MLOPS

Etienne KOA

4 mars 2024

Figure 1 – Codons pour toujours

Table des matières

1 Prerequis **10**

1.1 Aide Mémoie 10
1.2 AutoNLP 10
1.3 AutoEDA 10
1.4 AutoML 10
1.5 EDA Python AI 10
1.6 Feature Engineering 10
1.7 Feature Selection 10
1.8 Hyperparameter Tuning 10
1.9 Interpretable Machine Learining 10
1.10 Object Oriented Programming Project 10
1.11 Github Foundations 10
1.12 Mémoire M2 10

2 Défis et évolution du Machine Learning **11**

2.1 Introduction à l'apprentissage automatique . . . 11
2.2 Avantages de l'apprentissage automatique . . . 11
2.3 Principes fondamentaux du MLOps 11
2.4 Principes fondamentaux de DevOps et DataOps 11

3 Fondements du MLOps **12**

3.1 Problèmes résolus par MLOps 12
3.2 Composants MLOps 12
3.3 Boîte à outils MLOps 12
3.4 Étapes MLOps 12

4 Installation des outils et bibliothèques **13**

4.1 Comment installer des bibliothèques et préparer l'environnement 13
4.2 Principes de base de Jupyter Notebook 13

4.3 Installation de Docker et Ubuntu 13

5 Productivisation et structuration des projets ML 14
5.1 Cookiecutter pour gérer la structure du modèle Machine Learning 14
5.2 Bibliothèques et outils de gestion de projet du début à la fin 14
5.3 Poetry pour la gestion des dépendances 14
5.4 Makefile pour l'exécution automatisée des tâches 14
5.5 Hydra pour gérer les fichiers de configuration YAML 14
5.6 Hydra appliqué à un projet de Machine Learning 14
5.7 Vérifiez et corrigez automatiquement le code avant la validation dans Git 14
5.8 Révision du code avec Black et Flake8 dans le pré-commit 14
5.9 Révision du code avec Isort et Iterrogate dans l'intégration Pre-commit et Git 14
5.10 Générer automatiquement de la documentation pour le projet ML 14

6 MLOps Phase 1 : Conception de solutions 15
6.1 Conception et mise en œuvre de Volere 15

7 MLOps Phase 2 : Automatisation du cycle du modèle ML 16
7.1 Principes de base d'AutoML 16
7.2 Construire une maquette du début à la fin avec Pycaret 16
7.3 EDA et prétraitement avancé avec Pycaret . . . 16
7.4 Développement de modèles avancés (XGBoost, CatBoost, LightGBM) avec Pycaret) 16

7.5 Déploiement en production avec Pycaret 16

8 MLOps phase 2 : Versionnement et enregistrement du modèle avec MLFlow 17

8.1 Enregistrement et versioning des modèles avec MFlow 17
8.2 Enregistrement d'un modèle Scikit-Learn avec MLFlow 17
8.3 Enregistrement du modèle Pycaret auprès de MLFlow 17

9 Versionnement de l'ensemble de données avec DVC 18

9.1 Introduction au DVC 18
9.2 Commandes et processus DVC 18
9.3 Laboratoire pratique avec DVC 18
9.4 Pipelines DVC 18

10 Dépôt de code avec DagsHub, DVC, Git et MLFlow 19

10.1 Introduction à DagsHub pour le référentiel de code 19
10.2 EDA et prétraitement des données 19
10.3 Formation et évaluation du prototype du modèle ML 19
10.4 Création de compte DagsHub 19
10.5 Création de l'environnement Python et de l'ensemble de données 19
10.6 Hydra appliqué à un projet de Machine Learning 19
10.7 Déploiement du modèle dans DagsHub 19
10.8 Formation et versionnage du modèle ML 19
10.9 Utiliser DVC pour versionner les données et les modèles 19

10.10 Envoi de code, de données et de modèles à DagsHub 19
10.11 Expérimentation et enregistrement des expériences dans DagsHub 19
10.12 Utiliser DagsHub pour analyser et comparer des expériences et des modèles 19

11 Enregistrement et versioning automatisés avec Pycaret et DagsHub 20
11.1 Intégration Pycaret et Dagshub 20
11.2 Laboratoire pratique d'enregistrement d'un modèle et d'un ensemble de données avec Pycaret et DagsHub 20
11.3 Exercice pratique. Développement d'un modèle avec Pycaret et enregistrement dans MLFlow . 20
11.4 Solution. Développement d'un modèle avec Pycaret et inscription dans MLFlow 20
11.5 Exercice pratique. Générer un référentiel avec DagsHub 20
11.6 Solution. Générer un référentiel avec DagsHub . 20
11.7 Exercice pratique. Versionnement des données avec DVC 20
11.8 Solution. Versionnement des données avec DVC 20
11.9 Exercice pratique. Enregistrement du modèle sur un MLFlow partagé 20
11.10 Solution. Enregistrement du modèle sur un serveur MLFlow partagé 20

12 Interprétabilité du modèle 21
12.1 Bases de l'interprétabilité avec SHAP 21
12.2 Interprétation des modèles Scikit Learn avec SHAP 21

12.3 Interprétation de modèles avec SHAP dans Pycaret 21

13 Mise en production des modèles 22
13.1 Déploiement de modèles en production 22

14 MLOps phase 3 : Modèle servi via des API 23
14.1 Fondamentaux des API et FastAPI 23
14.2 Fonctions, méthodes et paramètres dans FastAPI 23
14.3 Méthode POST, Swagger et Pydantic dans FastAPI 23
14.4 Développement d'API pour le modèle Scikit-learn avec FastAPI 23
14.5 Développement d'API automatisé avec Pycaret 23

15 MLOps phase 3 : Service de modèles avec des applications Web 24
15.1 Servir le modèle via une application Web 24
15.2 Commandes Gradio de base 24
15.3 Développement d'une application web Gradio pour le Machine Learning 24
15.4 Développement d'applications Web automatisées avec Pycaret 24
15.5 Développement d'applications Web avec Streamlit 24
15.6 Laboratoire Développement d'applications Web avec Streamlit et Altair 24
15.7 Laboratory Streamlit et Pycaret pour développer un service web ML 24

16 Flask pour le développement d'applications 25
16.1 Fondamentaux de Flask 25
16.2 Construire un projet du début à la fin avec Flask 25

16.3 Développement back-end avec Flask et développement front-end avec HTML et CSS 25

17 Docker et conteneurs en Machine Learning 26
17.1 Fondamentaux de Flask 26
17.2 Construire un projet du début à la fin avec Flask 26
17.3 Développement back-end avec Flask et développement front-end avec HTML et CSS 26

18 BentoML pour le développement automatisé de services ML 27
18.1 Introduction à BentoML pour générer des services ML 27
18.2 Générer un service ML avec BentoML 27
18.3 Mettre le service en production avec BentoML et Docker 27
18.4 Introduction à BentoML pour générer des services ML 27
18.5 GPU, prétraitement, validation des données et modèles multiples dans BentoML 27
18.6 Différents outils pour développer des services ML 27
18.7 Exercice : Utiliser BentoML pour développer un service ML 27
18.8 Solution de l'exercice : Utiliser BentoML pour développer un service ML 27

19 Déployer sur Azure Cloud avec Azure Container et Azure SDK 28
19.1 Introduction à l'apprentissage automatique dans le cloud 28
19.2 Mettre l'application ML en production dans Azure Container avec Docker 28

19.3 SDK et Azure Blob Storage pour le déploiement de modèles sur Azure 28
19.4 Formation de modèles et déploiement en production dans Azure Blob Storage 28
19.5 Téléchargez le modèle Azure Blob Storage et obtenez des prédictions 28

20 Déploiement des services ML sur Heroku 29
20.1 Fondamentaux d'Heroku 29
20.2 Laboratoire pratique : déploiement d'un service ML sur Heroku 29

21 Section 21 : Intégration et livraison continues (CI/CD) avec Github Action et CML 30
21.1 Introduction aux actions GitHub 30
21.2 Flux de travail de base des actions GitHub . . . 30
21.3 Atelier pratique sur les actions GitHub 30
21.4 CI avec apprentissage automatique continu (CML) 30
21.5 Atelier pratique : Application des actions GitHub et du CML aux MLOps 30
21.6 Laboratoire pratique : suivi des performances avec les actions GitHub et CML. 30

22 Surveillance des modèles et des services avec Evidently AI 30
22.1 Introduction à la surveillance des modèles et services ML 30
22.2 Dérive des données, dérive des concepts et performances du modèle 30
22.3 es fondamentaux de Evidently AI 30
22.4 Dérive et qualité des données, dérive de la cible et qualité du modèle 30

22.5 Laboratoire pratique : Surveillance d'un modèle avec Evidently AI 30
22.6 Laboratoire pratique : Suivi du modèle en production 30
22.7 Exercice : Utiliser BentoML pour développer un service ML 30
22.8 Laboratoire pratique : Identification des dérives de données en production 30

23 Projet MLOps de bout en bout 31
23.1 Projet MLOps de bout en boutProjet MLOps de bout en bout 31
23.2 Développement du modèle ML 31
23.3 Validation de la qualité du code, du modèle et du prétraitement 31
23.4 Versionnement de projet avec MLFlow et DVC 31
23.5 Dépôt partagé avec DagsHub et MLFlow 31
23.6 Développement d'API avec BentoML 31
23.7 Développement d'applications avec Streamlit . 31
23.8 CI-CD : workflow de validation des données avec GitHub Actions 31
23.9 CI/CD : validation des fonctionnalités de l'application avec les actions GitHub 31
23.10CI/CD : déploiement automatisé d'applications avec GitHub Actions et Heroku 31

1 Prerequis

1.1 Aide Mémoie

1.2 AutoNLP

1.3 AutoEDA

1.4 AutoML

1.5 EDA Python AI

1.6 Feature Engineering

1.7 Feature Selection

1.8 Hyperparameter Tuning

1.9 Interpretable Machine Learining

1.10 Object Oriented Programming Project

1.11 Github Foundations

1.12 Mémoire M2

2 Défis et évolution du Machine Learning

2.1 Introduction à l'apprentissage automatique

2.2 Avantages de l'apprentissage automatique

2.3 Principes fondamentaux du MLOps

2.4 Principes fondamentaux de DevOps et DataOps

3 Fondements du MLOps

3.1 Problèmes résolus par MLOps

3.2 Composants MLOps

3.3 Boîte à outils MLOps

3.4 Étapes MLOps

4 Installation des outils et bibliothèques

4.1 Comment installer des bibliothèques et préparer l'environnement

4.2 Principes de base de Jupyter Notebook

4.3 Installation de Docker et Ubuntu

5 Productivisation et structuration des projets ML

5.1 Cookiecutter pour gérer la structure du modèle Machine Learning

5.2 Bibliothèques et outils de gestion de projet du début à la fin

5.3 Poetry pour la gestion des dépendances

5.4 Makefile pour l'exécution automatisée des tâches

5.5 Hydra pour gérer les fichiers de configuration YAML

5.6 Hydra appliqué à un projet de Machine Learning

5.7 Vérifiez et corrigez automatiquement le code avant la validation dans Git

5.8 Révision du code avec Black et Flake8 dans le pré-commit

5.9 Révision du code avec Isort et Iterrogate dans l'intégration Pre-commit et Git

5.10 Générer automatiquement de la documentation pour le projet ML

6 MLOps Phase 1 : Conception de solutions

6.1 Conception et mise en œuvre de Volere

7 MLOps Phase 2 : Automatisation du cycle du modèle ML

7.1 Principes de base d'AutoML

7.2 Construire une maquette du début à la fin avec Pycaret

7.3 EDA et prétraitement avancé avec Pycaret

7.4 Développement de modèles avancés (XGBoost, CatBoost, LightGBM) avec Pycaret)

7.5 Déploiement en production avec Pycaret

8 MLOps phase 2 : Versionnement et enregistrement du modèle avec MLFlow

8.1 Enregistrement et versioning des modèles avec MFlow

8.2 Enregistrement d'un modèle Scikit-Learn avec MLFlow

8.3 Enregistrement du modèle Pycaret auprès de MLFlow

9 Versionnement de l'ensemble de données avec DVC

9.1 Introduction au DVC

9.2 Commandes et processus DVC

9.3 Laboratoire pratique avec DVC

9.4 Pipelines DVC

10 Dépôt de code avec DagsHub, DVC, Git et MLFlow

10.1 Introduction à DagsHub pour le référentiel de code

10.2 EDA et prétraitement des données

10.3 Formation et évaluation du prototype du modèle ML

10.4 Création de compte DagsHub

10.5 Création de l'environnement Python et de l'ensemble de données

10.6 Hydra appliqué à un projet de Machine Learning

10.7 Déploiement du modèle dans DagsHub

10.8 Formation et versionnage du modèle ML

10.9 Utiliser DVC pour versionner les données et les modèles

10.10 Envoi de code, de données et de modèles à DagsHub

10.11 Expérimentation et enregistrement des expériences dans DagsHub

10.12 Utiliser DagsHub pour analyser et comparer des expériences et des modèles

11 Enregistrement et versioning automatisés avec Pycaret et DagsHub

11.1 Intégration Pycaret et Dagshub

11.2 Laboratoire pratique d'enregistrement d'un modèle et d'un ensemble de données avec Pycaret et DagsHub

11.3 Exercice pratique. Développement d'un modèle avec Pycaret et enregistrement dans MLFlow

11.4 Solution. Développement d'un modèle avec Pycaret et inscription dans MLFlow

11.5 Exercice pratique. Générer un référentiel avec DagsHub

11.6 Solution. Générer un référentiel avec DagsHub

11.7 Exercice pratique. Versionnement des données avec DVC

11.8 Solution. Versionnement des données avec DVC

11.9 Exercice pratique. Enregistrement du modèle sur un MLFlow partagé

11.10 Solution. Enregistrement du modèle sur un serveur MLFlow partagé

12 Interprétabilité du modèle

12.1 Bases de l'interprétabilité avec SHAP

12.2 Interprétation des modèles Scikit Learn avec SHAP

12.3 Interprétation de modèles avec SHAP dans Pycaret

13 Mise en production des modèles

13.1 Déploiement de modèles en production

14 MLOps phase 3 : Modèle servi via des API

14.1 Fondamentaux des API et FastAPI

14.2 Fonctions, méthodes et paramètres dans FastAPI

14.3 Méthode POST, Swagger et Pydantic dans FastAPI

14.4 Développement d'API pour le modèle Scikit-learn avec FastAPI

14.5 Développement d'API automatisé avec Pycaret

15 MLOps phase 3 : Service de modèles avec des applications Web

15.1 Servir le modèle via une application Web

15.2 Commandes Gradio de base

15.3 Développement d'une application web Gradio pour le Machine Learning

15.4 Développement d'applications Web automatisées avec Pycaret

15.5 Développement d'applications Web avec Streamlit

15.6 Laboratoire Développement d'applications Web avec Streamlit et Altair

15.7 Laboratory Streamlit et Pycaret pour développer un service web ML

16 Flask pour le développement d'applications

16.1 Fondamentaux de Flask

16.2 Construire un projet du début à la fin avec Flask

16.3 Développement back-end avec Flask et développement front-end avec HTML et CSS

17 Docker et conteneurs en Machine Learning

17.1 Fondamentaux de Flask

17.2 Construire un projet du début à la fin avec Flask

17.3 Développement back-end avec Flask et développement front-end avec HTML et CSS

18 BentoML pour le développement automatisé de services ML

18.1 Introduction à BentoML pour générer des services ML

18.2 Générer un service ML avec BentoML

18.3 Mettre le service en production avec BentoML et Docker

18.4 Introduction à BentoML pour générer des services ML

18.5 GPU, prétraitement, validation des données et modèles multiples dans BentoML

18.6 Différents outils pour développer des services ML

18.7 Exercice : Utiliser BentoML pour développer un service ML

18.8 Solution de l'exercice : Utiliser BentoML pour développer un service ML

19 Déployer sur Azure Cloud avec Azure Container et Azure SDK

19.1 Introduction à l'apprentissage automatique dans le cloud

19.2 Mettre l'application ML en production dans Azure Container avec Docker

19.3 SDK et Azure Blob Storage pour le déploiement de modèles sur Azure

19.4 Formation de modèles et déploiement en production dans Azure Blob Storage

19.5 Téléchargez le modèle Azure Blob Storage et obtenez des prédictions

20 Déploiement des services ML sur Heroku

20.1 Fondamentaux d'Heroku

20.2 Laboratoire pratique : déploiement d'un service ML sur Heroku

21 Section 21 : Intégration et livraison continues (CI/CD) avec Github Action et CML

21.1 Introduction aux actions GitHub

21.2 Flux de travail de base des actions GitHub

21.3 Atelier pratique sur les actions GitHub

21.4 CI avec apprentissage automatique continu (CML)

21.5 Atelier pratique : Application des actions GitHub et du CML aux MLOps

21.6 Laboratoire pratique : suivi des performances avec les actions GitHub et CML.

22 Surveillance des modèles et des services avec Evidently AI

22.1 Introduction à la surveillance des modèles et services ML

22.2 Dérive des données, dérive des concepts et performances du modèle

22.3 es fondamentaux de Evidently AI

22.4 Dérive et qualité des données, dérive de la cible et qualité du modèle

22.5 Laboratoire pratique : Surveillance d'un modèle avec Evidently AI

22.6 Laboratoire pratique : Suivi du modèle en production

22.7 Exercice : Utiliser BentoML pour développer un service ML

22.8 Laboratoire pratique : Identification des dérives de données en production

23 Projet MLOps de bout en bout

23.1 Projet MLOps de bout en boutProjet MLOps de bout en bout

23.2 Développement du modèle ML

23.3 Validation de la qualité du code, du modèle et du prétraitement

23.4 Versionnement de projet avec MLFlow et DVC

23.5 Dépôt partagé avec DagsHub et MLFlow

23.6 Développement d'API avec BentoML

23.7 Développement d'applications avec Streamlit

23.8 CI-CD : workflow de validation des données avec GitHub Actions

23.9 CI/CD : validation des fonctionnalités de l'application avec les actions GitHub

23.10 CI/CD : déploiement automatisé d'applications avec GitHub Actions et Heroku