MLOps Basics PDF

MLOps is the set of practices for taking machine learning models from experiments to reliable production systems and keeping them working. It covers versioning of code, data and models, automated training and deployment pipelines, serving, monitoring for drift and errors, and retraining. This page explains the life cycle and the tools you can use in a simple setup.

Key takeaways

- A model in a notebook is not a product; production needs repeatable pipelines.
- Version code, data, configuration and models so that any result can be reproduced.
- Deployment choices include batch jobs, real-time APIs and edge devices.
- Monitor data drift, model quality and system health after launch.
- Retrain on a schedule or on triggers, and test before replacing the live model.

What is MLOps?

MLOps, short for machine learning operations, is the discipline of combining machine learning, software engineering and operations practices to build, deploy, monitor and maintain models reliably in production.

Why do models need operations?

Many models that work in a notebook never reach users, or break soon after. Data changes, code is hard to rerun, nobody knows which dataset trained which model and no one is watching quality. MLOps applies DevOps ideas, such as automation, testing and monitoring, to machine learning, adding what is special about ML: data and models change behaviour, not only code.

What is the ML life cycle?

1. **Problem and success measure:** define the business goal and a metric. 2. **Data:** collect, validate and version it. 3. **Experiments:** train models, track settings and results. 4. **Evaluate:** test on held-out data and fairness and robustness checks. 5. **Package and deploy:** serve the model. 6. **Monitor:** track quality and health. 7. **Retrain:** update the model when needed. The cycle repeats.

What should be versioned?

Item	Why
Code	Reproduce training and serving logic

Item	Why
Data and features	Know exactly what trained a model
Configuration and parameters	Repeat experiments
Model files	Roll back to a previous model
Environment	Same libraries and versions

Use Git for code, a data versioning approach or dated snapshots for datasets, and a model registry to store models with their metadata and stage (testing, production).

What is experiment tracking?

Record each run's parameters, data version, metrics and output files, so you can compare runs and reproduce the best one. Tools such as MLflow provide tracking and a registry. Even a simple table in a spreadsheet is better than memory.

What is a training pipeline?

A pipeline automates steps: fetch data, validate it, create features, train, evaluate, compare with the current model and register the result. Running it from code, not by hand, makes it repeatable and testable. Use validation checks such as schema, missing values and range checks so that bad data stops the pipeline early.

How do I deploy a model?

Batch: run predictions on a schedule and store results, good for reports and nightly scores. **Real-time API:** a service returns predictions on request, good for interactive use. **Streaming:** process events as they arrive. **Edge:** run on a device. Package the model and code in a container, keep a clear input and output format, and test the service as you would any API.

```
Request:  POST /predict  {"tenure_months": 8, "plan": "basic", "tickets": 3}
Response: {"churn_probability": 0.41, "model_version": "2026-10-01"}
```

Include the model version in responses and logs. The numbers above are illustrative.

How do I release safely?

Use the same practices as software: automated tests, a staging environment, a gradual rollout such as a canary or shadow mode (the new model runs beside the old without affecting users) and a quick rollback. Compare the new model with the current one on real traffic before switching.

What should I monitor?

System health: latency, errors, memory and cost. **Data quality and drift:** are input distributions changing from training time? **Model quality:** when true outcomes are known (even later), track

accuracy or other metrics. **Business metrics:** the result the model was built to improve. Set alerts and agree who responds.

What are drift types?

Data drift is a change in the input distribution. **Concept drift** is a change in the relationship between inputs and the target, for example customer behaviour shifting after a price change. Both can reduce quality without any code change.

When should I retrain?

On a fixed schedule, when monitoring shows a quality drop, or when new data of enough volume arrives. Always evaluate the new model against the current one and keep the old model available for rollback.

What are good beginner practices?

Keep notebooks for exploration but move stable logic into scripts or packages, write tests for data and code, document the model's purpose and limits, secure data and credentials and keep the setup as simple as the problem requires.

Which tools are common?

Experiment tracking and model registries (for example MLflow), workflow tools (Airflow or similar), containers (Docker), orchestration (Kubernetes), cloud ML platforms from major providers, feature stores for shared features and monitoring tools for data and models. Learn the idea behind each tool class first, then try one tool in a small project. Choose the simplest setup that meets your needs.

What small project can I try?

Train a simple model, track runs with a tracking tool, save the best model to a registry, wrap it in a small web service in a container, write two tests for the service, log every prediction with the model version and build a script that compares input data this week with the training data. This gives you a complete, small MLOps story to explain in an interview.

Frequently asked questions

What is MLOps?

MLOps combines machine learning, software engineering and operations practices to build, deploy, monitor and maintain models reliably in production.

What is data drift?

Data drift is a change in the distribution of input data compared with training time, which can reduce model quality.

Why version data and models?

Versioning lets you reproduce results, know what trained a model and roll back to a previous version if needed.

What is shadow deployment?

The new model runs alongside the current one on real traffic without affecting users, so you can compare results before switching.

Do small teams need MLOps?

Yes, in a lighter form: version control, a repeatable training script, basic monitoring and a way to roll back.

Sources and further reading

- MLflow documentation
- scikit-learn documentation
- Docker documentation
- Google Cloud: MLOps guide

About this PDF

These PDFs are study material for learning and revision. They are not a certificate, a degree or a promise of a job, and technology changes, so check official documentation for the latest details. Spot an error? Write to info@uncodemy.com and the trainer team will review it.

More free resources and courses: uncodemy.com | info@uncodemy.com | +91 98183 66550