3ème Année 2ITE
Semestre 5
2020 - 2021

Etude comparative entre les outils d'automatisation et d'orchestration des tâche ML: Airflow vs. Luigi vs. Argo vs. MLFlow vs. KubeFlow

Visualisation et fouille Big Data

Présenté par
ARJANE Khadija

Supervisé par
Prof. F. KALLOUBI

Solution !

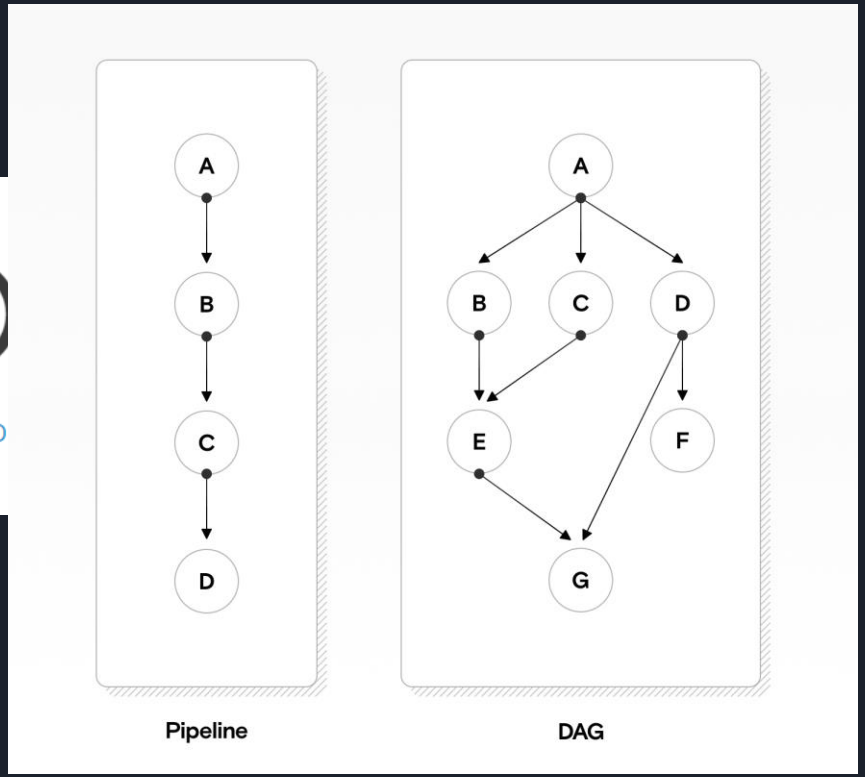
Ce réseau peut être modélisé comme un DAG - un graphe acyclique dirigé, qui modélise chaque tâche et les dép

Un
dép

Les
de c
elles dependent les unes des autres. L'outil execute ensuite ces tâches dans les délais, dans le bon ordre, en réessayant celles qui échouent avant d'exécuter les suivantes. Il surveille également les progrès et avertit votre équipe en cas d'échec.

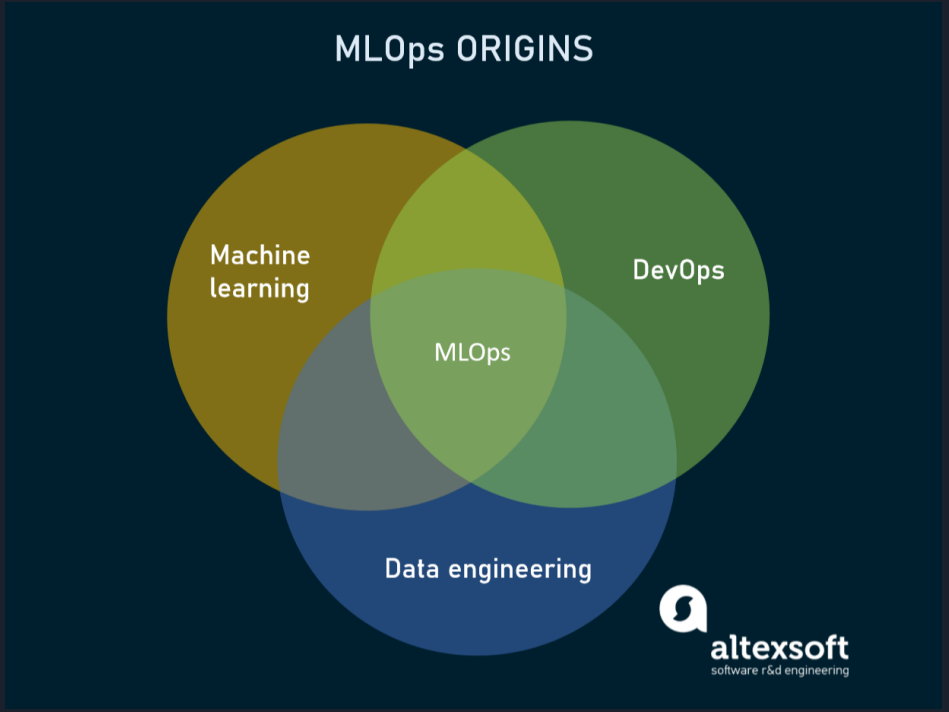
UNE MÉTHODOLOGIE ÉPROUVÉE

MODÈLE DE GESTION D



Qu'est-ce que MLOps ?

Fusion des termes «apprentissage automatique» et «opérations», MLOps est un ensemble de méthodes utilisées pour automatiser le cycle de vie des algorithmes d'apprentissage automatique en production de la formation initiale du modèle au déploiement en passant par le recyclage sur de nouvelles données.

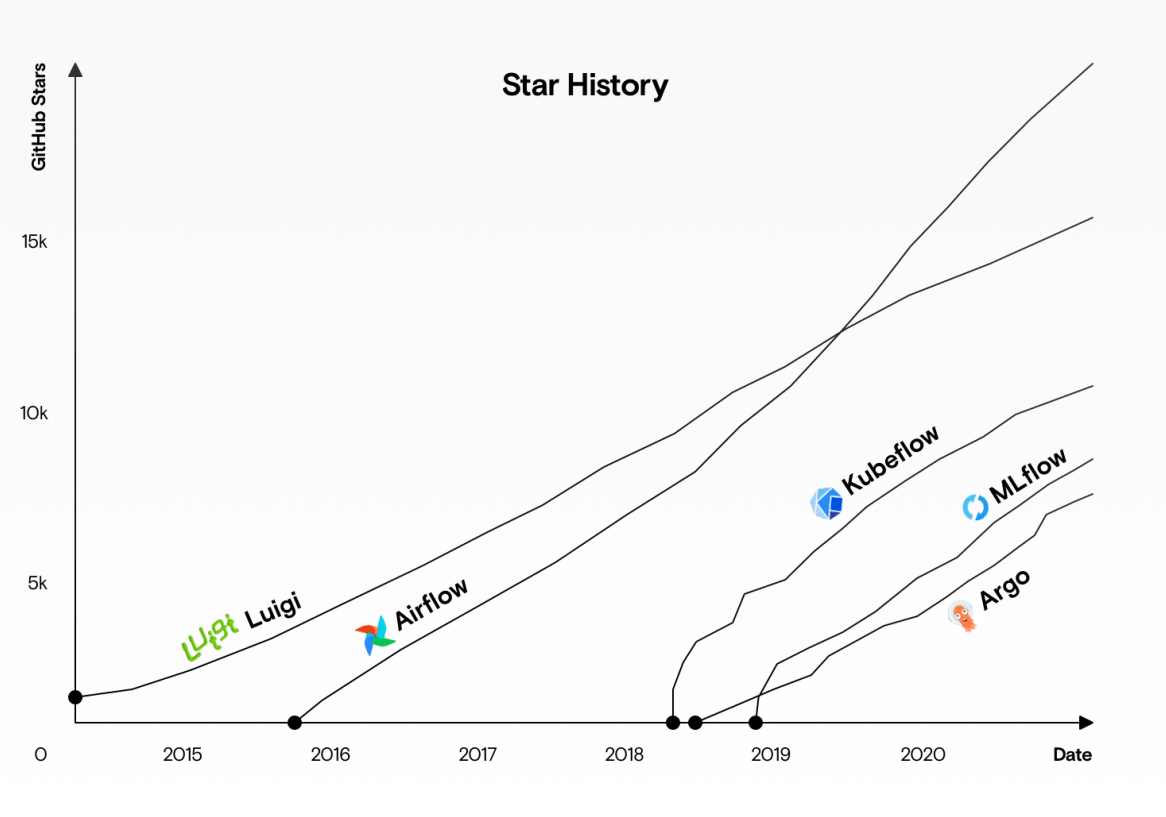


pourquoi MLOPS est utile?

L'adoption de MLOps promet aux organisations les avantages suivants:

- **Plus de temps pour le développement de nouveaux modèles**
- **Réduction du temps de mise sur le marché des modèles ML**
- **Meilleure expérience utilisateur**
- **Meilleure qualité des prévisions**

Etude comparative



Dis-moi lequel utiliser ?

- **Apache Airflow** si vous voulez l'outil le plus complet et le plus mature et que vous pouvez consacrer du temps à apprendre comment il fonctionne, à le configurer et à le maintenir.

- **Luigi** si vous avez besoin de quelque chose avec une courbe d'apprentissage plus facile que Airflow. Il a moins de fonctionnalités, mais il est plus facile de décoller.
- **Argo** si vous êtes déjà profondément investi dans l'écosystème Kubernetes et que vous souhaitez gérer toutes vos tâches en tant que pods, en les définissant en YAML au lieu de Python.
- **KubeFlow** si vous souhaitez utiliser Kubernetes tout en définissant vos tâches avec Python au lieu de YAML.
- **MLFlow** si vous vous souciez davantage du suivi des expériences ou du suivi et du déploiement de modèles à l'aide des modèles prédéfinis de MLFlow que de la recherche d'un outil capable de s'adapter à vos flux de travail personnalisés existants.

Tableau de comparaison

	Maturity	Popularity	Simplicity	Breadth	Language
Apache Airflow	B	A	C	A	Python
Luigi	B	A	A	B	Python
Argo	C	B	B	B	YAML
Kubeflow	C	B	B	C	Python
MLFlow	C	B	A	C	Python

Maturité: en fonction de l'âge du projet et du nombre de correctifs et de commits;
Popularité: basée sur l'adoption et les étoiles GitHub;
Simplicité: basée sur la facilité d'intégration et d'adoption;
Ampleur: en fonction de la spécialisation et de l'adaptabilité de chaque projet;
Langue: en fonction de la principale façon dont vous interagissez avec l'outil.

Luigi

Luigi est une bibliothèque basée sur Python pour l'orchestration générale des tâches, et peut être installée avec des outils de gestion de packages Python, tels que pip et conda. Luigi est beaucoup plus simple. Il est contenu dans un seul composant, Avec Luigi, vous devez écrire plus de code personnalisé pour exécuter des tâches selon un calendrier.

- Utilisez Luigi si vous avez une petite équipe et que vous devez vous lancer rapidement.
- Utilisez Luigi si vous avez besoin d'orchestrer une variété de tâches différentes, du nettoyage des données au déploiement du modèle.
- Il utilise Python et les DAG pour définir les tâches et les dépendances.

Argo

Argo est construit sur Kubernetes et chaque tâche est exécutée en tant que pod Kubernetes distinct. Cela peut être pratique si vous utilisez déjà Kubernetes pour la plupart de votre infrastructure, mais cela ajoutera de la complexité si vous ne l'êtes pas.Argo est une extension Kubernetes et est installé à l'aide de Kubernetes.

- Utilisez Argo si vous êtes déjà investi dans Kubernetes et que vous savez que toutes vos tâches seront des pods.
- Argo exécute chaque tâche comme un pod Kubernetes.
- Utilisez Argo si vous êtes déjà investi dans Kubernetes et que vous souhaitez exécuter une grande variété de tâches écrites dans différentes piles.
- Il utilise Yaml et les DAG pour définir les tâches et les dépendances.

KubeFlow

Kubeflow est un outil basé sur Kubernetes spécifiquement pour les workflows d'apprentissage automatique. Kubeflow se compose de deux composants distincts: Kubeflow et Kubeflow Pipelines. Ce dernier est axé sur le déploiement de modèles et CI / CD, et il peut être utilisé indépendamment des principales fonctionnalités de Kubeflow. Kubeflow se concentre spécifiquement sur les tâches d'apprentissage automatique, telles que le suivi des expériences. Les deux outils vous permettent de définir des tâches à l'aide de Python, mais Kubeflow exécute des tâches sur Kubernetes. Kubeflow est divisé en Kubeflow et Kubeflow Pipelines: ce dernier composant vous permet de spécifier des DAG, mais il est plus axé sur le déploiement et le service de modèles que sur les tâches générales.

- Utilisez Kubeflow si vous utilisez déjà Kubernetes et que vous souhaitez orchestrer des tâches d'apprentissage automatique courantes telles que le suivi des expériences et la formation de modèles.
- Utilisez Kubeflow si vous utilisez déjà Kubernetes et souhaitez davantage de modèles prêts à l'emploi pour les solutions d'apprentissage automatique.
- Utilisez Kubeflow si vous utilisez déjà Kubernetes et souhaitez davantage de modèles prêts à l'emploi pour les solutions d'apprentissage automatique.

MLFlow

MLFlow est un outil plus spécialisé pour vous aider à gérer et suivre votre cycle de vie et vos expériences d'apprentissage automatique. vous pouvez importer MLFlow directement dans votre code d'apprentissage automatique et utiliser sa fonction d'assistance pour consigner des informations (telles que les paramètres que vous utilisez) et artefacts (tels que les modèles entraînés). Vous pouvez également utiliser MLFlow comme outil de ligne de commande pour servir des modèles créés avec des outils courants (tels que scikit-learn) ou les déployer sur des plates-formes courantes (telles qu'AzureML ou Amazon SageMaker). MLFlow est une bibliothèque Python que vous pouvez importer dans votre code de machine learning existant et un outil de ligne de commande que vous pouvez utiliser pour entraîner et déployer des modèles de machine learning écrits en scikit-learn sur Amazon SageMaker ou AzureML.

- Utilisez MLFlow si vous voulez une manière avisée et prête à l'emploi de gérer vos expériences et déploiements de machine learning.
- Utilisez MLFlow si vous souhaitez une manière avisée de gérer votre cycle de vie de machine learning avec des plates-formes cloud gérées.

AirFlow

Airflow est une plateforme générique d'orchestration de tâches, il a une communauté plus large et quelques fonctionnalités supplémentaires, mais une courbe d'apprentissage beaucoup plus raide. Plus précisément, Airflow est beaucoup plus puissant en matière de planification et fournit une interface utilisateur de calendrier pour vous aider à configurer le moment où vos tâches doivent s'exécuter.

- Utilisez Airflow si vous avez une équipe plus nombreuse et que vous pouvez prendre un premier coup de productivité en échange de plus de puissance une fois que vous avez dépassé la courbe d'apprentissage.
- Utilisez Airflow si vous avez besoin d'un vaste écosystème mature qui peut exécuter une variété de tâches différentes.
- Utilisez Airflow si vous avez des exigences plus complexes et souhaitez plus de contrôle sur la façon dont vous gérez le cycle de vie de votre machine learning.
- Il utilise Python et les DAG pour définir les tâches et les dépendances.

Conclusion

Pas de solution miracle!
Bien que tous ces outils aient des points de focalisation différents et des forces différentes, aucun outil ne vous offrira un processus sans maux de tête dès la sortie de la boîte. Avant de vous demander quel outil choisir, il est généralement important de vous assurer que vous disposez de bons processus, y compris une bonne culture d'équipe, des rétrospectives sans blâme et des objectifs à long terme.

Alors pour notre manipulation nous allons utilliser Apache Airflow parce que il est à la fois l'outil le plus populaire et celui avec la plus large gamme de fonctionnalités, et en terme de performance il est **beaucoup plus puissant et riche de fonctionnalités.**