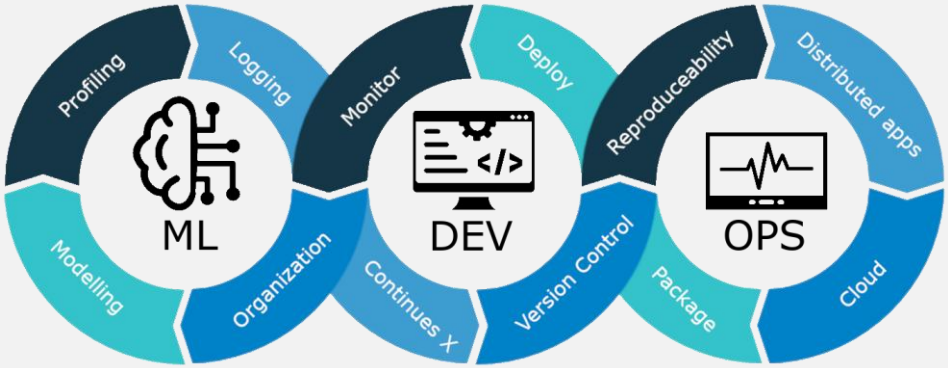
Intégration continue dans MLOps

UP ASI
Département informatique
Bureau: E204



Définition

- Pour les projets ML, l'intégration continue couvre la création et la vérification des aspects du processus de développement du modèle.
- Dans ces mêmes projets, l'intégration continue assure aussi **l'intégration et la validation automatique du code et des éléments**, en y **ajoutant l'évaluation régulière de la qualité et de la consistance des données ainsi que des performances des modèles de ML**.

Étapes

→ Plusieurs étapes peuvent être incluses dans la phase CI d'un projet ML parmi lesquelles nous pouvons citer :

- Le déclenchement automatique de la partie CI suite à n'importe quelles modifications au niveau du code/données/fichiers de conf/etc…
- La gestion du code source
- La vérification de la qualité du code
- Le formatage automatique du code
- La vérification de la sécurité du code
- Les tests unitaires et les tests fonctionnels
- La validation des données
- La vérification de la Qualité du Modèle (Validation et Evaluation)
- La gestion des modèles et du cycle de vie

Étapes

→ Déclenchement automatique

- Le déclenchement automatique de la partie **intégration continue (CI)** dans un pipeline **MLOps** repose sur l'automatisation du processus de validation du code, de tests et de déploiement de modèles dès qu'un changement est apporté dans le code ou les données.
- Ce processus implique l'intégration de différents outils et services pour garantir que chaque modification de code (ou de données) entraîne une série d'actions prédéfinies pour vérifier et valider la qualité du code, des modèles et des données avant de déployer en production.

- **Outils Courants** : Crontab (pour linux) / webhooks (git)/ etc ...

Étapes

01- La gestion du Code Source (Versioning)

- Cette étape consiste à garantir que le code soit bien versionné et stocké dans des dépôts (repositories).

Outils courants: GitHub / GitLab / BitBucket / AWS Code Commit

Étapes

02- La vérification de la qualité du code

- Pour les projets ML, le **langage dominant est python**.
- Pour la vérification du code python, nous optons surtout pour le **linting** (Le linting est une pratique permettant de détecter les erreurs, les mauvaises pratiques ou les incohérences dans le code avant qu'il ne soit exécuté).
- Un **linter** va analyser le code pour s'assurer qu'il respecte certaines règles de style, de syntaxe et de bonnes pratiques.

Étapes

02- La vérification de la qualité du code

- L'objectif est de détecter les erreurs de syntaxe, les variables non utilisées d'une part et d'assurer une cohérence de style de code pour le rendre plus lisible et maintenable d'autre part.

Outils courants : Pylint / Flake8 / MyPy

Étapes

03- La vérification de la sécurité du code

- Les tests de sécurité sont essentiels pour identifier les vulnérabilités dans le code.
- L'objectif est d'identifier les vulnérabilités de sécurité dans le code (comme l'injection SQL, les fuites d'informations sensibles, les erreurs de gestion des exceptions, etc.).
- Il faut s'assurer que le code est résistant aux attaques des hackers.
- Cette étape permet de prévenir les problèmes de sécurité (hotspots)

Étapes

03- La vérification de la sécurité du code

- Avant que le code n'atteigne l'environnement de production final.

Outils courants : Bandit, Safety, PyUp, SonarQube

Bandit

PyUp

sonarqube

Étapes

04- Les tests unitaires et les tests fonctionnels

- Les tests unitaires vérifient que chaque partie du code fonctionne correctement de manière isolée.
- Pour le code de machine learning, cela inclut souvent des tests sur les fonctions de prétraitement, les transformations de données, ou même des composants du modèle.
- L'essentiel est de vérifier que chaque fonction du code se comporte comme prévu.

Étapes

04- Les tests unitaires et les tests fonctionnels

- Assurer que les composants du système ne génèrent pas de régressions lorsque le code est modifié.

Outils courants : Pytest, unittest, nose2

Étapes

05- Validation des données (Data Validation)

- Dans les projets de machine learning, les données sont au cœur du processus.
- La validation des données permet de s'assurer que les données utilisées dans le modèle respectent des critères spécifiques de qualité avant l'entraînement.
- Nous devons nous assurer que les données utilisées pour l'entraînement, la validation et le test sont complètes et cohérentes.

Étapes

05- Validation des données (Data Validation)

- Il faut aussi vérifier que les nouvelles données introduites dans le pipeline n'ont pas d'anomalies susceptibles de perturber l'entraînement.

Outils courants : TensorFlow Data Validation : Un outil pour vérifier la qualité des données dans les projets utilisant TensorFlow.

TensorFlow

Étapes

06- Vérification de la Qualité du Modèle (Validation et Evaluation)

- Cette étape permet de vérifier que le modèle respecte les critères de performance avant d'être déployé. En mesurant des métriques sur la précision du modèle et en s'assurant que la modification des données n'affecte pas les performances du modèle.

Outils courants : MLflow, TensorBoard, etc.

Étapes

07- Gestion des modèles et du cycle de vie

- Dans cette phase, nous sommes amenés à gérer les différentes versions des modèles, d'assurer leur déploiement, et de suivre leur performance.

Outils courants : MLFlow / KubeFlow

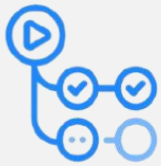
Outils

- Pour l 'orchestration des différentes étapes d'un pipeline d'un projet ML, plusieurs outils peuvent être utilisés :
  - ✓**Jenkins :** permet de créer des pipelines de manière flexible à l'aide de plugins et de scripts.
  - ✓**GitLab CI/CD :** automatiser les tests et déploiements et synchronisé avec les projets dont le code est disponible sur gitLab
  - ✓**Azure DevOps :** Un service Microsoft pour automatiser le processus CI/CD.

Jenkins

Outils

- Pour l 'orchestration des différentes étapes d'un pipeline d'un projet ML, plusieurs outils peuvent être utilisés :
  - ✓**GitHub Actions :** permet d'automatiser les flux de travail directement depuis GitHub. Il offre la possibilité de définir des actions pour construire, tester et déployer des applications.
  - ✓**Argo Workflows :** moteur d'orchestration des tâches basé sur Kubernetes
  - ✓**Bitbucket Pipelines**
  - ✓**CircleCI, Travis CI**
  - ✓ **Makefile**, etc …



GitHub Actions

Bitbucket Pipelines

Makefile
THE ORIGINAL BUILD TOOL

argo workflows

circleci

Travis CI

Outils - MakeFile

- Un Makefile est un fichier utilisé par l'outil make pour l'automatisation et la gestion de tâches répétitives.
- Il est plus simple, et est souvent utilisé dans des projets plus petits ou moins complexes, là où l'automatisation ne nécessite pas l'intégration avec des outils externes ou des processus complexes.
- Il permet d'automatiser des tâches locales, telles que la vérification du code, le prétraitement des données, l'entraînement des modèles, mais il manque de fonctionnalités intégrées pour des processus de déploiement et de suivi.

Intégration continue dans MLOps

Si vous avez des questions, n'hésitez pas à nous contacter :

Département Informatique

UP Architectures des Systèmes d'Information

Bureau E204 /E304