Кейс 1: Развертывание модели машинного обучения с использованием Docker

Задание:

1. Создать простую модель машинного обучения (например, классификатор на основе scikit-learn).
2. Написать скрипт для развертывания модели в Docker контейнере.
3. Запустить контейнер и протестировать API для предсказаний.

Ожидаемый результат:

- Модель создана и обучена.
- Скрипт для развертывания модели в Docker контейнере написан.
- Контейнер запущен и API для предсказаний работает.

Кейс 2: Автоматизация CI/CD для моделей машинного обучения с использованием Jenkins

Задание:

1. Настроить Jenkins для автоматической сборки и тестирования модели машинного обучения при каждом коммите в репозиторий.
2. Автоматически развертывать обновленную модель в тестовую среду.

Ожидаемый результат:

- Jenkins настроен для автоматической сборки и тестирования модели.
- Обновленная модель автоматически развертывается в тестовую среду.

Кейс 3: Мониторинг модели машинного обучения с использованием Prometheus и Grafana

Задание:

1. Настроить сбор метрик производительности модели (например, время ответа, количество запросов) с использованием Prometheus.
2. Настроить дашборд в Grafana для визуализации метрик.

Ожидаемый результат:

- Метрики производительности модели собираются с использованием Prometheus.
- Дашборд в Grafana отображает метрики.

Кейс 4: Управление версиями моделей с использованием MLflow

Задание:

1. Установить и настроить MLflow для отслеживания экспериментов с моделями машинного обучения.
2. Записывать параметры, метрики и артефакты модели в MLflow при каждом запуске эксперимента.

Ожидаемый результат:

- MLflow установлен и настроен.
- Параметры, метрики и артефакты моделей записываются в MLflow.

Кейс 5: Автоматическое обучение и развертывание моделей с использованием Kubeflow

Задание:

1. Установить Kubeflow на кластере Kubernetes.
2. Настроить пайплайн для автоматического обучения и развертывания модели.

Ожидаемый результат:

- Kubeflow установлен и настроен.
- Пайплайн для автоматического обучения и развертывания модели работает корректно.

Кейс 6: Обеспечение репродуктивности экспериментов

Задание:

1. Написать скрипт, который фиксирует случайные сиды для всех библиотек (numpy, tensorflow, pytorch и т.д.).
2. Обеспечить сохранение всех зависимостей и конфигураций для повторного запуска эксперимента.

Ожидаемый результат:

- Скрипт для фиксации случайных сидов написан.
- Все зависимости и конфигурации сохранены для обеспечения репродуктивности.

Кейс 7: Управление данными для обучения модели

Задание:

1. Настроить и использовать инструмент для версионирования данных (например, DVC).
2. Создать и управлять несколькими версиями датасетов для экспериментов.

Ожидаемый результат:

- DVC настроен и используется для версионирования данных.
- Созданы и управляются несколько версий датасетов.

Кейс 8: Тестирование моделей машинного обучения

Задание:

1. Написать юнит-тесты для проверки корректности работы функций предобработки данных и модели.
2. Настроить интеграцию с CI/CD пайплайном для автоматического запуска тестов.

Ожидаемый результат:

- Юнит-тесты для функций предобработки данных и модели написаны.
- Тесты автоматически запускаются при каждом коммите.

Кейс 9: Обеспечение масштабируемости модели

Задание:

1. Настроить модель для работы в распределенной среде с использованием Spark или Dask.
2. Провести тестирование производительности модели на больших данных.

Ожидаемый результат:

- Модель настроена для работы в распределенной среде.
- Тестирование производительности проведено и результаты зафиксированы.

Кейс 10: Управление жизненным циклом модели с использованием Seldon Core

Задание:

1. Установить Seldon Core на кластере Kubernetes.
2. Настроить автоматическое развертывание модели и управление ее версиями.

Ожидаемый результат:

- Seldon Core установлен и настроен.
- Модель автоматически развертывается и управляется в кластере Kubernetes.