



# Оркестрация и CI/CD для ML

**Ирина Степановна Трубчик**

<https://t.me/+PsC-JDrwrvsxNmVi>

Лекция 5

# Цели занятия

1

Что такое оркестрация пайплайнов в ML

2

Современные системы: Apache Airflow, Prefect, Kubeflow Pipelines, DVC

3

Как связать ML пайплайн с CI/CD



- 1. какие задачи автоматизировали вручную?*
- 2. Какие проблемы возникали?*





## **Оркестрация — это координация и автоматизация выполнения задач в сложных системах**

**Андреас Кляйн, ведущий инженер по данным  
в Google** Андреас Кляйн — эксперт в области MLOps  
и оркестрации, автор статей и докладов  
на международных конференциях по машинному  
обучению и DevOps.



# Зачем нужна оркестрация



- Проблемы роста: скрипты, ручные шаги, повторяемые ошибки
- Решения: DAG (граф задач), автоматизация, расписания, зависимости, повторяемость
- Бонусы: воспроизводимость, мониторинг, fail-fast

# Airflow/Prefect: базовые понятия

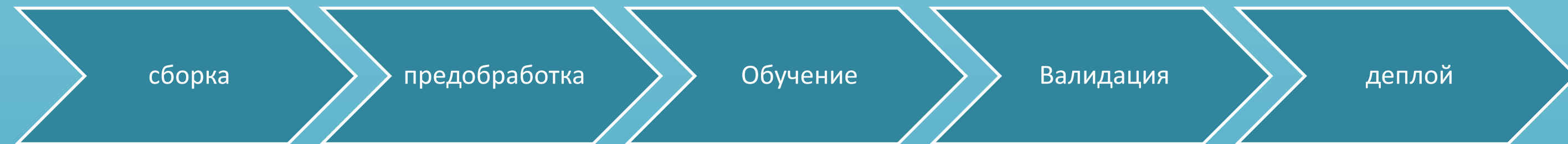
**1** DAG: Directed Acyclic Graph, задачи-узлы

**2** Операторы: Bash, Python, Docker, MLflow

**3** Сенсоры, SLA, ретрай, алерты



# Скетч-пример DAG

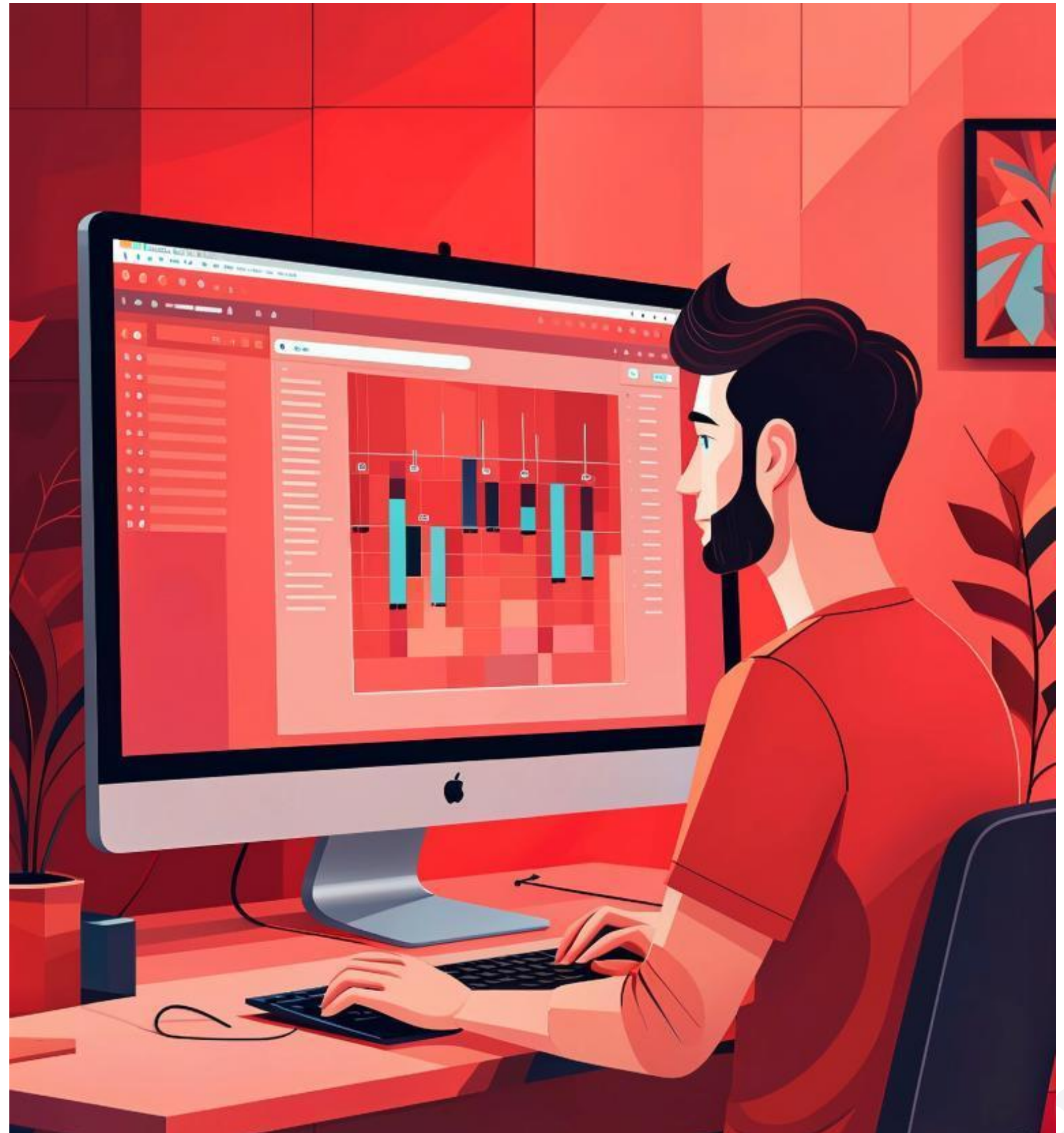


какие задачи могут быть зависимыми?



# Инструменты оркестрации

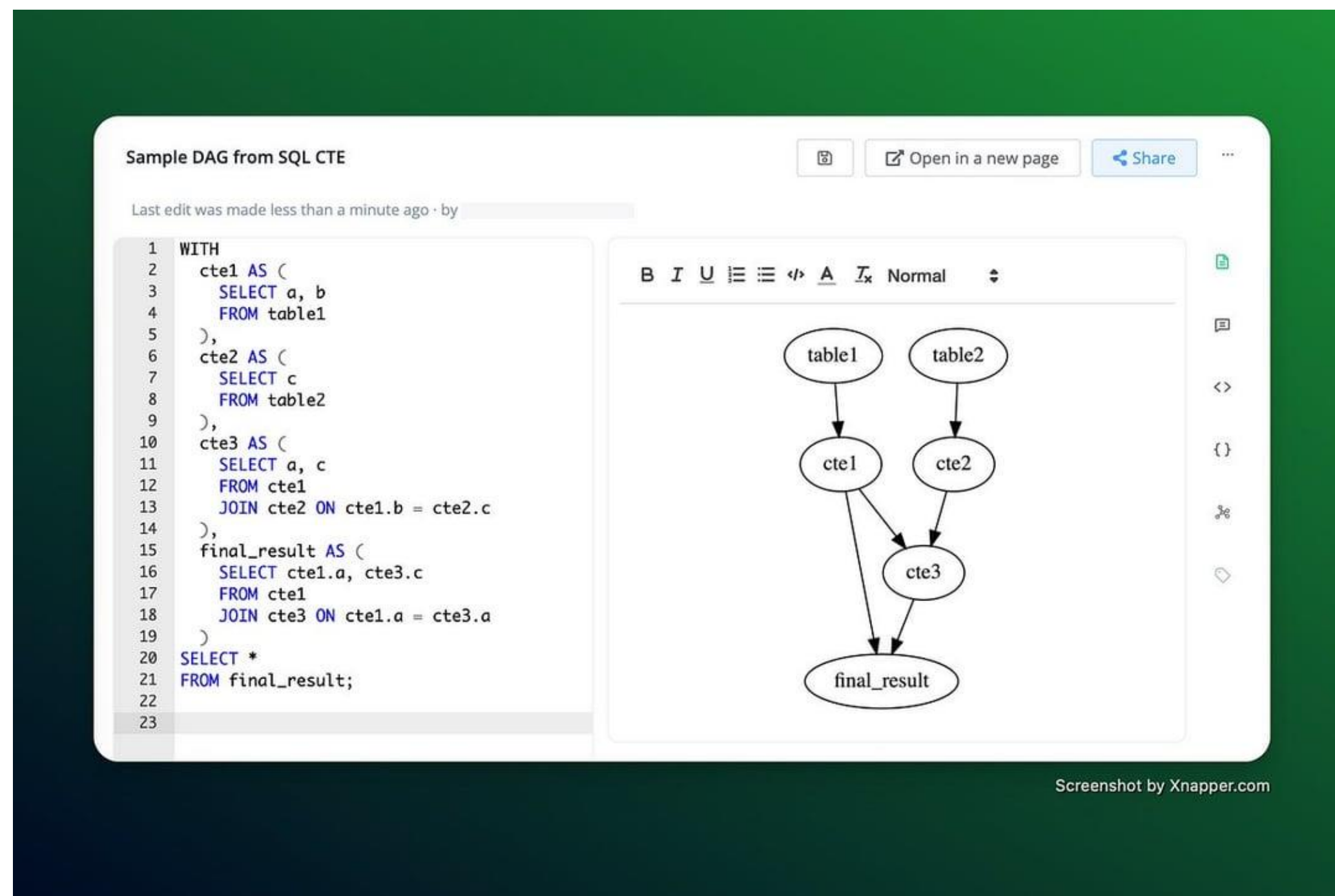
- Apache Airflow: инструмент для планирования и мониторинга рабочих процессов, поддерживает визуализацию и управление задачами.
- Kubeflow: платформа для оркестрации ML на Kubernetes, обеспечивает масштабируемость и управление ресурсами.
- MLFlow: инструмент для управления жизненным циклом ML-моделей, включая эксперименты, модели и развертывание.
- Argo Workflows: инструмент для оркестрации контейнерных рабочих процессов на Kubernetes, поддерживает сложные пайплайны и параллельное выполнение.



# Apache Airflow DAG

Apache Airflow является одним из самых популярных инструментов оркестрации с поддержкой DAG для MLOps. Основные особенности:

- Программное определение workflow на Python с операторами для различных задач
- Веб-интерфейс для мониторинга и управления пайплайнами
- Богатая экосистема операторов для интеграции с ML-инструментами
- Поддержка MLflow для отслеживания экспериментов и управления моделями





# Пример DAG для ML

# Airflow DAG-фрагмент

```
from airflow import DAG
```

```
from airflow.operators.python_operator import PythonOperator
```

```
from datetime import datetime
```

```
with DAG('flight_delay_ml_pipeline', start_date=datetime(2023, 1, 1)) as dag:
```

```
    preprocess = PythonOperator(
        task_id='preprocess_data',
        python_callable=preprocess_fn
    )
```

```
    train = PythonOperator(
        task_id='train_model',
        python_callable=train_fn
    )
```

```
    validate = PythonOperator(
        task_id='validate_model',
        python_callable=validate_fn
    )
```

```
    deploy = PythonOperator(
        task_id='deploy',
        python_callable=deploy_fn
    )
```

```
    preprocess >> train >> validate >> deploy
```

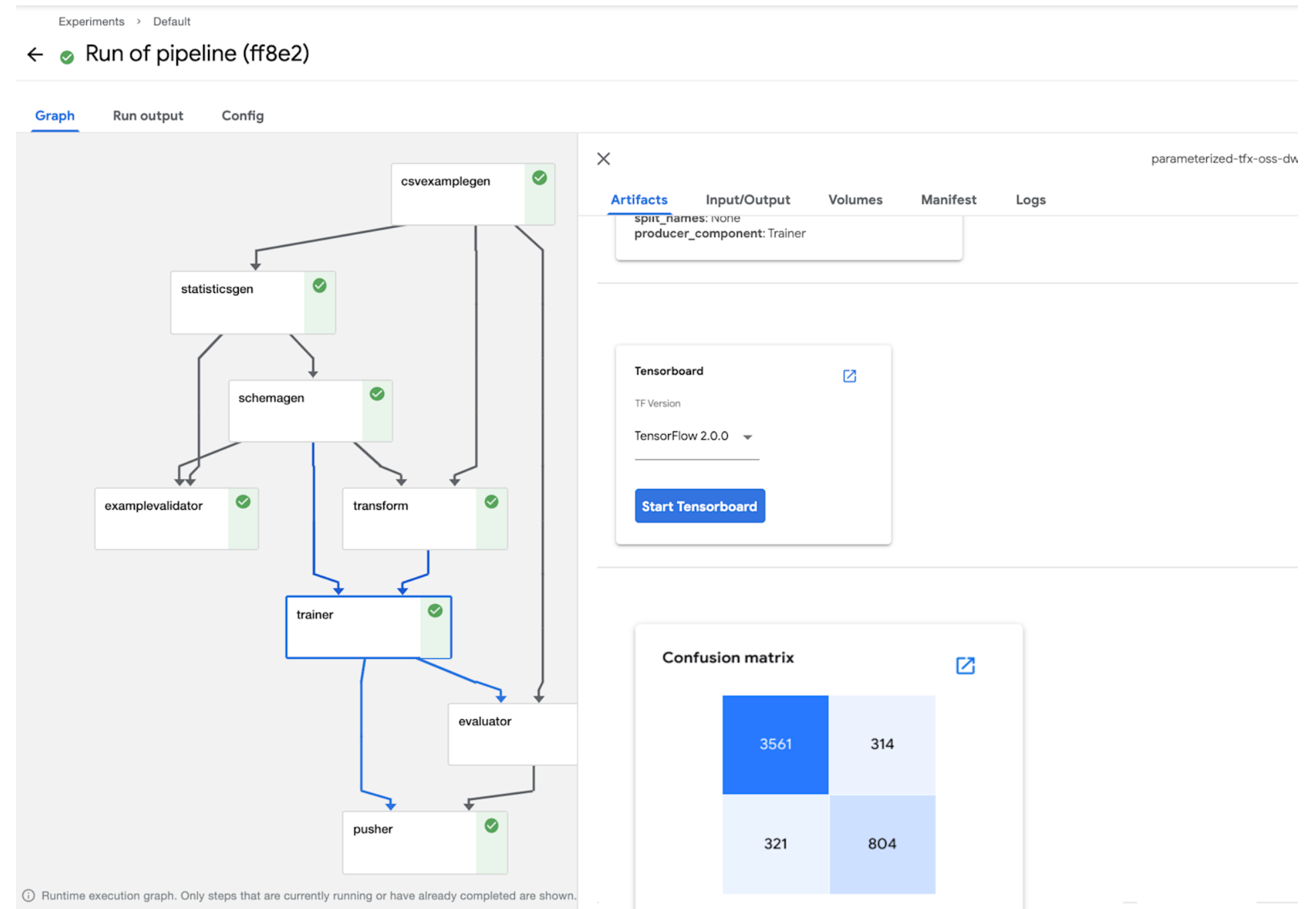
какие стадии требуют данных,  
а какие зависят от метрик



# Kubeflow Pipelines DAG

Kubeflow предоставляет облачно-нативную платформу для ML с фокусом на Kubernetes:

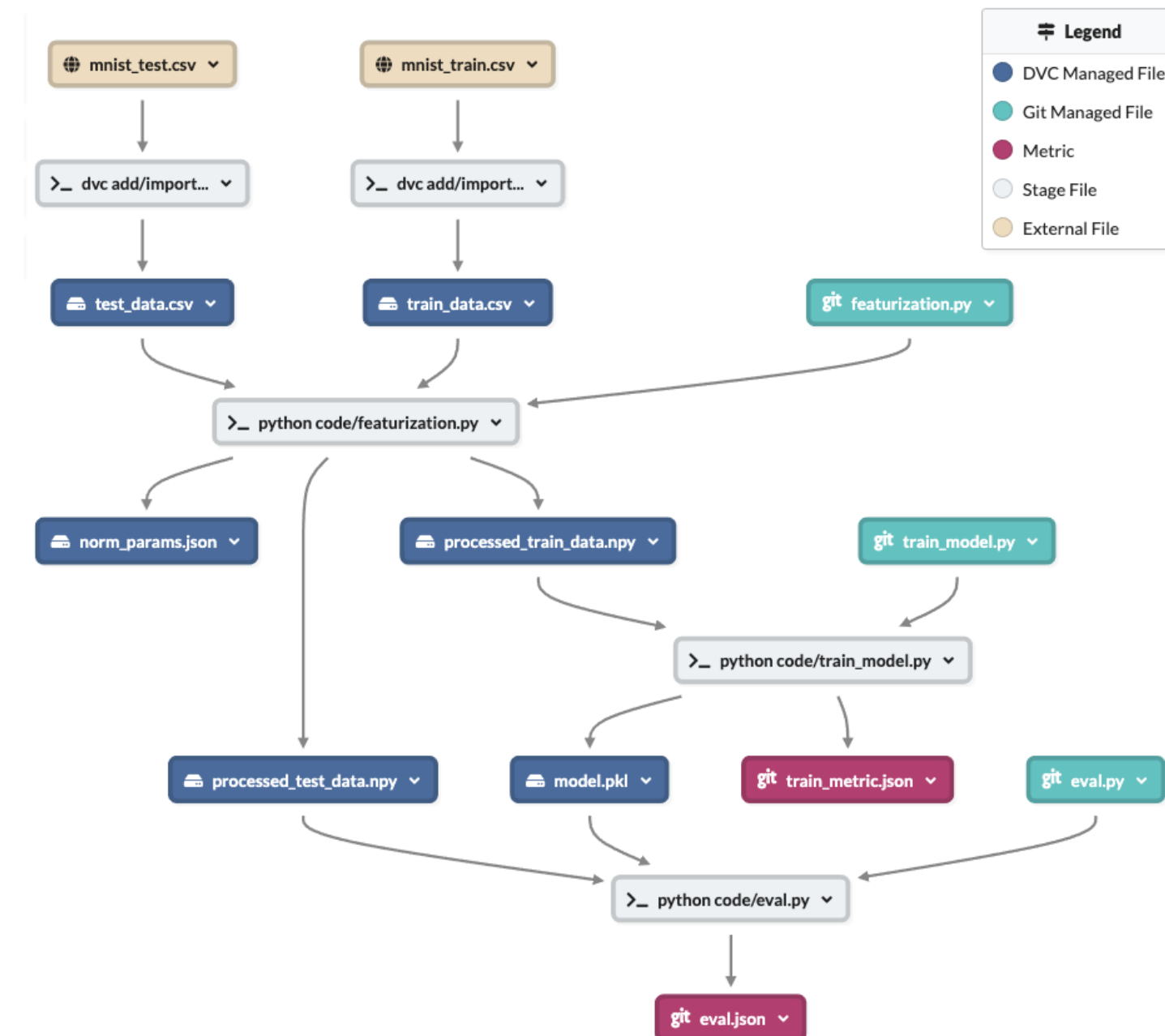
- Контейнеризированные компоненты для каждого этапа ML-пайплайна
- Автоматическое масштабирование и управление ресурсами
- Интеграция с Jupyter notebooks для интерактивной разработки
- Артефакты и метаданные для воспроизводимости экспериментов



# DVC Pipelines

Data Version Control (DVC) создает DAG для управления данными и ML-пайплайнами:

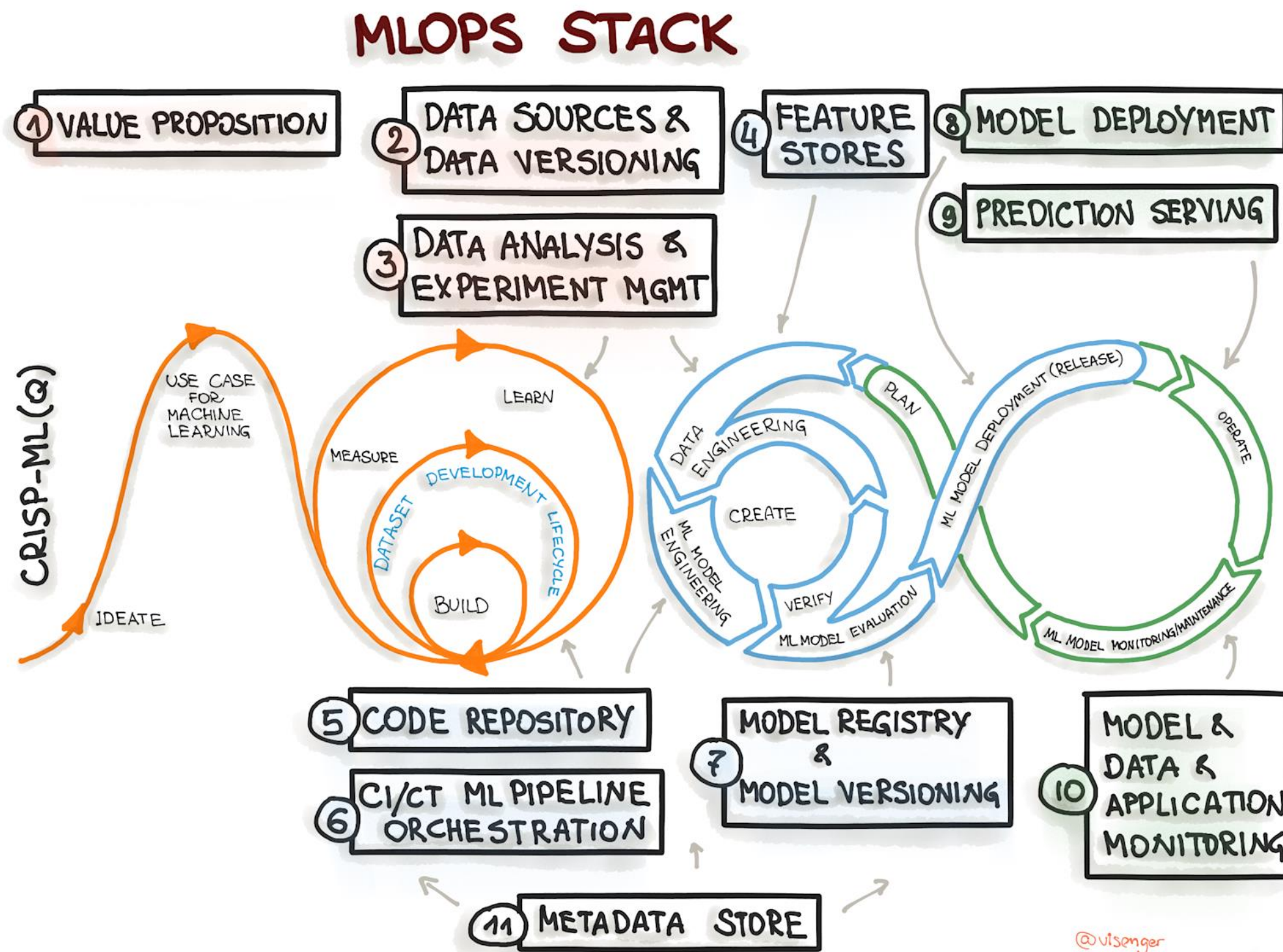
- Версионирование данных и кода совместно
- Воспроизводимость экспериментов через dvc repro
- Визуализация зависимостей с помощью dvc dag
- Интеграция с Git для отслеживания изменений



# MLflow с Airflow

Интеграция MLflow и Airflow обеспечивает полный жизненный цикл ML:

- Отслеживание экспериментов через MLflow Tracking
- Оркестрация через Airflow DAG
- Управление моделями с MLflow Model Registry
- Автоматизированное развертывание



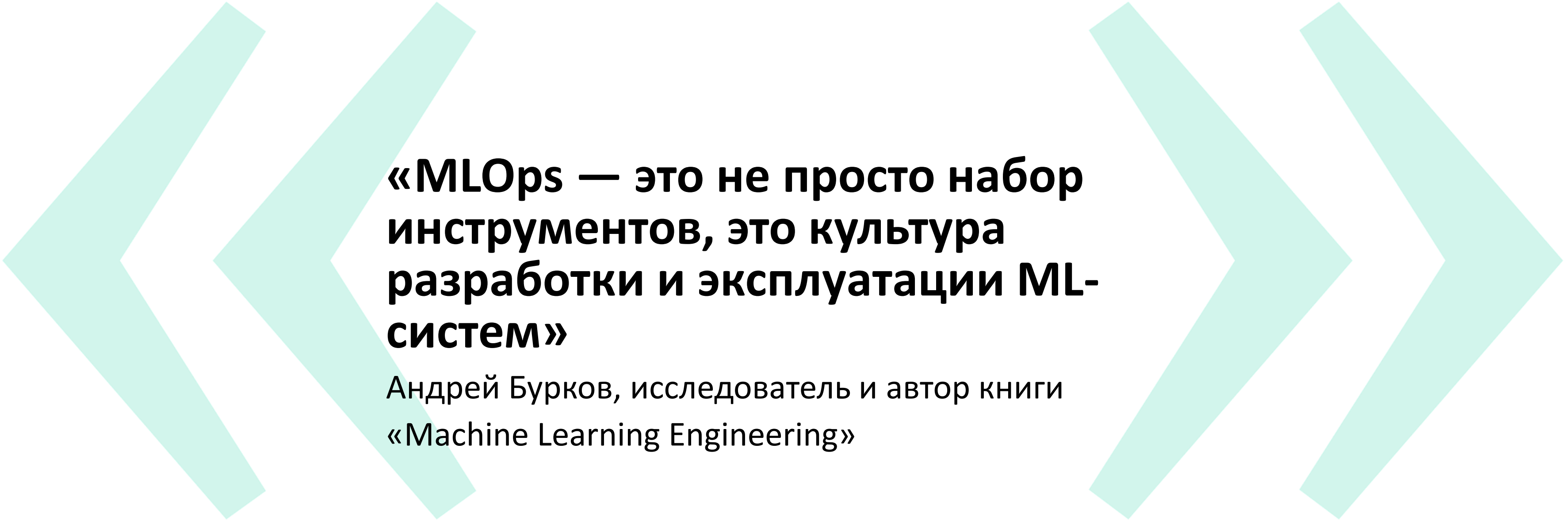




# Практические аспекты CI/CD

- Автоматизация тестирования моделей: интеграция юнит-тестов и интеграционных тестов для проверки качества моделей.
- Автоматическое развертывание: использование инструментов CI/CD для автоматического развертывания моделей в production.
- Мониторинг и логирование: настройка мониторинга производительности моделей и логирования событий для быстрого выявления и устранения проблем.





**«MLOps — это не просто набор инструментов, это культура разработки и эксплуатации ML-систем»**

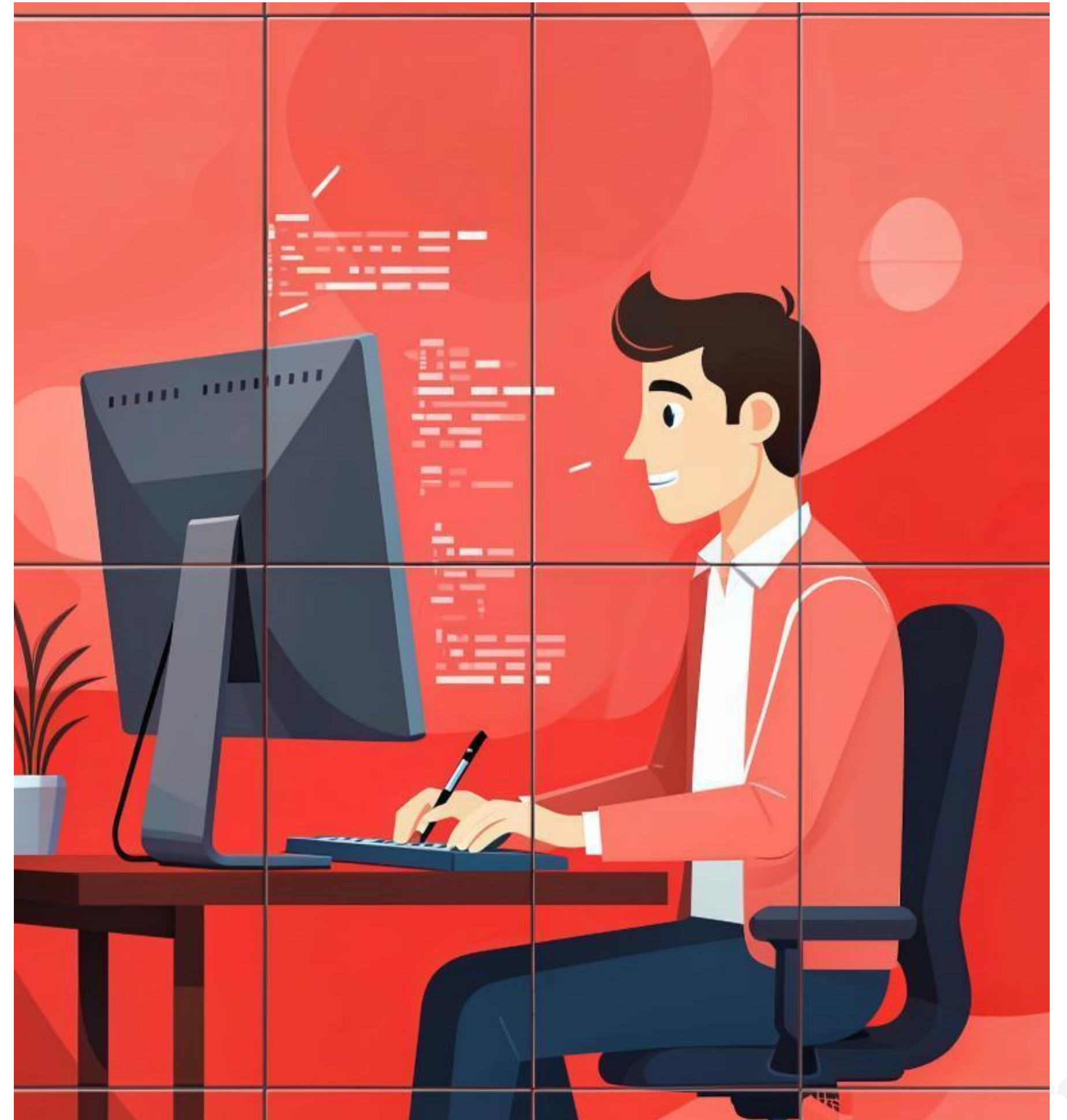
Андрей Бурков, исследователь и автор книги  
«Machine Learning Engineering»





# Лучшие практики

- Модульность и повторное использование: создание универсальных компонентов для повторного использования в разных проектах.
- Тестирование на каждом этапе: автоматическое тестирование моделей и пайплайнов на каждом этапе разработки.
- Документация и стандартизация: создание подробной документации и стандартов для обеспечения прозрачности и воспроизводимости.



# CI/CD в ML: принципы

- 1 Автоматизация сборки, тестов, деплоя
- 2 GitHub Actions, GitLab CI/CD, Jenkins, Argo
- 3 Триггеры: PR, push, обновление модели, дрейф данных
- 4 Варианты сборочных пайплайнов:  
build image → run tests → push to registry → deploy



# Пример CI-конфига (GitHub Actions)

```
name: ML Pipeline CI
```

```
on:
```

```
  push:
```

```
    paths:
```

- src/\*\*
- models/\*\*
- dvc.yaml

```
jobs:
```

```
  build-test-deploy:
```

```
    runs-on: ubuntu-latest
```

```
    steps:
```

- uses: actions/checkout@v3
- name: Install Python  
 uses: actions/setup-python@v4  
 with:  
 python-version: '3.10'
- name: Install dependencies  
 run: pip install -r requirements.txt
- name: DVC repro  
 run: dvc repro
- name: Run tests  
 run: pytest tests/
- name: Build Docker image  
 run: docker build -t ml-app .
- name: Push image  
 run: docker push \${ secrets.REGISTRY }/ml-app:\${ github.sha }
- name: Deploy



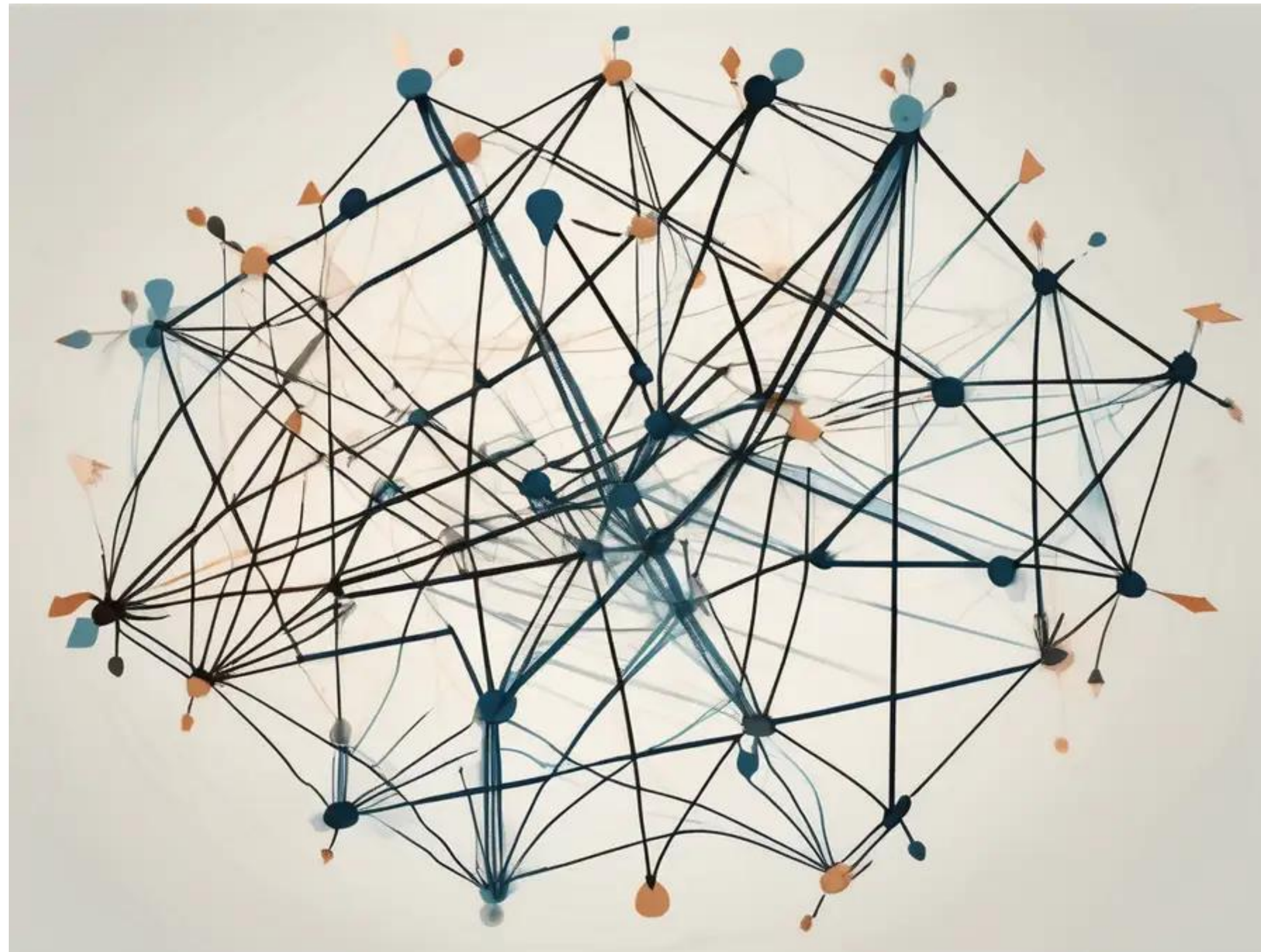


# Ключевые преимущества DAG в MLOps

Все представленные инструменты используют концепцию DAG для:

- Управления зависимостями между задачами обработки данных, обучения и развертывания
- Параллельного выполнения независимых задач для повышения эффективности
- Отслеживания провенанса данных от источника до результата
- Обеспечения воспроизводимости ML-экспериментов

**Абстрактное 3D-представление орграфа с вершинами и связями в сложно переплетённой архитектуре**



**Выбор конкретного инструмента зависит от ваших требований:**

**Airflow для универсальной оркестрации,**

**Kubeflow для Kubernetes-окружений,**

**DVC для версионирования данных, или**

**Prefect для современной разработки на Python.**







## Примеры проектов

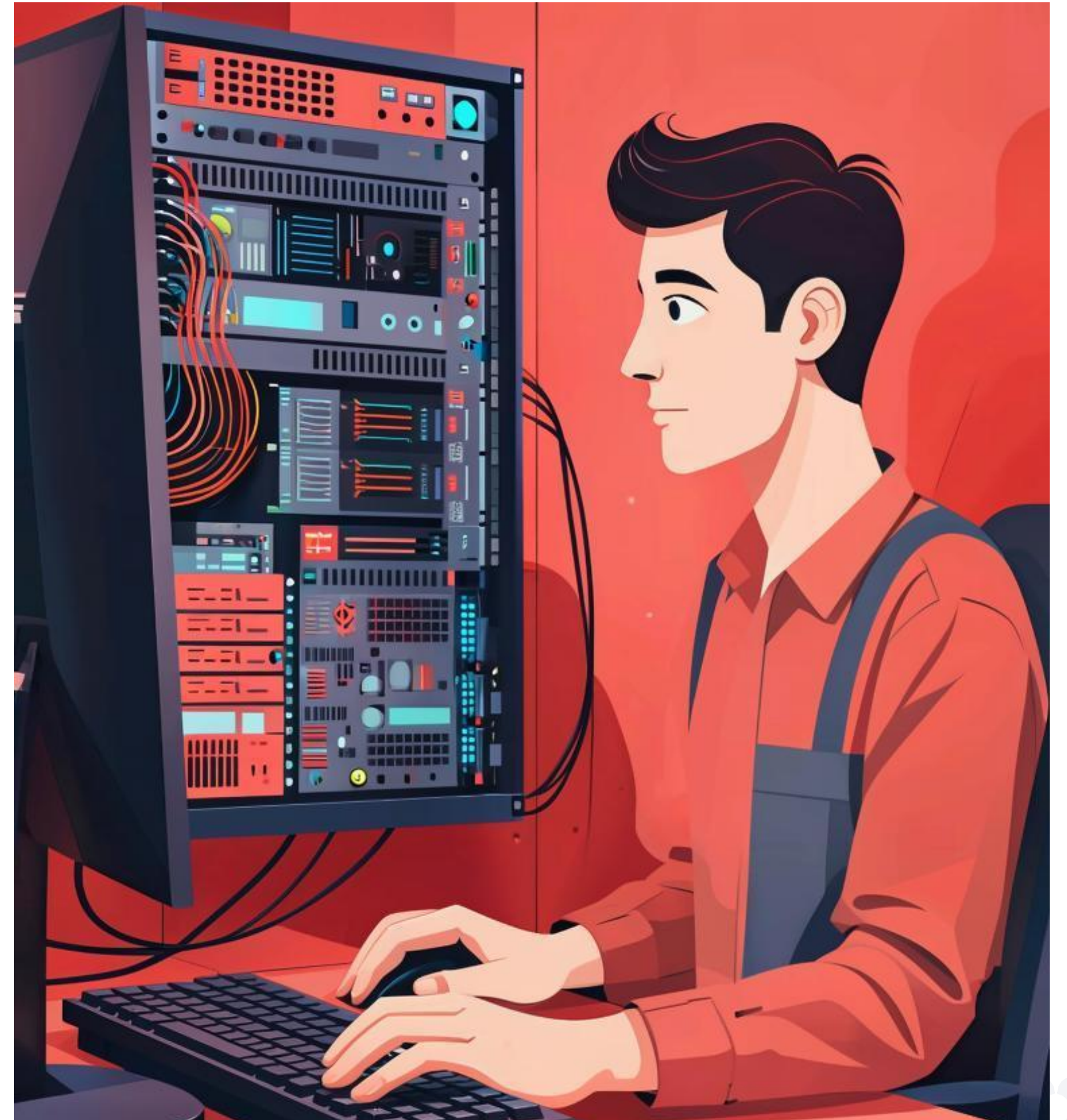
- Пример 1: Автоматизация ML-пайплайна с использованием Apache Airflow и Docker. Автоматизация развертывания моделей и мониторинга производительности.
- Пример 2: Интеграция CI/CD в ML-проекты с помощью GitLab CI и Kubernetes. Автоматическое тестирование и развертывание моделей в продакшн.





# Тенденции и будущее

- Автоматизация и автономия: внедрение инструментов для автоматического управления ML-пайплайнами и минимизация человеческого вмешательства.
- Интеграция с DevOps: объединение практик DevOps и MLOps для повышения эффективности и качества разработки ML-систем.
- Рост популярности MLOps: увеличение числа компаний, внедряющих MLOps для управления процессами машинного обучения, что приводит к росту спроса на специалистов в этой области.



1. Составьте DAG для процесса ML задачи (предобработка → обучение → инференс → мониторинг)
2. Настройте CI/CD workflow для тестового ml-репозитория
3. Реализуйте минимум один автоматический retrain



# Контрольные вопросы

1. В чем отличие ML CI/CD от классического DevOps?
2. Как связаны оркестрация и reproducibility?
3. Как бороться с дрейфом модели в продакшене?
4. Лучшие ресурсы: [airflow.apache.org](https://airflow.apache.org), [prefect.io](https://prefect.io), [docs.dvc.org](https://docs.dvc.org), [github.com/actions](https://github.com/actions)



# Материалы и ссылки

## 1. Airflow (оркестрация пайплайнов):

Официальная документация: <https://airflow.apache.org/docs/>

Разделы: Getting Started, DAGs, Operators, Sensors, Monitoring

## 2. Prefect (lightweight оркестрация):

Документация: <https://docs.prefect.io/>

## 3. Kubeflow Pipelines:

Гайд по ML pipeline в Kubernetes:

<https://www.kubeflow.org/docs/components/pipelines/>





# Материалы и ссылки

## 4. DVC (Data Version Control для ML-проекта):

Getting started + pipeline examples: <https://dvc.org/doc/start>

Интеграция с CI/CD: <https://dvc.org/doc/user-guide/ci-cd>

## 5. DevOps и ML-бест практики:

MLOps Specialization (Coursera):

<https://www.coursera.org/specializations/production-machine-learning>

Google MLOps whitepaper:

<https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning>

## 6. Визуализации пайплайнов:

DVC pipeline DAG: <https://docs.dvc.org/img/pipeline-dag.png>

Airflow DAG пример:

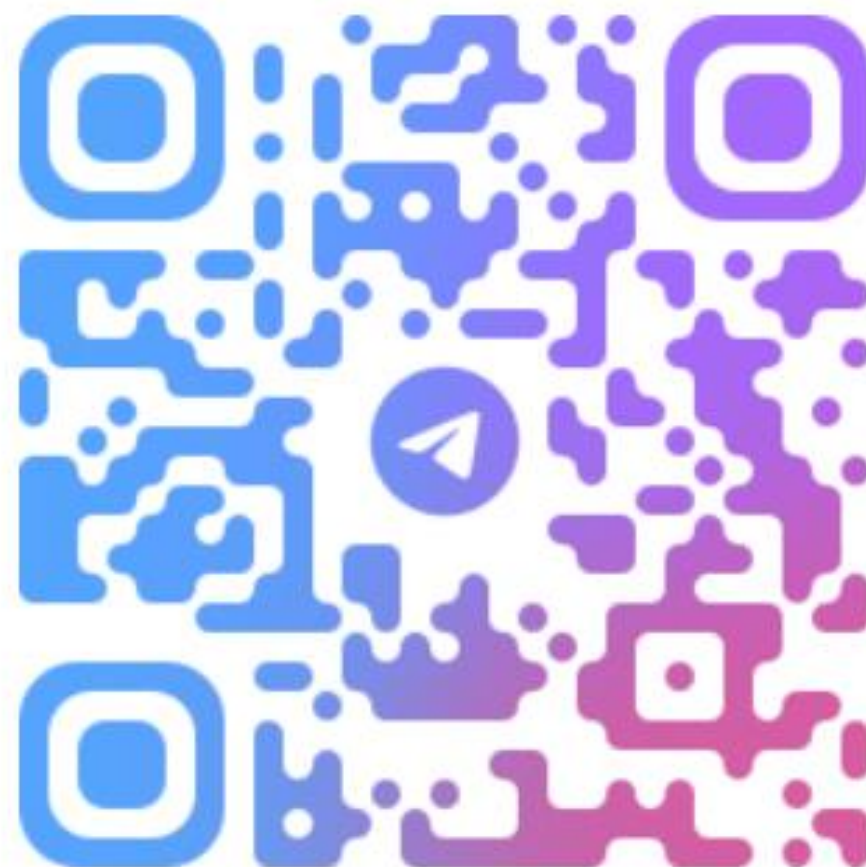
[https://airflow.apache.org/docs/apache-airflow/latest/\\_images/dag\\_structure.png](https://airflow.apache.org/docs/apache-airflow/latest/_images/dag_structure.png)



# Вопросы



Телеграм <https://t.me/+PsC-JDrwrvsxNmVi>



**СКИФ**

**(<https://do.skif.donstu.ru/course/view.php?id=7508>)**

