ML Project MLOps Pipeline Structure

Full MLOps setup with DVC, MLflow, and CI/CD integration. For production ML systems requiring reproducibility and automation.

#ml #python #mlops #dvc #mlflow #cicd

PNG PDF Copy Prompt

Project Directory

```
ml-pipeline/
  src/
    ml_pipeline/
      __init__.py
      data/
        make_dataset.py
        validate.py  Great Expectati…
      features/
        build_features.py
      models/
        train.py
        predict.py
        evaluate.py
  pipelines/  DVC pipeline de…
    dvc.yaml  Pipeline DAG
    params.yaml  Hyperparameters
  data/  DVC-tracked
    raw/.gitkeep
    processed/.gitkeep
    .gitignore  /*.csv, /*.parq…
  models/  DVC-tracked art…
    .gitignore
  configs/
    train.yaml  Hydra config
    model/
    data/
  tests/
    test_data.py
    test_model.py
  pyproject.toml
  dvc.yaml  Root pipeline c…
  dvc.lock  Pipeline state
  mlflow.yaml  MLflow tracking…
  Makefile  Common commands
  .dvc/  DVC internals
  README.md
```

Why This Structure?

Built for reproducible ML. DVC versions data and models alongside code. MLflow tracks experiments and registers models. The `dvc.yaml` pipeline ensures `dvc repro` rebuilds only changed stages. Configs via Hydra allow easy hyperparameter sweeps.

Key Directories

`pipelines/` - `dvc.yaml` defines stage dependencies

`configs/` - Hydra configs for experiment variations

`data/` - DVC-tracked, actual files in remote storage

`models/` - Trained models tracked by DVC

Getting Started

1. `pip install -e '.[dev]'` and `dvc init`
2. `dvc remote add -d storage s3://bucket/path`
3. `dvc pull` (fetch data from remote)
4. `dvc repro` (run full pipeline)
5. `mlflow ui` (view experiments at localhost:5000)

When To Use This

- Production ML systems
- Teams needing reproducible experiments
- Projects with large datasets
- Automated retraining pipelines
- Model versioning and A/B testing

Trade-offs

`Significant setup` - DVC remote, MLflow server required

`Learning curve` - DVC pipelines, Hydra, MLflow concepts

`Overkill for small projects` - Use notebook-first instead

Best Practices

- Tag releases with `dvc metrics diff` outputs
- Use `params.yaml` for all hyperparameters
- Run `dvc repro` in CI before merge
- Store data checksums, not data, in git