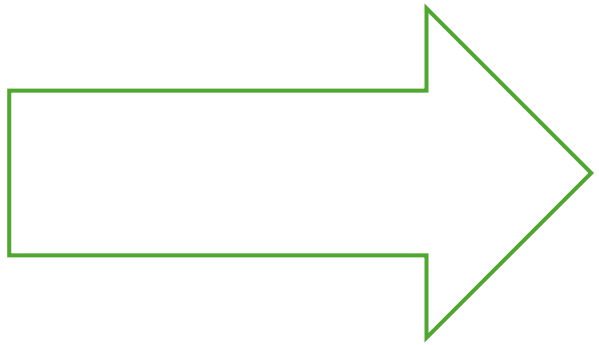
DVC

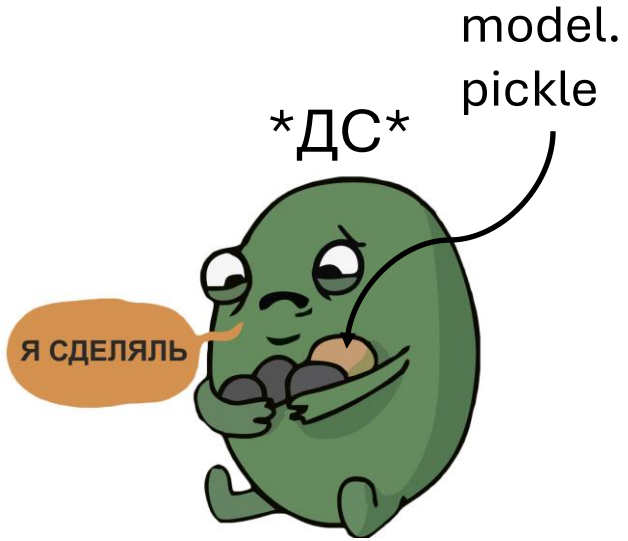
Лекция №4

Типичная картина: Recap



Код построения модели:

- не воспроизводим
- не обобщаемый
- не универсальный



Код использования модели:

- хороший
- воспроизводимый

3 этапа в жизни модели

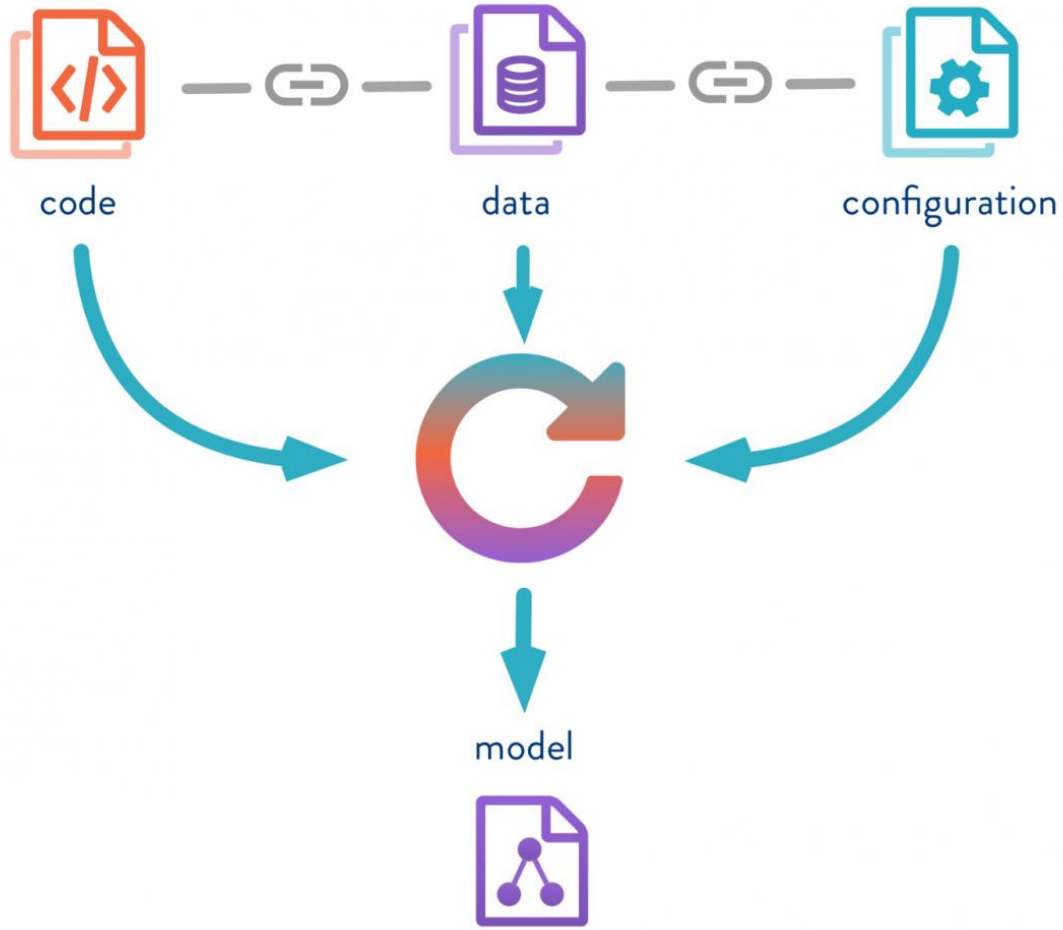
Дс

1) Обучение
2) 
3) Переобучение

Бэкенд

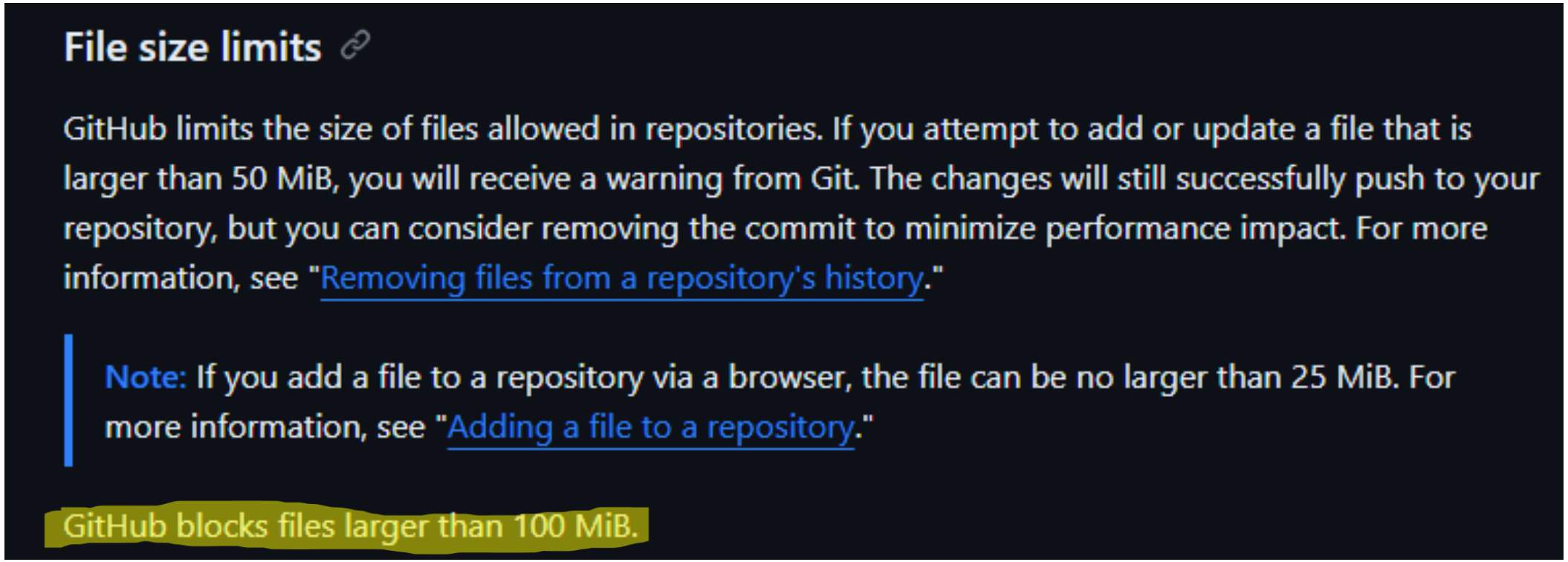
1) 
2) Выкатка и эксплуатация
3) 

Рецепт МЛ модели



- docs
- models
- notebooks
- references
- reports
- task
- test
- .gitignore
- Makefile
- README.md
- pyproject.toml
- requirements.txt
- setup.cfg

Как хранить данные?



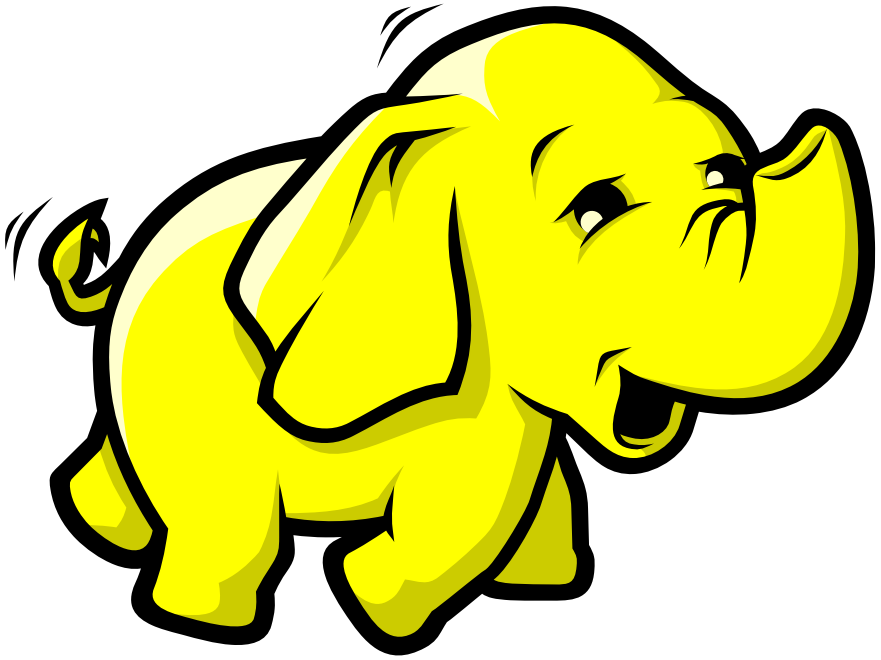
File size limits

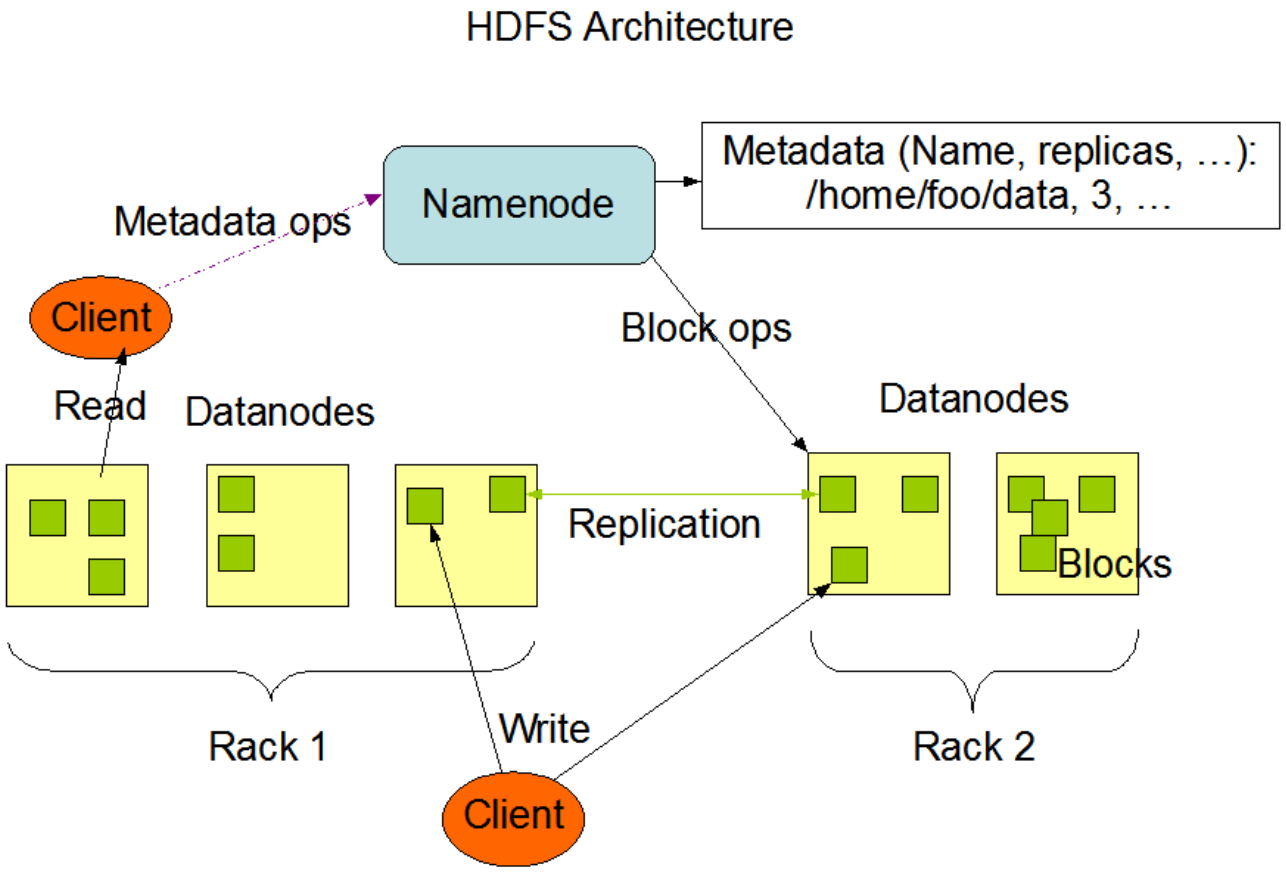
GitHub limits the size of files allowed in repositories. If you attempt to add or update a file that is larger than 50 MiB, you will receive a warning from Git. The changes will still successfully push to your repository, but you can consider removing the commit to minimize performance impact. For more information, see "Removing files from a repository's history."

> **Note:** If you add a file to a repository via a browser, the file can be no larger than 25 MiB. For more information, see "Adding a file to a repository."

GitHub blocks files larger than 100 MiB.

Куда складывать файлы?





Куда еще складывать данные?

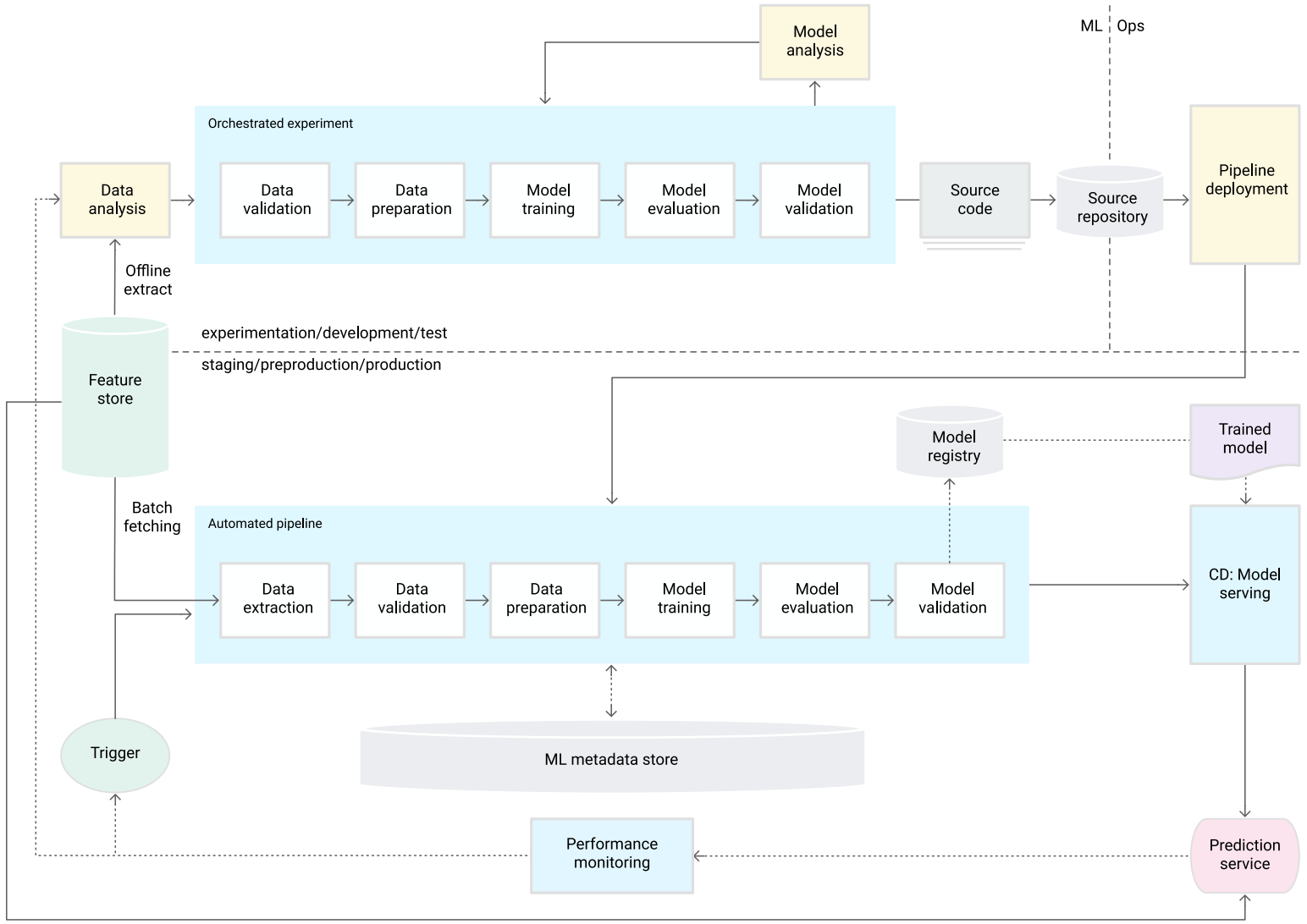
Что необходимо знать для воспроизводимого пайплайна

Сохраняем:

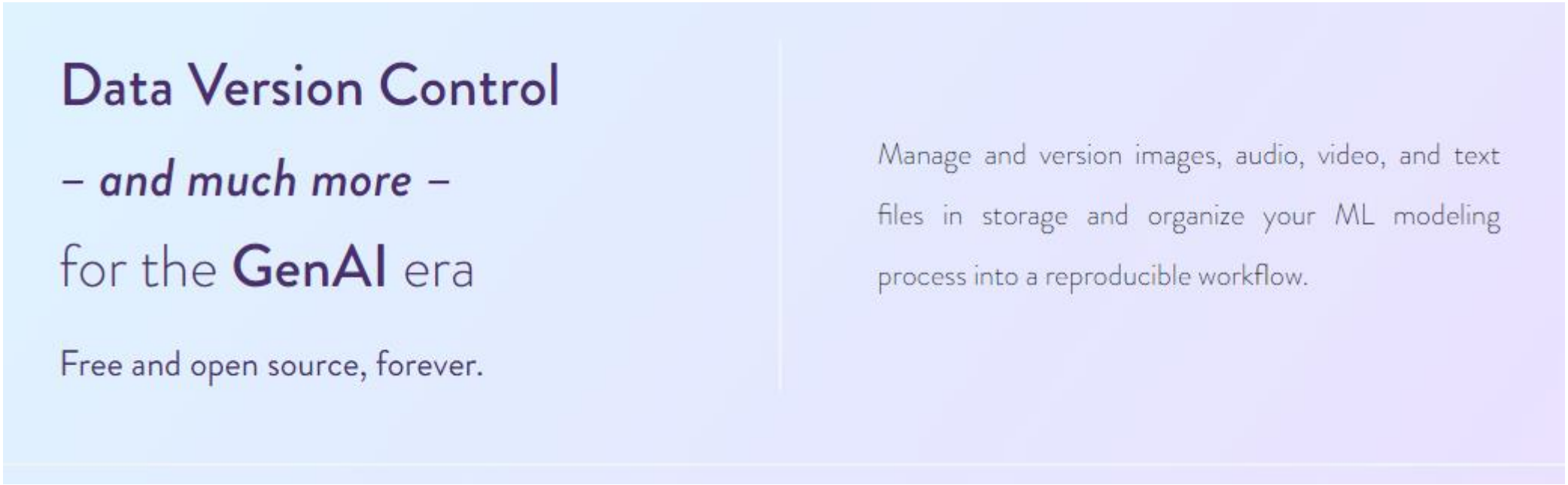
1. Файл модели
2. Ссылку на git repo и коммит (либо номер релиза)
3. Параметры модели
4. Метрики

Называем папку model_ver_1 и складываем на хранилище

MLOps Level 1

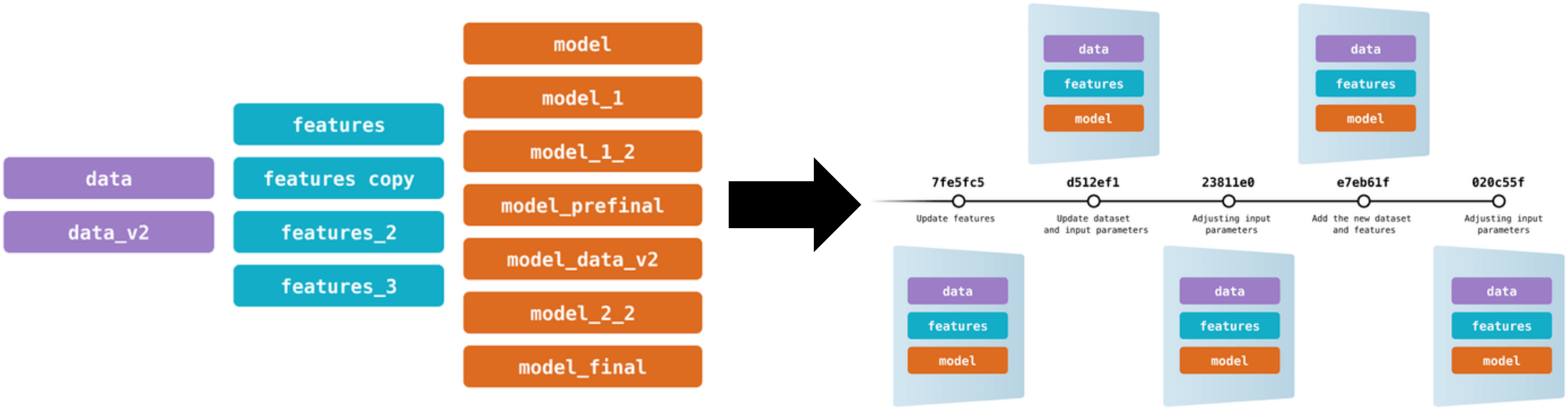


DVC

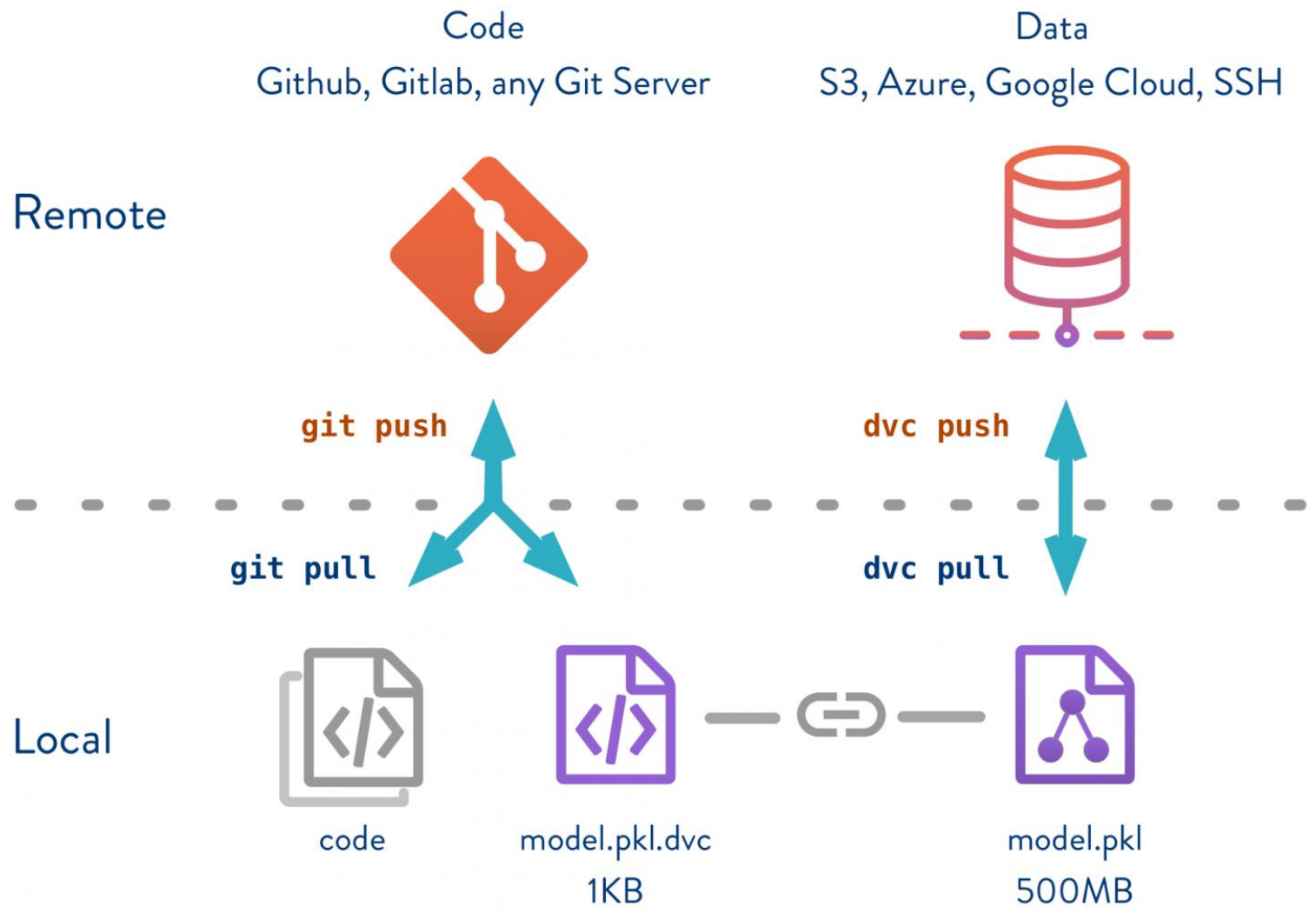


https://dvc.org

Версионирование данных



Версионирование данных



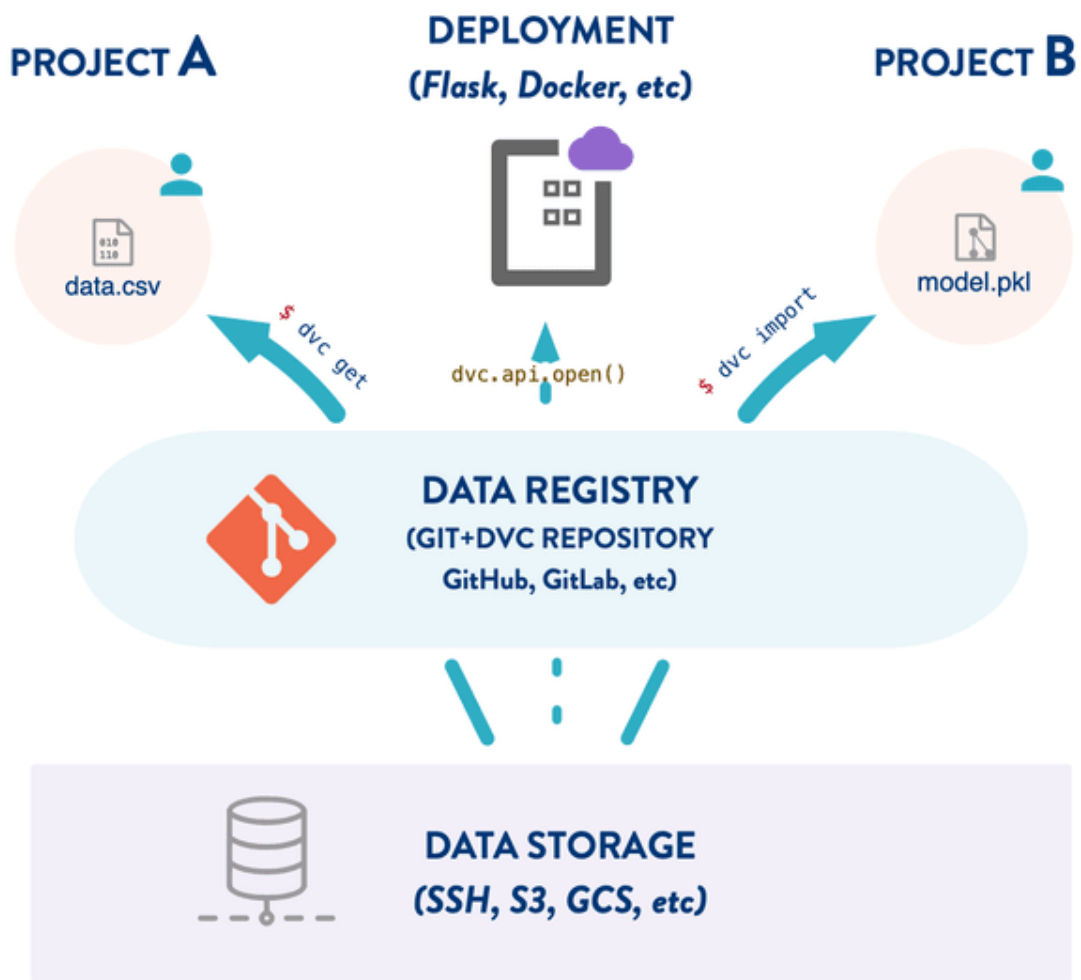
Версионирование данных

```
$ git init
$ dvc init

$ dvc add file.txt
$ git add file.txt.dvc

$ dvc push
$ git commit -m <comment>
$ git push
```

Версионирование данных



Выбор хранилища

```
$ dvc remote add -d temp /tmp/dvcstore
```

```
# .dvc/config
['remote "temp"']
    url = /tmp/dvcstore
[core]
    remote = myremote
```

! .dvc/config надо пушить в гит
!! всю папку .dvc пушить нельзя

Supported storage types

The following are the supported types of storage protocols and platforms.

Cloud providers

- Amazon S3 (AWS) and S3-compatible e.g. MinIO
- Microsoft Azure Blob Storage
- Google Cloud Storage (GCP)
- Google Drive
- Aliyun OSS

Self-hosted / On-premises

- SSH; Like `scp`
- HDFS & WebHDFS
- HTTP
- WebDAV

Пайплайны вычислений

```
$ dvc stage add -n train \
                -p train.seed,train.n_est,train.min_split \
                -d src/train.py -d data/features \
                -o model.pkl \
                python src/train.py data/features model.pkl
```

```
$ dvc dag
          +---------+
          | prepare |
          +---------+
               *
               *
               *
        +-----------+
        | featurize |
        +-----------+
               *
               *
               *
          +-------+
          | train |
          +-------+
```

Файлы: dvc.yaml

Содержит:

1. Информация об этапах пайплайна
2. Параметры запуска
3. Описание артефактов
4. Описание метрик

Поддерживает:

- foreach
- matrix

https://dvc.org/doc/user-guide/project-structure/dvcyaml-files

dvc.yaml

```yaml
params:
  - params.yaml
stages:
  make_dataset:
    cmd: 'python src/dataset.py params.yaml'
    deps:
    - src/dataset.py
    params:
    - data_params
    outs:
    - ${data_params.train_data_path}
    - ${data_params.test_data_path}
  train_model:
    cmd: 'python src/modeling/train.py params.yaml'
    deps:
    - src/modeling/train.py
    - ${data_params.train_data_path}
    - ${data_params.test_data_path}
    params:
    - train_params
    outs:
    - ${train_params.model_path}
    metrics:
    - ${train_params.metrics_path}

```

Домашнее задание №3

1. Создать локальное хранилище для DVC
2. Переписать полную тренировку модели через пайплайн DVC
3. Запушить необходимые файлы .dvc в гит

Не допускается никакого хардкода

Все, что можно сконфигурировать, должно лежать в конфигах (но без фанатизма)