren:

- `docker compose {up, down, -d}` zum Starten/Stoppen eines oder mehrerer Services
- `docker run ...` zum Ausführen eines Befehls in einem laufenden Container

In den Übunden wird der genaue Befehl jeweils angegeben.

12 Minute Intro

https://youtu.be/YFl2mCHdv24

Repository

Die Übungen der kommenden zwei Tage findest du hier:

https://github.com/tobycheese/hslu_cas_ml_hs_2024

- Du musst das Repository nicht zwingend clonen, da du es direkt aus github in codespaces öffnen wirst
- Du kannst es trotzdem clonen, damit du eine lokale Kopie hast (oder du lädst es am Schluss des Kurses von codespaces)
- Jedoch **bitte nicht forken**, da ich das Repository nach Abschluss des Kurses wieder private setzen möchte.

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN

Codespaces Reminder

Wichtig: Da du mit Codespaces arbeitest, sichere deine Arbeit, bevor du morgen Abend die Maschine löschst.

Bereit für die erste Übung

- Nun starten wir gemeinsam die Übungsumgebung
- https://github.com/tobycheese/hslu_cas_ml_hs_2024
- Danach findest du die einzelnen Übungen unter `exercises/html/`
- Eine Übersicht findest du in `index.html`